Neural Networks 23 (2010) 89–107

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

Clustering: A neural network approach☆

K.-L. Du *

Department of Electrical and Computer Engineering, Concordia University, 1455 de Maisonneuve West, Montreal, Canada, H3G 1M8

ARTICLE INFO

Article history:
Received 10 September 2007
Accepted 13 August 2009

Keywords:
Clustering
Neural network
Competitive learning
Competitive learning network
Vector quantization

ABSTRACT

Clustering is a fundamental data analysis method. It is widely used for pattern recognition, feature extraction, vector quantization (VQ), image segmentation, function approximation, and data mining. As an unsupervised classification technique, clustering identifies some inherent structures present in a set of objects based on a similarity measure. Clustering methods can be based on statistical model identification (McLachlan & Basford, 1988) or competitive learning. In this paper, we give a comprehensive overview of competitive learning based clustering methods. Importance is attached to a number of competitive learning based clustering neural networks such as the self-organizing map (SOM), the learning vector quantization (LVQ), the neural gas, and the ART model, and clustering algorithms such as the C-means, mountain/subtractive clustering, and fuzzy C-means (FCM) algorithms. Associated topics such as the under-utilization problem, fuzzy clustering, robust clustering, clustering based on non-Euclidean distance measures, supervised clustering, hierarchical clustering as well as cluster validity are also described. Two examples are given to demonstrate the use of the clustering methods.

1. Introduction

Vector quantization (VQ) is a classical method for approximating a continuous probability density function (PDF) $p(\mathbf{x})$ of the vector variable $\mathbf{x} \in R^n$ by using a finite number of prototypes. A set of feature vectors $\mathbf{x}$ is represented by a finite set of prototypes $\{\mathbf{c}_1, \ldots, \mathbf{c}_K\} \subset R^n$, referred to as the codebook. Codebook design can be performed by using clustering. Once the codebook is specified, approximation of $\mathbf{x}$ is to find the reference vector $\mathbf{c}$ from the codebook that is closest to $\mathbf{x}$ (Kohonen, 1989, 1997). This is the nearest-neighbor paradigm, and the procedure is actually simple competitive learning (SCL).

The codebook can be designed by minimizing the expected squared quantization error

$$E = \int \|\mathbf{x} - \mathbf{c}\|^2 p(\mathbf{x}) \mathrm{d}\mathbf{x} \tag{1}$$

where $\mathbf{c}$ is a function of $\mathbf{x}$ and $\mathbf{c}_i$. Given the sample $\mathbf{x}_t$, an iterative approximation scheme for finding the codebook is derived by Kohonen (1997)

$$\mathbf{c}_i(t+1) = \mathbf{c}_i(t) + \eta(t)\delta_{wi}[\mathbf{x}_t - \mathbf{c}_i(t)] \tag{2}$$

where the subscript w corresponds to the winning prototype, which is the prototype closest to $\mathbf{x}_t$, δ_{wi} is the Kronecker delta, with δ_{wi} taking 1 for $w = i$ and 0 otherwise, and $\eta > 0$ is a small learning rate that satisfies the classical Robbins–Monro conditions, that is, $\sum \eta(t) = \infty$ and $\sum \eta^2(t) < \infty$. Typically, η is selected to be decreasing monotonically in time. For example, one can select $\eta(t) = \eta_0\left(1 - \frac{t}{T}\right)$, where $\eta_0 \in (0, 1]$ and T is the iteration bound. This is the SCL based VQ.

Voronoi tessellation, also called Voronoi diagram, is useful for demonstrating VQ results. The space is partitioned into a finite number of regions bordered by hyperplanes. Each region is represented by a codebook vector, which is the nearest neighbor to any point within the region. All vectors in each region constitute a Voronoi set. For a smooth underlying probability density $p(\mathbf{x})$ and a large K, all regions in an optimal Voronoi partition have the same within-region variance σ_k (Gersho, 1979).

Given a competitive learning based clustering method, learning is first conducted to adjust the algorithmic parameters; after the learning phase is completed, the network is ready for generalization. When a new input pattern $\mathbf{x}$ is presented to the map, the map gives the corresponding output $\mathbf{c}$ based on the nearest-neighborhood rule. Clustering is a fundamental data analysis method, and is widely used for pattern recognition, feature extraction, VQ, image segmentation, and data mining. In this paper, we provide a comprehensive introduction to clustering. Various clustering techniques based on competitive learning are described. The paper is organized as follows. In Section 2, we give an introduction to competitive learning. In Section 3, the Kohonen network and the self-organizing map (SOM) are treated. Section 4 is dedicated to learning vector quantization (LVQ). Sections 5–7 deal with the C-means, mountain/subtractive, and neural gas clustering methods, respectively. ART and ARTMAP models are treated in Section 8.

☆ This work was supported by the NSERC of Canada.

* Tel.: +1 514 8482424x7015.
E-mail addresses: kldu@ieee.org, kldu@ece.concordia.ca.

0893-6080/$ – see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.neunet.2009.08.007

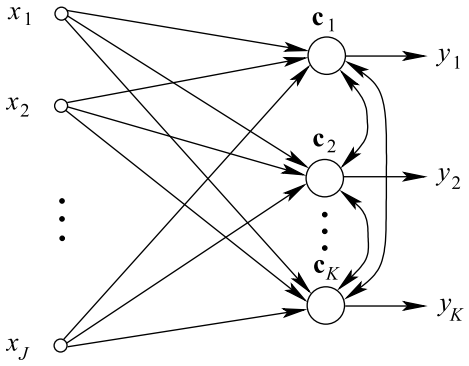

Fig. 1. Architecture of the competitive learning network. The output selects one of the prototypes $\mathbf{c}_i$ by setting $y_i = 1$ and all $y_j = 0, j \neq i$.

Fuzzy clustering is described in Section 9, and supervised clustering are described in Section 10. In Section 11, the under-utilization problem as well as strategies for avoiding this problem is narrated. Robust clustering is treated in Section 12, and clustering using non-Euclidean distance measures is coped with in Section 13. Hierarchical clustering and its hybridization with partitional clustering are described in Section 14. Constructive clustering methods and other clustering methods are introduced in Sections 15 and 16, respectively. Some cluster validity criteria are given in Section 17. Two examples are given in Section 18 to demonstrate the use of the clustering methods. We wind up by a summary in Section 19.

2. Competitive learning

Competitive learning can be implemented using a two-layer (J–K) neural network, as shown in Fig. 1. The input and output layers are fully connected. The output layer is called the competition layer, wherein lateral connections are used to perform lateral inhibition.

Based on the mathematical statistics problem called *cluster analysis*, competitive learning is usually derived by minimizing the mean squared error (MSE) functional (Tsypkin, 1973)

$$E = \frac{1}{N}\sum_{p=1}^{N} E_p \tag{3}$$

$$E_p = \sum_{k=1}^{K} \mu_{kp} \left\|\mathbf{x}_p - \mathbf{c}_k\right\|^2 \tag{4}$$

where N is the size of the pattern set, and μ_{kp} is the connection weight assigned to prototype $\mathbf{c}_k$ with respect to $\mathbf{x}_p$, denoting the membership of pattern p into cluster k. When $\mathbf{c}_k$ is the closest (winning) prototype to $\mathbf{x}_p$ in the Euclidean metric, $\mu_{kp} = 1$; otherwise $\mu_{kp} = 0$. The SCL is derived by minimizing (3) under the assumption that the weights are obtained by the nearest prototype condition. Thus

$$E_p = \min_{1\le k\le K} \left\|\mathbf{x}_p - \mathbf{c}_k\right\|^2 \tag{5}$$

which is the squared Euclidean distance between the input $\mathbf{x}_p$ and its closest prototype $\mathbf{c}_k$.

Based on the criterion (4) and the gradient-descent method, assuming $\mathbf{c}_w = \mathbf{c}_w(t)$ to be the winning prototype of $\mathbf{x} = \mathbf{x}_t$, we get the SCL as

$$\mathbf{c}_w(t+1) = \mathbf{c}_w(t) + \eta(t)\left[\mathbf{x}_t - \mathbf{c}_w(t)\right] \tag{6}$$

$$\mathbf{c}_i(t+1) = \mathbf{c}_i(t), \quad i \neq w \tag{7}$$

where $\eta(t)$ can be selected according to the Robbins–Monro conditions. The process is known as winner-take-all (WTA). The WTA mechanism plays an important role in most unsupervised learning networks. If each cluster has its own learning rate as $\eta_i = \frac{1}{N_i}$, N_i being the number of samples assigned to the ith cluster, the algorithm achieves the minimum output variance (Yair, Zeger, & Gersho, 1992).

Many WTA models were implemented based on the continuous-time Hopfield network topology (Dempsey & McVey, 1993; Majani, Erlanson, & Abu-Mostafa, 1989; Sum et al., 1999; Tam, Sum, Leung, & Chan, 1996), or based on the cellular neural network (CNN) (Chua & Yang, 1988) model with linear circuit complexity (Andrew, 1996; Seiler & Nossek, 1993). There are also some circuits for realizing the WTA function (Lazzaro, Lyckebusch, Mahowald, & Mead, 1989; Tam et al., 1996). k-winners-take-all (k-WTA) is a process of selecting the k largest components from an N-dimensional vector. It is a key task in decision making, pattern recognition, associative memories, or competitive learning networks. k-WTA networks are usually based on the continuous-time Hopfield network (Calvert & Marinov, 2000; Majani et al., 1989; Yen, Guo, & Chen, 1998), and k-WTA circuits (Lazzaro et al., 1989; Urahama & Nagao, 1995) can be implemented using the Hopfield network based on the penalty method and have infinite resolution.

3. The Kohonen network

Von der Malsburg's model (von der Malsburg, 1973) and Kohonen's self-organization map (SOM) (Kohonen, 1982, 1989) are two topology-preserving competitive learning models that are inspired by the cortex of mammals. The SOM is popular for VQ, clustering analysis, feature extraction, and data visualization.

The Kohonen network has the same structure as the competitive learning network. The output layer is called the Kohonen layer. Lateral connections are used as a form of feedback whose magnitude is dependent on the lateral distance from a specific neuron, which is characterized by a neighborhood parameter. The Kohonen network defined on R^n is a one-, two-, or higher-dimensional grid A of neurons characterized by prototypes $\mathbf{c}_k \in R^n$ (Kohonen, 1989, 1990). Input patterns are presented sequentially through the input layer, without specifying the desired output. The Kohonen network is called the SOM when the lateral feedback is more sophisticated than the WTA rule. For example, the lateral feedback used in the SOM can be selected as the Mexican hat function, which is found in the visual cortex. The SOM is more successful in classification and pattern recognition.

3.1. The self-organizing map

The SOM computes the Euclidean distance of the input pattern $\mathbf{x}$ to each neuron k, and find the winning neuron, denoted neuron w with prototype $\mathbf{c}_w$, using the nearest-neighbor rule. The winning node is called the excitation center.

For all the input vectors that are closest to $\mathbf{c}_w$, update all the prototype vectors by the Kohonen learning rule (Kohonen, 1990)

$$\mathbf{c}_k(t+1) = \mathbf{c}_k(t) + \eta(t)h_{kw}(t)\left[\mathbf{x}_t - \mathbf{c}_k(t)\right], \quad k = 1, \ldots, K \tag{8}$$

where $\eta(t)$ satisfies the Robbins–Monro conditions, and $h_{kw}(t)$ is the excitation response or neighbor function, which defines the response of neuron k when $\mathbf{c}_w$ is the excitation center. If $h_{kw}(t)$ takes δ_{kw}, (8) reduces to the SCL. $h_{kw}(t)$ can be selected as a function that decreases with the increasing distance between $\mathbf{c}_k$ and $\mathbf{c}_w$, and typically as the Gaussian function

$$h_{wk}(t) = h_0 \mathrm{e}^{-\frac{\|\mathbf{c}_k - \mathbf{c}_w\|^2}{\sigma^2(t)}} \tag{9}$$

where the constant $h_0 > 0, \sigma(t)$ is a decreasing function of t with a popular choice, $\sigma(t) = \sigma_0 \mathrm{e}^{-\frac{t}{\tau}}$, σ_0 being a positive constant and τ a time constant (Obermayer, Ritter, & Schulten, 1991). The Gaussian

function is biologically more reasonable than a rectangular one. The SOM using the Gaussian neighborhood converges more quickly than that using a rectangular one (Lo & Bavarian, 1991).

$\mathbf{c}_k(0)$ can be selected as random values, or from available samples, or any ordered initial state. The algorithm terminated when the map achieves an equilibrium with a given accuracy or when a specified number of iterations is reached. In the convergence phase, h_{wk} can be selected as time-invariant, and each prototype can be updated by using an individual learning rate η_k Kohonen (1997)

$$\eta_k(t+1) = \frac{\eta_k(t)}{1 + h_{wk}\eta_k(t)}. \tag{10}$$

Normalization of $\mathbf{x}$ is suggested since the resulting reference vectors tend to have the same dynamic range. This may improve the numerical accuracy (Kohonen, 1990).

The SOM (Kohonen, 1989) is a clustering network with a set of heuristic procedures: it is not based on the minimization of any known objective function. It suffers from several major problems, such as forced termination, unguaranteed convergence, non-optimized procedure, and the output being often dependent on the sequence of data. The Kohonen network is closely related to the C-means clustering (Lippman, 1987). There are some proofs for the convergence of the one-dimensional SOM based on the Markov chain analysis (Flanagan, 1996), but no general proof of convergence for multi-dimensional SOM is available (Flanagan, 1996; Kohonen, 1997).

The SOM performs clustering while preserving topology. It is useful for VQ, clustering, feature extraction, and data visualization. The Kohonen learning rule is a major development of competitive learning. The SOM is related to adaptive C-means, but performs a topological feature map which is more complex than just cluster analysis. After training, the input vectors are spatially ordered in the array. The Kohonen learning rule provides a codebook in which the distortion effects are automatically taken into account. The SOM is especially powerful for the visualization of high-dimensional data. It converts complex, nonlinear statistical relations between high-dimensional data into simple geometric relations at a low-dimensional display. The SOM can be used to decompose complex information processing systems into a set of simple subsystems (Gao, Ahmad, & Swamy, 1991). A fully analog integrated circuit of the SOM has been designed in Mann and Gilbert (1989). A comprehensive survey of SOM applications is given in Kohonen (1996).

However, the SOM is not a good choice in terms of clustering performance compared to other popular clustering algorithms such as the C-means, the neural gas, and the ART 2A (He, Tan, & Tan, 2004; Martinetz, Berkovich, & Schulten, 1993). For large output dimensions, the number of nodes in the adaptive grid increases exponentially with the number of function parameters. The prespecified standard grid topology may not be able to match the structure of the distribution, leading to poor topological mappings.

3.2. Extensions of the self-organizing Map

Adaptive subspace SOM (ASSOM) (Kohonen, 1996, 1997; Kohonen, Oja, Simula, Visa, & Kangas, 1996) is a modular neural network model comprising an array of topologically ordered SOM submodels. ASSOM creates a set of local subspace representations by competitive selection and cooperative learning. Each submodel is responsible for describing a specific region of the input space by its local principal subspace, and represents a manifold such as a linear subspace with a small dimensionality, whose basis vectors are determined adaptively. ASSOM not only inherits the topological representation property of the SOM, but provides learning results which reasonably describe the kernels of various transformation groups like the PCA. The hyperbolic SOM (HSOM) (Ritter, 1999) implements its lattice by a regular triangulation of the hyperbolic plane. The hyperbolic lattice provides more freedom to map a complex information space such as language into spatial relations.

Extraction of knowledge from databases is an essential task of data analysis and data mining. The multi-dimensional data may involve quantitative and qualitative (nominal, ordinal) variables such as categorical data, which is the case in survey data. The SOM can be viewed as an extension of principal component analysis (PCA) due to its topology-preserving property. For qualitative variables, the SOM has been generalized for multiple correspondence analysis (Cottrell, Ibbou, & Letremy, 2004).

The SOM is designed for real-valued vectorial data analysis, and it is not suitable for non-vectorial data analysis such as the structured data analysis. Examples of structured data are temporal sequences such as time series, language, and words, spatial sequences like the DNA chains, and tree or graph structured data arising from natural language parsing and from chemistry. Prominent unsupervised self-organizing methods for non-vectorial data are the temporal Kohonen map (TKM), the recurrent SOM (RSOM), the recursive SOM (RecSOM), the SOM for structured data (SOMSD), and the merge SOM (MSOM). All these models introduce recurrence into the SOM, and have been reviewed and compared in Hammer, Micheli, Sperduti, and Strickert (2004) and Strickert and Hammer (2005).

4. Learning vector quantization

The k-nearest-neighbor (k-NN) algorithm (Duda & Hart, 1973) is a conventional classification technique. It is also used for outlier detection. It generalizes well for large training sets, and the training set can be extended at any time. The theoretical asymptotic classification error is upper-bounded by twice the Bayes error. However, it uses a large storage space, and has a computational complexity of $O\left(N^2\right)$. It also takes a long time for recall.

LVQ (Kohonen, 1990) employs the same network architecture as the competitive learning network. The unsupervised LVQ is essentially the SCL based VQ. There are two families of the LVQ-style models, supervised models such as the LVQ1, th LVQ2, and the LVQ3 (Kohonen, 1989) as well as unsupervised models such as the LVQ (Kohonen, 1989) and the incremental C-means (MacQueen, 1967). The supervised LVQ is based on the known classification of feature vectors, and can be treated as a supervised version of the SOM. The LVQ is used for VQ and classification, as well as for fine tuning the SOM (Kohonen, 1989, 1990). LVQ algorithms define near-optimal decision borders between classes, even in the sense of classical Bayesian decision theory.

The supervised LVQ minimizes the functional (3), where $\mu_{kp} = 1$ if neuron k is the winner and zero otherwise, when pattern pair p is presented. It works on a set of N pattern pairs $\left(\mathbf{x}_p, \mathbf{y}_p\right)$, where $\mathbf{x}_p \in R^J$ is the input vector and $\mathbf{y}_p \in R^K$ is the binary target vector coding the class membership, that is, only one entry of $\mathbf{y}_p$ takes the value unity while all its other entries are zero. Assuming that the pth pattern is presented at time t, the LVQ1 is given as (Kohonen, 1990)

$$\begin{aligned}
&\mathbf{c}_w(t+1) = \mathbf{c}_w(t) + \eta(k)\left[\mathbf{x}_t - \mathbf{c}_w(t)\right], \quad y_{p,w} = 1\\
&\mathbf{c}_w(t+1) = \mathbf{c}_w(t) - \eta(t)\left[\mathbf{x}_t - \mathbf{c}_w(k)\right], \quad y_{p,w} = 0\\
&\mathbf{c}_i(t+1) = \mathbf{c}_i(t), \quad i \neq w
\end{aligned} \tag{11}$$

where w is the index of the winning neuron, $\mathbf{x}_t = \mathbf{x}_p$ and $\eta(t)$ is defined as in earlier formulations. When it is used to fine-tune the SOM, one should start with a small $\eta(0)$, usually less than 0.1. This algorithm tends to reduce the point density of $\mathbf{c}_i$ around the Bayesian decision surfaces. The OLVQ1 is an optimized version of

the LVQ1 (Kohonen, Kangas, Laaksonen, & Torkkola, 1992). In the OLVQ1, each codebook vector $\mathbf{c}_i$ is assigned an individual adaptive learning rate η_i. The OLVQ1 converges at a rate up to one order of magnitude faster than the LVQ1.

LVQ2 and LVQ3 comply better with the Bayesian decision surface. In LVQ1, only one codebook vector $\mathbf{c}_i$ is updated at each step, while LVQ2 and LVQ3 change two codebook vectors simultaneously. Different LVQ algorithms can be combined in the clustering process. However, both LVQ2 and LVQ3 have the problem of reference vector divergence (Sato & Yamada, 1995). In a generalization of the LVQ2 (Sato & Yamada, 1995), this problem is eliminated by applying gradient descent on a nonlinear cost function. Some applications of the LVQ were reviewed in Kohonen et al. (1996).

Addition of training counters to individual neurons can effectively record the training statistics of the LVQ (Odorico, 1997). This allows for dynamic self-allocation of the neurons to classes during the course of training. At the generalization stage, these counters provide an estimate of the reliability of classification of the individual neurons. The method is especially valuable in handling strongly overlapping class distributions in the pattern space.

5. *C*-means clustering

The most well-known data clustering technique is the statistical *C*-means, also known as the *k*-means (MacQueen, 1967; Moody & Darken, 1989; Tou & Gonzalez, 1976). The *C*-means algorithm approximates the maximum likelihood (ML) solution for determining the location of the means of a mixture density of component densities. The *C*-means clustering is closely related to the SCL, and is a special case of the SOM. The algorithm partitions the set of N input patterns into K separate subsets $\mathcal{C}_k$, each containing N_k input patterns by minimizing the MSE

$$E(\mathbf{c}_1, \ldots, \mathbf{c}_K) = \frac{1}{N} \sum_{k=1}^{K} \sum_{\mathbf{x}_n \in \mathcal{C}_k} \|\mathbf{x}_n - \mathbf{c}_k\|^2 \tag{12}$$

where $\mathbf{c}_k$ is the prototype or center of the cluster $\mathcal{C}_k$. By minimizing E with respect to $\mathbf{c}_k$, the optimal location of $\mathbf{c}_k$ is obtained as the mean of the samples in the cluster, $\mathbf{c}_k = \frac{1}{N_k} \sum_{\mathbf{x}_i \in \mathcal{C}_k} \mathbf{x}_i$.

The *C*-means can be implemented in either the batch mode (Linde, Buzo, & Gray, 1980; Moody & Darken, 1989) or the incremental mode (MacQueen, 1967). The batch *C*-means (Linde et al., 1980), also called the *Linde–Buzo–Gray*, *LBG* or *generalized Lloyd algorithm*, is applied when the whole training set is available. The incremental *C*-means is suitable for a training set that is obtained on-line. In the batch *C*-means, the initial partition is arbitrarily defined by placing each input pattern into a randomly selected cluster, and the prototypes are defined to be the average of the patterns in the individual clusters. When the *C*-means is performed, at each step the patterns keep changing from one cluster to the closest cluster $\mathbf{c}_k$ according to the nearest-neighbor rule and the prototypes are then recalculated as the mean of the samples in the clusters. In the incremental *C*-means, each cluster is initialized with a random pattern as its prototype; the *C*-means updates the prototypes upon the presentation of each new pattern. The incremental *C*-means gives the new prototype as

$$\mathbf{c}_k(t+1) = \begin{cases} \mathbf{c}_k(t) + \eta(t)\,(\mathbf{x}_t - \mathbf{c}_k(t)), & k = w \\ \mathbf{c}_k(t), & k \neq w \end{cases} \tag{13}$$

where w is the index of the winning neuron, $\eta(t)$ is defined as in earlier formulations. The general procedure for the *C*-means clustering is to repeat the redistribution of patterns among the clusters using criterion (12) until there is no further change in the prototypes of the clusters. After the algorithm converges, one can calculate the variance vector $\vec{\sigma}_k$ for each cluster.

As a gradient-descent technique, the *C*-means achieves a local optimum solution that depends on the initial selection of the cluster prototypes. The number of clusters must also be prespecified. Numerous improvements on the *C*-means have been made. The local minimum problem can be eliminated by using global optimization methods such as the genetic algorithm (GA) (Bandyopadhyay & Maulik, 2002; Krishna & Murty, 1999), the simulated annealing (SA) (Bandyopadhyay, Maulik, & Pakhira, 2001), and a hybrid SA and evolutionary algorithm (EA) system (Delport, 1996). In Chinrunrueng and Sequin (1995), the incremental *C*-means is improved by biasing the clustering towards an optimal Voronoi partition (Gersho, 1979) via a cluster variance-weighted MSE as the objective function, and by adjusting the learning rate dynamically according to the current variances in all partitions. The method always converges to an optimal or near-optimum configuration. The enhanced LBG (Patane & Russo, 2001) avoids bad local minima by incorporation of the concept of utility of a codeword. The enhanced LBG outperforms the LBG with utility (LBG-U) (Fritzke, 1997b) both in terms of accuracy and the number of required iterations. The LBG-U is also based on the LBG and the concept of utility.

When an initial prototype is in a region with few training patterns, this results in a large cluster. This disadvantage can be remedied by a modified *C*-means (Wilpon & Rabiner, 1985). The clustering starts from one cluster. It splits the cluster with the largest intracluster distance into two. After each splitting, the *C*-means is applied until the existing clusters are convergent. This procedure is continued until K clusters are obtained.

The relation between the PCA and the *C*-means has been established in Ding and He (2004). Principal components have been proved to be the continuous solutions to the discrete cluster membership indicators for the *C*-means clustering, with a clear simplex cluster structure (Ding & He, 2004). PCA based dimensionality reductions are particularly effective for the *C*-means clustering. Lower bounds for the *C*-means objective function (12) are derived as the total variance minus the eigenvalues of the data covariance matrix (Ding & He, 2004).

In the two-stage clustering procedure (Vesanto & Alhoniemi, 2000), the SOM is first used to cluster the data set, and the prototypes produced are further clustered using an agglomerative clustering algorithm or the *C*-means. The clustering results using the SOM as an intermediate step are comparable to that of direct clustering of the data, but with a significantly reduced computation time.

6. Mountain and subtractive clusterings

The mountain clustering (Yager & Filev, 1994a, 1994b) is a simple and effective method for estimating the number of clusters and the initial locations of the cluster centers. The method grids the data space and computes a potential value for each grid point based on its distance to the actual data points. Each grid point is a potential cluster center. The potential for each grid is calculated based on the density of the surrounding data points. The grid with the highest potential is selected as the first cluster center and then the potential values of all the other grids are reduced according to their distances to the first cluster center. The next cluster center is located at the grid point with the highest remaining potential. This process is repeated until the remaining potential values of all the grids fall below a threshold. However, the grid structure causes the complexity to grow exponentially with the dimension of the problem.

The subtractive clustering (Chiu, 1994a), as a modified mountain clustering, uses all the data points to replace all the grid points as potential cluster centers. This effectively reduces the number of grid points to N (Chiu, 1994a). The potential measure for each data point $\mathbf{x}_i$ is defined as a function of the Euclidean distances to all the other input data points

$$P(i) = \sum_{j=1}^{N} \mathrm{e}^{-\alpha \|\mathbf{x}_i - \mathbf{x}_j\|^2}, \quad i = 1, \ldots, N \tag{14}$$

where the constant $\alpha = \frac{4}{r_a^2}$, r_a being a normalized radius defining the neighborhood. A data point surrounded by many neighboring

data points has a high potential value. Thus, the mountain and subtractive clustering techniques are less sensitive to noise than other clustering algorithms, such as the C-means and the fuzzy C-means (FCM) (Bezdek, 1981).

After the data point with the highest potential, $\mathbf{x}_u$, is selected as the kth cluster center, that is, $\mathbf{c}_k = \mathbf{x}_u$ with $\overline{P}(k) = P(u)$ as its potential value, the potential of each data point $\mathbf{x}_i$ is modified by subtracting a term associated with $\mathbf{c}_k$

$$P(i) = P(i) - \overline{P}(k)\mathrm{e}^{-\beta\|\mathbf{x}_i - \mathbf{c}_k\|^2} \tag{15}$$

where the constant $\beta = \frac{4}{r_b^2}$, r_b being a normalized radius defining the neighborhood. In order to avoid closely located cluster centers, r_b is set greater than r_a, typically $r_b = 1.25r_a$. The algorithm continues until the remaining potentials of all the data points are below some fraction of the potential of the first cluster center

$$\overline{P}(k) = \max_i P(i) < \varepsilon\overline{P}(1) \tag{16}$$

where ε is selected within (0, 1). A small ε leads to a large number of hidden nodes, while a large ε generates a small network structure. Typically, ε is selected as 0.15.

The training data $\mathbf{x}_i$ is recommended to be scaled before applying the method for easy selection of α and β. Since it is difficult to select a suitable ε for all data patterns, additional criteria for accepting/rejecting cluster centers can be used. One method is to select two thresholds (Chiu, 1994a, 1994b), namely, $\overline{\varepsilon}$ and $\underline{\varepsilon}$. Above $\overline{\varepsilon}$, $\mathbf{c}_k$ is definitely accepted as a cluster center, while below $\underline{\varepsilon}$ it is definitely rejected. If $\overline{P}(k)$ falls between the two thresholds, a trade-off between a reasonable potential and its distance to the existing cluster centers must been examined.

Unlike the C-means and the FCM, which require iterations of many epochs, the subtractive clustering requires only one pass of the training data. Besides, the number of clusters does not need to be prespecified. The subtractive clustering is a deterministic method: For the same neural network structure, the same network parameters are always obtained. Both the C-means and the FCM require $O(KNT)$ computations, where T is the total number of epochs and each computation requires the calculation of the distance and the memberships. The computational load for the subtractive clustering is $O\left(N^2 + KN\right)$, each computation involving the calculation of the exponential function. Thus, for small- or medium-size training sets, the subtractive clustering is relatively fast, but it requires more training time when $N \gg KT$ (Dave & Krishnapuram, 1997).

The subtractive clustering provides only rough estimates of the cluster centers, since the cluster centers obtained are situated at some data points. Moreover, since α and β are not determined from the data set and no cluster validity is used, the clusters produced may not appropriately represent the clusters. The result by the subtractive clustering can be used for initializing iterative optimization based clustering algorithms such as the C-means and the FCM.

The subtractive clustering can be improved by performing a search over α and β, which makes it essentially equivalent to the least-biased fuzzy clustering algorithm (Beni & Liu, 1994). The least-biased fuzzy clustering, based on the deterministic annealing approach (Rose, 1998; Rose, Gurewitz, & Fox, 1990), tries to minimize the clustering entropy of each cluster under the assumption of unbiased centroids. In Angelov and Filev (2004), an on-line clustering method has been implemented based on a first-order Cauchy type potential function. In Pal and Chakraborty (2000), the mountain and subtractive clustering methods are improved by tuning the prototypes obtained using the gradient-descent method to maximize the potential function. By modifying the potential function, the mountain method can also be used to detect other types of clusters like circular shells (Pal & Chakraborty, 2000). In Kim, Lee, Lee, and Lee (2005), a kernel-induced distance is used to replace the Euclidean distance in the potential function. This enables to cluster the data that is linearly inseparable in the original space into homogeneous groups in the transformed high-dimensional space, where the data separability is increased.

7. Neural gas

The neural gas (NG) (Martinetz et al., 1993) is a VQ model which minimizes a known cost function and converges to the C-means quantization error via a soft-to-hard competitive model transition. The soft-to-hard annealing process helps the algorithm escape from local minima. The NG is a topology-preserving network, and can be treated as an extension to the C-means. It has a fixed number of processing units, K, with no lateral connection.

A data optimal topological ordering is achieved by using neighborhood ranking within the input space at each training step. To find its neighborhood rank, each neuron compares its distance to the input vector with those of all the other neurons to the input vector. Neighborhood ranking provides the training strategy with mechanisms related to robust statistics, and the NG does not suffer from the prototype under-utilization problem (Rumelhart & Zipser, 1985). At step t, the Euclidean distances between an input vector $\mathbf{x}_t$ and all the prototype vectors $\mathbf{c}_k(t)$ are calculated by $d_k(\mathbf{x}_t) = \|\mathbf{x}_t - \mathbf{c}_k(t)\|$, $k = 1, \ldots, K$, and $\mathbf{d}(t) = (d_1(\mathbf{x}_t), \ldots, d_K(\mathbf{x}_t))^{\mathrm{T}}$. Each prototype $\mathbf{c}_k(t)$ is assigned a rank $r_k(t)$, which takes an integer value from $0, \ldots, K-1$, with 0 for the smallest and $K-1$ for the largest $d_k(\mathbf{x}_t)$.

The prototypes are updated by

$$\mathbf{c}_k(t+1) = \mathbf{c}_k(t) + \eta h(r_k(t))(\mathbf{x}_t - \mathbf{c}_k(t)) \tag{17}$$

where $h(r) = \mathrm{e}^{-\frac{r}{\rho(t)}}$ realizes a soft competition, $\rho(t)$ being the neighborhood width. When $\rho(t) \to 0$, (17) reduces to the C-means update rule (13). During the iteration, both $\rho(t)$ and $\eta(t)$ decrease exponentially, $\eta(t) = \eta_0\left(\frac{\eta_\mathrm{f}}{\eta_0}\right)^{\frac{t}{T_\mathrm{f}}}$ and $\rho(t) = \rho_0\left(\frac{\rho_\mathrm{f}}{\rho_0}\right)^{\frac{t}{T_\mathrm{f}}}$, where η_0 and ρ_0 are the initial decay parameters, η_f and ρ_f are the final decay parameters, and T_f is the maximum number of iterations. The prototypes $\mathbf{c}_k$ are initialized by randomly assigning vectors from the training set.

Unlike the SOM, which uses predefined static neighborhood relations, the NG determines a dynamical neighborhood relation as learning proceeds. The NG is an efficient and reliable clustering algorithm, which is not sensitive to the neuron initialization. The NG converges faster to a smaller MSE E than the C-means, the maximum-entropy clustering (Rose et al., 1990), and the SOM. This advantage comes at the price of a higher computational effort. In serial implementation, the complexity for the NG is $O(K \log K)$ while the other three methods all have a complexity of $O(K)$. Nevertheless, in parallel implementation all the four algorithms have a complexity of $O(\log K)$ (Martinetz et al., 1993). The NG can be derived from a gradient-descent procedure on a potential function associated with the framework of fuzzy clustering (Bezdek, 1981).

To accelerate the sequential NG, a truncated exponential function is used as the neighborhood function and the neighborhood ranking is implemented without evaluating and sorting all the distances (Choy & Siu, 1998b). In Rovetta and Zunino (1999), an improved NG and its analog VLSI subcircuitry have been developed based on partial sorting. The approach reduces the training time by up to two orders of magnitude, without reducing the performance.

In the Voronoi tessellation, when the prototype of each Voronoi region is connected to all the prototypes of its bordering Voronoi regions, a Delaunay triangulation is obtained. Competitive Hebbian learning (Martinetz, 1993; Martinetz & Schulten, 1994) is a method that generates a subgraph of the Delaunay triangulation, called

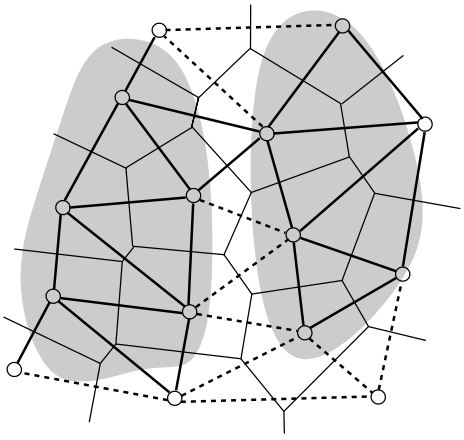

Fig. 2. An illustration of the Delaunay triangulation and the induced Delaunay triangulation. The Delaunay triangulation is represented by a mix of thick and thick dashed lines, the induced Delaunay triangulation by thick lines, Voronoi tessellation by thin lines, prototypes by circles, and a data distribution $P(\mathbf{x})$ by shaded regions. To generate the induced Delaunay triangulation, two prototypes are connected only if at least a part of the common border of their Voronoi polygons lies in a region where $P(\mathbf{x}) > 0$.

the *induced Delaunay triangulation* by masking the Delaunay triangulation with a data distribution $P(\mathbf{x})$. This is shown in Fig. 2. The induced Delaunay triangulation is optimally topology-preserving in a general sense (Martinetz, 1993). Given a number of prototypes in R^l, competitive Hebbian learning successively adds connections among them by evaluating input data drawn from $P(\mathbf{x})$. The method does not change the prototypes, but only generates topology according to these prototypes. For each input $\mathbf{x}$, its two closest prototypes are connected by an edge. This leads to the induced Delaunay triangulation, which is limited to those regions of the input space R^l, where $P(\mathbf{x}) > 0$. The topology-representing network (Martinetz & Schulten, 1994) is obtained by alternating the learning steps of the NG and the competitive Hebbian learning, where the NG is used to distribute a certain number of prototypes and the competitive Hebbian learning is then used to generate the topology. An edge aging scheme is used to remove obsolete edges. Competitive Hebbian learning avoids the topological defects observed for the SOM.

8. ART networks

Adaptive resonance theory (ART) (Grossberg, 1976) is biologically motivated and is a major advance in the competitive learning paradigm. The theory leads to a series of real-time unsupervised network models for clustering, pattern recognition, and associative memory (Carpenter & Grossberg, 1987a, 1987b, 1988, 1990; Carpenter, Grossberg, & Rosen, 1991a, 1991b; Carpenter, Grossberg, Markuzon, Reynolds, & Rosen, 1992). These models are capable of stable category recognition in response to arbitrary input sequences with either fast or slow learning. ART models are characterized by systems of differential equations that formulate stable self-organizing learning methods. Instar and outstar learning rules are the two learning rules used. The ART has the ability to adapt, yet not forget the past training, and it overcomes the so-called stability–plasticity dilemma (Carpenter & Grossberg, 1987a; Grossberg, 1976). At the training stage, the stored prototype of a category is adapted when an input pattern is sufficiently similar to the prototype. When novelty is detected, the ART adaptively and autonomously creates a new category with the input pattern as the prototype. The similarity is characterized by a vigilance parameter $\rho \in (0, 1]$. A large ρ leads to many finely divided categories, while a smaller ρ gives fewer categories. The stability and plasticity properties as well as the ability to efficiently process dynamic data make the ART attractive for clustering large, rapidly changing sequences of input patterns, such as in the case of data mining (Massey, 2003). However, the ART approach does not correspond to the C-means algorithm for cluster analysis and VQ in the global optimization sense (Lippman, 1987).

8.1. ART models

ART model family includes a series of unsupervised learning models. ART networks employ a J–K recurrent architecture, which is a different form of Fig. 1. The input layer F1, called the *comparing layer*, has J neurons while the output layer F2, called the *recognizing layer*, has K neurons. F1 and F2 are fully interconnected in both directions. F2 acts as a WTA network. The feedforward weights connecting to the F2 neuron j are represented by the vector $\mathbf{w}_j$, while the feedback weights from the same neuron are represented by the vector $\mathbf{c}_j$ that stores the prototype of cluster j. The number of clusters K varies with the size of the problem.

The ART models are characterized by a set of short-term memory (STM) and long-term memory (LTM) time-domain nonlinear differential equations. The STM equations describe the evolution of the neurons and their interactions, while the LTM equations describe the change of the interconnection weights with time as a function of the system state. F1 stores the STM for the current input pattern, while F2 stores the prototypes of clusters as the LTM. There are three types of ART implementations: full mode, STM steady-state mode, and fast learning mode (Carpenter & Grossberg, 1987b; Serrano-Gotarredona & Linares-Barranco, 1996). In the full mode, both the STM and LTM differential equations are realized. The STM steady-state mode only implements the LTM differential equations, while the STM behavior is governed by nonlinear algebraic equations. In the fast learning mode, both the STM and the LTM are implemented by their steady-state nonlinear algebraic equations, and thus proper sequencing of STM and LTM events is required. The fast learning mode is inexpensive and is most popular.

Like the incremental C-means, the ART model family is sensitive to the order of presentation of the input patterns. ART models tend to build clusters of the same size, independently of the distribution of the data.

8.1.1. ART 1

The simplest and most popular ART model is the ART 1 (Carpenter & Grossberg, 1987a) for learning to categorize arbitrarily many, complex binary input patterns presented in an arbitrary order. A popular fast learning implementation is given by Du and Swamy (2006), Moore (1988), Massey (2003) and Serrano-Gotarredona and Linares-Barranco (1996). The ART 1 is stable for a finite training set. However, the order of the training patterns may influence the final prototypes and clusters. Unlike the SOM (Kohonen, 1982), the Hopfield network (Hopfield, 1982), and the neocognitron (Fukushima, 1980), the ART 1 can deal with arbitrary combinations of binary input patterns. In addition, the ART 1 has no restriction on memory capacity since its memory matrices are not square.

Other popular ART 1-based clustering algorithms are the improved ART 1 (IART 1) (Shih, Moh, & Chang, 1992), the adaptive Hamming net (AHN) (Hung & Lin, 1995), the fuzzy ART (Carpenter et al., 1991a, 1992; Carpenter & Ross, 1995), the fuzzy AHN (Hung & Lin, 1995), and the projective ART (PART) (Cao & Wu, 2002). The fuzzy ART (Carpenter et al., 1991a) simply extends the logical AND in the ART 1 to the fuzzy AND. Both the fuzzy ART and the fuzzy AHN have an analog architecture, and function like the ART 1 but for analog input patterns.

The ART models, typically governed by differential equations, have a high computational complexity for numerical implementations. Implementations using analog or optical hardware are more

desirable. A modified ART 1 in the fast learning mode has been derived for easy hardware implementation in Serrano-Gotarredona and Linares-Barranco (1996), and the method has also been extended for the full mode and the STM steady-state mode. A number of hardware implementations of the ART 1 in different modes are also surveyed in Serrano-Gotarredona and Linares-Barranco (1996).

8.1.2. ART 2

The ART 2 (Carpenter & Grossberg, 1987b) is designed to categorize analog or binary random input sequences. It is similar to the ART 1, but has a more complex F1 field so as to allow the ART 2 to stably categorize sequences of analog inputs that can be arbitrarily close to one another. The F1 field includes a combination of normalization and noise suppression, as well as the comparison of the bottom-up and top-down signals needed for the reset mechanism. The clustering behavior of the ART 2 was found to be similar to that of the C-means clustering (Burke, 1991).

The ART 2 is computationally expensive and has difficulties in parameter selection. The ART 2A (Carpenter et al., 1991b) employs the same architecture as the ART 2, and can accurately reproduce the behavior of the ART 2 in the fast learning limit. The ART 2A is two to three orders of magnitude faster than the ART 2, and also suggests efficient parallel implementations. The ART 2A is also fast at intermediate learning rates, which captures many desirable properties of slow learning of the ART 2 such as noise tolerance. In Carpenter and Grossberg (1987b), F2 initially contains a number of uncommitted nodes, which get committed one by one upon the input presentation. An implementation of the ART 2A, with F2 being initialized as the null set and dynamically growing during learning, is given in Du and Swamy (2006); He et al. (2004). The ART 2A with an intermediate learning rate η copes better with noisy inputs than it does with a fast learning rate, and the emergent category structure is less dependent on the input presentation order (Carpenter et al., 1991b). The ART-C 2A (He et al., 2004) applies a constraint reset mechanism on the ART 2A to allow a direct control on the number of output clusters generated, by adaptively adjusting the value of ρ. The ART 2A and the ART-C 2A have clustering quality comparable to that of the C-means and the SOM, but with less computational time He et al. (2004).

8.1.3. Other ART models

The ART 3 (Carpenter & Grossberg, 1990) carries out parallel searches by testing hypotheses about distributed recognition codes in a multilevel network hierarchy. The ART 3 introduces a search process for ART architectures that can robustly cope with sequences of asynchronous analog input patterns in real time. The distributed ART (dART) (Carpenter, 1997) combines the stable fast learning capability of ART systems with the noise tolerance and code compression capabilities of the multilayer perceptron (MLP). With a WTA code, the unsupervised dART model reduces to the fuzzy ART (Carpenter et al., 1991a). Other ART-based algorithms include the efficient ART (EART) family (Baraldi & Alpaydin, 2002), the simplified ART (SART) family (Baraldi & Alpaydin, 2002), the symmetric fuzzy ART (S-Fuzzy ART) (Baraldi & Alpaydin, 2002), the Gaussian ART (Williamson, 1996) as an instance of SART family, and the fully self-organizing SART (FOSART) Baraldi and Parmiggiani (1997).

8.2. ARTMAP models

ARTMAP models (Carpenter, Grossberg, & Reynolds, 1991; Carpenter et al., 1992; Carpenter & Ross, 1995), which are self-organizing and goal-oriented, are a class of supervised learning methods. The ARTMAP, also called predictive ART, autonomously learns to classify arbitrarily many, arbitrarily ordered vectors into recognition categories based on predictive success (Carpenter et al., 1991). Compared to the backpropagation (BP) learning (Rumelhart, Hinton, & Williams, 1986), the ARTMAP has a number of advantages such as being self-organizing, self-stabilizing, match learning, and real time. The ARTMAP learns orders of magnitude faster and is also more accurate than the BP. These are achieved by using an internal controller that jointly maximizes predictive generalization and minimizes predictive error by linking predictive success to category size on a trial-by-trial basis, using only local operations. However, the ARTMAP is very sensitive to the order of the training patterns compared to learning by the radial basis function network (RBFN) (Broomhead & Lowe, 1988).

The ARTMAP learns predetermined categories of binary input patterns in a supervised manner. It is based on a pair of ART modules, namely, ART_a and ART_b. ART_a and ART_b can be fast learning ART 1 modules coding binary input vectors. These modules are connected by an inter-ART module that resembles ART 1. The inter-ART module includes a map field that controls the learning of an associative map from ART_a recognition categories to ART_b recognition categories. The map field also controls match tracking of the ART_a vigilance parameter. The inter-ART vigilance resetting signal is a form of backpropagation of information. Given a stream of input–output pairs $\{(\mathbf{x}_p, \mathbf{y}_p)\}$. During training, ART_a receives a stream $\{\mathbf{x}_p\}$ and ART_b receives a stream $\{\mathbf{y}_p\}$. During generalization, when a pattern $\mathbf{x}$ is presented to ART_a, its prediction is produced at ART_b.

The fuzzy ARTMAP (Carpenter et al., 1992; Carpenter & Ross, 1995) can be taught to supervisedly learn predetermined categories of binary or analog input patterns. The fuzzy ARTMAP incorporates two fuzzy ART modules. The fuzzy ARTMAP is capable of fast, but stable, on-line recognition learning, hypothesis testing, and adaptive naming in response to an arbitrary stream of analog or binary input patterns. The fuzzy ARTMAP is also shown to be a universal approximator (Verzi, Heileman, Georgiopoulos, & Anagnostopoulos, 2003). Other members of the ARTMAP family are the ART-EMAP (Carpenter & Ross, 1995), the ARTMAP-IC (Carpenter & Markuzon, 1998), the Gaussian ARTMAP (Williamson, 1996), the distributed ARTMAP (dARTMAP) (Carpenter, 1997), the default ARTMAP (Carpenter, 2003), and the simplified fuzzy ARTMAP (Kasuba, 1993; Vakil-Baghmisheh & Pavesic, 2003). The distributed vs. the WTA-coding representation is a primary factor differentiating the various ARTMAP networks. The relations of some of the ARTMAP variants are given by Carpenter (2003): fuzzy ARTMAP $\subset$ default ARTMAP $\subset$ ARTMAP-IC $\subset$ dARTMAP.

9. Fuzzy clustering

Fuzzy clustering is an important class of clustering algorithms. Fuzzy clustering helps to find natural vague boundaries in data. Preliminaries of fuzzy sets and logic are given in Buckley and Eslami (2002) and Du and Swamy (2006).

9.1. Fuzzy C-means clustering

The discreteness of each cluster makes the C-means analytically and algorithmically intractable. Partitioning the dataset in a fuzzy manner avoids this problem. The FCM clustering (Bezdek, 1974, 1981), also known as the fuzzy ISODATA (Dunn, 1974), treats each cluster as a fuzzy set, and each feature vector is assigned to multiple clusters with some degree of certainty measured by the membership function. The FCM optimizes the following objective function (Bezdek, 1974, 1981)

$$E = \sum_{j=1}^{K} \sum_{i=1}^{N} \mu_{ji}^{m} \left\| \mathbf{x}_i - \mathbf{c}_j \right\|^2 \tag{18}$$

where the membership matrix $\mathbf{U} = \{\mu_{ji}\}$, $\mu_{ji} \in [0, 1]$ denoting the membership of $\mathbf{x}_i$ into cluster j. The condition must be valid

$$\sum_{j=1}^{K} \mu_{ji} = 1, \quad i = 1, \ldots, N. \tag{19}$$

The weighting parameter $m \in (1, \infty)$ is called the fuzzifier. m determines the fuzziness of the partition produced, and reduces the influence of small membership values. When $m \to 1+$, the resulting partition asymptotically approaches a hard or crisp partition. On the other hand, the partition becomes a maximally fuzzy partition if $m \to \infty$.

By minimizing (18) subject to (19), the optimal solution is derived as

$$\mu_{ji} = \frac{\left(\frac{1}{\|\mathbf{x}_i - \mathbf{c}_j\|^2}\right)^{\frac{1}{m-1}}}{\sum_{l=1}^{K}\left(\frac{1}{\|\mathbf{x}_i - \mathbf{c}_l\|^2}\right)^{\frac{1}{m-1}}}, \tag{20}$$

$$\mathbf{c}_j = \frac{\sum_{i=1}^{N}\left(\mu_{ji}\right)^m \mathbf{x}_i}{\sum_{i=1}^{N}\left(\mu_{ji}\right)^m} \tag{21}$$

for $i = 1, \ldots, N$, $j = 1, \ldots, K$. Eq. (20) corresponds to a softmax rule and (21) is similar to the mean of the data points in a cluster. Both equations are dependent on each other. The iterative alternating optimization procedure terminates when the change in the prototypes is sufficiently small (Bezdek, 1981; Karayiannis & Mi, 1997). The FCM clustering with a high degree of fuzziness diminishes the probability of getting stuck at local minima (Bezdek, 1981). A typical value for m is 1.5 or 2.0.

The FCM needs to store $\mathbf{U}$ and all $\mathbf{c}_i$'s, and the alternating estimation of $\mathbf{U}$ and $\mathbf{c}_i$'s causes a computational and storage burden for large-scale data sets. The computation can be accelerated by combining their updates (Kolen & Hutcheson, 2002), and consequently the storage of $\mathbf{U}$ is avoided. The single iteration time of the accelerated method is $O(K)$, while that of the FCM is $O(K^2)$ (Kolen & Hutcheson, 2002). The C-means is a special case of the FCM, when μ_{ji} is unity for only one class and zero for all the other classes. Like the C-means, the FCM may find a local optimum solution, and the result is dependent on the initialization of $\mathbf{U}$ or $\mathbf{c}_j(0)$.

There are many variants of the FCM. The penalized FCM (Yang, 1993) is a convergent generalized FCM obtained by adding a penalty term associated with μ_{ji}. The compensated FCM (Lin, 1999) speeds up the convergence of the penalized FCM by modifying the penalty. A weighted FCM (Tsekouras, Sarimveis, Kavakli, & Bafas, 2004) is used for fuzzy modeling towards developing a Takagi–Sugeno–Kang (TSK) fuzzy model of optimal structure. All these and many other existing generalizations of the FCM can be analyzed in a unified framework called the generalized FCM (GFCM) (Yu & Yang, 2005), by using the Lagrange multiplier method from an objective function comprising a generalization of the FCM criterion and a regularization term. The multistage random sampling FCM (Cheng, Goldgof, & Hall, 1998) reduces the clustering time normally by a factor of 2 to 3, with a quality of the final partitions equivalent to that created by the FCM. The FCM has been generalized by introducing the generalized Boltzmann distribution to escape local minima (Richardt, Karl, & Muller, 1998). Existing global optimization techniques can be incorporated into the FCM to provide globally optimum solutions. The ε-insensitive FCM (εFCM) is an extension to the FCM by introducing the robust statistics using Vapnik's ε-insensitive estimator to reduce the effect of outliers (Leski, 2003a). The εFCM is based on L_1-norm clustering (Kersten, 1999). Other robust extensions to the FCM includes the L_p-norm clustering ($0 < p < 1$) (Hathaway & Bezdek, 2000) and the L_1-norm clustering (Kersten, 1999). The FCM has also been extended for clustering other data types, such as symbolic data (El-Sonbaty & Ismail, 1998).

For a blend of unlabeled and labeled patterns, the FCM with partial supervision (Pedrycz & Waletzky, 1997) can be applied and the method is derived following the same procedure as that of the FCM. The classification information is added to the objective function, and a weighting factor balances the supervised and unsupervised terms within the objective function (Pedrycz & Waletzky, 1997). The conditional FCM (Pedrycz, 1998) develops clusters preserving homogeneity of the clustered patterns with regard to their similarity in the input space, as well as their respective values assumed in the output space. It is a supervised clustering. The conditional FCM is based on the FCM, but requires the output variable of a cluster to satisfy a particular condition, which can be treated as a fuzzy set, defined via the corresponding membership. This results in a reduced computational complexity for classification problems by splitting the problem into a series of condition-driven clustering problems. A family of generalized weighted conditional FCM algorithms are derived in Leski (2003b).

9.2. Other fuzzy clustering algorithms

Many other clustering algorithms are based on the concept of fuzzy membership. The Gustafson–Kessel algorithm (Gustafson & Kessel, 1979) extends the FCM by using the Mahalanobis distance, and is suited for hyperellipsoidal clusters of equal volume. The algorithm takes typically five times as long as the FCM to complete cluster formation (Karayiannis & Randolph-Gips, 2003). The adaptive fuzzy clustering (AFC) (Anderson, Bezdek, & Dave, 1982) also employs the Mahalanobis distance, and is suitable for ellipsoidal or linear clusters. The Gath–Geva algorithm (Gath & Geva, 1989) is derived from a combination of the FCM and fuzzy ML estimation. The method incorporates the hypervolume and density criteria as cluster validity measures and performs well in situations of large variability of cluster shapes, densities, and number of data points in each cluster.

The C-means and the FCM are based on the minimization of the trace of the (fuzzy) within-cluster scatter matrix. The minimum scatter volume (MSV) and minimum cluster volume (MCV) algorithms are two iterative clustering algorithms based on determinant (volume) criteria (Krishnapuram & Kim, 2000). The MSV algorithm minimizes the determinant of the sum of the scatter matrices of the clusters, while the MCV minimizes the sum of the volumes of the individual clusters. The behavior of the MSV is similar to that of the C-means, whereas the MCV is more versatile. The MCV in general gives better results than the C-means, MSV, and Gustafson–Kessel algorithms, and is less sensitive to initialization than the expectation–maximization (EM) algorithm (Dempster, Laird, & Rubin, 1977). Volume prototypes extend the cluster prototypes from points to regions in the clustering space (Kaymak & Setnes, 2002). A cluster represented by a volume prototype implies that all data points close to a cluster center belong fully to that cluster. In Kaymak and Setnes (2002), the Gustafson–Kessel algorithm and the FCM have been extended by using the volume prototypes and similarity-driven merging of clusters.

There are various fuzzy clustering methods that are based on the Kohonen network, the LVQ, the ART models, and the Hopfield network.

9.2.1. Kohonen network and learning vector quantization based fuzzy clustering

The fuzzy SOM (Huntsberger & Ajjimarangsee, 1990) modifies the SOM by replacing the learning rate with fuzzy membership

of the nodes in each class. The fuzzy LVQ (FLVQ) (Bezdek & Pal, 1995), originally named the fuzzy Kohonen clustering network (FKCN) (Bezdek, Tsao, & Pal, 1992), is a batch algorithm that combines the ideas of fuzzy membership values for learning rates, the parallelism of the FCM, and the structure and self-organizing update rules of the Kohonen network. Soft competitive learning in clustering has the same function as fuzzy clustering (Baraldi & Blonda, 1999). The soft competition scheme (SCS) (Yair et al., 1992) is a sequential, deterministic version of LVQ, obtained by modifying the neighborhood mechanism of the Kohonen learning rule and incorporating the stochastic relaxation technique. The SCS consistently provides better codebooks than the incremental C-means (Linde et al., 1980), even for the same computation time, and is relatively insensitive to the choice of the initial codebook. The learning rates of the FLVQ and SCS algorithms have opposite tendencies (Bezdek & Pal, 1995). The SCS has difficulty in selecting good parameters (Bezdek & Pal, 1995). Other extensions to the FLVQ, LVQ, and FCM algorithms are the extended FLVQ family learning schemes (Karayiannis & Bezdek, 1997), the non-Euclidean FLVQ (NEFLVQ) and the non-Euclidean FCM (NEFCM) (Karayiannis & Randolph-Gips, 2003), the generalized LVQ (GLVQ) (Pal, Bezdek, & Tsao, 1993), the generalized LVQ family (GLVQ-F) (Karayiannis, Bezdek, Pal, Hathaway, & Pai, 1996), the family of fuzzy algorithms for LVQ (FALVQ) (Karayiannis, 1997; Karayiannis & Pai, 1996), entropy-constrained fuzzy clustering (ECFC) algorithms, and entropy-constrained LVQ (ECLVQ) algorithms (Karayiannis, 1999).

9.2.2. ART networks based fuzzy clustering

In Section 8.1, we have mentioned some fuzzy ART models such as the fuzzy ART, the S-fuzzy ART, and the fuzzy AHN, as well as some fuzzy ARTMAP models such as the fuzzy ARTMAP, the ART-EMAP, default ARTMAP, the ARTMAP-IC, and the dARTMAP. The supervised fuzzy min–max classification network (Simpson, 1992) as well as the unsupervised fuzzy min–max clustering network (Simpson, 1993) is a kind of combination of fuzzy logic and the ART 1 (Carpenter & Grossberg, 1987a). The operations in these models require only complements, additions and comparisons that are most suitable for parallel hardware execution. Some clustering and fuzzy clustering algorithms including the SOM (Kohonen, 1989), the FLVQ (Bezdek & Pal, 1995), the fuzzy ART (Carpenter et al., 1991a), the growing neural gas (GNG) (Fritzke, 1995a), and the FOSART (Baraldi & Parmiggiani, 1997) are surveyed and compared in Baraldi and Blonda (1999).

9.2.3. Hopfield network based fuzzy clustering

The clustering problem can be cast as a problem of minimization of the MSE between the training patterns and the cluster centers. This optimization problem can be solved using the Hopfield network (Lin, 1999; Lin, Cheng, & Mao, 1996). In the fuzzy Hopfield network (FHN) (Lin et al., 1996) and the compensated fuzzy Hopfield network (CFHN) (Lin, 1999), the training patterns are mapped to a Hopfield network of a two-dimensional neuron array, where each column represents a cluster and each row a training pattern. The state of each neuron corresponds to a fuzzy membership function. A fuzzy clustering strategy is included in the Hopfield network to eliminate the need for finding the weighting factors in the energy function. This energy function is called the scatter energy function, and is formulated based on the within-class scatter matrix. These models have inherent parallel structures. In the FHN (Lin et al., 1996), an FCM strategy is imposed for updating the neuron states. The CFHN (Lin, 1999) integrates the compensated FCM into the learning scheme and updating strategies of the Hopfield network to avoid the NP-hard problem (Swamy & Thulasiraman, 1981) and to accelerate the convergence for the clustering procedure. The CFHN learns more rapidly and more effectively than clustering using the Hopfield network, the FCM, and the penalized FCM (Yang, 1993). The CFHN has been used for VQ in image compression (Liu & Lin, 2000), so that the parallel implementation for codebook design is feasible.

10. Supervised clustering

When output patterns are used in clustering, this leads to supervised clustering. The locations of the cluster centers are determined by both the input pattern spread and the output pattern deviations. For classification problems, the class membership of each training pattern is available and can be used for clustering, thus significantly improving the decision accuracy. Examples of supervised clustering include the LVQ family (Kohonen, 1990), the ARTMAP family (Carpenter et al., 1991), the conditional FCM (Pedrycz, 1998), the supervised C-means (Al-Harbi & Rayward-Smith, 2006), and the C-means plus k-NN based clustering Bruzzone and Prieto (1998).

Supervised clustering can be implemented by augmenting the input pattern with its output pattern, $\widetilde{\mathbf{x}}_i = \left[\mathbf{x}_i^{\mathrm{T}}, \mathbf{y}_i^{\mathrm{T}}\right]^{\mathrm{T}}$, so as to obtain an improved distribution of the cluster centers by an unsupervised clustering (Chen, Chen, & Chang, 1993; Pedrycz, 1998; Runkler & Bezdek, 1999; Uykan, Guzelis, Celebi, & Koivo, 2000). A scaling factor β is introduced to balance between the similarities in the input and output spaces $\widetilde{\mathbf{x}} = \left[\mathbf{x}_i^{\mathrm{T}}, \beta\mathbf{y}_i^{\mathrm{T}}\right]^{\mathrm{T}}$ (Pedrycz, 1998). By applying the FCM, the new cluster centers $\mathbf{c}_j = \left[\mathbf{c}_{x,j}^{\mathrm{T}}, \mathbf{c}_{y,j}^{\mathrm{T}}\right]^{\mathrm{T}}$ are obtained. The resulting cluster codebook vectors are projected onto the input space to obtain the centers.

Based on the enhanced LBG (Patane & Russo, 2001), the clustering for function approximation (CFA) (Gonzalez, Rojas, Pomares, Ortega, & Prieto, 2002) algorithm is a supervised clustering method designed for function approximation. The CFA increases the density of the prototypes in the input areas where the target function presents a more variable response, rather than just in the zones with more input examples (Gonzalez et al., 2002). The CFA minimizes the variance of the output response of the training examples belonging to the same cluster. In Staiano, Tagliaferri, and Pedrycz (2006), a prototype regression function is built as a linear combination of local linear regression models, one for each cluster, and is then inserted into the FCM. Thus, the prototypes are adjusted according to both the input distribution and the regression function in the output space.

11. The under-utilization problem

Conventional competitive learning based clustering like the C-means or the LVQ suffers from a severe initialization problem called prototype under-utilization or dead-unit problem, since some prototypes, called dead units (Grossberg, 1987; Rumelhart & Zipser, 1985), may never win the competition. This problem is caused by the fact that only the winning prototype is updated for every input. Initializing the prototypes with random input vectors can reduce the probability of the under-utilization problem, but does not eliminate it. Many efforts have been made to solve the under-utilization problem.

11.1. Competitive learning with conscience

In the leaky learning strategy (Grossberg, 1987; Rumelhart & Zipser, 1985), all the prototypes are updated. The winning prototype is updated by employing a fast learning rate, while all the losing prototypes move towards the input vector with a much slower learning rate. Each processing unit is assigned with a threshold, and then increase the threshold if the unit wins, or decrease it otherwise (Rumelhart & Zipser, 1985).

The conscience strategy realizes a similar idea by reducing the winning rate of the frequent winners (Desieno, 1988). The frequent winner receives a bad conscience by adding a penalty

term to its distance from the input signal. This leads to an entropy maximization, that is, each unit wins at an approximately equal probability. Thus, the probability of under-utilized neurons being selected as winners is increased.

The popular frequency sensitive competitive learning (FSCL) (Ahalt, Krishnamurty, Chen, & Melton, 1990) reduces the under-utilization problem by introducing a distortion measure that ensures all codewords in the codebook to be updated with a similar probability. The codebooks obtained by the FSCL algorithm have sufficient entropy so that Huffman coding of the VQ indices would not provide significant additional compression. In the FSCL, each prototype incorporates a count of the number of times it has been the winner, $u_j, j = 1, \ldots, K$. The distance measure is modified to give prototypes with a lower count value a chance to win the competition. The only difference with the VQ algorithm is that the winning neuron is found by Ahalt et al. (1990)

$$\mathbf{c}_w(t) = \arg_{\mathbf{c}_j} \min_{j=1,\ldots,K} \left\{ u_j(t-1) \left\| \mathbf{x}_t - \mathbf{c}_j(t-1) \right\| \right\} \tag{22}$$

$u_w(t) = u_w(t-1) + 1$, $u_i(t) = u_i(t-1)$ for $i \neq w$, where w is the index of the winning neuron and $u_i(0) = 0, i = 1, \ldots, K$. In (22), $u_j \left\| \mathbf{x}_t - \mathbf{c}_j \right\|$ can be generalized as $F(u_j) \left\| \mathbf{x}_t - \mathbf{c}_j \right\|$. When selecting the fairness function as $F\left(u_j\right) = u_j^{\beta_0 e^{-t/T_0}}$, β_0 and T_0 being constants, the FSCL emphasizes the winning uniformity of codewords initially and gradually turns into competitive learning as training proceeds to minimize the MSE function.

In the multiplicatively biased competitive learning (MBCL) model (Choy & Siu, 1998a), the competition among the neurons is biased by a multiplicative term. The MBCL avoids neuron under-utilization with probability one, as time goes to infinity. The FSCL (Ahalt et al., 1990; Krishnamurthy, Ahalt, Melton, & Chen, 1990) is a member of the MBCL family. In the MBCL, only one weight vector is updated per step. The fuzzy FSCL (FFSCL) (Chung & Lee, 1994) combines the frequency sensitivity with fuzzy competitive learning. Since both the FSCL and the FFSCL use a non-Euclidean distance to determine the winner, the problem of shared clusters may occur: a number of prototypes move into the same cluster as learning proceeds.

11.2. Rival-penalized competitive learning

The problem of shared clusters is considered in the rival-penalized competitive learning (RPCL) algorithm (Xu, Krzyzak, & Oja, 1993). The RPCL adds a new mechanism to the FSCL by creating a rival penalizing force. For each input, the winning unit is modified to adapt to the input, the second-place winner called the rival is also updated by a smaller learning rate along the opposite direction, and all the other prototypes remain unchanged

$$\mathbf{c}_i(t+1) = \begin{cases} \mathbf{c}_i(t) + \eta_w \left(\mathbf{x}_t - \mathbf{c}_i(t)\right), & i = w \\ \mathbf{c}_i(t) - \eta_r \left(\mathbf{x}_t - \mathbf{c}_i(t)\right), & i = r \\ \mathbf{c}_i(t), & \text{otherwise} \end{cases} \tag{23}$$

where w and r are the indices of winning and rival prototypes, which are decided by (22), and η_w and η_r are their respective learning rates, $\eta_w(t) \gg \eta_r$.

This actually pushes the rival away from the sample pattern so as to prevent it from interfering the competition. The RPCL automatically allocates an appropriate number of prototypes for an input data set, and all the extra candidate prototypes will finally be pushed to infinity. It provides a better performance than the FSCL. The RPCL can be regarded as an unsupervised extension of the supervised LVQ2 (Kohonen, 1990). It simultaneously modifies the weight vectors of both the winner and its rival, when the winner is in a wrong class but the rival is in a correct class for an input vector (Xu et al., 1993). The lotto type competitive learning (LTCL) (Luk & Lien, 1998) can be treated as a generalization of the RPCL, where instead of just penalizing the nearest rival, all the losers are penalized equally. The generalized LTCL (Luk & Lien, 1999) modifies the LTCL by allowing more than one winner, which are divided into tiers, with each tier being rewarded differently.

The RPCL may, however, encounter the over-penalization or under-penalization problem (Zhang & Liu, 2002). The STepwise Automatic Rival-penalized (STAR) C-means (Cheung, 2003) is a generalization of the C-means based on the FSCL (Ahalt et al., 1990) and a Kullback–Leibler divergence based criterion. The STAR C-means has a mechanism similar to the RPCL, but penalizes the rivals in an implicit way, whereby avoiding the problem of the RPCL.

11.3. Soft competitive learning

The winner-take-most rule relaxes the WTA rule by allowing more than one neuron as winners to a certain degree. The is the soft competitive learning. Examples are the SCS (Yair et al., 1992), the SOM (Kohonen, 1989), the NG (Martinetz et al., 1993), the GNG (Fritzke, 1995a), maximum-entropy clustering (Rose et al., 1990), the GLVQ (Pal et al., 1993), the FCM (Bezdek, 1981), the fuzzy competitive learning (FCL) (Chung & Lee, 1994), and fuzzy clustering algorithms. The FCL algorithms (Chung & Lee, 1994) are a class of sequential algorithms obtained by fuzzifying competitive learning algorithms, such as the SCL and the FSCL. The enhanced sequential fuzzy clustering (ESFC) (Zheng & Billings, 1999) is a modification to the FCL to better overcome the under-utilization problem. The SOM (Kohonen, 1989) employs the winner-take-most strategy at the early stages and approaches a WTA method as time goes on. Due to the soft competitive strategy, these algorithms are less likely to be trapped at local minima and to generate dead units than hard competitive alternatives (Baraldi & Blonda, 1999).

The maximum-entropy clustering (Rose et al., 1990) circumvents the under-utilization problem and local minima in the error function by using soft competitive learning and deterministic annealing. The prototypes are updated by

$$\mathbf{c}_i(t+1) = \mathbf{c}_i(t) + \eta(t) \left[\frac{e^{-\beta \left\| \mathbf{x}_t - \mathbf{c}_i(t) \right\|^2}}{\sum_{j=1}^{K} e^{-\beta \left\| \mathbf{x}_t - \mathbf{c}_j(t) \right\|^2}} \right] \left(\mathbf{x}_t - \mathbf{c}_i(t)\right) \tag{24}$$

where η is the learning rate, $\frac{1}{\beta}$ anneals from a large number to zero, and the term within the bracket turns out to be the Boltzmann distribution. The SCS (Yair et al., 1992) employs a similar soft competitive strategy, but β is fixed as unity.

The winner-take-most criterion, however, detracts some prototypes from their corresponding clusters, and consequently becomes biased toward the global mean of the clusters, since all the prototypes are attracted to each input pattern (Liu, Glickman, & Zhang, 2000).

12. Robust clustering

Outliers in a data set affects the result of clustering. The influence of outliers can be eliminated by using the robust statistics approach (Huber, 1981). This idea has also been incorporated into many robust clustering methods (Bradley, Mangasarian, & Steet, 1996; Dave & Krishnapuram, 1997; Frigui & Krishnapuram, 1999; Hathaway & Bezdek, 2000; Kersten, 1999; Leski, 2003a). The C-median clustering (Bradley et al., 1996) is derived by solving a bilinear programming problem that utilizes the L_1-norm distance. The fuzzy C-median (Kersten, 1999) is a robust FCM method that uses the L_1-norm with the exemplar estimation based on the fuzzy median. Robust clustering algorithms can be derived by optimizing an objective function E_T, which comprises of the cost E for the conventional algorithms and a constraint term E_c for describing the noise.

12.1. Noise clustering

In the noise clustering approach (Dave, 1991), all outliers are collected into a separate, amorphous noise cluster, whose prototype has the same distance δ from all the data points, while all the other points are collected into K clusters. The threshold δ is relatively large compared to the distances of the good points to their respective cluster prototypes. If a noisy point is far away from all the K clusters, it is attracted to the noise cluster. In the noise clustering approach (Dave, 1991), the constraint term is given by

$$E_c = \sum_{i=1}^{N} \delta^2 \left(1 - \sum_{j=1}^{K} \mu_{ji}\right)^m. \tag{25}$$

Optimizing on E yields

$$\mu_{ji} = \frac{\left(\frac{1}{\|\mathbf{x}_i - \mathbf{c}_j\|^2}\right)^{\frac{1}{m-1}}}{\sum_{k=1}^{K}\left(\frac{1}{\|\mathbf{x}_i - \mathbf{c}_k\|^2}\right)^{\frac{1}{m-1}} + \left(\frac{1}{\delta^2}\right)^{\frac{1}{m-1}}}. \tag{26}$$

The second term in the denominator, due to outliers, lowers μ_{ji}. The formula for the prototypes is the same as that in the FCM. Thus, the noise clustering can be treated as a robustified FCM. When all the K clusters have a similar size, the noise clustering is very effective. However, a single threshold is too restrictive if the cluster size varies widely in the data set.

12.2. Possibilistic C-means

Unlike fuzzy clustering, the possibilistic C-means (PCM) (Krishnapuram & Keller, 1993) does not require the sum of the memberships of a data point across the clusters to be unity. The membership functions represent a possibility of belonging rather than a relative degree of membership between clusters. Thus, the derived degree of membership does not decrease as the number of clusters increases. Without this constraint, the modified objective function is decomposed into many individual objective functions, one for each cluster, which can be optimized separately.

The constraint term for the PCM is given by a sum associated with the fuzzy complements of all the K clusters

$$E_c = \sum_{j=1}^{K} \beta_j \sum_{i=1}^{N} \left(1 - \mu_{ji}\right)^m \tag{27}$$

where β_j are suitable positive numbers. The individual objective functions are given as

$$E_T^j = \sum_{i=1}^{N} \mu_{ji}^m \left\|\mathbf{x}_i - \mathbf{c}_j\right\|^2 + \beta_j \sum_{i=1}^{N} \left(1 - \mu_{ji}\right)^m, \quad j = 1, \ldots, K. \tag{28}$$

Optimizing (28) with respect to μ_{ji} yields the solution

$$\mu_{ji} = \frac{1}{1 + \left(\frac{\|\mathbf{x}_i - \mathbf{c}_j\|^2}{\beta_j}\right)^{\frac{1}{m-1}}}. \tag{29}$$

For outliers, μ_{ji} is small. Some heuristics for selecting β_j are given in Krishnapuram and Keller (1993).

Given a number of clusters K, the FCM will arbitrarily split or merge real clusters in the data set to produce exactly the specified number of clusters, while the PCM can find those natural clusters in the data set. When K is smaller than the number of actual clusters, only K good clusters are found, and the other data points are treated as outliers. When K is larger than the number of actual clusters, all the actual clusters can be found and some clusters will coincide. In the noise clustering, there is only one noise cluster, while in the PCM there are K noise clusters. The PCM behaves as a collection of K independent noise clustering algorithms, each searching a single cluster. The performance of the PCM, however, relies heavily on initialization of cluster prototypes and estimation of β_j, and the PCM tends to converge to coincidental clusters (Dave & Krishnapuram, 1997).

12.3. Other robust clustering problems

A family of robust clustering algorithms have been obtained by treating outliers as the fuzzy complement (Yang & Wang, 2004). Assuming that a noise cluster exists outside each data cluster, the fuzzy complement of μ_{ji} can be viewed as the membership of $\mathbf{x}_i$ in the noise cluster with a distance β_j. Based on this idea, many different implementations of the probabilistic approach can be proposed (Dave & Krishnapuram, 1997; Yang & Wang, 2004), and a general form of E_c is obtained as a generalization of that for the PCM (Yang & Wang, 2004). The alternating cluster estimation method (Runkler & Bezdek, 1999) is a simple extension of the general method (Dave & Krishnapuram, 1997; Yang & Wang, 2004). The fuzzy robust C-spherical shells algorithm (Yang & Wang, 2004) searches the clusters that belongs to the spherical shells by combining the concept of the fuzzy complement and the fuzzy C-spherical shells algorithm (Krishnapuram, Nasraoui, & Frigui, 1992). The hard robust clustering algorithm (Yang & Wang, 2004) is an extension of the GLVQ-F algorithm (Karayiannis et al., 1996). All these robust algorithms are highly dependent on the initial values and adjustment of β_j.

The robust competitive agglomeration (RCA) algorithm (Frigui & Krishnapuram, 1999) combines the advantages of both the hierarchical and partitional clustering techniques. The objective function also contains a constraint term. An optimum number of clusters is determined via a process of competitive agglomeration, while the knowledge of the global shape of the clusters is incorporated via the use of prototypes. Robust statistics like the M-estimator (Huber, 1981) is incorporated to combat the outliers. Overlapping clusters are handled by using fuzzy memberships.

Clustering of a vectorial data set with missing entries belongs to robust clustering. In Hathaway and Bezdek (2001), four strategies, namely the whole data, partial distance, optimal completion and nearest prototype strategies, are discussed for implementing the FCM for incomplete data. The introduction of the concept of noise clustering into relational clustering techniques leads to their robust versions (Dave & Sen, 2002). A review of robust clustering methods is given in Dave and Krishnapuram (1997).

13. Clustering using non-Euclidean distance measures

Due to the Euclidean distance measure, conventional clustering methods favor hyperspherically shaped clusters of equal size, but have the undesirable property of splitting big and elongated clusters (Duda & Hart, 1973). The Mahalanobis distance can be used to look for hyperellipsoid shaped clusters. However, the C-means algorithm using the Mahalanobis distance tends to produce unusually large or unusually small clusters (Mao & Jain, 1996). The hyperellipsoidal clustering (HEC) network (Mao & Jain, 1996) integrates PCA and clustering into one network, and can adaptively estimate the hyperellipsoidal shape of each cluster. The HEC implements clustering using a regularized Mahalanobis distance that is a linear combination of the Mahalanobis and Euclidean distances. The regularized distance achieves a trade-off between the hyperspherical and hyperellipsoidal cluster shapes to prevent the HEC network from producing unusually large or unusually small clusters. The Mahalanobis distance is used in the Gustafson–Kessel algorithm (Gustafson & Kessel, 1979) and the AFC (Anderson et al., 1982). The symmetry based C-means (Su & Chou, 2001) employs the C-means as a coarse search for the K cluster centroid and an ensuing fine-tuning procedure based on the

point-symmetry distance as the dissimilarity measure. The method can effectively find clusters with symmetric shapes, such as the human face.

A number of algorithms for detecting circles and hyperspherical shells have been proposed as extensions of the C-means and FCM algorithms. These include the fuzzy C-shells (Dave, 1990), fuzzy C-ring (Man & Gath, 1994), hard C-spherical shells (Krishnapuram et al., 1992), unsupervised C-spherical shells (Krishnapuram et al., 1992), fuzzy C-spherical shells (Krishnapuram et al., 1992), and possibilistic C-spherical shells (Krishnapuram & Keller, 1993) algorithms. All these algorithms are based on iterative optimization of objective functions similar to that for the FCM, but defines the distance from a prototype $\vec{\lambda}_i = (\mathbf{c}_i, r_i)$ to the point $\mathbf{x}_j$ as

$$d_{j,i}^2 = d^2\left(\mathbf{x}_j, \vec{\lambda}_i\right) = \left(\left\|\mathbf{x}_j - \mathbf{c}_i\right\| - r_i\right)^2 \tag{30}$$

where $\mathbf{c}_i$ and r_i are the center and radius of the hypersphere, respectively. The optimal number of substructures in the data set can be effectively estimated by using some validity criteria such as spherical shell thickness (Krishnapuram et al., 1992), fuzzy hypervolume and fuzzy density (Gath & Geva, 1989; Man & Gath, 1994).

By using different distance measures, many clustering algorithms can be derived for detecting clusters of various shapes such as lines and planes (Bezdek, 1981; Dave & Krishnapuram, 1997; Frigui & Krishnapuram, 1999; Kaymak & Setnes, 2002; Zhang & Liu, 2002), circles and spherical shells (Krishnapuram et al., 1992; Pal & Chakraborty, 2000; Zhang & Liu, 2002), ellipses (Frigui & Krishnapuram, 1999; Gath & Hoory, 1995), curves, curved surfaces, ellipsoids (Bezdek, 1981; Frigui & Krishnapuram, 1999; Gath & Geva, 1989; Kaymak & Setnes, 2002; Mao & Jain, 1996), rectangles, rectangular shells and polygons (Hoeppner, 1997). Relational data can be clustered by using the non-Euclidean relational FCM (NERFCM) (Hathaway & Bezdek, 1994, 2000). Fuzzy clustering for relational data is reviewed in Dave and Sen (2002).

14. Hierarchical clustering

Existing clustering algorithms are broadly classified into partitional, hierarchical, and density based clustering. Clustering methods discussed thus far belong to partitional clustering.

14.1. Partitional, hierarchical, and density based clustering

Partitional clustering can be either hard or fuzzy one. Fuzzy clustering can deal with overlapping cluster boundaries. Partitional clustering is dynamic, where points can move from one cluster to another. Knowledge of the shape or size of the clusters can be incorporated by using appropriate prototypes and distance measures. Partitional clustering is susceptible to local minima of its objective function, and the number of clusters K is usually required to be prespecified. Also, it is sensitive to noise and outliers. Partitional clustering has a typical complexity of $O(N)$.

Hierarchical clustering consists of a sequence of partitions in a hierarchical structure, which can be represented as a clustering tree called *dendrogram*. Hierarchical clustering takes the form of either agglomerative or divisive technique. New clusters are formed by reallocating the membership degree of one point at a time, based on a certain measure of similarity or distance. Agglomerative clustering is suitable for data with dendritic substructure. Outliers can be easily identified in hierarchical clustering, since they merge with other points less often due to their larger distances from the other points and the number of outliers is typically much less than that in a cluster. The number of clusters K need not be specified, and the local minimum problem arising from initialization does not occur. However, prior knowledge of the shape or size of the clusters cannot be incorporated, and overlapping clusters cannot be separated. Moreover, hierarchical clustering is static, and points committed to a given cluster cannot move to a different cluster. Hierarchical clustering has a typical complexity of $O\left(N^2\right)$, making it impractical for larger data set. Divisive clustering reverses the procedure, but is computationally more expensive (Xu & Wunsch II, 2005).

Density based clustering groups objects of a data set into clusters based on density conditions. Clusters are dense regions of objects in the data space and are separated by regions of low density. The method is robust against outliers since an outlier affects clustering only in the neighborhood of this data point. It can handle outliers and discover clusters of arbitrary shape. Density based clustering has a complexity of the same order as hierarchical clustering. The DBSCAN (Ester, Kriegel, Sander, & Xu, 1996) is a widely known density based clustering algorithm. In the DBSCAN, a region is defined as the set of points that lie in the ϵ-neighborhood of some point p. Cluster label propagation from p to the other points in a region $\mathcal{R}$ happens if $|\mathcal{R}|$, the cardinality of $\mathcal{R}$, exceeds a given threshold for the minimal number of points.

14.2. Distance measures and cluster representations

The inter-cluster distance is usually characterized by the single-linkage or the complete-linkage technique. The single-linkage technique calculates the inter-cluster distance using the two closest data points in different clusters. The method is more suitable for finding well-separated stringy clusters. In contrast, the complete-linkage technique defines the inter-cluster distance as the farthest distance between any two data points in different clusters. Other more complicated methods are group-average-linkage, median-linkage, and centroid-linkage techniques.

A cluster is conventionally represented by its centroid or prototype. This is desirable only for spherically shaped clusters, but causes cluster splitting for a large or arbitrarily shaped cluster, since the centroids of its subclusters can be far apart. At the other extreme, if all data points in a cluster are used as its representatives, the clustering algorithm is extremely sensitive to noise and outliers. This all-points representation can cluster arbitrary shapes. The scatter-points representation (Guha, Rastogi, & Shim, 2001), as a trade-off between the two extremes, represents each cluster by a fixed number of points that are generated by selecting well-scattered points from the cluster and then shrinking them toward the center of the cluster by a specified fraction. This reduces the adverse effects of the outliers since the outliers are typically farther away from the mean and are thus shifted by a larger distance due to shrinking. The scatter-points representation achieves robustness to outliers, and identifies clusters that have non-spherical shape and wide variations in size.

14.3. Agglomerative clustering

Agglomerative clustering starts from N clusters, each containing one data point. A series of nested merging is performed until all the data points are grouped into one cluster. The algorithm processes a set of N^2 numerical relationships between the N data points, and agglomerates according to their similarity or distance. Agglomerative clustering is based on a local connectivity criterion. The run time is $O\left(N^2\right)$. Dendrogram is used to illustrate the clusters produced by agglomerative clustering. Agglomerative clustering can be based on the centroid (Zhang, Ramakrishnan, & Livny, 1996), all-points (Zahn, 1971), or scatter-points (Guha et al., 2001) representation. For large data sets, storage or multiple input/output scans of the data points is a bottleneck for the existing clustering algorithms. Some strategies can be applied to combat this problem (Guha et al., 2001; Vesanto & Alhoniemi, 2000; Wang & Rau, 2001; Zhang et al., 1996).

The conventional minimum spanning tree (MST) algorithm (Zahn, 1971) is a graph-theoretical technique (Swamy & Thulasiraman, 1981; Thulasiraman & Swamy, 1992). It uses the all-points representation. The method first finds an MST for the input data. Then, by removing the longest $K-1$ edges, K clusters are obtained. The MST algorithm is good at clustering arbitrary shapes. The method, however, is very sensitive to the outliers, and it may merge two clusters due to a chain of outliers between them. The BIRCH method (Zhang et al., 1996) first performs an incremental and approximate preclustering phase in which dense regions of points are represented by compact summaries, and a centroid based hierarchical algorithm is then used to cluster the set of summaries. The outliers are eliminated from the summaries via the identification of the sparsely distributed data points in the feature space. The BIRCH needs only a little more than one scan of the data. However, the method fails to identify clusters with non-spherical shapes or a wide variation in size by splitting larger clusters and merging smaller clusters. The CURE method (Guha et al., 2001) is a robust clustering algorithm based on the scatter-points representation. To handle large databases, the CURE employs a combination of random sampling and partitioning. The complexity of the CURE is not worse than that of centroid based hierarchical algorithms. The CURE provides a better performance with less execution time compared to the BIRCH (Guha et al., 2001). It can discover clusters with interesting shapes and is less sensitive to the outliers than the MST. The CHAMELEON (Karypis, Han, & Kumar, 1999) first creates a graph, where each node represents a pattern and all the nodes are connected according to the k-NN paradigm. The graph is recursively partitioned into many small unconnected subgraphs, each partitioning yielding two subgraphs of roughly equal size. Agglomerative clustering is applied to the subclusters. Two subclusters are merged only when the interconnectivity as well as the closeness of the individual clusters is very similar. The CHAMELEON automatically adapts to the characteristics of the clusters being merged. The method is more effective than the CURE in discovering clusters of arbitrary shapes and varying densities (Karypis et al., 1999).

14.4. Hybridization of hierarchical and partitional clusterings

The advantages of both the hierarchical and the partitional clustering have been incorporated into many methods (Frigui & Krishnapuram, 1999; Geva, 1999; Su & Liu, 2005; Vesanto & Alhoniemi, 2000; Wang & Rau, 2001). The VQ-clustering and VQ-agglomeration methods (Wang & Rau, 2001) involve a VQ process followed, respectively, by clustering and agglomerative clustering that treat the codewords as initial prototypes. Each codeword is associated with a gravisphere that has a well defined attraction radius. The agglomeration algorithm requires that each codeword be moved directly to the centroid of its neighboring codewords. A similar two-stage clustering procedure that uses the SOM for VQ and an agglomerative clustering or the C-means for further clustering is given in Vesanto and Alhoniemi (2000). The performance results of these two-stage methods are comparable to those of direct methods, with a significantly reduced execution time (Vesanto & Alhoniemi, 2000; Wang & Rau, 2001). A two-stage procedure given in Su and Liu (2005) can cluster data with arbitrary shapes, where an ART-like algorithm partitions data into a set of small multi-dimensional hyperellipsoids and an agglomerative algorithm sequentially merges those hyperellipsoids. Dendrograms and the so-called tables of relative frequency counts are then used to pick some trustable clustering results from a lot of different clustering results. In the hierarchical unsupervised fuzzy clustering (HUFC) (Geva, 1999), PCA is applied to each cluster for optimal feature extraction. This method is effective for data sets with a wide dynamic variation in both the covariance matrix and the number of members in each class. The robust competitive agglomeration (RCA) (Frigui & Krishnapuram, 1999) finds the optimum number of clusters by competitive agglomeration, and achieves noise immunity by integrating robust statistics.

15. Constructive clustering techniques

Conventional partitional clustering algorithms assume a network with a fixed number of clusters (nodes) K. However, selecting the appropriate value of K is a difficult task without a prior knowledge of the input data. Constructive clustering can solve this difficulty.

A simple strategy for determining K is to perform clustering for a range of K, and select the value of K that minimizes a cluster validity measure. This procedure is computationally intensive when the actual number of clusters is large. Examples of such strategy are the scatter based FSCL clustering (Sohn & Ansari, 1998) and a method using the distortion errors plus a codebook complexity term as the cost function (Buhmann & Kuhnel, 1993). The ISODATA (Ball & Hall, 1967) can be treated as a variant of the incremental C-means (MacQueen, 1967) by incorporating some heuristics for merging and splitting clusters, and for handling outliers; thus, it realizes a variable number of clusters K.

Self-creating mechanism in the competitive learning process can adaptively determine the natural number of clusters. The self-creating and organizing neural network (SCONN) (Choi & Park, 1994) employs adaptively modified node thresholds to control its self-growth. For a new input, the winning node is updated if it is active; otherwise a new node is created from the winning node. Activation levels of all the nodes decrease with time, so that the weight vectors are distributed at the final stage according to the input distribution. Nonuniform VQ is realized by decreasing the activation levels of the active nodes and increasing those of the other nodes to estimate the asymptotic point density automatically. The SCONN avoids the under-utilization problem, and has VQ accuracy and speed advantage over the SOM and the batch C-means (Linde et al., 1980).

The growing cell structures (GCS) network (Fritzke, 1994a) can be viewed as a modification of the SOM by integrating node recruiting/pruning functions. It assigns each nodes with a local accumulated statistical variable called *signal counter* u_i. For each new pattern, only the winning node increases its signal counter u_w by 1, and then all the signal counters u_i decay with a forgetting factor. After a fixed number of iterations, a new node is inserted between the node with the largest signal counter and its farthest neighbor. The algorithm occasionally prunes a node with its signal counter below a specified threshold during a complete epoch. The growing grid network (Fritzke, 1995b) is strongly related to the GCS. As opposed to the GCS, the growing grid has a strictly rectangular topology. By inserting complete rows or columns of units, the grid may adapt its height/width ratio to the given pattern distribution. The branching competitive learning (BCL) network (Xiong, Swamy, Ahmad, & King, 2004) adopts the same technique for recruiting and pruning nodes as the GCS except that a new geometrical criterion is applied to the winning node before updating its signal counter u_w.

The GNG model (Fritzke, 1995a, 1997a) is based on the GCS (Fritzke, 1994a) and the NG (Martinetz et al., 1993). The GNG is capable of generating and removing neurons and lateral connections dynamically. Lateral connections are generated by the competitive Hebbian learning rule. The GNG achieves robustness against noise and performs perfect topology-preserving mapping. The GNG with utility criterion (GNG-U) (Fritzke, 1997a) integrates an on-line criterion to identify and delete useless neurons, and can thus track nonstationary data input. A similar on-line clustering method is given in Furao and Hasegawa (2005). The dynamic cell structures (DCS) model (Bruske & Sommer, 1995) uses a modified Kohonen learning rule to adjust the prototypes and the competitive Hebbian rule so as to establish a dynamic lateral connection structure. Applying the DCS to the GCS yields the DCS-GCS algorithm, which has a behavior similar to that of the GNG. The life-long learning cell

structures (LLCS) algorithm (Hamker, 2001) is an on-line clustering and topology representation method. It employs a strategy similar to that of the ART, and incorporates similarity based unit pruning and aging based edge pruning procedures.

The self-splitting competitive learning (SSCL) (Zhang & Liu, 2002) can find the natural number of clusters based on the one-prototype-take-one-cluster (OPTOC) paradigm and a validity measure for self-splitting. The OPTOC enables each prototype to situate at the centroid of one natural cluster when the number of clusters is greater than that of the prototypes. The SSCL starts with a single prototype and splits adaptively until all the clusters are found. During the learning process, one prototype is chosen to split into two prototypes according to the validity measure, until the SSCL achieves an appropriate number of clusters.

16. Miscellaneous clustering methods

There are also numerous density based and graph theory based clustering algorithms. Here, we mention some algorithms associated with competitive learning and neural networks. The LBG has been implemented by storing the data points via a *k-d* tree, achieving typically an order of magnitude faster than the LBG (Kanungo et al., 2002). The expectation–maximization (EM) clustering (Bradley, Fayyad, & Reina, 1998) represents each cluster using a probability distribution, typically a Gaussian distribution. Each cluster is represented by a mean and a $J_1 \times J_1$ covariance matrix, where J_1 is the dimension of an input vector. Each pattern belongs to all the clusters with the probabilities of membership determined by the distributions of the corresponding clusters. Thus, the EM clustering can be treated as a fuzzy clustering technique. The EM technique is derived by maximizing the log likelihood of the probability density function of the mixture model. The *C*-means is equivalent to the classification EM (CEM) algorithm corresponding to the uniform spherical Gaussian model (Celeux & Govaert, 1992; Xu & Wunsch II, 2005). Kernel based clustering first nonlinearly maps the patterns into an arbitrarily high-dimensional feature space, and clustering is then performed in the feature space. Some examples are the kernel *C*-means (Scholkopf, Smola, & Muller, 1998), kernel subtractive clustering (Kim et al., 2005), variants of kernel *C*-means based on the SOM and the ART (Corchado & Fyfe, 2000), a kernel based algorithm that minimizes the trace of the within-class scatter matrix (Girolami, 2002), and support vector clustering (SVC) (Ben-Hur, Horn, Siegelmann, & Vapnik, 2001; Camastra & Verri, 2005; Chiang & Hao, 2003). The SVC can effectively deal with the outliers.

17. Cluster validity

An optimal number of clusters or a good clustering algorithm is only in the sense of a certain cluster validity criterion. Many cluster validity measures are defined for this purpose.

17.1. Measures based on maximal compactness and maximal separation of clusters

A good clustering algorithm should generate clusters with small intracluster deviations and large inter-cluster separations. Cluster compactness and cluster separation are two measures for the performance of clustering. A popular cluster validity measure is defined as (Davies & Bouldin, 1979; Du & Swamy, 2006)

$$E_{\mathrm{WBR}} = \frac{1}{K}\sum_{k=1}^{K}\max_{l\neq k}\left\{\frac{d_{\mathrm{WCS}}(\mathbf{c}_k)+d_{\mathrm{WCS}}(\mathbf{c}_l)}{d_{\mathrm{BCS}}(\mathbf{c}_k,\mathbf{c}_l)}\right\} \tag{31}$$

where the within-cluster scatter for cluster k, denoted $d_{\mathrm{WCS}}(\mathbf{c}_k)$, and the between-cluster separation for clusters k and l, denoted $d_{\mathrm{BCS}}(\mathbf{c}_k,\mathbf{c}_l)$, are calculated by

$$d_{\mathrm{WCS}}(\mathbf{c}_k) = \frac{\sum_i \|\mathbf{x}_i-\mathbf{c}_k\|}{N_k}, \quad d_{\mathrm{BCS}}(\mathbf{c}_k,\mathbf{c}_l) = \|\mathbf{c}_k-\mathbf{c}_l\| \tag{32}$$

N_k being the number of data points in cluster k. The best clustering minimizes E_{WBR}. This index indicates good clustering results for spherical clusters (Vesanto & Alhoniemi, 2000). Alternative criteria for the cluster compactness, cluster separation, and overall cluster quality measures are given in He et al. (2004). In Xie and Beni (1991), the ratio of compactness and separation is used as a cluster validity criterion for fuzzy clustering. Entropy cluster validity measures based on class conformity are given in Boley (1998); He et al. (2004). Some cluster validity measures are described and compared in Bezdek and Pal (1998).

17.2. Measures based on minimal hypervolume and maximal density of clusters

A good partitioning of the data usually leads to a small total hypervolume and a large average density of the clusters. Cluster validity measures can be thus selected as the hypervolume and average density of the clusters. The fuzzy hypervolume criterion (Gath & Geva, 1989; Krishnapuram et al., 1992) is defined as the sum of the volumes of all the clusters, V_i, and $V_i = [\det(\mathbf{F}_i)]^{\frac{1}{2}}$, where $\mathbf{F}_i$, the fuzzy covariance matrix of the ith cluster, is defined by Gustafson and Kessel (1979)

$$\mathbf{F}_i = \frac{1}{\sum_{j=1}^{N}\mu_{ij}^m}\sum_{j=1}^{N}\mu_{ij}^m(\mathbf{x}_j-\mathbf{c}_i)(\mathbf{x}_j-\mathbf{c}_i)^{\mathrm{T}}. \tag{33}$$

The average fuzzy density criterion (Gath & Geva, 1989) is defined as the average of the fuzzy density in each cluster, $\frac{S_i}{V_i}$, where S_i sums the membership degrees of only those members within a hyperellipsoid defined by $\mathbf{F}_i$. The fuzzy hypervolume criterion typically has a clear extremum; the average fuzzy density criterion is not desirable when there is a substantial cluster overlapping and a large variation in the compactness of the clusters (Gath & Geva, 1989). A partitioning that results in both dense and loose clusters may lead to a large average fuzzy density.

For shell clustering, the hypervolume and average density measures are still applicable. However, the distance vector between a pattern and a prototype needs to be redefined. In the case of spherical shell clustering, the displacement or distance vector between a pattern $\mathbf{x}_j$ and a prototype $\vec{\lambda}_i = (\mathbf{c}_i, r_i)$ is defined by

$$\mathbf{d}_{ji} = (\mathbf{x}_j-\mathbf{c}_i) - r_i\frac{\mathbf{x}_j-\mathbf{c}_i}{\|\mathbf{x}_j-\mathbf{c}_i\|}. \tag{34}$$

The fuzzy hypervolume and average fuzzy density measures for spherical shell clustering are obtained by replacing the distance vector $(\mathbf{x}_j-\mathbf{c}_i)$ in (33) by $\mathbf{d}_{ji}$. For shell clustering, the shell thickness measure can be used to describe the compactness of a shell. In the case of fuzzy spherical shell clustering, the fuzzy shell thickness of a cluster is defined in Krishnapuram et al. (1992). The average shell thickness of all clusters can be used as a cluster validity measure for shell clustering.

18. Computer simulations

In this section, we give two examples to illustrate the application of clustering algorithms.

18.1. An artificial example

Given a data set of 1000 random data points in the two-dimensional space: In each of the two half rings there are 500 uniformly random data points. We use the SOM to realize VQ

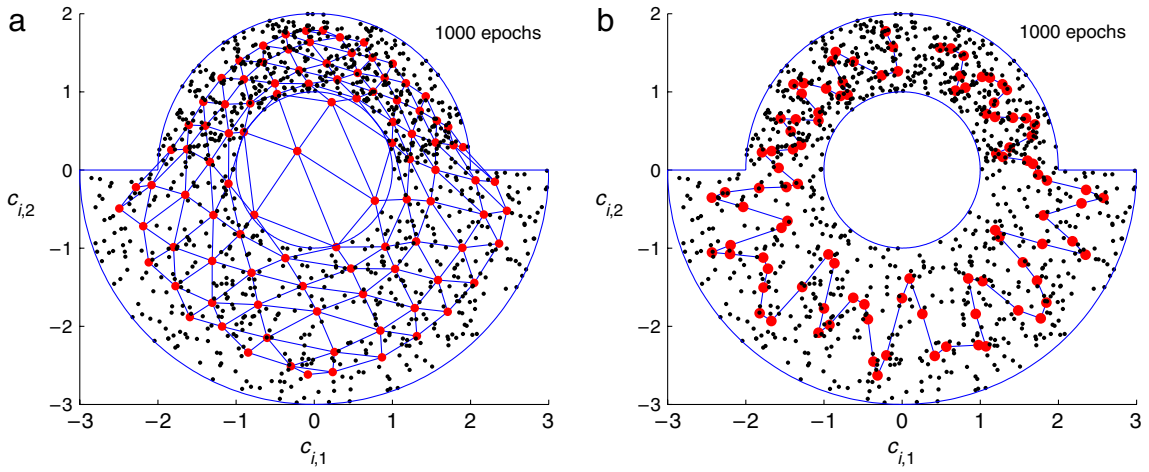

Fig. 3. Random data points in the two-dimensional space. In each of the two quarters, there are 1000 uniformly random points. (a) The out cells are arranged in a 10 × 10 grid. (b) The output cells are arranged in a one-dimensional grid of 100 cells.

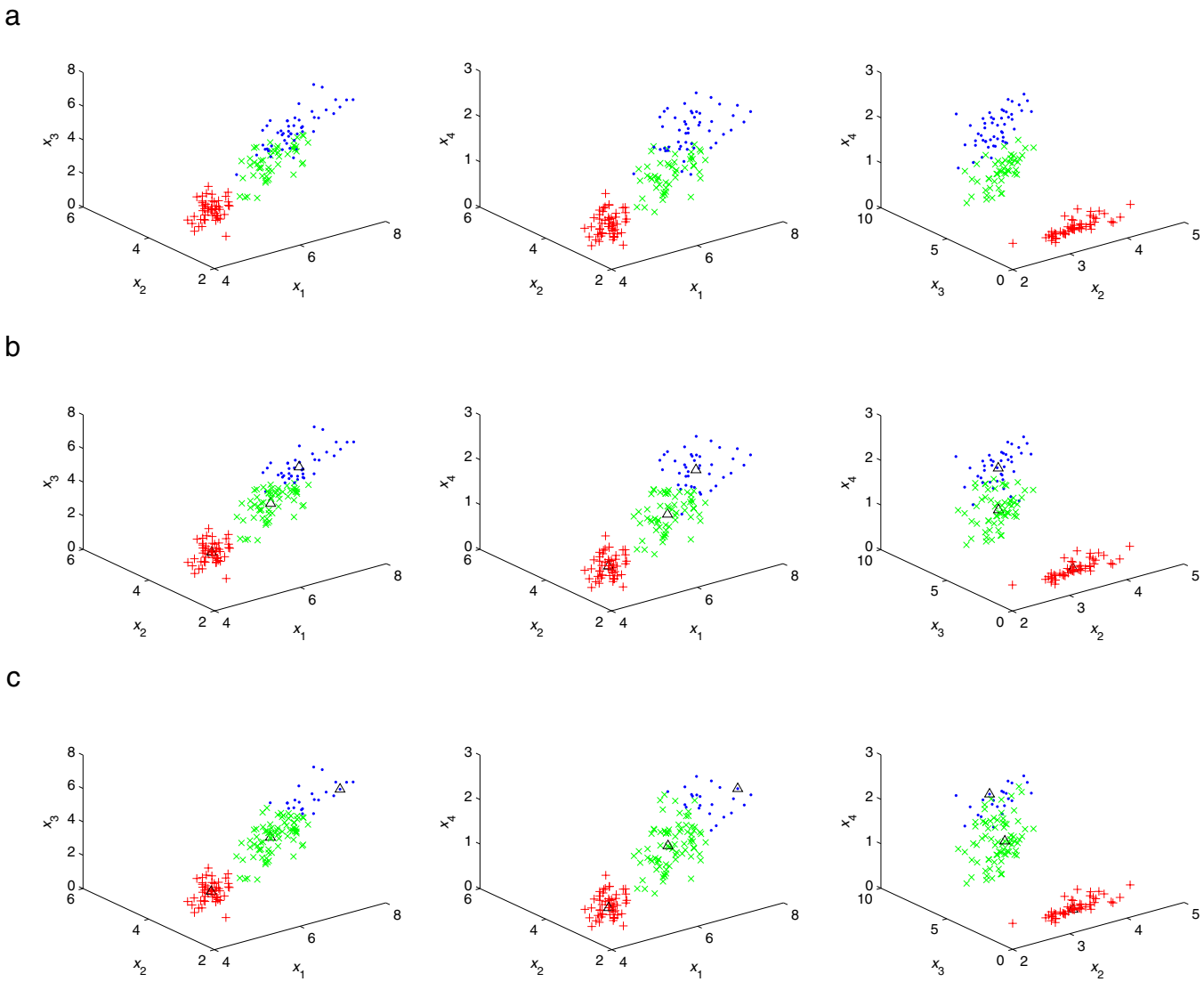

Fig. 4. The iris classification: (a) The iris data set and the class information. (b) The clustering result by the FCM. (c) The clustering result by the subtractive clustering.

and topology-preserving by producing a grid of cells. Simulation is based on the Matlab Neural Network Toolbox. In the first group of simulations, the output cells are arranged in a 10 × 10 grid, and the hexagonal neighborhood topology is used. The training result for 1000 epochs is shown in Fig. 3a. When the 100 output cells are arranged in one dimension, the training result for 1000 epochs is shown in Fig. 3b. Given a test point, the trained network can always find the prototype based on the nearest-neighbor paradigm.

18.2. Iris classification

In the iris data set, 150 patterns are classified into 3 classes. Each pattern has four numeric attributes, denoted by x_i, $i = 1, 2, 3, 4$, and each class has 50 patterns. The Iris data set and the corresponding classification are shown in Fig. 4a. We now use the FCM and subtractive clustering methods to cluster the data set. For the FCM, we select the number of clusters $K = 3$; for subtractive clustering, we select the radius for all the clusters as 0.8, and this also leads to $K = 3$. The clustering results for the FCM and the subtractive clustering are, respectively, shown in Fig. 4b, c. A comparison of Fig. 4b, c with Fig. 4a reveals that there are 14 classification errors for the FCM and 24 classification errors for the subtractive clustering. Thus the classification accuracy is 90.67% for the FCM, and 84.00% for the subtractive clustering.

19. Summary

Clustering is one of the most important data analysis methods. In this paper, we provide a state-of-the-art survey and introduction to neural network based clustering. Various aspects of clustering are addressed. Two examples are given to illustrate the application of clustering. Interested readers are referred to Jain, Murty, and Flynn (1999), Xu and Wunsch II (2005) and Du and Swamy (2006) for more information on clustering and their applications. Other topics such as global search based clustering are also reviewed in Xu and Wunsch II (2005).

Clustering has become an important tool for data mining, also known as knowledge discovery in databases (KDD) (Jain et al., 1999), which emerges as a rapidly growing area. The wealth of information in huge databases or the world wide web (WWW) has aroused tremendous interest in the area of data mining. Data mining is to automatically search large stores of data for consistent patterns and/or relationships between variables so as to predict future behavior. The process of data mining consists of three phases, namely, data preprocessing and exploration, model selection and validation, as well as final deployment. Clustering, neuro-fuzzy systems and rough sets, and evolution based global optimization methods are usually used for data mining (Du & Swamy, 2006). Neuro-fuzzy systems and rough sets are ideal tools for knowledge representation. Data mining needs first to discover the structural features in a database, and exploratory techniques through self-organization such as clustering are particularly promising. Some of the data mining approaches that use clustering are database segmentation, predictive modeling, and visualization of large databases (Jain et al., 1999). Structured databases have well defined features and data mining can easily succeed with good results. Web mining is more difficult since the WWW is a less structured database (Etzioni, 1996). The topology-preserving property for the SOM makes it particularly suitable for web information processing.

References

Ahalt, S. C., Krishnamurty, A. K., Chen, P., & Melton, D. E. (1990). Competitive learning algorithms for vector quantization. *Neural Networks*, *3*(3), 277–290.

Al-Harbi, S. H., & Rayward-Smith, V. J. (2006). Adapting k-means for supervised clustering. *Applied Intelligence*, *24*, 219–226.

Anderson, I.A., Bezdek, J.C., & Dave, R. (1982). Polygonal shape description of plane boundaries. In: Troncale L (ed). *Systems science and science*, 1, SGSR, Louisville, KY (pp. 295–301).

Andrew, L. (1996). Implementing the robustness of winner-take-all cellular neural network. *IEEE Transactions on Circuits and Systems–II*, *43*(4), 329–334.

Angelov, P. P., & Filev, D. P. (2004). An approach to online identification of Takagi-Sugeno fuzzy models. *IEEE Transactions on Systems, Man and Cybernetics–B*, *34*(1), 484–498.

Ball, G. H., & Hall, D. J. (1967). A clustering technique for summarizing multivariate data. *Behavioural Science*, *12*, 153–155.

Bandyopadhyay, S., Maulik, U., & Pakhira, M. K. (2001). Clustering using simulated annealing with probabilistic redistribution. *International Journal of Pattern Recognition and Artifical Intelligence*, *15*(2), 269–285.

Bandyopadhyay, S., & Maulik, U. (2002). An evolutionary technique based on k-means algorithm for optimal clustering. *Information Sciences*, *146*, 221–237.

Baraldi, A., & Blonda, P. (1999). A survey of fuzzy clustering algorithms for pattern recognition–Part II. *IEEE Transactions on Systems Man and Cybernetics–B*, *29*(6), 786–801.

Baraldi, A., & Alpaydin, E. (2002). Constructive feedforward ART clustering networks–Part I; Part II. *IEEE Transactions on Neural Networks*, *13*(3), 645–677.

Baraldi, A., & Parmiggiani, F. (1997). Novel neural network model combining radial basis function, competitive Hebbian learning rule, and fuzzy simplified adaptive resonance theory. In *Proc SPIE*, vol. 3165, (pp. 98–112).

Ben-Hur, A., Horn, D., Siegelmann, H., & Vapnik, V. (2001). Support vector clustering. *Journal of Machine Learning Research*, *2*, 125–137.

Beni, G., & Liu, X. (1994). A least biased fuzzy clustering method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *16*(9), 954–960.

Bezdek, J. (1974). Cluster validity with fuzzy sets. *Journla of Cybernetics*, *3*(3), 58–71.

Bezdek, J. (1981). *Pattern recognition with fuzzy objective function algorithms*. New York: Plenum Press.

Bezdek, J. C., & Pal, N. R. (1995). Two soft relatives of learning vector quantization. *Neural Networks*, *8*(5), 729–743.

Bezdek, J. C., & Pal, N. R. (1998). Some new indexes or cluster validity. *IEEE Transactions on Systems Man and Cybernetics*, *28*(3), 301–303.

Bezdek, J.C., Tsao, E.C., & Pal, N.R. (1992). Fuzzy Kohonen clustering networks. In *Proc 1st IEEE int conf fuzzy syst*, (pp. 1035–1043).

Boley, D. (1998). Principal direction divisive partitioning. *Data Mining and Knowledge Discovery*, *2*(4), 325–344.

Bradley, P.S., Fayyad, U.M., & Reina, C.A. (1998). Scaling EM (Expectation-maximization) clustering to large databases. MSR-TR-98-35, Microsoft Research.

Bradley, P. S., Mangasarian, O. L., & Steet, W. N. (1996). Clustering via Concave minimization. In D. S. Touretzky, M. C. Mozer, & M. E. Hasselmo (Eds.), *Advances in neural information processing systems*: *vol. 8* (pp. 368–374). Cambridge (MA): MIT Press.

Broomhead, D. S., & Lowe, D. (1988). Multivariable functional interpolation and adaptive networks. *Complex Systems*, *2*, 321–355.

Bruske, J., & Sommer, G. (1995). Dynamic Cell Structure. In G. Tesauro, D. S. Touretzky, & T. K. Leen (Eds.), *Advances in neural information processing systems*: *vol. 7* (pp. 497–504). Cambridge (MA): MIT Press.

Bruzzone, L., & Prieto, D. F. (1998). Supervised training technique for radial basis function neural networks. *Electronics Letters*, *34*(11), 1115–1116.

Buckley, J. J., & Eslami, E. (2002). *An introduction to fuzzy logic and fuzzy sets*. Heidelberg: Physica-Verlag.

Buhmann, J., & Kuhnel, H. (1993). Vector quantization with complexity costs. *IEEE Transactions on Information Theory*, *39*(4), 1133–1145.

Burke, L. I. (1991). Clustering characterization of adaptive resonance. *Neural Networks*, *4*(4), 485–491.

Calvert, B. D., & Marinov, C. A. (2000). Another K-winners-take-all analog neural network. *IEEE Transactions on Neural Networks*, *11*(4), 829–838.

Camastra, F., & Verri, A. (2005). A novel kernel method for clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *27*(5), 801–805.

Cao, Y., & Wu, J. (2002). Projective ART for clustering data sets in high dimensional spaces. *Neural Networks*, *15*, 105–120.

Carpenter, G. A. (1997). Distributed learning, recognition, and prediction by ART and ARTMAP neural networks. *Neural Networks*, *10*, 1473–1494.

Carpenter, G.A. (2003). Default ARTMAP. In *Proc int joint conf neural netw*, vol. 2 (pp. 1396–1401).

Carpenter, G. A., & Grossberg, S. (1987a). A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision, Graphics, Image Processing*, *37*, 54–115.

Carpenter, G. A., & Grossberg, S. (1987b). ART 2: Self-organization of stable category recognition codes for analog input patterns. *Applied Optics*, *26*, 4919–4930.

Carpenter, G. A., & Grossberg, S. (1988). The ART of adaptive pattern recognition by a self-organizing neural network. *Computer*, *21*, 77–88.

Carpenter, G. A., & Grossberg, S. (1990). ART 3: Hierarchical search using chemical transmitters in self-organizing pattern recognition architectures. *Neural Networks*, *3*, 129–152.

Carpenter, G., Grossberg, S., & Rosen, D. B. (1991a). Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system. *Neural Networks*, *4*, 759–771.

Carpenter, G., Grossberg, S., & Rosen, D.B. (1991b). ART 2-A: An adaptive resonance algorithm for rapid category learning and recognition, *Proc int joint conf neural netw*, vol. 2, (pp. 151–156) Also: Neural Netw 4: 493–504.

Carpenter, G. A., Grossberg, S., & Reynolds, J. H. (1991). ARTMAP: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network. *Neural Networks*, *4*(5), 565–588.

Carpenter, G. A., Grossberg, S., Markuzon, N., Reynolds, J. H., & Rosen, D. B. (1992). Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps. *IEEE Transactions on Neural Networks*, *3*, 698–713.

Carpenter, G. A., & Markuzon, N. (1998). ARTMAP-IC and medical diagnosis: Instance counting and inconsistent cases. *Neural Networks*, *11*, 323–336.

Carpenter, G. A., & Ross, W. D. (1995). ART-EMAP: A neural network architecture for object recognition by evidence accumulation. *IEEE Transactions on Neural Networks*, *6*(4), 805–818.

Celeux, G., & Govaert, G. (1992). A classification EM algorithm for clustering and two stochastic versions. *Computational Statistics and Data Analysis*, *14*, 315–332.

Chen, C. L., Chen, W. C., & Chang, F. Y. (1993). Hybrid learning algorithm for Gaussian potential function networks. *Proceedings of IEE–D*, *140*(6), 442–448.

Cheng, T. W., Goldgof, D. B., & Hall, L. O. (1998). Fast fuzzy clustering. *Fuzzy Sets and Systems*, *93*, 49–56.

Cheung, Y. M. (2003). k*-Means: A new generalized k-means clustering algorithm. *Pattern Recognition Letters*, *24*, 2883–2893.

Chiang, J., & Hao, P. (2003). A new kernel-based fuzzy clustering approach: Support vector clustering with cell growing. *IEEE Transactions on Fuzzy Systems*, *11*(4), 518–527.

Chinrunrueng, C., & Sequin, C. H. (1995). Optimal adaptive k-means algorithm with dynamic adjustment of learning rate. *IEEE Transactions on Neural Networks*, *6*(1), 157–169.

Chiu, S. (1994a). Fuzzy model identification based on cluster estimation. *Journal Intelligent and Fuzzy Systems*, *2*(3), 267–278.

Chiu, S.L. (1994b). A cluster estimation method with extension to fuzzy model identification. In *Proc IEEE int conf fuzzy syst*, vol. 2, (pp. 1240–1245).

Choi, D. I., & Park, S. H. (1994). Self-creating and organizing neural network. *IEEE Transactions on Neural Networks*, *5*(4), 561–575.

Choy, C. S. T., & Siu, W. C. (1998a). A class of competitive learning models which avoids neuron underutilization problem. *IEEE Transactions on Neural Networks*, *9*(6), 1258–1269.

Choy, C. S. T., & Siu, W. C. (1998b). Fast sequential implementation of neural-gas network for vector quantization. *IEEE Transactions on Communications*, *46*(3), 301–304.

Chua, L. O., & Yang, L. (1988). Cellular neural network–Part I: Theory; Part II: Applications. *IEEE Transaction on Circuits Systems*, *35*, 1257–1290.

Chung, F. L., & Lee, T. (1994). Fuzzy competitive learning. *Neural Networks*, *7*(3), 539–551.

Corchado, J., & Fyfe, C. (2000). A comparison of kernel methods for instantiating case based reasoning systems. *Computing and Information Systems*, *7*, 29–42.

Cottrell, M., Ibbou, S., & Letremy, P. (2004). SOM-based algorithms for qualitative variables. *Neural Networks*, *17*, 1149–1167.

Dave, R. N. (1990). Fuzzy-shell clustering and applications to circle detection in digital images. *International Journal of General Systems*, *16*, 343–355.

Dave, R. N. (1991). Characterization and detection of noise in clustering. *Pattern Recognition Letters*, *12*, 657–664.

Dave, R. N., & Krishnapuram, R. (1997). Robust clustering methods: A unified view. *IEEE Transactions on Fuzzy Systems*, *5*(2), 270–293.

Dave, R. N., & Sen, S. (2002). Robust fuzzy clustering of relational data. *IEEE Transactions on Fuzzy Systems*, *10*(6), 713–727.

Davies, D. L., & Bouldin, D. W. (1979). A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *1*(4), 224–227.

Dempsey, G. L., & McVey, E. S. (1993). Circuit implementation of a peak detector neural network. *IEEE Transactions on Circuits and Systems–II*, *40*, 585–591.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, *39*(1), 1–38.

Delport, V. (1996). Codebook design in vector quantisation using a hybrid system of parallel simulated annealing and evolutionary selection. *Electronics Letters*, *32*(13), 1158–1160.

Desieno, D. (1988). Adding a conscience to competitive learning. In *Proc IEEE int conf neural netw*, vol. 1 (pp. 117–124).

Ding, C., & He, X. (2004). Cluster structure of k-means clustering via principal component analysis. In: *Proc 8th Pacific-Asia conf on advances in knowledge discov data mining (PAKDD 2004)*, (pp. 414–418).

Du, K. L., & Swamy, M. N. S. (2006). *Neural networks in a softcomputing framework*. London: Springer.

Duda, R. O., & Hart, P. E. (1973). *Pattern classification and scene analysis*. New York: Wiley.

Dunn, J. C. (1974). Some recent investigations of a new fuzzy partitioning algorithm and its applicatiopn to pattern classification problems. *Journla of Cybernetics*, *4*, 1–15.

El-Sonbaty, Y., & Ismail, M. (1998). Fuzzy clustering for symbolic data. *IEEE Transactions on Fuzzy Systems*, *6*(2), 195–204.

Ester, M., Kriegel, H.P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc 2nd int conf knowledge discovery & data mining (KDD-96)*, (pp. 226–231).

Etzioni, O. (1996). The World-Wide Web: Quagmire or gold mine?. *Communications of the ACM*, *39*(11), 65–68.

Flanagan, J. A. (1996). Self-organization in Kohonen's SOM. *Neural Networks*, *9*(7), 1185–1197.

Frigui, H., & Krishnapuram, R. (1999). A robust competitive clustering algorithm with applications in computer vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *21*(5), 450–465.

Fritzke, B. (1994a). Growing cell structures—A self-organizing neural networks for unsupervised and supvised learning. *Neural Networks*, *7*(9), 1441–1460.

Fritzke, B. (1995a). A growing neural gas network learns topologies. In G. Tesauro, D. S. Touretzky, & T. K. Leen (Eds.), *Advances in neural information processing systems*: *7* (pp. 625–632). Cambridge, MA: MIT Press.

Fritzke, B. (1995b). Growing grid–A self-organizing network with constant neighborhood range and adaptation strength. *Neural Processing Letters*, *2*(5), 9–13.

Fritzke, B. (1997a). A self-organizing network that can follow nonstationary distributions. In W. Gerstner, A. Germond, M. Hasler, & J. D. Nicoud (Eds.), *LNCS*: *vol. 1327*. *Proc Int conf artificial neural netw, Lausanne, Switzerland* (pp. 613–618). Berlin: Springer.

Fritzke, B. (1997b). The LBG-U method for vector quantization–An improvement over LBG inspired from neural networks. *Neural Processing Letters*, *5*(1), 35–45.

Fukushima, K. (1980). Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, *36*, 193–202.

Furao, S., & Hasegawa, O. (2005). An incremental network for on-line unsupervised classification and topology learning. Neural Netw (in print).

Gao, K., Ahmad, M.O., & Swamy, M.N.S. (1991). Nonlinear signal processing with self-organizing neural networks. In *Proc IEEE int symp circuits syst*, vol. 3 (pp. 1404–1407).

Gath, I., & Geva, A. B. (1989). Unsupervised optimal fuzzy clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *11*(7), 773–781.

Gath, I., & Hoory, D. (1995). Fuzzy clustering of elliptic ring-shaped clusters. *Pattern Recognition Letters*, *16*, 727–741.

Gersho, A. (1979). Asymptotically optimal block quantization. *IEEE Transactions on Information Theory*, *25*(4), 373–380.

Geva, A. B. (1999). Hierarchical unsupervised fuzzy clustering. *IEEE Transactions on Fuzzy Systems*, *7*(6), 723–733.

Girolami, M. (2002). Mercer kernel based clustering in feature space. *IEEE Transactions on Neural Networks*, *13*(3), 780–784.

Gonzalez, J., Rojas, I., Pomares, H., Ortega, J., & Prieto, A. (2002). A new clustering technique for function approximation. *IEEE Transactions on Neural Networks*, *13*(1), 132–142.

Grossberg, S. (1976). Adaptive pattern classification and universal recording: I. Parallel development and coding of neural feature detectors; II. Feedback, expectation, olfaction, and illusions. *Biological Cybernetics*, *23*, 121–134. 187–202.

Grossberg, S. (1987). Competitive learning: From iterative activation to adaptive resonance. *Cognitive Science*, *11*, 23–63.

Guha, S., Rastogi, R., & Shim, K. (2001). CURE: An efficient clustering algorithm for large databases. *Information Systems*, *26*(1), 35–58.

Gustafson, D.E., & Kessel, W. (1979). Fuzzy clustering with a fuzzy covariance matrix. In *Proc IEEE conf decision contr* (pp. 761–766).

Hamker, F. H. (2001). Life-long learning cell structures—Continuously learning without catastrophic interference. *Neural Networks*, *14*, 551–573.

Hammer, B., Micheli, A., Sperduti, A., & Strickert, M. (2004). Recursive self-organizing network models. *Neural Networks*, *17*, 1061–1085.

Hathaway, R. J., & Bezdek, J. C. (1994). NERF c-means: Non-Euclidean re-lational fuzzy clustering. *Pattern Recognition*, *27*, 429–437.

Hathaway, R. J., & Bezdek, J. C. (2000). Generalized fuzzy c-means clustering strategies using L_p norm distances. *IEEE Transactions on Fuzzy Systems*, *8*(5), 576–582.

Hathaway, R. J., & Bezdek, J. C. (2001). Fuzzy c-means clustering of incomplete data. *IEEE Transactions on Systems Man and Cybernetics–B*, *31*(5), 735–744.

He, J., Tan, A. H., & Tan, C. L. (2004). Modified ART 2A growing network capable of generating a fixed number of nodes. *IEEE Transactions on Neural Networks*, *15*(3), 728–737.

Hoeppner, F. (1997). Fuzzy shell clustering algorithms in image processing: Fuzzy C-rectangular and 2-rectangular shells. *IEEE Transactions on Fuzzy Systems*, *5*(4), 599–613.

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Science*, *79*, 2554–2558.

Huber, P. J. (1981). *Robust statistics*. New York: Wiley.

Hung, C., & Lin, S. (1995). Adaptive Hamming net: A fast-learning ART 1 model without searching. *Neural Networks*, *8*(4), 605–618.

Huntsberger, T. L., & Ajjimarangsee, P. (1990). Parallel self-organizing feature maps for unsupervised pattern recognition. *International Journal of General Systems*, *16*, 357–372.

Jain, A., Murty, M., & Flynn, P. (1999). Data clustering: A review. *ACM Computing Surveys*, *31*(3), 264–323.

Kanungo, T., Mount, D. M., Netanyahu, N. S., Piatko, C. D., Silverman, R., & Wu, A. Y. (2002). An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *24*(7), 881–892.

Kasuba, T. (1993). Simplified fuzzy ARTMAP. *AI Expert*, *8*(11), 18–25.

Karayiannis, N. B. (1997). A methodology for constructing fuzzy algorithms for learning vector quantization. *IEEE Transactions on Neural Networks*, *8*(3), 505–518.

Karayiannis, N. B. (1999). An axiomatic approach to soft learning vector quantization and clustering. *IEEE Transactions on Neural Networks*, *10*(5), 1153–1165.

Karayiannis, N. B., & Bezdek, J. C. (1997). An integrated approach to fuzzy learning vector quantization and fuzzy c-means clustering. *IEEE Transactions on Fuzzy Systems*, *5*(4), 622–628.

Karayiannis, N. B., Bezdek, J. C., Pal, N. R., Hathaway, R. J., & Pai, P. I. (1996). Repair to GLVQ: A new family of competitive learning schemes. *IEEE Transactions on Neural Networks*, *7*(5), 1062–1071.

Karayiannis, N. B., & Pai, P. I. (1996). Fuzzy algorithms for learning vector quantization. *IEEE Transactions on Neural Networks*, *7*, 1196–1211.

Karayiannis, N. B., & Mi, G. W. (1997). Growing radial basis neural networks: Merging supervised and unsupervised learning with network growth techniques. *IEEE Transactions on Neural Networks*, *8*(6), 1492–1506.

Karayiannis, N. B., & Randolph-Gips, M. M. (2003). Soft learning vector quantization and clustering algorithms based on non-Euclidean norms: Multinorm algorithms. *IEEE Transactions on Neural Networks*, *14*(1), 89–102.
Karypis, G., Han, E. H., & Kumar, V. (1999). Chameleon: Hierarchical clustering using dynamic modeling cover feature. *Computer*, *12*, 68–75.
Kaymak, U., & Setnes, M. (2002). Fuzzy clustering with volume prototypes and adaptive cluster merging. *IEEE Transactions on Fuzzy Systems*, *10*(6), 705–712.
Kersten, P. R. (1999). Fuzzy order statistics and their application to fuzzy clustering. *IEEE Transactions on Fuzzy Systems*, *7*(6), 708–712.
Kim, D. W., Lee, K. Y., Lee, D., & Lee, K. H. (2005). A kernel-based subtractive clustering method. *Pattern Recognition Letters*, *26*, 879–891.
Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, *43*, 59–69.
Kohonen, T. (1989). *Self-organization and associative memory*. Berlin: Springer.
Kohonen, T. (1990). The self-organizing map. *Proceedings of IEEE*, *78*, 1464–1480.
Kohonen, T. (1996). Emergence of invariant-feature detectors in the adaptive-subspace self-organizing map. *Biological Cybernetics*, *75*, 281–291.
Kohonen, T. (1997). *Self-organizing maps*. Berlin: Springer.
Kohonen, T., Kangas, J., Laaksonen, J., & Torkkola, K. (1992). LVQPAK: A program package for the correct application of learning vector quantization algorithms. In *Proc int joint conf neural netw*, vol. 1 (pp. 725–730).
Kohonen, T., Oja, E., Simula, O., Visa, A., & Kangas, J. (1996). Engineering applications of the self-organizing map. *Proceedings of IEEE*, *84*(10), 1358–1384.
Kolen, J., & Hutcheson, T. (2002). Reducing the time complexity of the fuzzy *C*-means algorithm. *IEEE Transactions on Fuzzy Systems*, *10*(2), 263–267.
Krishna, K., & Murty, M. N. (1999). Genetic k-means algorithm. *IEEE Transactions on Systems Man and Cybernetics–B*, *29*(3), 433–439.
Krishnamurthy, A. K., Ahalt, S. C., Melton, D. E., & Chen, P. (1990). Neural networks for vector quantization of speech and images. *IEEE Journal on Selected Areas in Communications*, *8*(8), 1449–1457.
Krishnapuram, R., Nasraoui, O., & Frigui, H. (1992). The fuzzy c spherical shells algorithm: A new approach. *IEEE Transactions on Neural Networks*, *3*(5), 663–671.
Krishnapuram, R., & Keller, J. M. (1993). A possibilistic approach to clustering. *IEEE Transactions on Fuzzy Systems*, *1*(2), 98–110.
Krishnapuram, R., & Kim, J. (2000). Clustering algorithms based on volume criteria. *IEEE Transactions on Fuzzy Systems*, *8*(2), 228–236.
Lazzaro, J., Lyckebusch, S., Mahowald, M. A., & Mead, C. A. (1989). Winner-take-all networks of O(n) complexity. In D. S. Touretzky (Ed.)., *Advances in neural information processing systems*: *vol. 1* (pp. 703–711). San Mateo, CA: Morgan Kaufmann.
Leski, J. (2003a). Towards a robust fuzzy clustering. *Fuzzy Sets and Systems*, *137*, 215–233.
Leski, J. M. (2003b). Generalized weighted conditional fuzzy clustering. *IEEE Transactions on Fuzzy Systems*, *11*(6), 709–715.
Lin, J. S. (1999). Fuzzy clustering using a compensated fuzzy Hopfield network. *Neural Processing Letters*, *10*, 35–48.
Lin, J. S., Cheng, K. S., & Mao, C. W. (1996). A fuzzy Hopfield neural network for medical image segmentation. *IEEE Transactions on Nuclear Sciences*, *43*(4), 2389–2398.
Linde, Y., Buzo, A., & Gray, R. M. (1980). An algorithm for vector quantizer design. *IEEE Transactions on Communications*, *28*, 84–95.
Lippman, R. P. (1987). An introduction to computing with neural nets. *IEEE ASSP Magazine*, *4*(2), 4–22.
Liu, Z. Q., Glickman, M., & Zhang, Y. J. (2000). Soft-competitive learning paradigms. In Z. Q. Liu, & S. Miyamoto (Eds.), *Soft computing and human-centered machines* (pp. 131–161). New York: Springer-Verlag.
Liu, S. H., & Lin, J. S. (2000). A compensated fuzzy Hopfield neural network for codebook design in vector quantization. *International Journal of Pattern Recognition and Artifical Intelligence*, *14*(8), 1067–1079.
Lo, Z. P., & Bavarian, B. (1991). On the rate of convergence in topology preserving neural networks. *Biological Cybernetics*, *65*, 55–63.
Luk, A., & Lien, S. (1998). Learning with lotto-type competition. In *Proc int joint conf neural netw*, vol. 2 (pp. 1143–1146).
Luk, A., & Lien, S. (1999). Lotto-type competitive learning and its stability. In: *Proc int joint conf neural netw*, vol. 2 (pp. 1425–1428).
MacQueen, J.B. (1967). Some methods for classification and analysis of multivariate observations. In *Proc 5th Berkeley symp on math statistics and probability*, Univ of California Press, Berkeley (pp. 281–297).
Majani, E., Erlanson, R., & Abu-Mostafa, Y. (1989). On the k-winners-take-all network. In D. S. Touretzky (Ed.)., *Advances in neural information processing systems*: *vol. 1* (pp. 634–642). San Mateo, CA: Morgan Kaufmann.
Man, Y., & Gath, I. (1994). Detection and separation of ring-shaped clusters using fuzzy clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *16*(8), 855–861.
Mann, J. R., & Gilbert, S. (1989). An analog self-organizing neural network chip. In D. S. Touretzky (Ed.)., *Advances in neural information processing systems*: *vol. 1* (pp. 739–747). San Mateo, CA: Morgan Kaufmann.
Mao, J., & Jain, A. K. (1996). A self-organizing network for hyperellipsoidal clustering (HEC). *IEEE Transactions on Neural Networks*, *7*(1), 16–29.
Martinetz, T.M. (1993). Competitive Hebbian learning rule forms perfectly topology preserving maps. In *Proc int conf artif neural netw (ICANN)* (pp. 427–434).
Martinetz, T. M., Berkovich, S. G., & Schulten, K. J. (1993). Neural-gas network for vector quantization and its application to time-series predictions. *IEEE Transactions on Neural Networks*, *4*(4), 558–569.
Martinetz, T. M., & Schulten, K. J. (1994). Topology representing networks. *Neural Networks*, *7*, 507–522.
Massey, L. (2003). On the quality of ART1 text clustering. *Neural Networks*, *16*, 771–778.
McLachlan, G., & Basford, K. (1988). *Mixture models: Inference and application to clustering*. New York: Marcel Dekker.
Moody, J., & Darken, C. J. (1989). Fast learning in networks of locally-tuned processing units. *Neural Computation*, *1*(2), 281–294.
Moore, B. (1988). ART and pattern clustering. In D. Touretzky, G. Hinton, & T. Sejnowski (Eds.), *Proc 1988 connectionist model summer school* (pp. 174–183). San Mateo, CA: Morgan Kaufmann.
Obermayer, K., Ritter, H., & Schulten, K. (1991). Development and spatial structure of cortical feature maps: A model study. In R. P. Lippmann, J. E. Moody, & D. S. Touretzky (Eds.), *Advances in neural information processing systems*: *vol. 3* (pp. 11–17). San Mateo, CA: Morgan Kaufmann.
Odorico, R. (1997). Learning vector quantization with training count (LVQTC). *Neural Networks*, *10*(6), 1083–1088.
Pal, N. R., Bezdek, J. C., & Tsao, E. C. K. (1993). Generalized clustering networks and Kohonen's self-organizing scheme. *IEEE Transactions on Neural Networks*, *4*(2), 549–557.
Pal, N. R., & Chakraborty, D. (2000). Mountain and subtractive clustering method: Improvements and generalizations. *International Journal of Intelligent Systems*, *15*, 329–341.
Patane, G., & Russo, M. (2001). The enhanced-LBG algorithm. *Neural Networks*, *14*, 1219–1237.
Pedrycz, W., & Waletzky, J. (1997). Fuzzy clustering with partial supervision. *IEEE Transacions on Systems Man and Cynernetics–B*, *27*(5), 787–795.
Pedrycz, W. (1998). Conditional fuzzy clustering in the design of radial basis function neural networks. *IEEE Transactions on Neural Networks*, *9*(4), 601–612.
Richardt, J., Karl, F., & Muller, C. (1998). Connections between fuzzy theory, simulated annealing, and convex duality. *Fuzzy Sets and Systems*, *96*, 307–334.
Ritter, H. (1999). Self-organizing maps in non-Euclidean spaces. In E. Oja, & S. Kaski (Eds.), *Kohonen maps* (pp. 97–108). Berlin: Springer.
Rose, K. (1998). Deterministic annealing for clustering, compression, classification, regression, and related optimization problems. *Proceedings of IEEE*, *86*(11), 2210–2239.
Rose, K., Gurewitz, E., & Fox, G. C. (1990). A deterministic annealing approach to clustering. *Pattern Recognition Letters*, *11*(9), 589–594.
Rovetta, S., & Zunino, R. (1999). Efficient training of neural gas vector quantizers with analog circuit implementation. *IEEE Transactions on Circuits and Systems–II*, *46*(6), 688–698.
Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning internal representations by error propagation. In D. E. Rumelhart, & J. L. McClelland (Eds.), *Parallel distributed processing: Explorations in the microstructure of cognition, 1: Foundation* (pp. 318–362). Cambridge: MIT Press.
Rumelhart, D. E., & Zipser, D. (1985). Feature discovery by competititve learning. *Cognitive Science*, *9*, 75–112.
Runkler, T. A., & Bezdek, J. C. (1999). Alternating cluster estimation: A new tool for clustering and function approximation. *IEEE Transactions on Fuzzy Systems*, *7*(4), 377–393.
Sato, A., & Yamada, K. (1995). Generalized learning vector quantization. In G. Tesauro, D. Touretzky, & T. Leen (Eds.), *Advances in neural information processing systems*: *vol. 7* (pp. 423–429). Cambridge, MA: MIT Press.
Scholkopf, B., Smola, A., & Muller, K. R. (1998). Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, *10*, 1299–1319.
Seiler, G., & Nossek, J. (1993). Winner-take-all cellular neural networks. *IEEE Transactions on Circuits and Systems–II*, *40*(3), 184–190.
Serrano-Gotarredona, T., & Linares-Barranco, B. (1996). A modified ART 1 algorithm more suitable for VLSI implementations. *Neural Networks*, *9*(6), 1025–1043.
Shih, F. Y., Moh, J., & Chang, F. C. (1992). A new ART-based neural architecture for pattern classification and image enhancement without prior knowledge. *Pattern Recognition*, *25*(5), 533–542.
Simpson, P. K. (1992). Fuzzy min–max neural networks—Part I. Classification. *IEEE Transactions on Neural Networks*, *3*, 776–786.
Simpson, P. K. (1993). Fuzzy min–max neural networks–Part II: Clustering. *IEEE Transactions on Fuzzy Systems*, *1*(1), 32–45.
Sohn, I., & Ansari, N. (1998). Configuring RBF neural networks. *Electronics Letters*, *34*(7), 684–685.
Staiano, A., Tagliaferri, R., & Pedrycz, W. (2006). Improving RBF networks performance in regression tasks by means of a supervised fuzzy clustering. Neurocomputation (in print).
Strickert, M., & Hammer, B. (2005). Merge SOM for temporal data. *Neurocomputation*, *64*, 39–71.
Su, M. C., & Chou, C. H. (2001). A modified version of the K-means algorithm with a distance based on cluster symmetry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *23*(6), 674–680.
Su, M. C., & Liu, Y. C. (2005). A new approach to clustering data with arbitrary shapes. *Pattern Recognition*, *38*, 1887–1901.
Sum, J. P. F., Leung, C. S., Tam, P. K. S., Young, G. H., Kan, W. K., & Chan, L. W. (1999). Analysis for a class of winner-take-all model. *IEEE Transactions on Neural Networks*, *10*(1), 64–71.
Swamy, M. N. S., & Thulasiraman, K. (1981). *Graphs, networks, and algorithms*. New York: Wiley.
Tam, P.K.S., Sum, J., Leung, C.S., & Chan, L.W. (1996). Network response time for a general class of WTA. In *Progress in neural information processing: Proc int conf neural info processing*, vol. 1 (pp. 492–495).
Thulasiraman, K., & Swamy, M. N. S. (1992). *Graphs: Theory and algorithms*. New York: Wiley.

Tou, J. T., & Gonzalez, R. C. (1976). *Pattern recognition principles*. Reading, MA: Addison Wesley.
Tsekouras, G., Sarimveis, H., Kavakli, E., & Bafas, G. (2004). A hierarchical fuzzy-clustering approach to fuzzy modeling. *Fuzzy Sets and Systems*, *150*(2), 245–266.
Tsypkin, Y. Z. (1973). *Foundations of the theory of learning*. New York: Academic.
Urahama, K., & Nagao, T. (1995). K-winners-take-all circuit with O(N) complexity. *IEEE Transactions on Neural Networks*, *6*, 776–778.
Uykan, Z., Guzelis, C., Celebi, M. E., & Koivo, H. N. (2000). Analysis of input–output clustering for determining centers of RBFN. *IEEE Transactions on Neural Networks*, *11*(4), 851–858.
Vakil-Baghmisheh, M. T., & Pavesic, N. (2003). A fast simplified fuzzy ARTMAP network. *Neural Processing Letters*, *17*, 273–316.
Verzi, S.J., Heileman, G.L., Georgiopoulos, M., & Anagnostopoulos, G.C. (2003). Universal approximation with fuzzy ART and fuzzy ARTMAP. In *Proc int joint conf neural netw*, vol. 3 (pp. 1987–1992).
Vesanto, J., & Alhoniemi, E. (2000). Clustering of the self-organizing map. *IEEE Transactions on Neural Networks*, *11*(3), 586–600.
von der Malsburg, C. (1973). Self-organizing of orientation sensitive cells in the striata cortex. *Kybernetik*, *14*, 85–100.
Wang, J. H., & Rau, J. D. (2001). VQ-agglomeration: A novel approach to clustering. *IEE Proceedings–Vision Image and Signal Processing*, *148*(1), 36–44.
Williamson, J. (1996). Gaussian ARTMAP: A neural network for fast incremental learning of noisy multidimensional maps. *Neural Networks*, *9*(5), 881–897.
Wilpon, J. G., & Rabiner, L. R. (1985). A modified K-means clustering algorithm for use in isolated work recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, *33*(3), 587–594.
Xie, X. L., & Beni, G. (1991). A validity measure for fuzzy clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *13*(8), 841–847.
Xiong, H., Swamy, M. N. S., Ahmad, M. O., & King, I. (2004). Branching competitive learning network: A novel self-creating model. *IEEE Transactions on Neural Networks*, *15*(2), 417–429.
Xu, L., Krzyzak, A., & Oja, E. (1993). Rival penalized competitive learning for clustering analysis, RBF net, and curve detection. *IEEE Transactions on Neural Networks*, *4*(4), 636–649.
Xu, R., & Wunsch, D., II (2005). Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, *16*(3), 645–678.
Yager, R., & Filev, D. (1994a). Generation of fuzzy rules by mountain clustering. *Journal of Intelligent Fuzzy Systems*, *2*(3), 209–219.
Yager, R. R., & Filev, D. (1994b). Approximate clustering via the mountain method. *IEEE Transactions on Systems Man and Cybernetics*, *24*(8), 1279–1284.
Yair, E., Zeger, K., & Gersho, A. (1992). Competitive learning and soft competition for vector quantizer design. *IEEE Transactions on Signal Processing*, *40*(2), 294–309.
Yang, M. S. (1993). On a class of fuzzy classification maximum likelihood procedures. *Fuzzy Sets and Systems*, *57*, 365–375.
Yang, T. N., & Wang, S. D. (2004). Competitive algorithms for the clustering of noisy data. *Fuzzy Sets and Systems*, *141*, 281–299.
Yen, J. C., Guo, J. I., & Chen, H. C. (1998). A new k-winners-take-all neural network and its array architecture. *IEEE Transactions on Neural Networks*, *9*(5), 901–912.
Yu, J., & Yang, M. S. (2005). Optimality test for generalized FCM and its application to parameter selection. *IEEE Transactions on Fuzzy Systems*, *13*(1), 164–176.
Zahn, C. T. (1971). Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Transactions on Computers*, *20*, 68–86.
Zhang, T., Ramakrishnan, R., & Livny, M. (1996). BIRCH: An efficient data clustering method for very large databases. In: *Proc ACM SIGMOD conf on management of data* (pp. 103–114).
Zhang, Y. J., & Liu, Z. Q. (2002). Self-splitting competitive learning: A new on-line clustering paradigm. *IEEE Transactions on Neural Networks*, *13*(2), 369–380.
Zheng, G. L., & Billings, S. A. (1999). An enhanced sequential fuzzy clustering algorithm. *International Journal of Systems Science*, *30*(3), 295–307.