Computational Intelligence in Clustering Algorithms, With Applications

Rui Xu[1] and Donald Wunsch II[2]

[1] Applied Computational Intelligence Laboratory, Dept. of Electrical & Computer Engineering, University of Missouri - Rolla, Rolla, MO 65409-0249 USA
rxu@umr.edu

[2] Applied Computational Intelligence Laboratory, Dept. of Electrical & Computer Engineering, University of Missouri - Rolla, Rolla, MO 65409-0249 USA
dwunsch@umr.edu

ABSTRACT - Cluster analysis plays an important role for understanding various phenomena and exploring the nature of obtained data. A remarkable diversity of ideas, in a wide range of disciplines, has been applied to clustering research. Here, we survey clustering algorithms in computational intelligence, particularly based on neural networks and kernel-based learning. We further illustrate their applications in five real world problems.

1 Introduction

Clustering, in contrast to supervised classification, involves problems where no labeled data are available [18], [22], [28], [45]. The goal is to separate a finite unlabeled data set into a finite and discrete set of "natural", hidden data structures, rather than provide an accurate characterization of unobserved samples generated from the same probability distribution [4], [18]. One of the important properties of clustering is the subjectivity, which precludes an absolute judgment as to the relative efficacy of all clustering algorithms [4], [46].

Clustering algorithms partition data into a certain number of clusters (groups, subsets, or categories). There is no universally agreed upon definition [28]. Most researchers describe a cluster by considering the internal homogeneity and the external separation [34], [40], [45], i.e., patterns in the same cluster should be similar to each other, while patterns in different clusters should not. Both the similarity and the dissimilarity should be examinable in a clear and meaningful way. Here, we give the simple mathematical descriptions of partitional clustering and hierarchical clustering, based on [40].

Given a set of N input patterns $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_j, \ldots, \mathbf{x}_N\}$, where $\mathbf{x}_j = (x_{j1}, x_{j2}, \ldots, x_{jd})^T \in \Re^d$ and each x_{ji} measure is said to be a feature (attribute, dimension, or variable),

- (Hard) partitional clustering attempts to seek a K-partition of $\mathbf{X}$, $C = \{C_1, \ldots, C_K\}(K \leq N)$, such that
  - $C_i \neq \phi, i = 1, \ldots, K$;
  - $\bigcup_{i=1}^{K} C_i = \mathbf{X}$;
  - $C_i \cap C_j = \phi, i, j = 1, \ldots, K$ and $i \neq j$.
- Hierarchical clustering attempts to construct a tree-like nested structure partition of $\mathbf{X}$, $H = \{H_1, \ldots, H_Q\}(Q \leq N)$, such that $C_i \in H_m, C_j \in H_l$, and $m > l$ imply $C_i \in C_j$ or $C_i \cap C_j = \phi$ for all $i, j \neq i, m, l = 1, \ldots, Q$.

Clustering consists of four basic steps:

1. *Feature selection or extraction.* As pointed out in [9] and [46], feature selection chooses distinguishing features from a set of candidates, while feature extraction utilizes some transformations to generate useful and novel features.
2. *Clustering algorithm design or selection.* The step is usually combined with the proximity measure selection and the criterion function construction. The proximity measure directly affects the formation of the resulting clusters. Once it is chosen, the clustering criterion construction makes the partition of clusters an optimization problem, which is well defined mathematically.
3. *Cluster validation.* Effective evaluation standards and criteria are important to provide the users with a degree of confidence for the clustering results derived from the used algorithms.
4. *Results interpretation.* Experts in the relevant fields interpret the data partition. Further analysis, even experiments, may be required to guarantee the reliability of extracted knowledge.

The remainder of the paper is organized as follows. In Sect. 2, we briefly review major clustering techniques rooted in machine learning, computer science, and statistics. More discussions on computational intelligence technologies based clustering are given in Sect. 3 and 4. We illustrate five important applications of the clustering algorithms in Sect. 5. We conclude the paper and summarize the potential challenges in Sect. 6.

2 Clustering Algorithms

Different objects and criteria usually lead to different taxonomies of clustering algorithms [28], [40], [45], [46]. A rough but widely agreed frame is to classify clustering techniques as hierarchical clustering and partitional clustering [28], [46], as described in Sec. 1.

Hierarchical clustering (HC) algorithms organize data objects with a sequence of partitions, either from singleton clusters to a cluster including all individuals or vice versa [28]. The results of HC are usually depicted by a binary tree or dendrogram. The root node of the dendrogram represents the whole data set and each leaf node is regarded as a data object. The intermediate nodes thus describe the extent that the objects are proximal to each other; and the height of the dendrogram usually expresses the distance between each pair of objects or clusters, or an object and a cluster. The ultimate clustering results can be obtained by cutting the dendrogram at different levels. This representation provides very informative descriptions and visualization for the potential data clustering structures, especially when real hierarchical relations exist in the data. However, classical HC algorithms lack robustness and are sensitive to noise and outliers. The computational complexity for most of HC algorithms is at least $O(N^2)$ and this high cost limits their application in large-scale data.

In contrast to hierarchical clustering, partitional clustering assigns a set of objects into a pre-specified K clusters without a hierarchical structure. The principally optimal partition is infeasible in practice, due to the expensive computation [28]. Therefore, heuristic algorithms have been developed in order to seek approximate solutions. One of the important factors in partitional clustering is the criterion function [40], and the sum of squared error function is one of the most widely used, which aims to minimize the cost function. The K-means algorithm is the best-known squared error-based clustering algorithm, which is very simple and can be easily implemented in solving many practical problems [54]. It can work very well for compact and hyperspherical clusters. The time complexity of K-means is $O(NKd)$, which makes it scale well for large data sets. The major disadvantages of K-means lie in its dependence on the initial partitions and the identification of the number of clusters, the convergence problem, and the sensitivity to noise. Many variants of K-means have been proposed to address these problems, as summarized in [87]. Particularly, the stochastic optimization methods, such as the genetic algorithms, can explore the solution space more flexibly and efficiently and find the approximate global optimum [38]. However, the potential price are the difficulty of parameter selection and expensive computational complexity [87].

Hard or crisp clustering only assigns an object to one cluster. However, a pattern may also be allowed to belong to all clusters with a degree of membership, $u_{i,j} \in [0, 1]$, which represents the j^{th} membership coefficient of the i^{th} object in the cluster and satisfies the following two constraints: $\sum_{i=1}^{c} u_{i,j} = 1, \forall j$ and $\sum_{j=1}^{N} u_{i,j} < N, \forall i$, as introduced in fuzzy set theory [89]. This is particularly useful when the boundaries among the clusters are not well separated and ambiguous. Moreover, the memberships may help us discover more sophisticated relations between a given object and the disclosed clusters. The typical example is Fuzzy c-Means algorithm, together with its numerous variants [8], [43], [87].

In the probabilistic view, data points in different clusters are assumed to be generated according to different probability distributions. The mixture probability density for the whole data set is expressed as $p(\mathbf{x}|\eta) = \sum_{i=1}^{K} p(\mathbf{x}|C_i, \eta_i)P(C_i)$, where $\eta = (\eta_1, \ldots, \eta_K)$ is the parameter vector, $P(C_i)$ is the prior probability and $\sum_{i=1}^{K} P(C_i) = 1$, and $p(\mathbf{x}|C_i, \eta_i)$ is the conditional probability density. The component density can be different types of functions, or the same family, but with different parameters. If these distributions are known, finding the clusters of a given data set is equivalent to estimating the parameters of several underlying models, where Maximum Likelihood (ML) estimation can be used [22]. In the case that the solutions of the likelihood equations of ML cannot be obtained analytically, the Expectation-Maximization (EM) algorithm can be utilized to approximate the ML estimates through an iterative procedure [56]. As long as the parameter vector is decided, the posterior probability for assigning a data point to a cluster can be easily calculated with Bayes's theorem.

3 Neural Networks-Based Clustering

In competitive neural networks, active neurons reinforce their neighborhood within certain regions, while suppressing the activities of other neurons (so-called on-center/off-surround competition). Typical examples include Learning Vector Quantization (LVQ) and Self-Organizing Feature Maps (SOFM) [48], [49]. Intrinsically, LVQ performs supervised learning, and is not categorized as a clustering algorithm [49], [61]. But its learning properties provide an insight to describe the potential data structure using the prototype vectors in the competitive layer. By pointing out the limitations of LVQ, including sensitivity to initiation and lack of a definite clustering object, Pal, Bezdek and Tsao proposed a general LVQ algorithm for clustering, known as GLVQ [61]. They constructed the clustering problem as an optimization process based on minimizing a loss function, which is defined on the locally weighted error between the input pattern and the winning prototype. They also showed the relations between LVQ and the online K-means algorithm.

The objective of SOFM is to represent high-dimensional input patterns with prototype vectors that can be visualized in a usually two-dimensional lattice structure [48], [49]. Each unit in the lattice is called a neuron, and adjacent neurons are connected to each other, which gives the clear topology of how the network fits itself to the input space. Input patterns are fully connected to all neurons via adaptable weights, and during the training process, neighboring input patterns are projected into the lattice, corresponding to adjacent neurons. In this sense, some authors prefer to think of SOFM as a method to displaying latent data structure in a visual way rather than a clustering approach [61]. Basic SOFM training goes through the following steps and a variety of variants of SOFM can be found in [49].

1. Define the topology of the SOFM; Initialize the prototype vectors $\mathbf{m}_i(0), i = 1, \ldots, K$ randomly;
2. Present an input pattern $\mathbf{x}$ to the network; Choose the winning node J that is closest to $\mathbf{x}$, i.e. $J = \arg\min_j\{\|\mathbf{x} - \mathbf{m}_j\|\}$;
3. Update prototype vectors $\mathbf{m}_i(t+1) = \mathbf{m}_i(t) + h_{ci}(t)[\mathbf{x} - \mathbf{m}_i(t)]$, where $h_{ci}(t)$ is the neighborhood function that is often defined as $h_{ci}(t) = \alpha(t)\exp(\frac{-\|\mathbf{r}_c - \mathbf{r}_i\|^2}{2\sigma^2(t)})$, where $\alpha(t)$ is the monotonically decreasing learning rate, $\mathbf{r}$ represents the position of corresponding neuron, and $\sigma(t)$ is the monotonically decreasing kernel width function, or

$$h_{ci}(t) = \begin{cases} \alpha(t) & \text{if node } c \text{ belongs to the neighborhood of the winning node } J \\ 0 & \text{otherwise} \end{cases}$$

4. Repeat steps 2 and 3 until no change of neuron position that is more than a small positive number is observed.

Adaptive resonance theory (ART) was developed, by Carpenter and Grossberg, as a solution to the plasticity and stability dilemma [11], [13]. ART can learn arbitrary input patterns in a stable, fast and self-organizing way, thus overcoming the effect of learning instability that plagues many other competitive networks. ART is not, as is popularly imagined, a neural network architecture. It is a learning theory, that resonance in neural circuits can trigger fast learning. As such, it subsumes a large family of current and future neural networks architectures, with many variants. ART1 is the first member, which only deals with binary input patterns [11], although it can be extended to arbitrary input patterns by a variety of coding mechanisms. ART2 extends the applications to analog input patterns [12] and ART3 introduces a new mechanism originating from elaborate biological processes to achieve more efficient parallel search in hierarchical structures [14]. By incorporating two ART modules, which receive input patterns (ART_a) and corresponding labels (ART_b) respectively, with an inter-ART module, the resulting ARTMAP system can be used for supervised classifications [15]. The match tracking strategy ensures the consistency of category prediction between two ART modules by dynamically adjusting the vigilance parameter of ART_a. A similar idea, omitting the inter-ART module, is known as LAPART [42].

The basic ART1 architecture consists of two-layer nodes (see Fig. 1), the feature representation field F_1 and the category representation field F_2. They are connected by adaptive weights, bottom-up weight matrix $\mathbf{W}^{12}$ and top-down weight matrix $\mathbf{W}^{21}$. The prototypes of clusters are stored in layer F_2. After it is activated according to the winner-takes-all competition, an expectation is reflected in layer F_1, and compared with the input pattern. The orienting subsystem with the specified vigilance parameter $\rho(0 \leq \rho \leq 1)$ determines whether the expectation and the input are closely matched, and therefore controls the generation of new clusters. It is clear that the larger ρ is, the more clusters are generated. Once weight adaptation occurs, both bottom-up and top-down weights are updated simultaneously. This is called

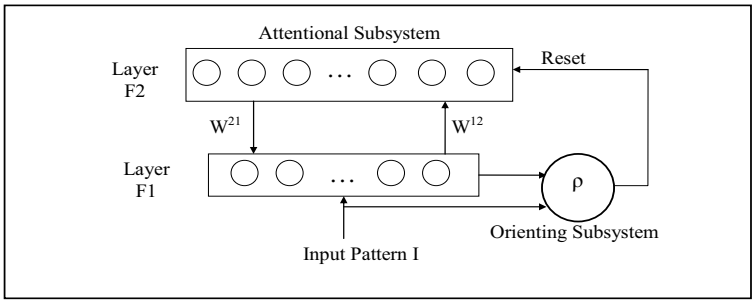

Fig. 1. ART1 Architecture.

resonance, from which the name comes. The ART1 algorithm can be described as follows:

1. Initialize weight matrices $\mathbf{W}^{12}$ and $\mathbf{W}^{21}$ as $W_{ij}^{12} = \alpha_j$, where α_j are sorted in a descending order and satisfies $0 < \alpha_j < 1/(\beta + |\mathbf{x}|)$ for $\beta > 0$ and any binary input pattern $\mathbf{x}$, and $W_{ji}^{21} = 1$;
2. For a new pattern $\mathbf{x}$, calculate the input from layer F_1 to layer F_2 as
$$T_j = \sum_{i=1}^{d} W_{ij}^{12} x_i = \begin{cases} |\mathbf{x}|\alpha_j & \text{if } j \text{ is uncommitted (first activated)} \\ \frac{|\mathbf{x} \cap \mathbf{W}_j^{21}|}{\beta + |\mathbf{W}_j^{21}|} & \text{if } j \text{ is committed} \end{cases},$$
where $\cap$ represents the logic AND operation.
3. Activate layer F_2 by choosing node J with the winner-takes-all rule $T_J = \max_j\{T_j\}$;
4. Compare the expectation from layer F_2 with the input pattern. If $\rho \leq \frac{|\mathbf{x} \cap \mathbf{W}_J^{21}|}{|\mathbf{x}|}$, go to step 5a, otherwise go to step 5b.
5. a Update the corresponding weights for the active node as $\mathbf{W}_J^{12}(new) = \frac{|\mathbf{x} \cap \mathbf{W}_J^{21}(old)|}{\beta + |\mathbf{W}_J^{21}(old)|}$ and $\mathbf{W}_J^{21}(new) = \mathbf{x} \cap \mathbf{W}_J^{21}(old)$;
   b Send a reset signal to disable the current active node by the orienting subsystem and return to step 3;
6. Present another input pattern, return to step 2 until all patterns are processed.

Note the relation between ART network and other clustering algorithms described in traditional and statistical language. Moore used several clustering algorithms to explain the clustering behaviors of ART1 and therefore induced and proved a number of important properties of ART1, notably its equivalence to varying K-means clustering [57]. She also showed how to adapt these algorithms under the ART1 framework. In [83] and [84], the ease with which ART may be used for hierarchical clustering is also discussed.

Fuzzy ART (FA) benefits the incorporation of fuzzy set theory and ART [16]. FA maintains similar operations to ART1 and uses the fuzzy set operators to replace the binary operators, so that it can work for all real data

sets. FA exhibits many desirable characteristics such as fast and stable learning and atypical pattern detection. The criticisms for FA are mostly focused on its inefficiency in handling noise and the deficiency of hyperrectangular representation for clusters [4], [5], [81]. Williamson described Gaussian ART (GA) to overcome these shortcomings, in which each cluster is modeled with Gaussian distribution and represented as a hyperellipsoid geometrically [81]. GA does not inherit the offline fast learning property of FA, as indicated by Anagnostopoulos et al. [3], who proposed Ellipsoid ART (EA) for hyperellipsoidal clusters to explore a more efficient representation of clusters, while keeping important properties of FA [3]. Baraldi and Alpaydin proposed Simplified ART (SART) following their general ART clustering networks frame, which is described through a feed-forward architecture combined with a match comparison mechanism [4]. As specific examples, they illustrated Symmetric Fuzzy ART (SFART) and Fully Self-Organizing SART (FOSART) networks. These networks outperform ART1 and FA according to their empirical studies [4].

Like ART family, there are other neural network-based constructive clustering algorithms that can adaptively and dynamically adjust the number of clusters rather than use a pre-specified and fixed number, as K-means and SOFM require [26], [62], [65], [90].

4 Kernel-Based Clustering

Kernel-based learning algorithms [60], [71], [80] are based on Cover's theorem. By nonlinearly transforming a set of complex and nonlinearly separable patterns into a higher-dimensional feature space, we can obtain the possibility to separate these patterns linearly [41]. The difficulty of curse of dimensionality can be overcome by the kernel trick, arising from Mercer's theorem [41]. By designing and calculating an inner-product kernel, we can avoid the time-consuming, sometimes even infeasible process, to explicitly describe the nonlinear mapping and compute the corresponding points in the transformed space.

In [72], Schölkopf, Smola and Müller depicted a kernel-K-means algorithm in the online mode. Suppose we have a set of patterns $\mathbf{x}_j \in \Re^d, j = 1, \ldots, N$, and a nonlinear map $\Phi : \Re^d \rightarrow F$. Here, F represents a feature space with arbitrarily high dimensionality. The object of the algorithm is to find K centers so that we can minimize the distance between the mapped patterns and their closest center $\|\Phi(\mathbf{x}) - \mathbf{m}_l\|^2 = \|\Phi(\mathbf{x}) - \sum_{j=1}^{N} \tau_{lj}\Phi(\mathbf{x}_j)\|^2 = k(\mathbf{x},\mathbf{x}) - 2\sum_{j=1}^{N} \tau_{lj} k(\mathbf{x},\mathbf{x}_j) + \sum_{i,j=1}^{N} \tau_{li}\tau_{lj} k(\mathbf{x}_i,\mathbf{x}_j)$, where $\mathbf{m}_l$ is the center for the l^{th} cluster and lies in a span of $\Phi(\mathbf{x}_1), \ldots, \Phi(\mathbf{x}_N)$, and $k(\mathbf{x},\mathbf{x}_j) = \Phi(\mathbf{x}) \cdot \Phi(\mathbf{x}_j)$ is the inner-product kernel.

Define the cluster assignment variable $C_{jl} = \begin{cases} 1 \text{ if } \mathbf{x}_j \text{ belongs to cluster } l \\ 0 \text{ otherwise} \end{cases}$, then the kernel-$K$-means algorithm can be formulated as below:

1. Initialize the centers $\mathbf{m}_l$ with the first $i, (i \geq K)$, observation patterns;
2. Take a new pattern $\mathbf{x}_{i+1}$ and calculate $C_{(i+1)h}$ as

$$C_{(i+1)h} = \begin{cases} 1 \text{ if } \|\Phi(\mathbf{x}_{i+1}) - \mathbf{m}_h\|^2 < \|\Phi(\mathbf{x}_{i+1}) - \mathbf{m}_j\|^2, \forall j \neq h \\ 0 \text{ otherwise} \end{cases};$$

3. Update the mean vector $\mathbf{m}_h$ whose corresponding $C_{(i+1)h}$ is 1,

$$\mathbf{m}_h^{new} = \mathbf{m}_h^{old} + \xi(\Phi(\mathbf{x}_{i+1}) - \mathbf{m}_h^{old}),$$

   where $\xi = C_{(i+1)h} / \sum_{j=1}^{i+1} C_{jh}$;
4. Adapt the coefficients τ_{hj} for each $\Phi(\mathbf{x}_j)$ as

$$\tau_{hj}^{new} = \begin{cases} \tau_{hj}^{old}(1-\xi) & \text{for } j \neq i+1 \\ \xi & \text{for } j = i+1 \end{cases};$$

5. Repeat the steps 2-4 until convergence is achieved.

Two variants of kernel-K-means were introduced in [20], motivated by SOFM and ART networks.

An alternative kernel-based clustering approach is in [30]. The problem was formulated to determine an optimal partition $\mathbf{\Gamma}$ to minimize the trace of within-group scatter matrix in the feature space,

$$\begin{aligned} \mathbf{\Gamma} &= \arg\min_{\mathbf{\Gamma}} Tr(S_W^{\Phi}) \\ &= \arg\min_{\mathbf{\Gamma}} Tr\{\frac{1}{N}\sum_{i=1}^{K}\sum_{j=1}^{N}\gamma_{ij}(\Phi(\mathbf{x}_j) - \mathbf{m}_i)(\Phi(\mathbf{x}_j) - \mathbf{m}_i)^T\} \\ &= \arg\min_{\mathbf{\Gamma}} \sum_{i=1}^{K} \xi_i R(\mathbf{x}|C_i) \end{aligned}$$

where $\xi_i = N_i/N$, $R(\mathbf{x}|C_i) = \frac{1}{N_i^2}\sum_{l=1}^{N}\sum_{j=1}^{N}\gamma_{il}\gamma_{ij}k(\mathbf{x}_l, \mathbf{x}_j)$, and N_i is the total number of patterns in the i^{th} cluster. The kernel function utilized in this case is the radial basis function.

Ben-Hur et al. presented a new clustering algorithm, Support Vector Clustering (SVC), in order to find a set of contours used as the cluster boundaries in the original data space [6]. These contours can be formed by mapping back the smallest enclosing sphere, which contains all the data points in the transformed feature space. Chiang and Hao extended the idea by considering each cluster corresponding to a sphere, instead of just one sphere in SVC [19]. They adopted a mechanism similar to ART to dynamically generate clusters. When an input is presented, clusters compete based on some pre-specified distance function. A validation test is performed to ensure the eligibility of the cluster to represent the input pattern. A new cluster is created as a result of the failure of all clusters available to the vigilance test. Furthermore, the distance

between the input pattern and the cluster center and the radius of the sphere provide a way to calculate the fuzzy membership function.

Kernel-based clustering algorithms have many advantages:

1. It is more possible to obtain a linearly separable hyperplane in the high-dimensional, or even infinite feature space;
2. They can form arbitrary clustering shapes other than hyperellipsoid and hypersphere;
3. Kernel-based clustering algorithms, like SVC, have the capability of dealing with noise and outliers;
4. For SVC, there is no requirement for prior knowledge to determine the system topological structure. In [30], Girolami performed eigenvalue decomposition on the kernel matrix in the high-dimensional feature space and used the dominant K components in the decomposition summation as an indication of the possible existence of K clusters.

5 Applications

Clustering has been applied in a wide variety of fields [28], [46]. We illustrate the applications of clustering algorithms in five interesting and important aspects, as described through Sect. 5.1 to 5.5.

5.1 Traveling Salesman Problem

The Traveling Salesman Problem (TSP) is one of the most studied examples in NP-complete problems. Given a complete undirected graph $G = (V, E)$, where V is a set of vertices and E is a set of edges with an associated non-negative integer cost, the most general form of the TSP is equivalent to finding any Hamiltonian cycle, which is a tour over G that begins and ends at the same vertex and visits other vertices exactly once. The more common form of the problem is the optimization problem of trying to find the shortest Hamiltonian cycle, and in particular, the most common is the Euclidean version, where the vertices and edges all lie in the plane. Mulder and Wunsch applied a divide-and-conquer clustering technique, with ART networks, to scale the problem to a million cities [59], and later, to 25 million cities [85]. The divide and conquer paradigm gives the flexibility to hierarchically break large problems into arbitrarily small clusters depending on what trade-off between accuracy and speed is desired. In addition, the sub-problems provide an excellent opportunity to take advantage of parallel systems for further optimization. As the first stage of the algorithm, ART is used to cluster the cities. The clusters were then each passed to a version of the Lin-Kernighan algorithm. The last step combines the subtours back into one complete tour. Tours with good quality for up to 25 million cities were obtained within 13,500 seconds on a 2GHz AMD Athlon MP processor with 512M of DDR RAM.

5.2 Bioinformatics - Gene Expression Data Analysis

Genome sequencing projects have achieved great advance in recent years. However, these successes can only be seen as the first step towards understanding the functions of genes and proteins and the interactions among cellular molecules. DNA microarray technologies provide an effective way to measure expression levels of tens of thousands of genes simultaneously under different conditions, which makes it possible to investigate gene activities of the whole genome [24], [53]. We demonstrate the applications of clustering algorithms in analyzing the explosively increasing gene expression data through both genes and tissues clustering.

Cluster analysis, for grouping functionally similar genes, gradually became popular after the successful application of the average linkage hierarchical clustering algorithm for the expression data of budding yeast *Saccharomyces cerevisiae* and reaction of human fibroblasts to serum by Eisen et al. [25]. They used the Pearson correlation coefficient to measure the similarity between two genes, and provided a very informative visualization of the clustering results. Their results demonstrate that functionally similar genes tend to reside in the same clusters formed by their expression pattern. Tomayo et al. made use of SOFM to cluster gene expression data and its application in hematopoietic differentiation provided new insight for further research [77]. Since many genes usually display more than one function, fuzzy clustering may be more effective in exposing these relations [21]. Gene expression data is also important to elucidate the genetic regulation mechanism in a cell. Spellman et al. clustered 800 genes according to their expression during the yeast cell cycle [75]. Analyses of 8 major gene clusters unravel the connection between co-expression and co-regulation. Tavazoie et al. partitioned 3,000 genes into 30 clusters with the K-means algorithm [78]. For each cluster, 600 base pairs upstream sequences of the genes were searched for potential motifs. 18 motifs were found from 12 clusters in their experiments and 7 of them can be verified according to previous empirical results. Fig. 2 (a) and (b) illustrate the application of hierarchical clustering and SOFM for the small round blue-cell tumors (SRBCTs) data set, which consists of the measurement of the expression levels of 2,308 genes across 83 samples [47]. Hierarchical clustering was performed by the program CLUSTER and the results were visualized by the program TreeView, developed by Eisen in Stanford University. The software package GeneCluster, developed by Whitehead Institute/MIT Center for Genome Research, was used for SOFM analysis.

In addition to genes clustering, tissues clustering are valuable in identifying samples that are in the different disease states, discovering or predicting different cancer types, and evaluating the effects of novel drugs and therapies [1], [31], [70]. Golub et al. described the restriction of traditional cancer classification methods and divided cancer classification as class discovery and class prediction. They utilized SOFM to discriminate two types of human acute leukemias: acute myeloid leukemia (AML) and acute lymphoblastic leukemia

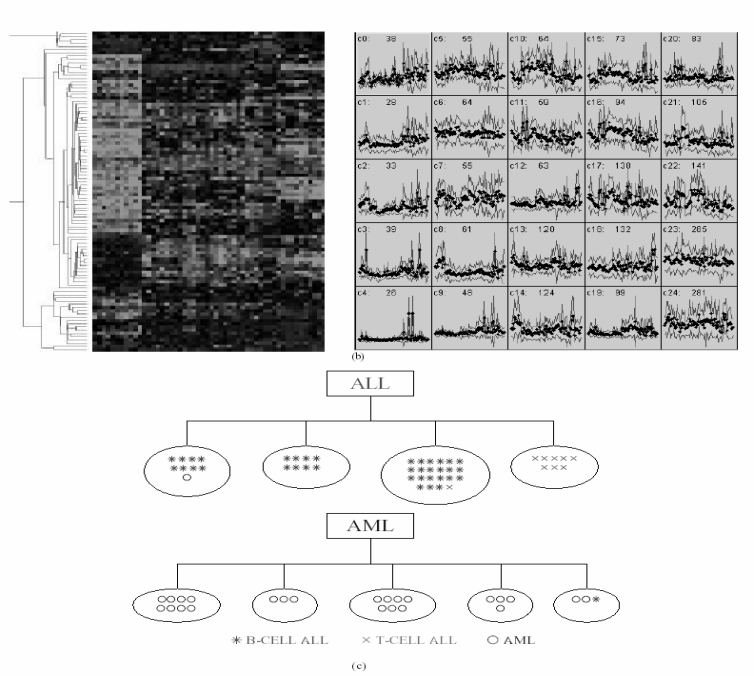

Fig. 2. Clustering for Gene Expression Data. (a) Hierarchical clustering result for the 100 selected genes from the SRBCT data set. The gene expression matrix is visualized through a color scale; (b) SOFM clustering result for all the 2308 genes of SRBCT data set. A 5x5 SOFM is used and 25 clusters are formed. Each cluster is represented by the average values; (c) EA clustering result for ALL/AML data set. EA effectively separates the two ALL subsets.

(ALL) [31]. Two subsets of ALL, with quite different origin of lineage, can be well separated. This result is also confirmed by the analysis with Ellipsoidal ART network, as illustrated in Fig. 2 (c) [86]. Alizadeh et al. successfully distinguished two molecularly distinct subtypes of diffuse large B-cell lymphoma, which cause high percentage failure in clinical treatment, based on their gene expression profiles [1]. Scherf et al. constructed a gene expression database to study the relationship between genes and drugs for 60 human cancer cell lines, which provides an important criterion for therapy selection and drug discovery [70]. Moreover, gene expression profiles are extended for patient survival analysis. Rosenwald et al. used hierarchical clustering to divide diffuse large-B-cell lymphoma, and the Kaplan-Meier estimates of the survival probabilities for each group show significant difference [66].

Furthermore, bi-clustering concept has been raised, referring to the clustering of both the genes (rows) and samples or conditions (columns) simultaneously [17]. Therefore, it is more effective in specifying a set of genes related

to some certain experimental conditions or cellular processes. A good survey paper on bi-clustering can be found in [55].

5.3 Bioinformatics - DNA or Protein Sequences Clustering

In recent decades, DNA and protein sequences grew explosively [23], [37]. For example, the recent statistics released on June 15, 2005 (Release 148.0) shows that there are 49,398,852,122 bases from 45,236,251 reported sequences in GenBank database [29]. The information hidden in the sequences offers a cue to identify functions of genes and proteins. In contrast to sequence comparison and search, cluster analysis provides a more effective way to discover complicated relations among these sequences. We summarize the following clustering applications for DNA and protein sequences:

1. Function recognition of uncharacterized genes or proteins [36];
2. Structure identification of large-scale DNA or protein databases [69], [74];
3. Redundancy decrease of large-scale DNA or protein databases [52];
4. Domain identification [27], [35];
5. EST (Expressed Sequence Tag) clustering [10].

Since biology sequential data are expressed in an alphabetic form, conventional measure methods are not appropriate. If a sequence comparison is regarded as a process of transforming a given sequence to another with a series of substitution, insertion, and deletion operations, the distance between the two sequences can be defined by virtue of the minimum number of required operations, known as edit distance [37], [68]. These edit operations are weighted according to some prior domain knowledge and the distance herein is equivalent to the minimum cost to complete the transformation. In this sense, the similarity or distance between two sequences can be reformulated as an optimal alignment problem, which fits well in the framework of dynamic programming [23]. However, for the basic alignment algorithms, the computation complexity is $O(NM)$, which is incapable of dealing with tons of nucleic acids and amino acids in the current DNA or protein databases [23]. In practice, sequence comparison or proximity measure is achieved via some heuristics, such as BLAST and FASTA with their variants [2], [63]. The key idea of these methods is to identify regions that may have potentially high matches, with a list of pre-specified high-scoring words, at an early stage. Therefore, further search only needs to focus on these regions with expensive but accurate algorithms.

Generally, there are three strategies for clustering DNA or protein sequence data. Clustering algorithms can either directly operate on a proximity measure or are based on feature extraction. They also can be constructed according to the statistical models to describe the dynamics of each group of sequences. Somervuo and Kohonen illustrated an application of SOFM to cluster protein sequences in SWISSPROT database [74]. FASTA was used to calculate the sequence similarity. Based on the similarity measure of gapped

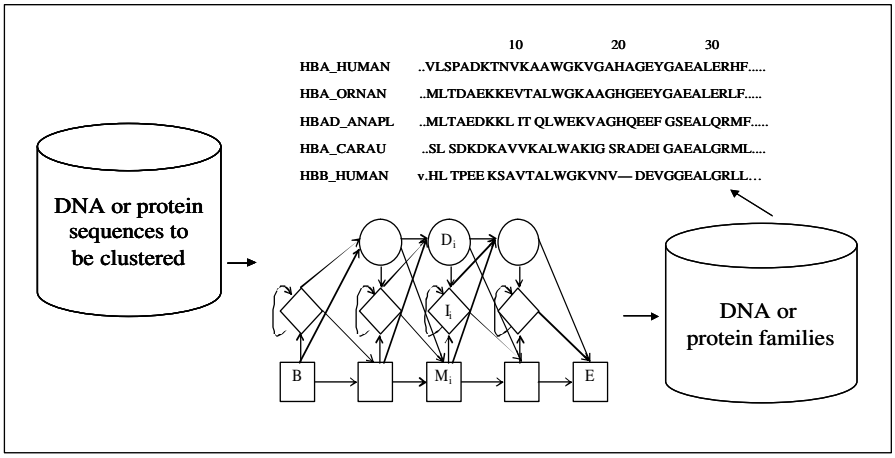

Fig. 3. DNA or Protein Clustering with HMMs. The result shown here is the part of the alignment of 9 globin sequences obtained from SWISS-PROT protein sequences databank.

BLAST, Sasson et al. utilized an agglomerative hierarchical clustering paradigm to cluster all protein sequences in SWISSPROT [69]. In contrast with the proximity-based methods, Guralnik and Karypis transformed protein or DNA sequences into a new feature space, based on the detected sub-patterns working as the sequence features, and clustered with the K-means algorithm [36]. The method is immune from all-against-all expensive sequence comparison. However, it is largely dependent on the feature selection process, which may mislead the analysis. Krogh demonstrated the power of hidden Markov models (HMMs) [64] in biological sequences modeling and clustering of protein families [51]. Fig. 3 depicts a typical clustering analysis of protein or DNA sequences with HMMs, in which match states (M), insert states (I), and delete states (D) are represented as rectangles, diamonds, and circles, respectively [23], [51]. These states correspond to substitution, insertion, and deletion in edit operations. For convenience, a begin state (B) and an end (E) state are added to the model. Either 4-letter nucleotide alphabets or 20-letter amino acid alphabets are generated from match and insert states according to some emission probability distributions. Delete states do not produce any symbols, and are used to skip the match states. K HMMs are required in order to describe K clusters, or families (subfamilies), which are regarded as a mixture model and proceeded with an EM learning algorithm. This paradigm models clusters directly from original data without additional process that may cause information loss. They provide more intuitive ways to capture the dynamics of data and more flexible means to deal with variable length sequences. However, determining the number of model components remains a complicated and uncertain process [73]. Also, the model selected is required to have sufficient complexity, in order to interpret the characteristics of data.

5.4 Dimensionality Reduction - Human Face Expression Recognition

Nowadays, it is more common to analyze data with very high dimensionality, which causes the problem curse of dimensionality [7], [41]. Fortunately, in practice, many high-dimensional data usually have an intrinsic dimensionality that is much lower than the original dimension [18]. Although strictly speaking, dimension reduction methods do not belong to clustering algorithms, they are still very important in cluster analysis. Dimensionality reduction not only reduces the computational cost and makes the high-dimensional data processible, but provides users with a clear picture and good visual examination of the data of interest. However, dimensionality reduction methods inevitably cause some information loss, and may damage the interpretability of the results, even distort the real clusters.

Unlike the typical linear components extraction techniques, like principle component analysis [22] and independent component analysis [44], Locally Linear Embedding (LLE) algorithm focuses on nonlinear dimensionality reduction [67]. LLE emphasizes the local linearity of the manifold and assumes that the local relations in the original data space (D-dimensional) are also preserved in the projected low-dimensional space (L-dimensional). This is represented through a weight matrix, describing how each point is related to the reconstruction of another data point. Therefore, the procedure for dimensional reduction can be constructed as the problem that finding L-dimensional vectors $\mathbf{y}_i$ so that the criterion function $\sum_i |\mathbf{y}_i - \sum_j w_{ij}\mathbf{y}_j|$ is minimized. This process makes LLE different from other nonlinear projection techniques, such as Multidimensional Scaling (MDS) [88] and the isometric feature mapping algorithm (ISOMAP), which extends MDS and aims to estimate the shortest path between a pair of points on a manifold, by virtue of the measured input-space distances [79]. It is worth mentioning another method, elastic maps, which seek an optimal configuration of nodes, in a sense of minimum energy, to approximate the data points [32],[33].

An application for human face expression recognition by LLE is illustrated in [67]. The data set includes 2,000 face images from the same individual with different expressions. Each input pattern is a 560-dimensional vector, corresponding to the 20x28 grayscale of the images. The faces are mapped into a two-dimensional space, consisting of the first two constructed coordinates of LLE. The result shows that LLE can effectively find and capture the data structure.

5.5 Document Clustering

Document clustering, particularly web document clustering over Internet, has become more and more important as a result of the requirement for automatic creation of documents hierarchy, information retrieval from documents collections, and search engine results analysis. Steinbach et al. compared the

performance of agglomerative hierarchical clustering and K-means clustering (with one of its variants) on 8 document data sets [76]. Kohonen et al. demonstrated the effectiveness of SOFM for clustering of a large set of documental data, in which 6,840,568 patent abstracts were projected onto a SOFM with 1,002,240 nodes [50].

Different from methods based on individual words analysis, Hammouda and Kamel proposed a phase-based incremental web document clustering system [39]. Each document consists of a set of sentences, each of which includes a sequence of words and is weighted based on the occurrence in the documents, i.e., title, keywords, figure caption, etc., and is indexed through a Document Index Graph (DIG) model. Each node in DIG corresponds to a unique word and each directed edge between a pair of words indicates the order of their occurrence in the document. The similarity measure considers four components, i.e., the number, length, frequencies, and weights of the matching phrases in two documents. The online similarity histogram-based clustering algorithm aims to maintain a high coherency in each cluster, based on the histogram of the cluster's document similarities. A new document is added into a cluster only if it increases the calculated histogram ratio or does not cause a significant decrease of the ratio while still above some minimum threshold.

6 Conclusions

As an important tool for data exploration, cluster analysis examines unlabeled data and includes a series of steps. Clustering algorithms evolve from different research communities, attempt to solve different problems, and have their own pros and cons. Particularly, clustering algorithms, based on computational intelligence technologies, play an important role and attract more intensive efforts. However, there is no universal clustering algorithm that can be applied to solve all problems. In this sense, it is not accurate to say 'best' in the context of clustering algorithms and it is important to select the appropriate methods based on the specific applications. Though we have already seen many examples of successful applications of cluster analysis, there still remain many open problems due to the existence of many inherent uncertain factors. As a conclusion, we summarize the paper with a list of some important issues and research trends for clustering algorithms, however, some more detailed requirements for specific applications will affect these properties.

1. Generate arbitrary shapes of clusters rather than be confined to some particular shape;
2. Handle large volume of data as well as high-dimensional features with acceptable time and storage complexities;
3. Detect and remove possible outliers and noise;
4. Decrease the reliance of algorithms on users-dependent parameters;
5. Have the capability of dealing with newly occurring data without relearning from the scratch;

6. Be immune to the effects of order of input patterns;
7. Provide some insight for the number of potential clusters without prior knowledge;
8. Show good data visualization and provide users with results that can simplify further analysis;
9. Be capable of handling both numerical and categorical data or be easily adaptable to some other data type.

ACKNOWLEDGMENT

We would like to thank the Eisen Laboratory in Stanford University for use of their CLUSTER and TreeView software and Whitehead Institute/MIT Center for Genome Research for use of their GeneCluster software. We thank S. Mulder for the part on the traveling salesman problem. Partial support for this research from the National Science Foundation, and from the M.K. Finley Missouri endowment, is gratefully acknowledged.

References

1. A. Alizadeh, M. Eisen, R. Davis, C. Ma, I. Lossos, A. Rosenwald, J. Boldrick, H. Sabet, T. Tran, X. Yu, J. Powell, L. Yang, G. Marti, T. Moore, J. Hudson, L. Lu, D. Lewis, R. Tibshirani, G. Sherlock, W. Chan, T. Greiner, D. Weisenburger, J. Armitage, R. Warnke, R. Levy, W. Wilson, M. Grever, J. Byrd, D. Botstein, P. Brown, and L. Staudt: Distinct types of diffuse large B-cell Lymphoma identified by gene expression profiling, *Nature*, 2000, 503–511.
2. S. Altschul, W. Gish, W. Miller, E. Myers, and D. Lipman: Basic local alignment search tool, *Journal of Molecular Biology*, 1990, 403–410.
3. G. Anagnostopoulos and M. Georgiopoulos: Ellipsoid ART and ARTMAP for incremental unsupervised and supervised learning, in: *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN'01)*, 2001, 1221–1226.
4. A. Baraldi and E. Alpaydin: Constructive feedforward ART clustering networks - Part I and II, *IEEE Transactions on Neural Networks*, 2002, 645–677.
5. A. Baraldi and P. Blonda: A survey of fuzzy clustering algorithms for pattern recognition - Part I and II, *IEEE Transactions on Systems, Man, And Cybernetics - Part B: Cybernetics*, 1999, 778–801.
6. A. Ben-Hur, D. Horn, H. Siegelmann and V. Vapnik: Support vector clustering, *Journal of Machine Learning Research*, 2001, 125–137.
7. K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft: When is nearest neighbor meaningful, in: *Proceedings of 7th International Conference on Database Theory*, 1999, 217–235.
8. J. Bezdek: *Pattern recognition with fuzzy objective function algorithms*, Plenum Press, New York, 1981.
9. C. Bishop: *Neural networks for pattern recognition*, Oxford University Press, New York, 1995.
10. J. Burke, D. Davison and W. Hide: d2_Cluster: A validated method for clustering EST and full-length cDNA sequences, *Genome Research*, 1999, 1135–1142.

11. G. Carpenter and S. Grossberg: A massively parallel architecture for a self-organizing neural pattern recognition machine, *Computer Vision, Graphics, and Image Processing*, 1987, 54–115.
12. G. Carpenter and S. Grossberg: ART2: Self-organization of stable category recognition codes for analog input patterns, *Applied Optics*, 1987, 4919–4930.
13. G. Carpenter and S. Grossberg: The ART of adaptive pattern recognition by a self-organizing neural network, *IEEE Computer*, 1988, 77–88.
14. G. Carpenter and S. Grossberg: ART3: Hierarchical search using chemical transmitters in self-organizing pattern recognition Architectures, *Neural Networks*, 1990, 129–152.
15. G. Carpenter, S. Grossberg, and J. Reynolds: ARTMAP: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network, *Neural Networks*, 1991, 169–181.
16. G. Carpenter, S. Grossberg, and D. Rosen: Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance System, *Neural Networks*, 1991, 759–771.
17. Y. Cheng and G. Church: Biclustering of Expression Data, in: *Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology (ISMB '00)*, 2000, 93–103.
18. V. Cherkassky, and F. Mulier: *Learning from data: concepts, theory, and methods*, John Wiley & Sons, Inc., 1998.
19. J. Chiang and P. Hao: A new kernel-based fuzzy clustering approach: support vector clustering with cell growing, *IEEE Transactions on Fuzzy Systems*, 2003, 518–527.
20. J. Corchado and C. Fyfe: A comparison of kernel methods for instantiating case based reasoning systems, *Computing and Information Systems*, 2000, 29–42.
21. D. Dembélé and P. Kastner: Fuzzy c-means method for clustering microarray data, *Bioinformatics*, 2003, 973–980.
22. R. Duda, P. Hart, and D. Stork: *Pattern classification*, 2nd edition, John Wiley & Sons, Inc., 2001.
23. R. Durbin, S. Eddy, A. Krogh, and G. Mitchison: *Biological sequence analysis: probabilistic models of proteins and nucleic acids*, Cambridge University Press, 1998.
24. M. Eisen and P. Brown: DNA arrays for analysis of gene expression, *Methods Enzymol*, 1999, 179–205.
25. M. Eisen, P. Spellman, P. Brown and D. Botstein: Cluster analysis and display of genome-wide expression patterns, *Proceedings of the National Academy of Science*, 1998, USA 95, 14863–14868.
26. T. Eltoft and R. deFigueiredo: A new neural network for cluster-detection-and-labeling, *IEEE Transactions on Neural Networks*, 1998, 1021–1035.
27. A. Enright and C. Ouzounis: GeneRAGE: A robust algorithm for sequence clustering and domain detection, *Bioinformatics*, 2000, 451–457.
28. B. Everitt, S. Landau, and M. Leese: *Cluster analysis*, Arnold, 2001.
29. GenBank Release Notes 148.0, 2005.
30. M. Girolami: Mercer kernel based clustering in feature space, *IEEE Transactions on Neural Networks*, 2002, 780–784.
31. T. Golub, D. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. Mesirov, H. Coller, M. Loh, J. Downing, M. Caligiuri, C. Bloomfield, and E. Lander: Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring, *Science*, 1999, 531–537.

32. A. Gorban, A. Pitenko, A. Zinovyev and D. Wunsch II: Visualization of Any Data with Elastic Map Method, *Proc. Artificial Neural Networks in Engineering*, 2001.
33. A. Gorban, A. Zinovyev and D. Wunsch II: Application of the Method of Elastic Maps in Analysis of Genetic Texts, *Proc. International Joint Conference on Neural Networks (IJCNN)*, 2003.
34. A. Gordon: *Classification*, 2nd, Chapman and Hall/CRC Press, 1999.
35. X. Guan and L. Du: Domain identification by clustering sequence alignments, *Bioinformatics*, 1998, 783–788.
36. V. Guralnik and G. Karypis: A scalable algorithm for clustering sequential data, in: *Proceedings of the 1st IEEE International Conference on Data Mining (ICDM 2001)*, 2001, 179–186.
37. D. Gusfield: *Algorithms on strings, trees, and sequences: Computer science and computational biology*, Cambridge University Press, 1997.
38. L. Hall, I. özyurt, and J. Bezdek: Clustering with a genetically optimized approach, *IEEE Transactions on Evolutionary Computation*, 1999, 103–112.
39. K. Hammouda, and M. Kamel: Efficient phrase-based document indexing for web document clustering, *IEEE Transactions on Knowledge and Data Engineering*, 2004, 1279–1296.
40. P. Hansen and B. Jaumard: Cluster analysis and mathematical programming, *Mathematical Programming*, 1997, 191–215.
41. S. Haykin: *Neural networks: A comprehensive foundation*, 2nd, Prentice Hall, 1999.
42. M. Healy, T. Caudell, and S. Smith: A neural architecture for pattern sequence verification through inferencing, *IEEE Transactions on Neural Networks*, 1993, 9–20.
43. F. Höppner, F. Klawonn, and R. Kruse: *Fuzzy cluster analysis: Methods for classification, data analysis and image recognition*, Wiley, New York, 1999.
44. A. Hyvärinen: Survey of independent component analysis, *Neural Computing Surveys*, 1999, 94–128.
45. A. Jain and R. Dubes: *Algorithms for clustering data*, Prentice Hall, Englewood Cliffs, 1988.
46. A. Jain, M. Murty, and P. Flynn: Data clustering: A review, *ACM Computing Surveys*, 1999, 264–323.
47. J. Khan, J. Wei, M. Ringnér, L. Saal, M. Ladanyi, F. Westermann, F. Berthold, M. Schwab, C. Antonescu, C. Peterson, and P. Meltzer: Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks, *Nature Medicine*, 2001, 673–679.
48. T. Kohonen: The self-organizing map, *Proceedings of the IEEE*, 1990, 1464–1480.
49. T. Kohonen: *Self-organizing maps*, 3rd edition, Springer, 2001.
50. T. Kohonen, S. Kaski, K. Lagus, J. Salojärvi, J. Honkela, V. Paatero, and A. Saarela: Self organization of a massive document collection, *IEEE Transactions on Neural Networks*, 2000, 574–585.
51. A. Krogh, M. Brown, I. Mian, K. Sjölander, and D. Haussler: Hidden Markov models in computational biology: Applications to protein modeling, *Journal of Molecular Biology*, 1994, 1501–1531.
52. W. Li, L. Jaroszewski, and A. Godzik: Clustering of highly homologous sequences to reduce the size of large protein databases, *Bioinformatics*, 2001, 282–283.

53. R. Lipshutz, S. Fodor, T. Gingeras, and D. Lockhart: High density synthetic oligonucleotide arrays, *Nature Genetics*, 1999, 20–24.
54. J. MacQueen: Some methods for classification and analysis of multivariate observations, in: *Proceedings of the Fifth Berkeley Symposium*, 1967, 281–297.
55. S. Madeira, and A. Oliveira: Biclustering algorithms for biological data analysis: a survey, *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2004, 24–45.
56. G. McLachlan and D. Peel: *Finite mixture models*, John Wiley & Sons, New York, 2000.
57. B. Moore: ART1 and pattern clustering, in: *Proceedings of the 1988 Connectionist Models Summer School*, Morgan Kaufmann, 1989, 174–185.
58. Y. Moreau, F. Smet, G. Thijs, K. Marchal, and B. Moor: Functional bioinformatics of microarray data: From expression to regulation, *Proceedings of the IEEE*, 2002, 1722–1743.
59. S. Mulder and D. Wunsch: Million city traveling salesman problem solution by divide and conquer clustering with adaptive resonance neural networks, *Neural Networks*, 2003, 827–832.
60. K. Müller, S. Mika, G. Rätsch, K. Tsuda and B. Schölkopf: An introduction to kernel-based learning algorithms, *IEEE Transactions on Neural Networks*, 2001, 181–201.
61. N. Pal, J. Bezdek, and E. Tsao: Generalized clustering networks and Kohonen's self-organizing scheme, *IEEE Transactions on Neural Networks*, 1993, 549–557.
62. G. Patanè and M. Russo: Fully automatic clustering system, *IEEE Transactions on Neural Networks*, 2002, 1285–1298.
63. W. Pearson: Improved tools for biological sequence comparison, *Proceedings of the National Academy of Science*, 1988, USA 85, 2444–2448.
64. L. Rabiner: A tutorial on hidden Markov models and selected applications in speech recognition, *Proceedings of the IEEE*, 1989, 257–286.
65. S. Ridella, S. Rovetta, and R. Zunino: Plastic algorithm for adaptive vector quantization, *Neural Computing and Applications*, 1998, 37–51.
66. A. Rosenwald, G. Wright, W. Chan, J. Connors, C. Campo, R. Fisher, R. Gascoyne, H. Muller-Hermelink, E. Smeland, and L. Staudt: The use of molecular profiling to predict survival after chemotherapy for diffuse large-B-cell lymphoma, *The New England Journal of Medicine*, 2002, 1937–1947.
67. S. Roweis and L. Saul: Nonlinear dimensionality reduction by locally linear embedding, *Science*, 2000, 2323–2326.
68. D. Sankoff and J. Kruskal: *Time warps, string edits, and macromolecules: the theory and practice of sequence comparison*, CSLI publications, 1999.
69. O. Sasson, N. Linial and M. Linial: The metric space of proteins - comparative study of clustering algorithms, *Bioinformatics*, 2002, s14–s21.
70. U. Scherf, D. Ross, M. Waltham, L. Smith, J. Lee, L. Tanabe, K. Kohn, W. Reinhold, T. Myers, D. Andrews, D. Scudiero, M. Eisen, E. Sausville, Y. Pommier, D. Botstein, P. Brown, and J. Weinstein: A gene expression database for the molecular pharmacology of cancer, *Nature Genetics*, 2000, 236–44.
71. B. Schölkopf and A. Smola: *Learning with kernels: Support vector machines, regularization, optimization, and beyond*, The MIT Press, Cambridge, MA, 2002.
72. B. Schölkopf, A. Smola and K. Müller: Nonlinear component analysis as a kernel eigenvalue problem, *Neural Computation*, 1998, 1299–1319.

73. P. Smyth: Clustering sequences with hidden Markov models, in: *Advances in Neural Information Processing*, M. Mozer, M. Jordan and T. Petsche (eds.), MIT Press, 1997, 648–654.
74. P. Somervuo and T. Kohonen: Clustering and visualization of large protein sequence databases by means of an extension of the self-organizing map, *LNAI 1967*, 2000, 76–85.
75. P. Spellman, G. Sherlock, M. Ma, V. Iyer, K. Anders, M. Eisen, P. Brown, D. Botstein, and B. Futcher: Comprehensive identification of cell cycle-regulated genes of the Yeast Saccharomyces Cerevisiae by microarray hybridization, *Mol. Biol. Cell*, 1998, 3273–3297.
76. M. Steinbach, G. Karypis, and V. Kumar: A comparison of document clustering techniques, in: *Proceedings of KDD Workshop on Text Mining*, 2000.
77. P. Tamayo, D. Slonim, J. Mesirov, Q. Zhu, S. Kitareewan, E. Dmitrovsky, E. Lander, and T. Golub: Interpreting patterns of gene expression with self-organizing maps: Methods and application to Hematopoietic differentiation, *Proceedings of the National Academy of Science*, 1999, USA 96, 2907–2912.
78. S. Tavazoie, J. Hughes, M. Campbell, R. Cho, and G. Church: Systematic determination of genetic network architecture, *Nature Genetics*, 1999, 281–285.
79. J. Tenenbaum, V. Silva, and J. Langford: A global geometric framework for nonlinear dimensionality reduction, *Science*, 2000, 2319–2323.
80. V. Vapnik: Statistical learning theory, John Wiley & Sons, New York, 1998.
81. J. Williamson: Gaussian ARTMAP: A neural network for fast incremental learning of noisy multidimensional maps, *Neural Networks*, 1996, 881–897.
82. S. Wu, A. Liew, H. Yan, and M. Yang: Cluster analysis of gene expression data based on self-splitting and merging competitive learning, *IEEE Transactions on Information Technology in Biomedicine*, 2004, 5–15.
83. D. Wunsch: An optoelectronic learning machine: Invention, experimentation, analysis of first hardware implementation of the ART1 neural network, Ph.D. dissertation, University of Washington, 1991.
84. D. Wunsch, T. Caudell, C. Capps, R. Marks, and R. Falk: An optoelectronic implementation of the adaptive resonance neural network, *IEEE Transactions on Neural Networks*, 1993, 673–684.
85. D. Wunsch, S. Mulder: Evolutionary Algorithms, Markov Decision Processes, Adaptive Critic Designs, and Clustering: Commonalities, Hybridization, and Performance, in: *Proceedings of IEEE International Conference on Intelligent Sensing and Information Processing*, 2004.
86. R. Xu, G. Anagnostopoulos and D. Wunsch: Tissue classification through analysis of gene expression data using a new family of ART architectures, in: *Proceedings of International Joint Conference on Neural Networks 02*, 2002, 300–304.
87. R. Xu, and D. Wunsch: Survey of clustering algorithms, *IEEE Transactions on Neural Networks*, 2005, 645–678.
88. F. Young and R. Hamer: *Multidimensional scaling: History, theory, and applications*, Hillsdale, NJ: Lawrence Erlbaum Associates, 1987.
89. L. Zadeh: Fuzzy sets, *Information and Control*, 1965, 338–353.
90. Y. Zhang and Z. Liu: Self-splitting competitive learning: A new on-line clustering paradigm, *IEEE Transactions on Neural Networks*, 2002, 369–380.