Survey of Clustering Algorithms

Rui Xu, *Student Member, IEEE* and Donald Wunsch II, *Fellow, IEEE*

***Abstract*—Data analysis plays an indispensable role for understanding various phenomena. Cluster analysis, primitive exploration with little or no prior knowledge, consists of research developed across a wide variety of communities. The diversity, on one hand, equips us with many tools. On the other hand, the profusion of options causes confusion. We survey clustering algorithms for data sets appearing in statistics, computer science, and machine learning, and illustrate their applications in some benchmark data sets, the traveling salesman problem, and bioinformatics, a new field attracting intensive efforts. Several tightly related topics, proximity measure, and cluster validation, are also discussed.**

***Index Terms*—Adaptive resonance theory (ART), clustering, clustering algorithm, cluster validation, neural networks, proximity, self-organizing feature map (SOFM).**

I. Introduction

WE ARE living in a world full of data. Every day, people encounter a large amount of information and store or represent it as data, for further analysis and management. One of the vital means in dealing with these data is to classify or group them into a set of categories or clusters. Actually, as one of the most primitive activities of human beings [14], classification plays an important and indispensable role in the long history of human development. In order to learn a new object or understand a new phenomenon, people always try to seek the features that can describe it, and further compare it with other known objects or phenomena, based on the similarity or dissimilarity, generalized as proximity, according to some certain standards or rules. "Basically, classification systems are either supervised or unsupervised, depending on whether they assign new inputs to one of a finite number of discrete supervised classes or unsupervised categories, respectively [38], [60], [75]. In supervised classification, the mapping from a set of input data vectors ($\mathbf{x} \in \Re^d$, where d is the input space dimensionality), to a finite set of discrete class labels ($y \in 1, \ldots C$, where C is the total number of class types), is modeled in terms of some mathematical function $y = y(\mathbf{x}, \mathbf{w})$, where $\mathbf{w}$ is a vector of adjustable parameters. The values of these parameters are determined (optimized) by an inductive learning algorithm (also termed inducer), whose aim is to minimize an empirical risk functional (related to an inductive principle) on a finite data set of input–output examples, $(\mathbf{x}_i, y_i), i = 1, \ldots, N$, where N is the finite cardinality of the available representative data set [38], [60], [167]. When the inducer reaches convergence or terminates, an induced classifier is generated [167].

In unsupervised classification, called clustering or exploratory data analysis, no labeled data are available [88], [150]. The goal of clustering 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 [23], [60]. This can make the task of clustering fall outside of the framework of unsupervised predictive learning problems, such as vector quantization [60] (see Section II-C), probability density function estimation [38] (see Section II-D), [60], and entropy maximization [99]. It is noteworthy that clustering differs from multidimensional scaling (perceptual maps), whose goal is to depict all the evaluated objects in a way that minimizes the topographical distortion while using as few dimensions as possible. Also note that, in practice, many (predictive) vector quantizers are also used for (nonpredictive) clustering analysis [60].

Nonpredictive clustering is a subjective process in nature, which precludes an absolute judgment as to the relative efficacy of all clustering techniques [23], [152]. As pointed out by Backer and Jain [17], "in cluster analysis a group of objects is split up into a number of more or less homogeneous subgroups on the basis of an often subjectively chosen measure of similarity (i.e., chosen subjectively based on its ability to create "interesting" clusters), such that the similarity between objects within a subgroup is larger than the similarity between objects belonging to different subgroups""[1].

Clustering algorithms partition data into a certain number of clusters (groups, subsets, or categories). There is no universally agreed upon definition [88]. Most researchers describe a cluster by considering the internal homogeneity and the external separation [111], [124], [150], 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 some simple mathematical descriptions of several types of clustering, based on the descriptions in [124].

Given a set of 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 measure x_{ji} 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
  1) $C_i \neq \phi, i = 1, \ldots, K$;
  2) $\bigcup_{i=1}^{K} C_i = \mathbf{X}$;
  3) $C_i \cap C_j = \phi, i, j = 1, \ldots, K$ and $i \neq j$.

Manuscript received March 31, 2003; revised September 28, 2004. This work was supported in part by the National Science Foundation and in part by the M. K. Finley Missouri Endowment.

The authors are with the Department of Electrical and Computer Engineering, University of Missouri-Rolla, Rolla, MO 65409 USA (e-mail: rxu@umr.edu; dwunsch@ece.umr.edu).

Digital Object Identifier 10.1109/TNN.2005.845141

[1]The preceding quote is taken verbatim from verbiage suggested by the anonymous associate editor, a suggestion which we gratefully acknowledge.

GOOGLE EXHIBIT 1017
Google v. K.Mizra
IPR2026-00304

1045-9227/$20.00 © 2005 IEEE

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

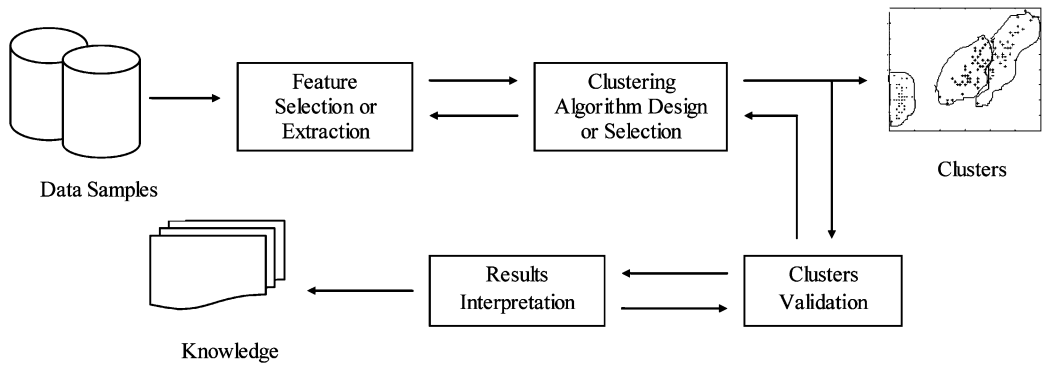


Fig. 1. Clustering procedure. The typical cluster analysis consists of four steps with a feedback pathway. These steps are closely related to each other and affect the derived clusters.

•) 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 $i, j \neq i, m, l = 1, \ldots, Q$.

For hard partitional clustering, each pattern only belongs 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 membership coefficient of the jth object in the ith cluster and satisfies the following two constraints:

$$\sum_{i=1}^{c} u_{i,j} = 1, \quad \forall j \quad \text{and} \quad \sum_{j=1}^{N} u_{i,j} < N, \quad \forall i$$

as introduced in fuzzy set theory [293]. This is known as fuzzy clustering, reviewed in Section II-G.

Fig. 1 depicts the procedure of cluster analysis with four basic steps.

1) *Feature selection or extraction*. As pointed out by Jain *et al.* [151], [152] and Bishop [38], feature selection chooses distinguishing features from a set of candidates, while feature extraction utilizes some transformations to generate useful and novel features from the original ones. Both are very crucial to the effectiveness of clustering applications. Elegant selection of features can greatly decrease the workload and simplify the subsequent design process. Generally, ideal features should be of use in distinguishing patterns belonging to different clusters, immune to noise, easy to extract and interpret. We elaborate the discussion on feature extraction in Section II-L, in the context of data visualization and dimensionality reduction. More information on feature selection can be found in [38], [151], and [250].

2) *Clustering algorithm design or selection*. The step is usually combined with the selection of a corresponding proximity measure and the construction of a criterion function. Patterns are grouped according to whether they resemble each other. Obviously, the proximity measure directly affects the formation of the resulting clusters. Almost all clustering algorithms are explicitly or implicitly connected to some definition of proximity measure. Some algorithms even work directly on the proximity matrix, as defined in Section II-A. Once a proximity measure is chosen, the construction of a clustering criterion function makes the partition of clusters an optimization problem, which is well defined mathematically, and has rich solutions in the literature. Clustering is ubiquitous, and a wealth of clustering algorithms has been developed to solve different problems in specific fields. However, there is no clustering algorithm that can be universally used to solve all problems. "It has been very difficult to develop a unified framework for reasoning about it (clustering) at a technical level, and profoundly diverse approaches to clustering" [166], as proved through an impossibility theorem. Therefore, it is important to carefully investigate the characteristics of the problem at hand, in order to select or design an appropriate clustering strategy.

3) *Cluster validation*. Given a data set, each clustering algorithm can always generate a division, no matter whether the structure exists or not. Moreover, different approaches usually lead to different clusters; and even for the same algorithm, parameter identification or the presentation order of input patterns may affect the final results. Therefore, 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. These assessments should be objective and have no preferences to any algorithm. Also, they should be useful for answering questions like how many clusters are hidden in the data, whether the clusters obtained are meaningful or just an artifact of the algorithms, or why we choose some algorithm instead of another. Generally, there are three categories of testing criteria: external indices, internal indices, and relative indices. These are defined on three types of clustering structures, known as partitional clustering, hierarchical clustering, and individual clusters [150]. Tests for the situation, where no clustering structure exists in the data, are also considered [110], but seldom used, since users are confident of the presence of clusters. External indices are based on some prespecified structure, which is the reflection of prior information on the data, and used as a standard to validate the clustering solutions. Internal tests are not dependent on external information (prior knowledge). On the contrary, they examine the clustering structure directly from the original data. Relative criteria place

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

the emphasis on the comparison of different clustering structures, in order to provide a reference, to decide which one may best reveal the characteristics of the objects. We will not survey the topic in depth and refer interested readers to [74], [110], and [150]. However, we will cover more details on how to determine the number of clusters in Section II-M. Some more recent discussion can be found in [22], [37], [121], [180], and [181]. Approaches for fuzzy clustering validity are reported in [71], [104], [123], and [220].

4) *Results interpretation.* The ultimate goal of clustering is to provide users with meaningful insights from the original data, so that they can effectively solve the problems encountered. Experts in the relevant fields interpret the data partition. Further analyzes, even experiments, may be required to guarantee the reliability of extracted knowledge.

Note that the flow chart also includes a feedback pathway. Cluster analysis is not a one-shot process. In many circumstances, it needs a series of trials and repetitions. Moreover, there are no universal and effective criteria to guide the selection of features and clustering schemes. Validation criteria provide some insights on the quality of clustering solutions. But even how to choose the appropriate criterion is still a problem requiring more efforts.

Clustering has been applied in a wide variety of fields, ranging from engineering (machine learning, artificial intelligence, pattern recognition, mechanical engineering, electrical engineering), computer sciences (web mining, spatial database analysis, textual document collection, image segmentation), life and medical sciences (genetics, biology, microbiology, paleontology, psychiatry, clinic, pathology), to earth sciences (geography. geology, remote sensing), social sciences (sociology, psychology, archeology, education), and economics (marketing, business) [88], [127]. Accordingly, clustering is also known as numerical taxonomy, learning without a teacher (or unsupervised learning), typological analysis and partition. The diversity reflects the important position of clustering in scientific research. On the other hand, it causes confusion, due to the differing terminologies and goals. Clustering algorithms developed to solve a particular problem, in a specialized field, usually make assumptions in favor of the application of interest. These biases inevitably affect performance in other problems that do not satisfy these premises. For example, the K-means algorithm is based on the Euclidean measure and, hence, tends to generate hyperspherical clusters. But if the real clusters are in other geometric forms, K-means may no longer be effective, and we need to resort to other schemes. This situation also holds true for mixture-model clustering, in which a model is fit to data in advance.

Clustering has a long history, with lineage dating back to Aristotle [124]. General references on clustering techniques include [14], [75], [77], [88], [111], [127], [150], [161], [259]. Important survey papers on clustering techniques also exist in the literature. Starting from a statistical pattern recognition viewpoint, Jain, Murty, and Flynn reviewed the clustering algorithms and other important issues related to cluster analysis [152], while Hansen and Jaumard described the clustering problems under a mathematical programming scheme [124]. Kolatch and He investigated applications of clustering algorithms for spatial database systems [171] and information retrieval [133], respectively. Berkhin further expanded the topic to the whole field of data mining [33]. Murtagh reported the advances in hierarchical clustering algorithms [210] and Baraldi surveyed several models for fuzzy and neural network clustering [24]. Some more survey papers can also be found in [25], [40], [74], [89], and [151]. In addition to the review papers, comparative research on clustering algorithms is also significant. Rauber, Paralic, and Pampalk presented empirical results for five typical clustering algorithms [231]. Wei, Lee, and Hsu placed the emphasis on the comparison of fast algorithms for large databases [280]. Scheunders compared several clustering techniques for color image quantization, with emphasis on computational time and the possibility of obtaining global optima [239]. Applications and evaluations of different clustering algorithms for the analysis of gene expression data from DNA microarray experiments were described in [153], [192], [246], and [271]. Experimental evaluation on document clustering techniques, based on hierarchical and K-means clustering algorithms, were summarized by Steinbach, Karypis, and Kumar [261].

In contrast to the above, the purpose of this paper is to provide a comprehensive and systematic description of the influential and important clustering algorithms rooted in statistics, computer science, and machine learning, with emphasis on new advances in recent years.

The remainder of the paper is organized as follows. In Section II, we review clustering algorithms, based on the natures of generated clusters and techniques and theories behind them. Furthermore, we discuss approaches for clustering sequential data, large data sets, data visualization, and high-dimensional data through dimension reduction. Two important issues on cluster analysis, including proximity measure and how to choose the number of clusters, are also summarized in the section. This is the longest section of the paper, so, for convenience, we give an outline of Section II in bullet form here:

II. Clustering Algorithms

- *A*. Distance and Similarity Measures (See also Table I)
- *B*. Hierarchical
  - — Agglomerative
    - ■ Single linkage, complete linkage, group average linkage, median linkage, centroid linkage, Ward's method, balanced iterative reducing and clustering using hierarchies (BIRCH), clustering using representatives (CURE), robust clustering using links (ROCK) . . .
  - — Divisive
    - ■ divisive analysis (DIANA), monothetic analysis (MONA) . . .
- *C*. Squared Error-Based (Vector Quantization)
  - — K-means, iterative self-organizing data analysis technique (ISODATA), genetic K-means algorithm (GKA), partitioning around medoids (PAM) . . .
- *D*. pdf Estimation via Mixture Densities
  - — Gaussian mixture density decomposition (GMDD), AutoClass . . .
- *E*. Graph Theory-Based
  - — Chameleon, Delaunay triangulation graph (DTG), highly connected subgraphs (HCS), clustering iden-

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

TABLE I
SIMILARITY AND DISSIMILARITY MEASURE FOR QUANTITATIVE FEATURES

Measures	Forms	Comments	Examples and Applications
Minkowski distance	$D_{ij} = \left(\sum_{l=1}^{d} \lvert x_{il} - x_{jl} \rvert^{1/n} \right)^{n}$	Metric. Invariant to any translation and rotation only for $n = 2$ (Euclidean distance). Features with large values and variances tend to dominate over other features.	Fuzzy c-means with measures based on Minkowski family [130].
Euclidean distance	$D_{ij} = \left(\sum_{l=1}^{d} \lvert x_{il} - x_{jl} \rvert^{1/2} \right)^{2}$	The most commonly used metric. Special case of Minkowski metric at $n = 2$. Tend to form hyperspherical clusters.	K-means algorithm [191]
City-block distance	$D_{ij} = \sum_{l=1}^{d} \lvert x_{il} - x_{jl} \rvert$	Special case of Minkowski metric at $n = 1$. Tend to form hyperrecutangular clusters.	Fuzzy ART [57]
Sup distance	$D_{ij} = \max_{1 \le l \le d} \lvert x_{il} - x_{jl} \rvert$	Special case of Minkowski metric at $n \to \infty$.	Fuzzy c-means with sup norm [39].
Mahalanobis distance	$D_{ij} = (\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{S}^{-1} (\mathbf{x}_i - \mathbf{x}_j)$, where $\mathbf{S}$ is the within-group covariance matrix.	Invariant to any nonsingular linear transformation. $\mathbf{S}$ is calculated based on all objects. Tend to form hyperellipsoidal clusters. When features are not correlated, squared Mahalanobis distance is equivalent to squared Euclidean distance. May cause some computational burden.	Ellipsoidal ART [13], Hyperellipsoidal clustering algorithm [194].
Pearson correlation	$D_{ij} = (1 - r_{ij})/2$, where $r_{ij} = \frac{\sum_{l=1}^{d} (x_{il} - \overline{x_i})(x_{jl} - \overline{x_j})}{\sqrt{\sum_{l=1}^{d} (x_{il} - \overline{x_i})^2 \sum_{l=1}^{d} (x_{jl} - \overline{x_j})^2}}$	Not a metric. Derived from correlation coefficient. Unable to detect the magnitude of differences of two variables.	Widely used as the measure for analyzing gene expression data [80].
Point symmetry distance	$D_{ir} = \min_{\substack{j=1,\ldots,N \\ \text{and } j \ne i}} \frac{\lVert (\mathbf{x}_i - \mathbf{x}_r) + (\mathbf{x}_j - \mathbf{x}_r) \rVert}{\lVert (\mathbf{x}_i - \mathbf{x}_r) \rVert + \lVert (\mathbf{x}_j - \mathbf{x}_r) \rVert}$	Not a metric. Compute the distance between an object $\mathbf{x}_i$ and a reference point $\mathbf{x}_r$. D_{ir} is minimized when a symmetric pattern exists.	SBKM (Symmetry-based K-means) [264].
Cosine similarity	$S_{ij} = \cos \alpha = \frac{\mathbf{x}_i^T \mathbf{x}_j}{\lVert \mathbf{x}_i \rVert \lVert \mathbf{x}_j \rVert}$	Independent of vector length. Invariant to rotation, but not to linear transformations.	The most commonly used measure in document clustering [261].

tification via connectivity kernels (CLICK), cluster affinity search technique (CAST) ...

- *F*. Combinatorial Search Techniques-Based
  - Genetically guided algorithm (GGA), TS clustering, SA clustering ...
- *G*. Fuzzy
  - Fuzzy c-means (FCM), mountain method (MM), possibilistic c-means clustering algorithm (PCM), fuzzy c-shells (FCS) ...
- *H*. Neural Networks-Based
  - Learning vector quantization (LVQ), self-organizing feature map (SOFM), ART, simplified ART (SART), hyperellipsoidal clustering network (HEC), self-splitting competitive learning network (SPLL) ...
- *I*. Kernel-Based
  - Kernel K-means, support vector clustering (SVC) ...
- *J*. Sequential Data
  - Sequence Similarity
  - Indirect sequence clustering
  - Statistical sequence clustering
- *K*. Large-Scale Data Sets (See also Table II)
  - CLARA, CURE, CLARANS, BIRCH, DBSCAN, DENCLUE, WaveCluster, FC, ART ...
- *L*. Data visualization and High-dimensional Data
  - PCA, ICA, Projection pursuit, Isomap, LLE, CLIQUE, OptiGrid, ORCLUS ...
- *M*. How Many Clusters?

Applications in two benchmark data sets, the traveling salesman problem, and bioinformatics are illustrated in Section III. We conclude the paper in Section IV.

II. CLUSTERING ALGORITHMS

Different starting points and criteria usually lead to different taxonomies of clustering algorithms [33], [88], [124], [150], [152], [171]. A rough but widely agreed frame is to classify clustering techniques as hierarchical clustering and partitional clustering, based on the properties of clusters generated [88], [152]. Hierarchical clustering groups data objects with a sequence of partitions, either from singleton clusters to a cluster including all individuals or vice versa, while partitional clustering directly divides data objects into some prespecified number of clusters without the hierarchical structure. We follow this frame in surveying the clustering algorithms in the literature. Beginning with the discussion on proximity measure, which is the basis for most clustering algorithms, we focus on hierarchical clustering and classical partitional clustering algorithms in Section II-B–D. Starting from part E, we introduce and analyze clustering algorithms based on a wide variety of theories and techniques, including graph theory, combinatorial search techniques, fuzzy set theory, neural networks, and kernels techniques. Compared with graph theory and fuzzy set

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

TABLE II
COMPUTATIONAL COMPLEXITY OF CLUSTERING ALGORITHMS

Cluster algorithm	Complexity	Capability of tackling high dimensional data
K-means	$O(NKd)$ (time) $O(N+K)$ (space)	No
Fuzzy c-means	Near $O(N)$	No
Hierarchical clustering*	$O(N^2)$ (time) $O(N^2)$ (space)	No
CLARA	$O(K(40+K)^2+K(N-K))^+$ (time)	No
CLARANS	Quadratic in total performance	No
BIRCH	$O(N)$ (time)	No
DBSCAN	$O(N \log N)$ (time)	No
CURE	$O(N^2_{sample} \log N_{sample})$ (time) $O(N_{sample})$ (space)	Yes
WaveCluster	$O(N)$ (time)	No
DENCLUE	$O(N \log N)$ (time)	Yes
FC	$O(N)$ (time)	Yes
CLIQUE	Linear with the number of objects, Quadratic with the number of dimensions	Yes
OptiGrid	Between $O(Nd)$ and $O(Nd \log N)$	Yes
ORCLUS	$O(K_0^3+K_0Nd+K_0^2d^3)$ (time) $O(K_0d^2)$ (space)	Yes

*Include single-linkage, complete-linkage, average-linkage, etc.
+Based on the heuristic for drawing a sample from the entire data set in [161].

theory, which had already been widely used in cluster analysis before the 1980s, the other techniques have been finding their applications in clustering just in the recent decades. In spite of the short history, much progress has been achieved. Note that these techniques can be used for both hierarchical and partitional clustering. Considering the more frequent requirement of tackling sequential data sets, large-scale, and high-dimensional data sets in many current applications, we review clustering algorithms for them in the following three parts. We focus particular attention on clustering algorithms applied in bioinformatics. We offer more detailed discussion on how to identify appropriate number of clusters, which is particularly important in cluster validity, in the last part of the section.

A. Distance and Similarity Measures

It is natural to ask what kind of standards we should use to determine the closeness, or how to measure the distance (dissimilarity) or similarity between a pair of objects, an object and a cluster, or a pair of clusters. In the next section on hierarchical clustering, we will illustrate linkage metrics for measuring proximity between clusters. Usually, a prototype is used to represent a cluster so that it can be further processed like other objects. Here, we focus on reviewing measure approaches between individuals due to the previous consideration.

A data object is described by a set of features, usually represented as a multidimensional vector. The features can be quantitative or qualitative, continuous or binary, nominal or ordinal, which determine the corresponding measure mechanisms.

A distance or dissimilarity function on a data set $\mathbf{X}$ is defined to satisfy the following conditions.

1) Symmetry. $D(\mathbf{x}_i, \mathbf{x}_j) = D(\mathbf{x}_j, \mathbf{x}_i)$;
2) Positivity. $D(\mathbf{x}_i, \mathbf{x}_j) \geq 0$ for all $\mathbf{x}_i$ and $\mathbf{x}_j$.

If conditions

3) Triangle inequality.

$$D(\mathbf{x}_i, \mathbf{x}_j) \leq D(\mathbf{x}_i, \mathbf{x}_k) + D(\mathbf{x}_k, \mathbf{x}_j) \text{ for all } \mathbf{x}_i, \mathbf{x}_j, \text{ and } \mathbf{x}_k$$

and (4) Reflexivity. $D(\mathbf{x}_i, \mathbf{x}_j) = 0$ iff $\mathbf{x}_i = \mathbf{x}_j$ also hold, it is called a metric.

Likewise, a similarity function is defined to satisfy the conditions in the following.

1) Symmetry. $S(\mathbf{x}_i, \mathbf{x}_j) = S(\mathbf{x}_j, \mathbf{x}_i)$;
2) Positivity. $0 \leq S(\mathbf{x}_i, \mathbf{x}_j) \leq 1$, for all $\mathbf{x}_i$ and $\mathbf{x}_j$.

If it also satisfies conditions

3)

$$S(\mathbf{x}_i, \mathbf{x}_j)S(\mathbf{x}_j, \mathbf{x}_k) \leq [S(\mathbf{x}_i, \mathbf{x}_j) + S(\mathbf{x}_j, \mathbf{x}_k)]S(\mathbf{x}_i, \mathbf{x}_k) \quad \text{for all } \mathbf{x}_i, \mathbf{x}_j \text{ and } \mathbf{x}_k$$

and (4) $S(\mathbf{x}_i, \mathbf{x}_j) = 1$ iff $\mathbf{x}_i = \mathbf{x}_j$, it is called a similarity metric.

For a data set with N input patterns, we can define an $N \times N$ symmetric matrix, called proximity matrix, whose (i, j)th element represents the similarity or dissimilarity measure for the ith and jth patterns $(i, j = 1, \ldots, N)$.

Typically, distance functions are used to measure continuous features, while similarity measures are more important for qualitative variables. We summarize some typical measures for continuous features in Table I. The selection of different measures is problem dependent. For binary features, a similarity measure is commonly used (dissimilarity measures can be obtained by simply using $D_{ij} = 1 - S_{ij}$). Suppose we use two binary subscripts to count features in two objects. n_{00} and n_{11} represent the number of simultaneous absence or presence of features in two objects, and n_{01} and n_{10} count the features present only in one object. Then two types of commonly used similarity measures for data points $\mathbf{x}_i$ and $\mathbf{x}_j$ are illustrated in the following.

- $$S_{ij} = \frac{n_{11} + n_{00}}{n_{11} + n_{00} + w(n_{10} + n_{01})}$$
  $w = 1$, simple matching coefficient
  $w = 2$, Rogers and Tanimoto measure.
  $w = 1/2$, Gower and Legendre measure.

  These measures compute the match between two objects directly. Unmatched pairs are weighted based on their contribution to the similarity.

- $$S_{ij} = \frac{n_{11}}{n_{11} + w(n_{10} + n_{01})}$$
  $w = 1$, Jaccard coefficient
  $w = 2$, Sokal and Sneath measure.
  $w = 1/2$, Gower and Legendre measure.

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

These measures focus on the co-occurrence features while ignoring the effect of co-absence.

For nominal features that have more than two states, a simple strategy needs to map them into new binary features [161], while a more effective method utilizes the matching criterion

$$S_{ij} = \frac{1}{d}\sum_{l=1}^{d} S_{ijl}$$

where

$$S_{ijl} = \begin{cases} 0, & \text{if } i \text{ and } j \text{ do not match} \\ 1, & \text{if } i \text{ and } j \text{ match} \end{cases}$$

[88]. Ordinal features order multiple states according to some standard and can be compared by using continuous dissimilarity measures discussed in [161]. Edit distance for alphabetic sequences is discussed in Section II-J. More discussion on sequences and strings comparisons can be found in [120] and [236].

Generally, for objects consisting of mixed variables, we can map all these variables into the interval (0, 1) and use measures like the Euclidean metric. Alternatively, we can transform them into binary variables and use binary similarity functions. The drawback of these methods is the information loss. A more powerful method was described by Gower in the form of $S_{ij} = (\sum_{l=1}^{d} \eta_{ijl} S_{ijl})/(\sum_{l=1}^{d} \eta_{ijl})$, where S_{ijl} indicates the similarity for the lth feature and η_{ijl} is a 0–1 coefficient based on whether the measure of the two objects is missing [88], [112].

B. Hierarchical Clustering

Hierarchical clustering (HC) algorithms organize data into a hierarchical structure according to the proximity matrix. 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, like the data from evolutionary research on different species of organizms. HC algorithms are mainly classified as agglomerative methods and divisive methods. Agglomerative clustering starts with N clusters and each of them includes exactly one object. A series of merge operations are then followed out that finally lead all objects to the same group. Divisive clustering proceeds in an opposite way. In the beginning, the entire data set belongs to a cluster and a procedure successively divides it until all clusters are singleton clusters. For a cluster with N objects, there are $2^{N-1} - 1$ possible two-subset divisions, which is very expensive in computation [88]. Therefore, divisive clustering is not commonly used in practice. We focus on the agglomerative clustering in the following discussion and some of divisive clustering applications for binary data can be found in [88]. Two divisive clustering algorithms, named MONA and DIANA, are described in [161].

The general agglomerative clustering can be summarized by the following procedure.

1) Start with N singleton clusters. Calculate the proximity matrix for the N clusters.
2) Search the minimal distance

$$D(C_i, C_j) = \min_{\substack{1 \le m, l \le N \\ m \ne l}} D(C_m, C_l)$$

where $D(*,*)$ is the distance function discussed before, in the proximity matrix, and combine cluster C_i and C_j to form a new cluster.
3) Update the proximity matrix by computing the distances between the new cluster and the other clusters.
4) Repeat steps 2)–3) until all objects are in the same cluster.

Based on the different definitions for distance between two clusters, there are many agglomerative clustering algorithms. The simplest and most popular methods include single linkage [256] and complete linkage technique [258]. For the single linkage method, the distance between two clusters is determined by the two closest objects in different clusters, so it is also called nearest neighbor method. On the contrary, the complete linkage method uses the farthest distance of a pair of objects to define inter-cluster distance. Both the single linkage and the complete linkage method can be generalized by the recurrence formula proposed by Lance and Williams [178] as

$$\begin{aligned} D(C_l, (C_i, C_j)) = {} & \alpha_i D(C_l, C_i) + \alpha_j D(C_l, C_j) \\ & + \beta D(C_i, C_j) + \gamma |D(C_l, C_i) - D(C_l, C_j)| \end{aligned}$$

where $D(*,*)$ is the distance function and $\alpha_i, \alpha_j, \beta$, and γ are coefficients that take values dependent on the scheme used.

The formula describes the distance between a cluster l and a new cluster formed by the merge of two clusters i and j. Note that when $\alpha_i = \alpha_j = 1/2, \beta = 0$, and $\gamma = -1/2$, the formula becomes

$$D(C_l, (C_i, C_j)) = \min(D(C_l, C_i), D(C_l, C_j))$$

which corresponds to the single linkage method. When $\alpha_i = \alpha_j = \gamma = 1/2$ and $\beta = 0$, the formula is

$$D(C_l, (C_i, C_j)) = \max(D(C_l, C_i), D(C_l, C_j))$$

which corresponds to the complete linkage method.

Several more complicated agglomerative clustering algorithms, including group average linkage, median linkage, centroid linkage, and Ward's method, can also be constructed by selecting appropriate coefficients in the formula. A detailed table describing the coefficient values for different algorithms is offered in [150] and [210]. Single linkage, complete linkage and average linkage consider all points of a pair of clusters, when calculating their inter-cluster distance, and are also called graph methods. The others are called geometric methods since they use geometric centers to represent clusters and determine their distances. Remarks on important features and properties of these methods are summarized in [88]. More inter-cluster

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

distance measures, especially the mean-based ones, were introduced by Yager, with further discussion on their possible effect to control the hierarchical clustering process [289].

The common criticism for classical HC algorithms is that they lack robustness and are, hence, sensitive to noise and outliers. Once an object is assigned to a cluster, it will not be considered again, which means that HC algorithms are not capable of correcting possible previous misclassification. 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 sets. Other disadvantages of HC include the tendency to form spherical shapes and reversal phenomenon, in which the normal hierarchical structure is distorted.

In recent years, with the requirement for handling large-scale data sets in data mining and other fields, many new HC techniques have appeared and greatly improved the clustering performance. Typical examples include CURE [116], ROCK [117], Chameleon [159], and BIRCH [295].

The main motivations of BIRCH lie in two aspects, the ability to deal with large data sets and the robustness to outliers [295]. In order to achieve these goals, a new data structure, clustering feature (CF) tree, is designed to store the summaries of the original data. The CF tree is a height-balanced tree, with each internal vertex composed of entries defined as $[\mathbf{CF}_i, \text{child}_i], i = 1, \ldots, B$, where $\mathbf{CF}_i$ is a representation of the cluster i and is defined as $\mathbf{CF}_i = (N_i, \mathbf{LS}, \text{SS})$, where N_i is the number of data objects in the cluster, $\mathbf{LS}$ is the linear sum of the objects, and SS is the squared sum of the objects, child_i is a pointer to the ith child node, and B is a threshold parameter that determines the maximum number of entries in the vertex, and each leaf composed of entries in the form of $[\mathbf{CF}_i], i = 1, \ldots, L$, where L is the threshold parameter that controls the maximum number of entries in the leaf. Moreover, the leaves must follow the restriction that the diameter $((\sum_{l=1}^{N_i} \sum_{m=1}^{N_i} (\mathbf{x}_l - \mathbf{x}_m)^2 / N_i(N_i - 1))^{1/2})$ of each entry in the leaf is less than a threshold T. The CF tree structure captures the important clustering information of the original data while reducing the required storage. Outliers are eliminated from the summaries by identifying the objects sparsely distributed in the feature space. After the CF tree is built, an agglomerative HC is applied to the set of summaries to perform global clustering. An additional step may be performed to refine the clusters. BIRCH can achieve a computational complexity of $O(N)$.

Noticing the restriction of centroid-based HC, which is unable to identify arbitrary cluster shapes, Guha, Rastogi, and Shim developed a HC algorithm, called CURE, to explore more sophisticated cluster shapes [116]. The crucial feature of CURE lies in the usage of a set of well-scattered points to represent each cluster, which makes it possible to find rich cluster shapes other than hyperspheres and avoids both the chaining effect [88] of the minimum linkage method and the tendency to favor clusters with similar sizes of centroid. These representative points are further shrunk toward the cluster centroid according to an adjustable parameter in order to weaken the effects of outliers. CURE utilizes random sample (and partition) strategy to reduce computational complexity. Guha *et al.* also proposed another agglomerative HC algorithm, ROCK, to group data with qualitative attributes [117]. They used a novel measure "link" to describe the relation between a pair of objects and their common neighbors. Like CURE, a random sample strategy is used to handle large data sets. Chameleon is constructed from graph theory and will be discussed in Section II-E.

Relative hierarchical clustering (RHC) is another exploration that considers both the internal distance (distance between a pair of clusters which may be merged to yield a new cluster) and the external distance (distance from the two clusters to the rest), and uses the ratio of them to decide the proximities [203]. Leung *et al.* showed an interesting hierarchical clustering based on scale-space theory [180]. They interpreted clustering using a blurring process, in which each datum is regarded as a light point in an image, and a cluster is represented as a blob. Li and Biswas extended agglomerative HC to deal with both numeric and nominal data. The proposed algorithm, called similarity-based agglomerative clustering (SBAC), employs a mixed data measure scheme that pays extra attention to less common matches of feature values [183]. Parallel techniques for HC are discussed in [69] and [217], respectively.

C. Squared Error—Based Clustering (Vector Quantization)

In contrast to hierarchical clustering, which yields a successive level of clusters by iterative fusions or divisions, partitional clustering assigns a set of objects into K clusters with no hierarchical structure. In principle, the optimal partition, based on some specific criterion, can be found by enumerating all possibilities. But this brute force method is infeasible in practice, due to the expensive computation [189]. Even for a small-scale clustering problem (organizing 30 objects into 3 groups), the number of possible partitions is 2×10^{14}. Therefore, heuristic algorithms have been developed in order to seek approximate solutions.

One of the important factors in partitional clustering is the criterion function [124]. The sum of squared error function is one of the most widely used criteria. Suppose we have a set of objects $\mathbf{x}_j \in \Re^d, j = 1, \ldots, N$, and we want to organize them into K subsets $C = \{C_1, \ldots, C_K\}$. The squared error criterion then is defined as

$$J(\mathbf{\Gamma}, \mathbf{M}) = \sum_{i=1}^{K} \sum_{j=1}^{N} \gamma_{ij} \|\mathbf{x}_j - \mathbf{m}_i\|^2$$

where

$\mathbf{\Gamma} =$ a partition matrix;
$\{\gamma_{ij}\}$

$$\gamma_{ij} = \begin{cases} 1 & \text{if } \mathbf{x}_j \in \text{cluster } i \\ 0 & \text{otherwise} \end{cases} \quad \text{with } \sum_{i=1}^{K} \gamma_{ij} = 1 \forall j;$$

$\mathbf{M} =$ cluster prototype or centroid (means) matrix;
$[\mathbf{m}_1, \ldots, \mathbf{m}_K]$
$\mathbf{m}_i =$ sample mean for the ith cluster;
$(1/N_i) \sum_{j=1}^{N} \gamma_{ij} \mathbf{x}_j$
N_i number of objects in the ith cluster.

Note the relation between the sum of squared error criterion and the scatter matrices defined in multiclass discriminant analysis [75],

$$S_T = S_W + S_B$$

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

where

S_T total scatter matrix;

$S_W =$within-class scatter matrix;

$\sum_{i=1}^{K}\sum_{j=1}^{N}\gamma_{ij}(\mathbf{x}_j-\mathbf{m}_i)(\mathbf{x}_j-\mathbf{m}_i)^T$

$S_B =$ between-class scatter matrix; and

$\sum_{i=1}^{K}N_i(\mathbf{m}_i-\mathbf{m})(\mathbf{m}_i-\mathbf{m})^T$

$\mathbf{m} =$ mean vector for the whole data set.

$(1/N)\sum_{i=1}^{K}N_i\mathbf{m}_i$

It is not difficult to see that the criterion based on the trace of S_W is the same as the sum of squared error criterion. To minimize the squared error criterion is equivalent to minimizing the trace of S_W or maximizing the trace of S_B. We can obtain a rich class of criterion functions based on the characteristics of S_W and S_B [75].

The K-means algorithm is the best-known squared error-based clustering algorithm [94], [191].

1) Initialize a K-partition randomly or based on some prior knowledge. Calculate the cluster prototype matrix $\mathbf{M}=[\mathbf{m}_1,\ldots,\mathbf{m}_K]$.
2) Assign each object in the data set to the nearest cluster C_w, i.e.

$$\mathbf{x}_j\in C_w,\quad \text{if } \|\mathbf{x}_j-\mathbf{m}_w\|<\|\mathbf{x}_j-\mathbf{m}_i\| \quad \text{for } j=1,\ldots,N,\quad i\neq w,\quad \text{and}\quad i=1,\ldots,K.$$

3) Recalculate the cluster prototype matrix based on the current partition.
4) Repeat steps 2)–3) until there is no change for each cluster.

The K-means algorithm is very simple and can be easily implemented in solving many practical problems. It can work very well for compact and hyperspherical clusters. The time complexity of K-means is $O(NKd)$. Since K and d are usually much less than N, K-means can be used to cluster large data sets. Parallel techniques for K-means are developed that can largely accelerate the algorithm [262]. The drawbacks of K-means are also well studied, and as a result, many variants of K-means have appeared in order to overcome these obstacles. We summarize some of the major disadvantages with the proposed improvement in the following.

1) There is no efficient and universal method for identifying the initial partitions and the number of clusters K. The convergence centroids vary with different initial points. A general strategy for the problem is to run the algorithm many times with random initial partitions. Peña, Lozano, and Larrañaga compared the random method with other three classical initial partition methods by Forgy [94], Kaufman [161], and MacQueen [191], based on the effectiveness, robustness, and convergence speed criteria [227]. According to their experimental results, the random and Kaufman's method work much better than the other two under the first two criteria and by further considering the convergence speed, they recommended Kaufman's method. Bradley and Fayyad presented a refinement algorithm that first utilizes K-means M times to M random subsets from the original data [43]. The set formed from the union of the solution (centroids of the K clusters) of the M subsets is clustered M times again, setting each subset solution as the initial guess. The starting points for the whole data are obtained by choosing the solution with minimal sum of squared distances. Likas, Vlassis, and Verbeek proposed a global K-means algorithm consisting of a series of K-means clustering procedures with the number of clusters varying from 1 to K [186]. After finding the centroid for only one cluster existing, at each $k, k=2,\ldots,K$, the previous $k-1$ centroids are fixed and the new centroid is selected by examining all data points. The authors claimed that the algorithm is independent of the initial partitions and provided accelerating strategies. But the problem on computational complexity exists, due to the requirement for executing K-means N times for each value of k.

   An interesting technique, called ISODATA, developed by Ball and Hall [21], deals with the estimation of K. ISODATA can dynamically adjust the number of clusters by merging and splitting clusters according to some predefined thresholds (in this sense, the problem of identifying the initial number of clusters becomes that of parameter (threshold) tweaking). The new K is used as the expected number of clusters for the next iteration.
2) The iteratively optimal procedure of K-means cannot guarantee convergence to a global optimum. The stochastic optimal techniques, like simulated annealing (SA) and genetic algorithms (also see part II.F), can find the global optimum with the price of expensive computation. Krishna and Murty designed new operators in their hybrid scheme, GKA, in order to achieve global search and fast convergence [173]. The defined biased mutation operator is based on the Euclidean distance between an object and the centroids and aims to avoid getting stuck in a local optimum. Another operator, the K-means operator (KMO), replaces the computationally expensive crossover operators and alleviates the complexities coming with them. An adaptive learning rate strategy for the online mode K-means is illustrated in [63]. The learning rate is exclusively dependent on the within-group variations and can be adjusted without involving any user activities. The proposed enhanced LBG (ELBG) algorithm adopts a roulette mechanism typical of genetic algorithms to become near-optimal and therefore, is not sensitive to initialization [222].
3) K-means is sensitive to outliers and noise. Even if an object is quite far away from the cluster centroid, it is still forced into a cluster and, thus, distorts the cluster shapes. ISODATA [21] and PAM [161] both consider the effect of outliers in clustering procedures. ISODATA gets rid of clusters with few objects. The splitting operation of ISODATA eliminates the possibility of elongated clusters typical of K-means. PAM utilizes real data points (medoids) as the cluster prototypes and avoids the effect of outliers. Based on the same consideration, a K-medoids algorithm is presented in [87]

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

by searching the discrete 1-medians as the cluster centroids.

4) The definition of "means" limits the application only to numerical variables. The K-medoids algorithm mentioned previously is a natural choice, when the computation of means is unavailable, since the medoids do not need any computation and always exist [161]. Huang [142] and Gupta *et al.* [118] defined different dissimilarity measures to extend K-means to categorical variables. For Huang's method, the clustering goal is to minimize the cost function $J(\mathbf{\Gamma}, \mathbf{Q}) = \sum_{i=1}^{K} \sum_{j=1}^{N} \gamma_{ij} D(\mathbf{x}_j, \mathbf{Q}_i)$, where

$$D(\mathbf{x}_i, \mathbf{x}_j) = \sum_{l=1}^{d} \delta(x_{il}, x_{jl})$$

and

$$\delta(x_{il}, x_{jl}) = \begin{cases} 1 & x_{il} \neq x_{jl} \\ 0 & x_{il} = x_{jl} \end{cases}$$

with a set of d-dimensional vectors $\mathbf{Q} = \{\mathbf{Q}_1, \ldots, \mathbf{Q}_K\}$, where $\mathbf{Q}_j = (Q_{j1}, \ldots, Q_{jd})$. Each vector $\mathbf{Q}_j$ is known as a mode and is defined to minimize the sum of distances $\sum_{i=1}^{N} D(\mathbf{x}_i, \mathbf{Q}_j)$. The proposed K-modes algorithm operates in a similar way as K-means.

Several recent advances on K-means and other squared-error based clustering algorithms with their applications can be found in [125], [155], [222], [223], [264], and [277].

D. Mixture Densities-Based Clustering (pdf Estimation via Mixture Densities)

In the probabilistic view, data objects are assumed to be generated according to several probability distributions. Data points in different clusters were generated by different probability distributions. They can be derived from different types of density functions (e.g., multivariate Gaussian or t-distribution), or the same families, but with different parameters. If the distributions are known, finding the clusters of a given data set is equivalent to estimating the parameters of several underlying models. Suppose the prior probability (also known as mixing probability) $P(C_i)$ for cluster $C_i, i = 1, \ldots, K$ (here, K is assumed to be known and methods for estimating K are discussed in Section II-M) and the conditional probability density $p(\mathbf{x} \,|\, C_i, \boldsymbol{\theta}_i)$ (also known as component density), where $\boldsymbol{\theta}_i$ is the unknown parameter vector, are known. Then, the mixture probability density for the whole data set is expressed as

$$p(\mathbf{x} \,|\, \boldsymbol{\theta}) = \sum_{i=1}^{K} p(\mathbf{x} \,|\, C_i, \boldsymbol{\theta}_i) P(C_i)$$

where $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_K)$, and $\sum_{i=1}^{K} P(C_i) = 1$. As long as the parameter vector $\boldsymbol{\theta}$ is decided, the posterior probability for assigning a data point to a cluster can be easily calculated with Bayes's theorem. Here, the mixtures can be constructed with any types of components, but more commonly, multivariate Gaussian densities are used due to its complete theory and analytical tractability [88], [297].

Maximum likelihood (ML) estimation is an important statistical approach for parameter estimation [75] and it considers the best estimate as the one that maximizes the probability of generating all the observations, which is given by the joint density function

$$p(\{\mathbf{x}_1, \ldots, \mathbf{x}_N\} \,|\, \boldsymbol{\theta}) = \prod_{j=1}^{N} p(\mathbf{x}_j \,|\, \boldsymbol{\theta})$$

or, in a logarithm form

$$l(\boldsymbol{\theta}) = \sum_{j=1}^{N} \ln p(\mathbf{x}_j \,|\, \boldsymbol{\theta}).$$

The best estimate can be achieved by solving the log-likelihood equations $(\partial l(\boldsymbol{\theta}))/(\partial \boldsymbol{\theta}_i) = 0$.

Unfortunately, since the solutions of the likelihood equations cannot be obtained analytically in most circumstances [90], [197], iteratively suboptimal approaches are required to approximate the ML estimates. Among these methods, the expectation-maximization (EM) algorithm is the most popular [196]. EM regards the data set as incomplete and divides each data point $\mathbf{x}_j$ into two parts $\mathbf{x}_j = \{\mathbf{x}_j^g, \mathbf{x}_j^m\}$, where $\mathbf{x}_j^g$ represents the observable features and $\mathbf{x}_j^m = (x_{j1}^m, \ldots, x_{jK}^m)$ is the missing data, where x_{ji}^m chooses value 1 or 0 according to whether $\mathbf{x}_j$ belongs to the component i or not. Thus, the complete data log-likelihood is

$$l(\boldsymbol{\theta}) = \sum_{j=1}^{N} \sum_{i=1}^{K} x_{ji}^m \log\left[P(C_i) p\left(\mathbf{x}_j^g \,\middle|\, \boldsymbol{\theta}_i\right)\right].$$

The standard EM algorithm generates a series of parameter estimates $\{\boldsymbol{\theta}^0, \boldsymbol{\theta}^1, \ldots, \boldsymbol{\theta}^T\}$, where T represents the reaching of the convergence criterion, through the following steps:

1) initialize $\boldsymbol{\theta}^0$ and set $t = 0$;
2) e-step: Compute the expectation of the complete data log-likelihood

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^t) = E[\log p(\mathbf{x}^g, \mathbf{x}^m \,|\, \boldsymbol{\theta}) \,|\, \mathbf{x}^g, \boldsymbol{\theta}^t];$$

3) m-step: Select a new parameter estimate that maximizes the Q-function, $\boldsymbol{\theta}^{t+1} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^t)$;
4) Increase $t = t + 1$; repeat steps 2)–3) until the convergence condition is satisfied.

The major disadvantages for EM algorithm are the sensitivity to the selection of initial parameters, the effect of a singular covariance matrix, the possibility of convergence to a local optimum, and the slow convergence rate [96], [196]. Variants of EM for addressing these problems are discussed in [90] and [196].

A valuable theoretical note is the relation between the EM algorithm and the K-means algorithm. Celeux and Govaert proved that classification EM (CEM) algorithm under a spherical Gaussian mixture is equivalent to the K-means algorithm [58].

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

Fraley and Raftery described a comprehensive mixture-model based clustering scheme [96], which was implemented as a software package, known as MCLUST [95]. In this case, the component density is multivariate Gaussian, with a mean vector $\boldsymbol{\mu}$ and a covariance matrix $\boldsymbol{\Sigma}$ as the parameters to be estimated. The covariance matrix for each component can further be parameterized by virtue of eigenvalue decomposition, represented as $\boldsymbol{\Sigma} = \lambda \boldsymbol{\Pi} \mathbf{A} \boldsymbol{\Pi}^T$, where λ is a scalar, $\boldsymbol{\Pi}$ is the orthogonal matrix of eigenvectors, and $\mathbf{A}$ is the diagonal matrix based on the eigenvalues of $\boldsymbol{\Sigma}$ [96]. These three elements determine the geometric properties of each component. After the maximum number of clusters and the candidate models are specified, an agglomerative hierarchical clustering was used to ignite the EM algorithm by forming an initial partition, which includes at most the maximum number of clusters, for each model. The optimal clustering result is achieved by checking the Bayesian information criterion (BIC) value discussed in Section II-M. GMDD is also based on multivariate Gaussian densities and is designed as a recursive algorithm that sequentially estimates each component [297]. GMDD views data points that are not generated from a distribution as noise and utilizes an enhanced model-fitting estimator to construct each component from the contaminated model. AutoClass considers more families of probability distributions (e.g., Poisson and Bernoulli) for different data types [59]. A Bayesian approach is used in AutoClass to find out the optimal partition of the given data based on the prior probabilities. Its parallel realization is described in [228]. Other important algorithms and programs include Multimix [147], EM based mixture program (EMMIX) [198], and Snob [278].

E. Graph Theory-Based Clustering

The concepts and properties of graph theory [126] make it very convenient to describe clustering problems by means of graphs. Nodes V of a weighted graph G correspond to data points in the pattern space and edges E reflect the proximities between each pair of data points. If the dissimilarity matrix is defined as

$$D_{ij} = \begin{cases} 1 & \text{if } D(\mathbf{x}_i, \mathbf{x}_j) < d_0 \\ 0 & \text{otherwise} \end{cases}$$

where d_0 is a threshold value, the graph is simplified to an unweighted threshold graph. Both the single linkage HC and the complete linkage HC can be described on the basis of the threshold graph. Single linkage clustering is equivalent to seeking maximally connected subgraphs (components) while complete linkage clustering corresponds to finding maximally complete subgraphs (cliques) [150]. Jain and Dubes illustrated and discussed more applications of graph theory (e.g., Hubert's algorithm and Johnson's algorithm) for hierarchical clustering in [150]. Chameleon [159] is a newly developed agglomerative HC algorithm based on the k-nearest-neighbor graph, in which an edge is eliminated if both vertices are not within the k closest points related to each other. At the first step, Chameleon divides the connectivity graph into a set of subclusters with the minimal edge cut. Each subgraph should contain enough nodes in order for effective similarity computation. By combining both the relative interconnectivity and relative closeness, which make Chameleon flexible enough to explore the characteristics of potential clusters, Chameleon merges these small subsets and, thus, comes up with the ultimate clustering solutions. Here, the relative interconnectivity (or closeness) is obtained by normalizing the sum of weights (or average weight) of the edges connecting the two clusters over the internal connectivity (or closeness) of the clusters. DTG is another important graph representation for HC analysis. Cherng and Lo constructed a hypergraph (each edge is allowed to connect more than two vertices) from the DTG and used a two-phase algorithm that is similar to Chameleon to find clusters [61]. Another DTG-based application, known as AMOEBA algorithm, is presented in [86].

Graph theory can also be used for nonhierarchical clusters. Zahn's clustering algorithm seeks connected components as clusters by detecting and discarding inconsistent edges in the minimum spanning tree [150]. Hartuv and Shamir treated clusters as HCS, where "highly connected" means the connectivity (the minimum number of edges needed to disconnect a graph) of the subgraph is at least half as great as the number of the vertices [128]. A minimum cut (mincut) procedure, which aims to separate a graph with a minimum number of edges, is used to find these HCSs recursively. Another algorithm, called CLICK, is based on the calculation of the minimum weight cut to form clusters [247]. Here, the graph is weighted and the edge weights are assigned a new interpretation, by combining probability and graph theory. The edge weight between node i and j is defined as shown in

$$e_{ij} = \log \frac{\Pr ob(i, j \text{ belong to the same cluster} \mid S_{ij})}{\Pr ob(i, j \text{ does not belong to the same cluster} \mid S_{ij})}$$

where S_{ij} represents the similarity between the two nodes. CLICK further assumes that the similarity values within clusters and between clusters follow Gaussian distributions with different means and variances, respectively. Therefore, the previous equation can be rewritten by using Bayes' theorem as

$$e_{ij} = \log \frac{p_0 \sigma_B}{(1-p_0)\sigma_W} + \frac{(S_{ij} - \mu_B)^2}{2\sigma_B^2} - \frac{(S_{ij} - \mu_W)^2}{2\sigma_W^2}$$

where p_0 is the prior probability that two objects belong to the same cluster and $\mu_B, \sigma_B^2, \mu_W, \sigma_W^2$ are the means and variances for between-cluster similarities and within-clusters similarities, respectively. These parameters can be estimated either from prior knowledge, or by using parameter estimation methods [75]. CLICK recursively checks the current subgraph, and generates a kernel list, which consists of the components satisfying some criterion function. Subgraphs that include only one node are regarded as singletons, and are separated for further manipulation. Using the kernels as the basic clusters, CLICK carries out a series of singleton adoptions and cluster merge to generate the resulting clusters. Additional heuristics are provided to accelerate the algorithm performance.

Similarly, CAST considers a probabilistic model in designing a graph theory-based clustering algorithm [29]. Clusters are modeled as corrupted clique graphs, which, in ideal conditions, are regarded as a set of disjoint cliques. The effect of noise is incorporated by adding or removing edges from the ideal

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

model, with a probability α. Proofs were given for recovering the uncorrupted graph with a high probability. CAST is the heuristic implementation of the original theoretical version. CAST creates clusters sequentially, and each cluster begins with a random and unassigned data point. The relation between a data point i and a cluster C_0 being built is determined by the affinity, defined as $a(i) = \sum_{j \in C_0} S_{ij}$, and the affinity threshold parameter t. When $a(i) \geq t|C_0|$, it means that the data point is highly related to the cluster and vice versa. CAST alternately adds high affinity data points or deletes low affinity data points from the cluster until no more changes occur.

F. Combinatorial Search Techniques-Based Clustering

The basic object of search techniques is to find the global or approximate global optimum for combinatorial optimization problems, which usually have NP-hard complexity and need to search an exponentially large solution space. Clustering can be regarded as a category of optimization problems. Given a set of data points $\mathbf{x}_j \in \Re^d, j = 1, \ldots, N$, clustering algorithms aim to organize them into K subsets $\{C_1, \ldots, C_K\}$ that optimize some criterion function. The possible partition for N points into K clusters is given by the formula [189]

$$P(N, K) = \frac{1}{K!} \sum_{m=1}^{K} (-1)^{K-m} C_K^m m^N.$$

As shown before, even for small N and K, the computational complexity is extremely expensive, not to mention the large-scale clustering problems frequently encountered in recent decades. Simple local search techniques, like hill-climbing algorithms, are utilized to find the partitions, but they are easily stuck in local minima and therefore cannot guarantee optimality. More complex search methods (e.g., evolutionary algorithms (EAs) [93], SA [165], and Tabu search (TS) [108] are known as stochastic optimization methods, while deterministic annealing (DA) [139], [234] is the most typical deterministic search technique) can explore the solution space more flexibly and efficiently.

Inspired by the natural evolution process, evolutionary computation, which consists of genetic algorithms (GAs), evolution strategies (ESs), evolutionary programming (EP), and genetic programming (GP), optimizes a population of structure by using a set of evolutionary operators [93]. An optimization function, called the fitness function, is the standard for evaluating the optimizing degree of the population, in which each individual has its corresponding fitness. Selection, recombination, and mutation are the most widely used evolutionary operators. The selection operator ensures the continuity of the population by favoring the best individuals in the next generation. The recombination and mutation operators support the diversity of the population by exerting perturbations on the individuals. Among many EAs, GAs [140] are the most popular approaches applied in cluster analysis. In GAs, each individual is usually encoded as a binary bit string, called a chromosome. After an initial population is generated according to some heuristic rules or just randomly, a series of operations, including selection, crossover and mutation, are iteratively applied to the population until the stop condition is satisfied.

Hall, Özyurt, and Bezdek proposed a GGA that can be regarded as a general scheme for center-based (hard or fuzzy) clustering problems [122]. Fitness functions are reformulated from the standard sum of squared error criterion function in order to adapt the change of the construction of the optimization problem (only the prototype matrix is needed)

$$J(\mathbf{M}) = \sum_{j=1}^{N} \min(D_{1j}, D_{2j}, \ldots, D_{Kj}) \quad \text{for hard clustering}$$

$$J(\mathbf{M}) = \sum_{j=1}^{N} \left(\sum_{i=1}^{K} D_{ij}^{1/(1-m)} \right)^{1-m} \quad \text{for fuzzy clustering}$$

where $D_{ij}, i = 1, \ldots, K, j = 1, \ldots, N$, is the distance between the ith cluster and the jth data object, and m is the fuzzification parameter.

GGA proceeds with the following steps.

1) Choose appropriate parameters for the algorithm. Initialize the population randomly with P individuals, each of which represents a $K \times d$ prototype matrix and is encoded as gray codes. Calculate the fitness value for each individual.
2) Use selection (tournament selection) operator to choose parental members for reproduction.
3) Use crossover (two-point crossover) and mutation (bitwise mutation) operator to generate offspring from the individuals chosen in step 2).
4) Determine the next generation by keeping the individuals with the highest fitness.
5) Repeat steps 2)–4) until the termination condition is satisfied.

Other GAs-based clustering applications have appeared based on a similar framework. They are different in the meaning of an individual in the population, encoding methods, fitness function definition, and evolutionary operators [67], [195], [273]. The algorithm CLUSTERING in [273] includes a heuristic scheme for estimating the appropriate number of clusters in the data. It also uses a nearest-neighbor algorithm to divide data into small subsets, before GAs-based clustering, in order to reduce the computational complexity. GAs are very useful for improving the performance of K-means algorithms. Babu and Murty used GAs to find good initial partitions [15]. Krishna and Murty combined GA with K-means and developed GKA algorithm that can find the global optimum [173]. As indicated in Section II-C, the algorithm ELBG uses the roulette mechanism to address the problems due to the bad initialization [222]. It is worthwhile to note that ELBG are equivalent to another algorithm, fully automatic clustering system (FACS) [223], in terms of quantization level detection. The difference lies in the input parameters employed (ELBG adopts the number of quantization levels, while FACS uses the desired distortion error). Except the previous applications, GAs can also be used for hierarchical clustering. Lozano and Larrañag discussed the properties of ultrametric distance [127] and reformulated the hierarchical clustering as an optimization

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

problem that tries to find the closest ultrametic distance for a given dissimilarity with Euclidean norm [190]. They suggested an order-based GA to solve the problem. Clustering algorithms based on ESs and EP are described and analyzed in [16] and [106], respectively.

TS is a combinatory search technique that uses the tabu list to guide the search process consisting of a sequence of moves. The tabu list stores part or all of previously selected moves according to the specified size. These moves are forbidden in the current search and are called tabu. In the TS clustering algorithm developed by Al-Sultan [9], a set of candidate solutions are generated from the current solution with some strategy. Each candidate solution represents the allocations of N data objects in K clusters. The candidate with the optimal cost function is selected as the current solution and appended to the tabu list, if it is not already in the tabu list or meets the aspiration criterion, which can overrule the tabu restriction. Otherwise, the remaining candidates are evaluated in the order of their cost function values, until all these conditions are satisfied. When all the candidates are tabu, a new set of candidate solutions are created followed by the same search process. The search process proceeds until the maximum number of iterations is reached. Sung and Jin's method includes more elaborate search processes with the packing and releasing procedures [266]. They also used a secondary tabu list to keep the search from trapping into the potential cycles. A fuzzy version of TS clustering can be found in [72].

SA is also a sequential and global search technique and is motivated by the annealing process in metallurgy [165]. SA allows the search process to accept a worse solution with a certain probability. The probability is controlled by a parameter, known as temperature T and is usually expressed as $\exp(\Delta J/T)$, where ΔJ is the change of the energy (cost function). The temperature T goes through an annealing schedule from initial high to ultimate low values, which means that SA attempts to explore solution space more completely at high temperatures while favors the solutions that lead to lower energy at low temperatures. SA-based clustering was reported in [47] and [245]. The former illustrated an application of SA clustering to evaluate different clustering criteria and the latter investigated the effects of input parameters to the clustering performance.

Hybrid approaches that combine these search techniques are also proposed. A tabu list is used in a GA clustering algorithm to preserve the variety of the population and avoid repeating computation [243]. An application of SA for improving TS was reported in [64]. The algorithm further reduces the possible moves to local optima.

The main drawback that plagues the search techniques-based clustering algorithms is the parameter selection. More often than not, search techniques introduce more parameters than other methods (like K-means). There are no theoretic guidelines to select the appropriate and effective parameters. Hall *et al.* provided some methods for setting parameters in their GAs-based clustering framework [122], but most of these criteria are still obtained empirically. The same situation exists for TS and SA clustering [9], [245]. Another problem is the computational complexity paid for the convergence to global optima. High computational requirement limits their applications in large-scale data sets.

G. Fuzzy Clustering

Except for GGA, the clustering techniques we have discussed so far are referred to as hard or crisp clustering, which means that each object is assigned to only one cluster. For fuzzy clustering, this restriction is relaxed, and the object can belong to all of the clusters with a certain degree of membership [293]. 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.

FCM is one of the most popular fuzzy clustering algorithms [141]. FCM can be regarded as a generalization of ISODATA [76] and was realized by Bezdek [35]. FCM attempts to find a partition (c fuzzy clusters) for a set of data points $\mathbf{x}_j \in \Re^d, j = 1, \ldots, N$ while minimizing the cost function

$$J(\mathbf{U}, \mathbf{M}) = \sum_{i=1}^{c} \sum_{j=1}^{N} (u_{i,j})^m D_{ij}$$

where

$\mathbf{U}=[u_{i,j}]_{c \times N}$	is the fuzzy partition matrix and $u_{i,j} \in [0,1]$ is the membership coefficient of the jth object in the ith cluster;
$\mathbf{M}=[\mathbf{m}_1,\ldots,\mathbf{m}_c]$	is the cluster prototype (mean or center) matrix;
$m \in [1, \infty)$	is the fuzzification parameter and usually is set to 2 [129];
$D_{ij}=D(\mathbf{x}_j, \mathbf{m}_i)$	is the distance measure between $\mathbf{x}_j$ and $\mathbf{m}_i$.

We summarize the standard FCM as follows, in which the Euclidean or L_2 norm distance function is used.

1) Select appropriate values for m, c, and a small positive number ε. Initialize the prototype matrix $\mathbf{M}$ randomly. Set step variable $t = 0$.
2) Calculate (at $t = 0$) or update (at $t > 0$) the membership matrix $\mathbf{U}$ by

$$u_{ij}^{(t+1)} = 1 \Bigg/ \left(\sum_{l=1}^{c} (D_{lj}/D_{ij})^{1/(1-m)} \right) \quad \text{for } i = 1, \ldots, c \quad \text{and} \quad j = 1, \ldots, N.$$

3) Update the prototype matrix $\mathbf{M}$ by

$$\mathbf{m}_i^{(t+1)} = \left(\sum_{j=1}^{N} \left(u_{ij}^{(t+1)} \right)^m \mathbf{x}_j \right) \Bigg/ \left(\sum_{j=1}^{N} \left(u_{ij}^{(t+1)} \right)^m \right) \quad \text{for } i = 1, \ldots, c.$$

4) Repeat steps 2)–3) until $\|\mathbf{M}^{(t+1)} - \mathbf{M}^{(t)}\| < \varepsilon$.

Numerous FCM variants and other fuzzy clustering algorithms have appeared as a result of the intensive investigation on the distance measure functions, the effect of weighting exponent on fuzziness control, the optimization approaches for fuzzy partition, and improvements of the drawbacks of FCM [84], [141].

Like its hard counterpart, FCM also suffers from the presence of noise and outliers and the difficulty to identify the initial partitions. Yager and Filev proposed a MM in order to estimate the

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

centers of clusters [290]. Candidate centers consist of a set of vertices that are formed by building a grid on the pattern space. The mountain function for a vertex $\mathbf{v}_i$ is defined as

$$M(\mathbf{v}_i) = \sum_{j=1}^{N} e^{-\alpha D(\mathbf{x}_j, \mathbf{v}_i)}$$

where $D(\mathbf{x}_j, \mathbf{v}_i)$ is the distance between the jth data object and the ith node, and α is a positive constant. Therefore, the closer a data object is to a vertex, the more the data object contributes to the mountain function. The vertex $\mathbf{v}_{m_1}$ with the maximum value of mountain function $M_{\mathbf{v}_{m_1}}$ is selected as the first center. A procedure, called mountain destruction, is performed to get rid of the effects of the selected center. This is achieved by subtracting the mountain function value for each of the rest vertices with an amount dependent on the current maximum mountain function value and the distance between the vertex and the center. The process iterates until the ratio between the current maximum and $M_{\mathbf{v}_{m_1}}$ is below some threshold. The connection of MM with several other fuzzy clustering algorithms was further discussed in [71]. Gath and Geva described an initialization strategy of unsupervised tracking of cluster prototypes in their 2-layer clustering scheme, in which FCM and fuzzy ML estimation are effectively combined [102].

Kersten suggested that city block distance (or L_1 norm) could improve the robustness of FCM to outliers [163]. Furthermore, Hathaway, Bezdek, and Hu extended FCM to a more universal case by using Minkowski distance (or L_p norm, $p \geq 1$) and seminorm $(0 < p < 1)$ for the models that operate either directly on the data objects or indirectly on the dissimilarity measures [130]. According to their empirical results, the object data based models, with L_1 and L_2 norm, are recommended. They also pointed out the possible improvement of models for other L_p norm with the price of more complicated optimization operations. PCM is another approach for dealing with outliers [175]. Under this model, the memberships are interpreted by a possibilistic view, i.e., "the compatibilities of the points with the class prototypes" [175]. The effect of noise and outliers is abated with the consideration of typicality. In this case, the first condition for the membership coefficient described in Section I is relaxed to $\max_i u_{i,j} > 0, \forall j$. Accordingly, the cost function is reformulated as

$$J(\mathbf{U}, \mathbf{M}) = \sum_{i=1}^{c} \sum_{j=1}^{N} (u_{i,j})^m D_{ij} + \sum_{i=1}^{c} \eta_i \sum_{j=1}^{N} (1 - u_{i,j})^m$$

where η_i are some positive constants. The additional term tends to give credits to memberships with large values. A modified version in order to find appropriate clusters is proposed in [294]. Davé and Krishnapuram further elaborated the discussion on fuzzy clustering robustness and indicated its connection with robust statistics [71]. Relations among some widely used fuzzy clustering algorithms were discussed and their similarities to some robust statistical methods were also reviewed. They reached a unified framework as the conclusion for the previous discussion and proposed generic algorithms for robust clustering.

The standard FCM alternates the calculation of the membership and prototype matrix, which causes a computational burden for large-scale data sets. Kolen and Hutcheson accelerated the computation by combining updates of the two matrices [172]. Hung and Yang proposed a method to reduce computational time by identifying more accurate cluster centers [146]. FCM variants were also developed to deal with other data types, such as symbolic data [81] and data with missing values [129].

A family of fuzzy c-shells algorithms has also appeared to detect different types of cluster shapes, especially contours (lines, circles, ellipses, rings, rectangles, hyperbolas) in a two-dimensional data space. They use the "shells" (curved surfaces [70]) as the cluster prototypes instead of points or surfaces in traditional fuzzy clustering algorithms. In the case of FCS [36], [70], the proposed cluster prototype is represented as a d-dimensional hyperspherical shell $\mathbf{m}(\mathbf{v}, r)$ ($d = 2$ for circles), where $\mathbf{v} \in \Re^d$ is the center, and $r \in \Re$ is the radius. A distance function is defined as $D(\mathbf{x}_i, \mathbf{m}_j) = (\|\mathbf{x}_i - \mathbf{v}_j\| - r_j)$ to measure the distance from a data object $\mathbf{x}_i$ to the prototype $\mathbf{m}_j$. Similarly, other cluster shapes can be achieved by defining appropriate prototypes and corresponding distance functions, example including fuzzy c-spherical shells (FCSS) [176], fuzzy c-rings (FCR) [193], fuzzy c-quadratic shells (FCQS) [174], and fuzzy c-rectangular shells (FCRS) [137]. See [141] for further details.

Fuzzy set theories can also be used to create hierarchical cluster structure. Geva proposed a hierarchical unsupervised fuzzy clustering (HUFC) algorithm [104], which can effectively explore data structure at different levels like HC, while establishing the connections between each object and cluster in the hierarchy with the memberships. This design makes HUFC overcome one of the major disadvantages of HC, i.e., HC cannot reassign an object once it is designated into a cluster. Fuzzy clustering is also closely related to neural networks [24], and we will see more discussions in the following section.

H. Neural Networks-Based Clustering

Neural networks-based clustering has been dominated by SOFMs and adaptive resonance theory (ART), both of which are reviewed here, followed by a brief discussion of other approaches.

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 LVQ and SOFM [168], [169]. Intrinsically, LVQ performs supervised learning, and is not categorized as a clustering algorithm [169], [221]. 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 [221] (also see [157] for its improved version GLVQ-F). 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

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

algorithm. Soft LVQ algorithms, e.g., fuzzy algorithms for LVQ (FALVQ), were discussed in [156].

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 [168], [169]. 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 [221]. Basic SOFM training goes through the following steps.

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., $J = \arg_j \min\{\|\mathbf{x} - \mathbf{m}_j\|\}$.
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

$$h_{ci}(t) = \alpha(t) \exp\left(\frac{-\|\mathbf{r}_c - \mathbf{r}_i\|^2}{2\sigma^2(t)}\right)$$

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

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

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

While SOFM enjoy the merits of input space density approximation and independence of the order of input patterns, a number of user-dependent parameters cause problems when applied in real practice. Like the K-means algorithm, SOFM need to predefine the size of the lattice, i.e., the number of clusters, which is unknown for most circumstances. Additionally, trained SOFM may be suffering from input space density misrepresentation [132], where areas of low pattern density may be over-represented and areas of high density under-represented. Kohonen reviewed a variety of variants of SOFM in [169], which improve drawbacks of basic SOFM and broaden its applications. SOFM can also be integrated with other clustering approaches (e.g., K-means algorithm or HC) to provide more effective and faster clustering. [263] and [276] illustrate two such hybrid systems.

ART was developed by Carpenter and Grossberg, as a solution to the plasticity and stability dilemma [51], [53], [113]. 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 [51], although it can be extended to arbitrary input patterns by a variety of coding mechanisms. ART2 extends the applications to analog input patterns [52] and ART3 introduces a new mechanism originating from elaborate biological processes to achieve more efficient parallel search in hierarchical structures [54]. By incorporating two ART modules, which receive input patterns $(\mathrm{ART_a})$ and corresponding labels $(\mathrm{ART_b})$, respectively, with an inter-ART module, the resulting ARTMAP system can be used for supervised classifications [56]. The match tracking strategy ensures the consistency of category prediction between two ART modules by dynamically adjusting the vigilance parameter of $\mathrm{ART_a}$. Also see fuzzy ARTMAP in [55]. A similar idea, omitting the inter-ART module, is known as LAPART [134].

The basic ART1 architecture consists of two-layer nodes, 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 ρ $(0 \leq \rho \leq 1)$ 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 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

$$T_j = \sum_{i=1}^{d} W_{ij}^{12} x_i = \begin{cases} |\mathbf{x}|\alpha_j & \text{if } j \text{ is an uncommitted node (first activated)} \\ \dfrac{|\mathbf{x} \cap \mathbf{W}_j^{21}|}{\beta + |\mathbf{W}_j^{21}|} & \text{if } j \text{ is a committed node} \end{cases}$$

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 $\rho \leq (|\mathbf{x} \cap \mathbf{W}_J^{21}|/|\mathbf{x}|)$, go to step 5a), otherwise go to step 5b).

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

5)
   a) Update the corresponding weights for the active node as $\mathbf{W}_J^{12}(\text{new}) = (|\mathbf{x} \cap \mathbf{W}_J^{21}(\text{old})|)/(\beta + |\mathbf{W}_J^{21}(\text{old})|)$ and $\mathbf{W}_J^{21}(\text{new}) = \mathbf{x} \cap \mathbf{W}_J^{21}(\text{old})$.;
   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 [204]. She also showed how to adapt these algorithms under the ART1 framework. In [284] and [285], 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 [57]. 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. Huang *et al.* investigated and revealed more properties of FA classified as template, access, reset, and the number of learning epochs [143]. The criticisms for FA are mostly focused on its inefficiency in dealing with noise and the deficiency of hyperrectangular representation for clusters in many circumstances [23], [24], [281]. Williamson described Gaussian ART (GA) to overcome these shortcomings [281], in which each cluster is modeled with Gaussian distribution and represented as a hyperellipsoid geometrically. GA does not inherit the offline fast learning property of FA, as indicated by Anagnostopoulos *et al.* [13], who proposed different ART architectures: hypersphere ART (HA) [12] for hyperspherical clusters and ellipsoid ART (EA) [13] for hyperellipsoidal clusters, to explore a more efficient representation of clusters, while keeping important properties of FA. Baraldi and Alpaydin proposed SART following their general ART clustering networks frame, which is described through a feedforward architecture combined with a match comparison mechanism [23]. 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 [23].

In addition to these, many other neural network architectures are developed for clustering. Most of these architectures utilize prototype vectors to represent clusters, e.g., cluster detection and labeling network (CDL) [82], HEC [194], and SPLL [296]. HEC uses a two-layer network architecture to estimate the regularized Mahalanobis distance, which is equated to the Euclidean distance in a transformed whitened space. CDL is also a two-layer network with an inverse squared Euclidean metric. CDL requires the match between the input patterns and the prototypes above a threshold, which is dynamically adjusted. SPLL emphasizes initiation independent and adaptive generation of clusters. It begins with a random prototype in the input space and iteratively chooses and divides prototypes until no further split is available. The divisibility of a prototype is based on the consideration that each prototype represents only one natural cluster, instead of the combinations of several clusters. Simpson employed hyperbox fuzzy sets to characterize clusters [100], [249]. Each hyperbox is delineated by a min and max point, and data points build their relations with the hyperbox through the membership function. The learning process experiences a series of expansion and contraction operations, until all clusters are stable.

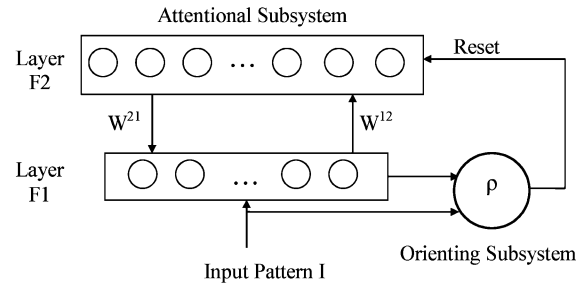


Fig. 2. ART1 architecture. Two layers are included in the attentional subsystem, connected via bottom-up and top-down adaptive weights. Their interactions are controlled by the orienting subsystem through a vigilance parameter.

I. Kernel-Based Clustering

Kernel-based learning algorithms [209], [240], [274] 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 [132]. The difficulty of curse of dimensionality can be overcome by the kernel trick, arising from Mercer's theorem [132]. 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 [241], 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 $\Phi : \Re^d \to F$. 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

$$\begin{aligned} &||\Phi(\mathbf{x}) - \mathbf{m}_l||^2 \\ &\quad = \left\| \Phi(\mathbf{x}) - \sum_{j=1}^{N} \tau_{lj} \Phi(\mathbf{x}_j) \right\|^2 \\ &\quad = k(\mathbf{x}, \mathbf{x}) - 2 \sum_{j=1}^{N} \tau_{lj} k(\mathbf{x}, \mathbf{x}_j) + \sum_{i,j=1}^{N} \tau_{li} \tau_{lj} k(\mathbf{x}_i, \mathbf{x}_j) \end{aligned}$$

where $\mathbf{m}_l$ is the center for the lth 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

$$C_{jl} = \begin{cases} 1, & \text{if } \mathbf{x}_j \text{ belongs to cluster } l \\ 0, & \text{otherwise.} \end{cases}$$

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

Then the kernel-K-means algorithm can be formulated as the following.

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 shown in the equation at the bottom of the page.
3) Update the mean vector $\mathbf{m}_h$ whose corresponding $C_{(i+1)h}$ is 1

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

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

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

5) Repeat steps 2)–4) until convergence is achieved.

Two variants of kernel- K-means were introduced in [66], motivated by SOFM and ART networks. These variants consider effects of neighborhood relations, while adjusting the cluster assignment variables, and use a vigilance parameter to control the process of producing mean vectors. The authors also illustrated the application of these approaches in case based reasoning systems.

An alternative kernel-based clustering approach is in [107]. The problem was formulated to determine an optimal partition $\mathbf{\Gamma}$ to minimize the trace of within-group scatter matrix in the feature space

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

where
$\xi_i = N_i/N, R(\mathbf{x}\,|\,C_i) = (1/N_i^2)\sum_{l=1}^{N}\sum_{j=1}^{N}\gamma_{il}\gamma_{ij}k(\mathbf{x}_l, \mathbf{x}_j)$, and N_i is the total number of patterns in the ith cluster.

Note that the kernel function utilized in this case is the radial basis function (RBF) and $R(\mathbf{x}\,|\,C_i)$ can be interpreted as a measure of the denseness for the ith cluster.

Ben-Hur *et al.* presented a new clustering algorithm, SVC, in order to find a set of contours used as the cluster boundaries in the original data space [31], [32]. These contours can be formed by mapping back the smallest enclosing sphere in the transformed feature space. RBF is chosen in this algorithm, and, by adjusting the width parameter of RBF, SVC can form either agglomerative or divisive hierarchical clusters. When some points are allowed to lie outside the hypersphere, SVC can deal with outliers effectively. An extension, called multiple spheres support vector clustering, was proposed in [62], which combines the concept of fuzzy membership.

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 [107], the kernel matrix can provide the means to estimate the number of clusters.

Meanwhile, there are also some problems requiring further consideration and investigation. Like many other algorithms, how to determine the appropriate parameters, for example, the width of Gaussian kernel, is not trivial. The problem of computational complexity may become serious for large data sets.

The process of constructing the sum-of-squared clustering algorithm [107] and K-means algorithm [241] presents a good example to reformulate more powerful nonlinear versions for many existing linear algorithms, provided that the scalar product can be obtained. Theoretically, it is important to investigate whether these nonlinear variants can keep some useful and essential properties of the original algorithms and how Mercer kernels contribute to the improvement of the algorithms. The effect of different types of kernel functions, which are rich in the literature, is also an interesting topic for further exploration.

J. Clustering Sequential Data

Sequential data are sequences with variable length and many other distinct characteristics, e.g., dynamic behaviors, time constraints, and large volume [120], [265]. Sequential data can be generated from: DNA sequencing, speech processing, text mining, medical diagnosis, stock market, customer transactions, web data mining, and robot sensor analysis, to name a few [78], [265]. In recent decades, sequential data grew explosively. For example, in genetics, the recent statistics released on October 15, 2004 (Release 144.0) shows that there are 43 194 602 655 bases from 38 941 263 sequences in GenBank database [103] and release 45.0 of SWISSPROT on October 25, 2004 contains 59 631 787 amino acids in 163 235 sequence entries [267]. Cluster analysis explores potential patterns hidden in the large number of sequential data in the context of unsupervised learning and therefore provides a crucial way to meet the current challenges. Generally, strategies for sequential clustering mostly fall into three categories.

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

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

1) Sequence Similarity: The first scheme is based on the measure of the distance (or similarity) between each pair of sequences. Then, proximity-based clustering algorithms, either hierarchical or partitional, can group sequences. Since many sequential data are expressed in an alphabetic form, like DNA or protein sequences, conventional measure methods are inappropriate. 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. A common analysis processes is alignment, illustrated in Fig. 3. The defined distance is known as edit distance or Levenshtein distance [120], [236]. These edit operations are weighted (punished or rewarded) 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.

Given two sequences, $\mathbf{x} = (x_1 \ldots x_i \ldots x_N)$ and $\mathbf{y} = (y_1 \ldots y_j \ldots y_M)$, the basic dynamic programming-based sequence alignment algorithm, also known as the Needleman-Wunsch algorithm, can be depicted by the following recursive equation [78], [212]:

$$S(i,j) = \max \begin{cases} S(i-1,j-1) + e(x_i, y_j) \\ S(i-1,j) + e(x_i, \phi), \\ S(i,j-1) + e(\phi, y_j) \end{cases}$$

where $S(i,j)$ is defined as the best alignment score between sequence segment $(x_1 \ldots x_i), (1 \leq i \leq N)$ of $\mathbf{x}$ and $(y_1 \ldots y_j), (1 \leq j \leq M)$ of $\mathbf{y}$, and $e(x_i, y_j), e(x_i, \phi)$, or $e(\phi, y_j)$ represent the cost for aligning x_i to y_j, aligning x_i to a gap (denoted as ϕ), or aligning y_j to a gap, respectively. The computational results for each position at i and j are recorded in an array with a pointer that stores current optimal operations and provides an effective path in backtracking the alignment. The Needleman-Wunsch algorithm considers the comparison of the whole length of two sequences and therefore performs a global optimal alignment. However, it is also important to find local similarity among sequences in many circumstances. The Smith-Waterman algorithm achieves that by allowing the beginning of a new alignment during the recursive computation, and the stop of an alignment anywhere in the dynamic programming matrix [78], [251]. This change is summarized in the following:

$$S(i,j) = \min \begin{cases} S(i-1,j-1) + e(x_i, y_j) \\ S(i-1,j) + e(x_i, \phi), \\ S(i,j-1) + e(\phi, y_j) \\ 0. \end{cases} .$$

For both the global and local alignment algorithms, the computation complexity is $O(NM)$, which is very expensive, especially for a clustering problem that requires an all-against-all pairwise comparison. A wealth of speed-up methods has been developed to improve the situation [78], [120]. We will see more discussion in Section III-E in the context of biological sequences analysis. Other examples include applications for speech recognition [236] and navigation pattern mining [131].

```
C L U S - - - - - - T E R I - N G
M M S M I I I I I M D D M I M D
C L A S S I F I C A T - - I O N -
```

M: Match; D: Deletion;
I: Insertion; S: Substitution

Fig. 3. Illustration of a sequence alignment. Series of edit operations is performed to change the sequence CLUSTERING into the sequence CLASSIFICATION.

2) Indirect Sequence Clustering: The second approach employs an indirect strategy, which begins with the extraction of a set of features from the sequences. All the sequences are then mapped into the transformed feature space, where classical vector space-based clustering algorithms can be used to form clusters. Obviously, feature extraction becomes the essential factor that decides the effectiveness of these algorithms. Guralnik and Karypis discussed the potential dependency between two sequential patterns and suggested both the global and the local approaches to prune the initial feature sets in order to better represent sequences in the new feature space [119]. Morzy *et al.* utilized the sequential patterns as the basic element in the agglomerative hierarchical clustering and defined a co-occurrence measure, as the standard of fusion of smaller clusters [207]. These methods greatly reduce the computational complexities and can be applied to large-scale sequence databases. However, the process of feature selection inevitably causes the loss of some information in the original sequences and needs extra attention.

3) Statistical Sequence Clustering: Typically, the first two approaches are used to deal with sequential data composed of alphabets, while the third paradigm, which aims to construct statistical models to describe the dynamics of each group of sequences, can be applied to numerical or categorical sequences. The most important method is hidden Markov models (HMMs) [214], [219], [253], which first gained its popularity in the application of speech recognition [229]. A discrete HMM describes an unobservable stochastic process consisting of a set of states, each of which is related to another stochastic process that emits observable symbols. Therefore, the HMM is completely specified by the following.

1) A finite set $V = \{V_1, \ldots, V_Q\}$ with Q states.
2) A discrete set $O = \{O_1, \ldots, O_M\}$ with M observation symbols.
3) A state transition distribution $\mathrm{A} = \{\alpha_{ij}\}$, where
$$\alpha_{ij} = P(j\text{th state at time } t+1 \,|\, i\text{th state at time } t).$$
4) A symbol emission distribution $\mathrm{B} = \{\beta_{il}\}$, where
$$\beta_{il} = P[V_l \text{ at } t \,|\, i\text{th state at } t].$$
5) An initial state distribution $\pi = \{\pi_i\}$, where
$$\pi_i = P[i\text{th state at } t = 1].$$

After an initial state is selected according to the initial distribution π, a symbol is emitted with emission distribution E. The next state is decided by the state transition distribution T and it also generates a symbol based on E. The process repeats until reaching the last state. Note that the procedure generates

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

a sequence of symbol observations instead of states, which is where the name "hidden" comes from. HMMs are well founded theoretically [229]. Dynamic programming and EM algorithm are developed to solve the three basic problems of HMMs as the following.

1) *Likelihood* (forward or backward algorithm). Compute the probability of an observation sequence given a model.
2) *State interpretation* (Vertbi algorithm). Find an optimal state sequence by optimizing some criterion function given the observation sequence and the model.
3) *Parameter estimation* (Baum–Welch algorithm). Design suitable model parameters to maximize the probability of observation sequence under the model.

The equivalence between an HMM and a recurrent back-propagation network was elucidated in [148], and a universal framework was constructed to describe both the computational and the structural properties of the HMM and the neural network.

Smyth proposed an HMM-based clustering model, which, similar to the theories introduced in mixture densities-based clustering, assumes that each cluster is generated based on some probability distribution [253]. Here, HMMs are used rather than the common Gaussian or t-distribution. In addition to the form of finite mixture densities, the mixture model can also be described by means of an HMM with the transition matrix

$$\mathrm{A} = \begin{pmatrix} \mathrm{A}_1 & & & 0 \\ & \mathrm{A}_2 & & \\ & & \cdots & \\ 0 & & & \mathrm{A}_K \end{pmatrix}$$

where $\mathrm{A}_i, i = 1, \ldots, K$ is the transition distribution for the ith cluster. The initial distribution of the HMM is determined based on the prior probability for each cluster. The basic learning process starts with a parameter initialization scheme to form a rough partition with the log-likelihood of each sequence serving as the distance measure. The partition is further refined by training the overall HMM over all sequences with the classical EM algorithm. A Monte-Carlo cross validation method was used to estimate the possible number of clusters. An application with a modified HMM model that considers the effect of context for clustering facial display sequences is illustrated in [138]. Oates *et al.* addressed the initial problem by pregrouping the sequences with the agglomerative hierarchical clustering, which operates on the proximity matrix determined by the dynamic time warping (DTW) technique [214]. The area formed between one original sequence and a new sequence, generated by warping the time dimension of another original sequence, reflects the similarity of the two sequences. Li and Biswas suggested several objective criterion functions based on posterior probability and information theory for structural selection of HMMs and cluster validity [182]. More recent advances on HMMs and other related topics are reviewed in [30].

Other model-based sequence clustering includes mixtures of first-order Markov chain [255] and a linear model like autoregressive moving average (ARMA) model [286]. Usually, they are combined with EM for parameter estimation [286]. Smyth [255] and Cadez *et al.* [50] further generalize a universal probabilistic framework to model mixed data measurement, which includes both conventional static multivariate vectors and dynamic sequence data.

The 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 [214], [253]. Also, the model selected is required to have sufficient complexity, in order to interpret the characteristics of data.

K. Clustering Large-Scale Data Sets

Scalability becomes more and more important for clustering algorithms with the increasing complexity of data, mainly manifesting in two aspects: enormous data volume and high dimensionality. Examples, illustrated in the sequential clustering section, are just some of the many applications that require this capability. With the further advances of database and Internet technologies, clustering algorithms will face more severe challenges in handling the rapid growth of data. We summarize the computational complexity of some typical and classical clustering algorithms in Table II with several newly proposed approaches specifically designed to deal with large-scale data sets. Several points can be generalized through the table.

1) Obviously, classical hierarchical clustering algorithms, including single-linkage, complete linkage, average linkage, centroid linkage and median linkage, are not appropriate for large-scale data sets due to the quadratic computational complexities in both execution time and store space.
2) K-means algorithm has a time complexity of $O(NKd)$ and space complexity of $O(N + K)$. Since N is usually much larger than both K and d, the complexity becomes near linear to the number of samples in the data sets. K-means algorithm is effective in clustering large-scale data sets, and efforts have been made in order to overcome its disadvantages [142], [218].
3) Many novel algorithms have been developed to cluster large-scale data sets, especially in the context of data mining [44], [45], [85], [135], [213], [248]. Many of them can scale the computational complexity linearly to the input size and demonstrate the possibility of handling very large data sets.
   a) Random sampling approach, e.g., CLARA clustering large applications (CLARA) [161] and CURE [116]. The key point lies that the appropriate sample sizes can effectively maintain the important geometrical properties of clusters. Furthermore, Chernoff bounds can provide estimation for the lower bound of the minimum sample size, given the low probability that points in each cluster are missed in the sample set [116]. CLARA represents each cluster with a

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

medoid while CURE chooses a set of well-scattered and center-shrunk points.

b) Randomized search approach, e.g., clustering large applications based on randomized search (CLARANS) [213]. CLARANS sees the clustering as a search process in a graph, in which each node corresponds to a set of K medoids. It begins with an arbitrary node as the current node and examines a set of neighbors, defined as the node consisting of only one different data object, to seek a better solution, i.e., any neighbor, with a lower cost, becomes the current node. If the maximum number of neighbors, specified by the user, has been reached, the current node is accepted as a winning node. This process iterates several times as specified by users. Though CLARANS achieves better performance than algorithms like CLARA, the total computational time is still quadratic, which makes CLARANS not quite effective in very large data sets.

c) Condensation-based approach, e.g., BIRCH [295]. BIRCH generates and stores the compact summaries of the original data in a CF tree, as discussed in Section II-B. This new data structure efficiently captures the clustering information and largely reduces the computational burden. BIRCH was generalized into a broader framework in [101] with two algorithms realization, named as BUBBLE and BUBBLE-FM.

d) Density-based approach, e.g., density based spatial clustering of applications with noise (DBSCAN) [85] and density-based clustering (DENCLUE) [135]. DBSCAN requires that the density in a neighborhood for an object should be high enough if it belongs to a cluster. DBSCAN creates a new cluster from a data object by absorbing all objects in its neighborhood. The neighborhood needs to satisfy a user-specified density threshold. DBSCAN uses a R^*-tree structure for more efficient queries. DENCLUE seeks clusters with local maxima of the overall density function, which reflects the comprehensive influence of data objects to their neighborhoods in the corresponding data space.

e) Grid-based approach, e.g., WaveCluster [248] and fractal clustering (FC) [26]. WaveCluster assigns data objects to a set of units divided in the original feature space, and employs wavelet transforms on these units, to map objects into the frequency domain. The key idea is that clusters can be easily distinguished in the transformed space. FC combines the concepts of both incremental clustering and fractal dimension. Data objects are incrementally added to the clusters, specified through an initial process, and represented as cells in a grid, with the condition that the fractal dimension of cluster needs to keep relatively stable.

4) Most algorithms listed previously lack the capability of dealing with data with high dimensionality. Their performances degenerate with the increase of dimensionality. Some algorithms, like FC and DENCLUE, have shown some successful applications in such cases, but these are still far from completely effective.

In addition to the aforementioned approaches, several other techniques also play significant roles in clustering large-scale data sets. Parallel algorithms can more effectively use computational resources, and greatly improve overall performance in the context of both time and space complexity [69], [217], [262]. Incremental clustering techniques do not require the storage of the entire data set, and can handle it in a one-pattern-at-a-time way. If the pattern displays enough closeness to a cluster according to some predefined criteria, it is assigned to the cluster. Otherwise, a new cluster is created to represent the object. A typical example is the ART family [51]–[53] discussed in Section II-H. Most incremental clustering algorithms are dependent on the order of the input patterns [51], [204]. Bradley, Fayyad, and Reina proposed a scalable clustering framework, considering seven relevant important characteristics in dealing with large databases [44]. Applications of the framework were illustrated for the K-means algorithm and EM mixture models [44], [45].

L. Exploratory Data Visualization and High-Dimensional Data Analysis Through Dimensionality Reduction

For most of the algorithms summarized in Table II, although they can deal with large-scale data, they are not sufficient for analyzing high-dimensional data. The term, "curse of dimensionality," which was first used by Bellman to indicate the exponential growth of complexity in the case of multivariate function estimation under a high dimensionality situation [28], is generally used to describe the problems accompanying high dimensional spaces [34], [132]. It is theoretically proved that the distance between the nearest points is no different from that of other points when the dimensionality of the space is high enough [34]. Therefore, clustering algorithms that are based on the distance measure may no longer be effective in a high dimensional space. Fortunately, in practice, many high-dimensional data usually have an intrinsic dimensionality that is much lower than the original dimension [60]. Dimension reduction is important in cluster analysis, which not only makes the high-dimensional data addressable and reduces the computational cost, but provides users with a clearer picture and visual examination of the data of interest. However, dimensionality reduction methods inevitably cause some loss of information, and may damage the interpretability of the results, even distort the real clusters.

One natural strategy for dimensionality reduction is to extract important components from original data, which can contribute to the division of clusters. Principle component analysis (PCA) or Karhunen-Loéve transformation is one of the typical approaches, which is concerned with constructing a linear combination of a set of vectors that can best describe the variance of data. Given the $N \times D$ input pattern matrix $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_j, \ldots, \mathbf{x}_N]^T$, the linear mapping $\mathbf{Y} = \mathbf{XV}$ projects $\mathbf{x}_i$ $(1 \leq i \leq N)$ into a low-dimensional $(L(L < D))$ subspace, where $\mathbf{Y}$ is the resulting $N \times L$ matrix and $\mathbf{V}$ is the $D \times L$ projection matrix whose columns are the eigenvectors that correspond to the l largest eigenvalues of the $D \times D$ covariance matrix $\mathbf{\Sigma}$, calculated from the whole data set (hence, the column vectors of $\mathbf{V}$ are orthonormal). PCA estimates the matrix $\mathbf{V}$ while minimizing the sum of squares of the error

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

of approximating the input vectors. In this sense, PCA can be realized through a three-layer neural network, called an auto-associative multilayer perceptron, with linear activation functions [19], [215]. In order to extract more complicated nonlinear data structure, nonlinear PCA was developed and one of the typical examples is kernel PCA. As methods discussed in Section II-I, kernel PCA first maps the input patterns into a feature space. The similar steps are then applied to solve the eigenvalue problem with the new covariance matrix in the feature space. In another way, extra hidden layers with nonlinear activation functions can be added into the auto-associative network for this purpose [38], [75].

PCA is appropriate for Gaussian distributions since it relies on second-order relationships in the covariance matrix, Other linear transforms, like independent component analysis (ICA) and projection pursuit, which use higher order statistical information, are more suited for non-Gaussian distributions [60], [151]. The basic goal of ICA is to find the components that are most statistically independent from each other [149], [154]. In the context of blind source separation, ICA aims to separate the independent source signals from the mixed observation signal. This problem can be formulated in several different ways [149], and one of the simplest form (without considering noise) is represented as $\mathbf{x} = \mathbf{A}\mathbf{s}$, where $\mathbf{x}$ is the N-dimensional observable vector, $\mathbf{s}$ is the L-dimensional source vector assumed to be statistically independent, and $\mathbf{A}$ is a nonsingular $N \times L$ mixing matrix. ICA can also be realized by virtue of multilayer perceptrons, and [158] illustrates one of such examples. The proposed ICA network includes whitening, separation, and basis vectors estimation layers, with corresponding learning algorithms. The authors also indicated its connection to the auto-associative multilayer perceptron. Projection pursuit is another statistical technique for seeking low-dimensional projection structures for multivariate data [97], [144]. Generally, projection pursuit regards the normal distribution as the least interesting projections and optimizes some certain indices that measure the degree of nonnormality [97]. PCA can be considered as a special example of projection pursuit, as indicated in [60]. More discussions on the relations among PCA, ICA, projection pursuit, and other relevant techniques are offered in [149] and [158].

Different from PCA, ICA, and projection pursuit, Multidimensional scaling (MDS) is a nonlinear projection technique [75], [292]. The basic idea of MDS lies in fitting original multivariate data into a low-dimensional structure while aiming to maintain the proximity information. The distortion is measured through some criterion functions, e.g., in the sense of sum of squared error between the real distance and the projection distance. The isometric feature mapping (Isomap) algorithm is another nonlinear technique, based on MDS [270]. Isomap estimates the geodesic distance between a pair of points, which is the shortest path between the points on a manifold, by virtue of the measured input-space distances, e.g., the Euclidean distance usually used. This extends the capability of MDS to explore more complex nonlinear structures in the data. Locally linear embedding (LLE) algorithm addresses the nonlinear dimensionality reduction problem from a different starting point [235]. 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. Another interesting nonlinear dimensionality reduction approach, known as Laplace eigenmap algorithm, is presented in [27].

As discussed in Section II-H, SOFM also provide good visualization for high-dimensional input patterns [168]. SOFM map input patterns into a one or usually two dimensional lattice structure, consisting of nodes associated with different clusters. An application for clustering of a large set of documental data is illustrated in [170], in which 6 840 568 patent abstracts were projected onto a SOFM with 1 002 240 nodes.

Subspace-based clustering addresses the challenge by exploring the relations of data objects under different combinations of features. clustering in quest (CLIQUE) [3] employs a bottom-up scheme to seek dense rectangular cells in all subspaces with high density of points. Clusters are generated as the connected components in a graph whose vertices stand for the dense units. The resulting minimal description of the clusters is obtained through the merge of these rectangles. OptiGrid [136] is designed to obtain an optimal grid-partitioning. This is achieved by constructing the best cutting hyperplanes through a set of projections. The time complexity for OptiGrid is in the interval of $O(Nd)$ and $O(Nd \log N)$. ORCLUS (arbitrarily ORiented projected CLUster generation) [2] defines a generalized projected cluster as a densely distributed subset of data objects in a subspace, along with a subset of vectors that represent the subspace. The dimensionality of the subspace is prespecified by users as an input parameter, and several strategies are proposed in guidance of its selection. The algorithm begins with a set of randomly selected K_0 seeds with the full dimensionality. This dimensionality and the number of clusters are decayed according to some factors at each iteration, until the number of clusters reaches the predefined values. Each repetition consists of three basic operations, known as assignment, vector finding, and merge. ORCLUS has the overall time complexity of $O(K_0^3 + K_0 Nd + K_0^2 d^3)$ and space complexity of $O(K_0 d^2)$. Obviously, the scalability to large data sets relies on the number of initial seeds K_0. A generalized subspace clustering model, pCluster was proposed in [279]. These pClusters are formed by a depth-first clustering algorithm. Several other interesting applications, including a Clindex (CLustering for INDEXing) scheme and wavelet transform, are shown in [184] and [211], respectively.

M. How Many Clusters?

The clustering process partitions data into an appropriate number of subsets. Although for some applications, users can determine the number of clusters, K, in terms of their expertise, under more circumstances, the value of K is unknown and needs to be estimated exclusively from the data themselves. Many clustering algorithms ask K to be provided as an input parameter, and it is obvious that the quality of resulting clusters is largely dependent on the estimation of K. A division with

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

too many clusters complicates the result, therefore, makes it hard to interpret and analyze, while a division with too few clusters causes the loss of information and misleads the final decision. Dubes called the problem of determining the number of clusters "the fundamental problem of cluster validity" [74].

A large number of attempts have been made to estimate the appropriate K and some of representative examples are illustrated in the following.

1) *Visualization of the data set*. For the data points that can be effectively projected onto a two-dimensional Euclidean space, which are commonly depicted with a histogram or scatterplot, direct observations can provide good insight on the value of K. However, the complexity of most real data sets restricts the effectiveness of the strategy only to a small scope of applications.

2) *Construction of certain indices (or stopping rules)*. These indices usually emphasize the compactness of intra-cluster and isolation of inter-cluster and consider the comprehensive effects of several factors, including the defined squared error, the geometric or statistical properties of the data, the number of patterns, the dissimilarity (or similarity), and the number of clusters. Milligan and Cooper compared and ranked 30 indices according to their performance over a series of artificial data sets [202]. Among these indices, the Caliñski and Harabasz index [74] achieve the best performance and can be represented as

$$\mathrm{CH}(K) = \frac{\mathrm{Tr}(S_B)}{K-1} \bigg/ \frac{\mathrm{Tr}(S_W)}{N-K}$$

where N is the total number of patterns and $\mathrm{Tr}(S_B)$ and $\mathrm{Tr}(S_W)$ are the trace of the between and within class scatter matrix, respectively. The K that maximizes the value of $\mathrm{CH}(K)$ is selected as the optimal. It is worth noting that these indices may be data dependent. The good performance of an index for certain data does not guarantee the same behavior with different data. As pointed out by Everitt, Landau, and Leese, "it is advisable not to depend on a single rule for selecting the number of groups, but to synthesize the results of several techniques" [88].

3) *Optimization of some criterion functions under probabilistic mixture-model framework*. In a statistical framework, finding the correct number of clusters (components) K, is equivalent to fitting a model with observed data and optimizing some criterion [197]. Usually, the EM algorithm is used to estimate the model parameters for a given K, which goes through a predefined range of values. The value of K that maximizes (or minimizes) the defined criterion is regarded as optimal. Smyth presented a Monte-Carlo cross-validation method, which randomly divides data into training and test sets M times according to a certain fraction β ($\beta = 0.5$ works well from the empirical results) [252]. The K is selected either directly based on the criterion function or some posterior probabilities calculated.

A large number of criteria, which combine concepts from information theory, have been proposed in the literature. Typical examples include,

- Akaike's information criterion (AIC) [4], [282]

$$\mathrm{AIC}(K) = \frac{-2(N-1-N_k-K/2)l(\hat{\boldsymbol{\theta}})}{N} + 3N_p$$

where N is the total number of patterns, N_k is the number of parameters for each cluster, N_p is the total number of parameters estimated, and $l(\hat{\boldsymbol{\theta}})$ is the maximum log-likelihood. K is selected with the minimum value of $\mathrm{AIC}(K)$.

- Bayesian inference criterion (BIC) [226], [242]

$$\mathrm{BIC}(K) = l(\hat{\boldsymbol{\theta}}) - (N_p/2)\log(N).$$

K is selected with the maximum value of $\mathrm{BIC}(K)$.

More criteria, such as minimum description length (MDL) [114], [233], minimum message length (MML) [114], [216], cross validation-based information criterion (CVIC) [254] and covariance inflation criterion (CIC) [272], with their characteristics, are summarized in [197]. Like the previous discussion for validation index, there is no criterion that is superior to others in general case. The selection of different criteria is still dependent on the data at hand.

4) *Other heuristic approaches based on a variety of techniques and theories*. 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 [107]. Kothari and Pitts described a scale-based method, in which the distance from a cluster centroid to other clusters in its neighborhood is considered (added as a regularization term in the original squared error criterion, Section II-C) [160]. The neighborhood of clusters work as a scale parameter and the K that is persistent in the largest interval of the neighborhood parameter is regarded as the optimal.

Besides the previous methods, constructive clustering algorithms can adaptively and dynamically adjust the number of clusters rather than use a prespecified and fixed number. ART networks generate a new cluster, only when the match between the input pattern and the expectation is below some prespecified confidence value [51]. A functionally similar mechanism is used in the CDL network [82]. The robust competitive clustering algorithm (RCA) describes a competitive agglomeration process that progresses in stages, and clusters that lose in the competition are discarded, and absorbed into other clusters [98]. This process is generalized in [42], which attains the number of clusters by balancing the effect between the complexity and the fidelity. Another learning scheme, SPLL iteratively divides cluster prototypes from a single prototype until no more prototypes satisfy the split criterion [296]. Several other constructive clustering algorithms, including the FACS and plastic neural gas, can be accessed in [223] and [232], respectively. Obviously, the problem of determining the number

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

of clusters is converted into a parameter selection problem, and the resulting number of clusters is largely dependent on parameter tweaking.

III. Applications

We illustrate applications of clustering techniques in three aspects. The first is for two classical benchmark data sets that are widely used in pattern recognition and machine learning. Then, we show an application of clustering for the traveling salesman problem. The last topic is on bioinformatics. We deal with classical benchmarks in Sections III-A and III-B and the traveling salesman problem in Section III-C. A more extensive discussion of bioinformatics is in Sections III-D and III-E.

A. Benchmark Data Sets—Iris

The iris data set [92] is one of the most popular data sets to examine the performance of novel methods in pattern recognition and machine learning. It can be downloaded from the UCI Machine Learning Repository at http://www.ics.uci.edu/~mlearn/MLRepository.html. There are three categories in the data set (i.e., iris setosa, iris versicolor and iris virginical), each having 50 patterns with four features [i.e., sepal length (SL), sepal width (SW), petal length (PL), and petal width (PW)]. Iris setosa can be linearly separated from iris versicolor and iris virginical, while iris versicolor and iris virginical are not linearly separable (see Fig. 4(a), in which only three features are used). Fig. 4(b) depicts the clustering result with a standard K-means algorithm. It is clear to see that K-means can correctly differentiate iris setosa from the other two iris plants. But for iris versicolor and virginical, there exist 16 misclassifications. This result is similar to those (around 15 errors) obtained from other classical clustering algorithms [221]. Table III summarizes some of the clustering results reported in the literature. From the table, we can see that many newly developed approaches can greatly improve the clustering performance on iris data set (around 5 misclassifications); some even can achieve 100% accuracy. Therefore, the data can be well classified with appropriate methods.

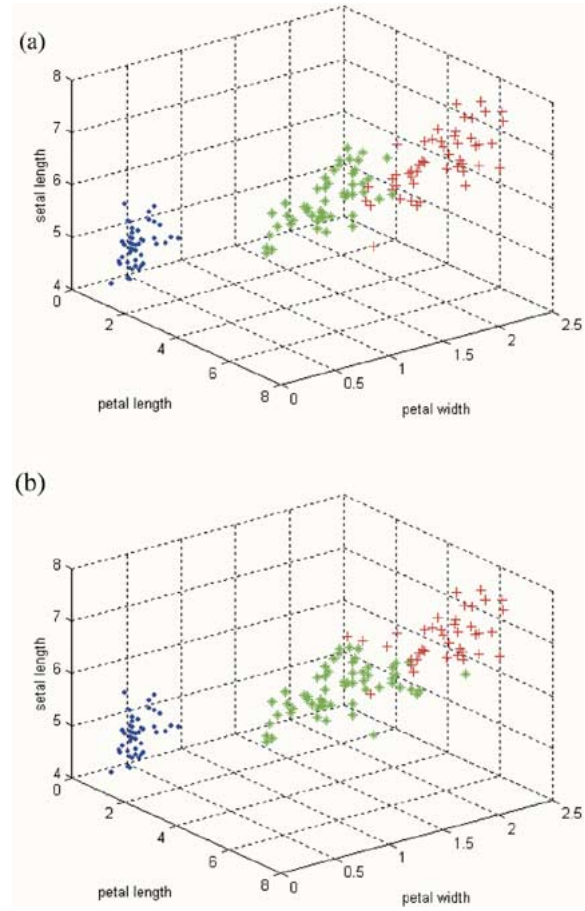


Fig. 4. (a) Iris data sets. There are three iris categories, each having 50 samples with 4 features. Here, only three features are used: PL, PW, and SL. (b) K-means clustering result with 16 classification errors observed.

TABLE III
Some Clustering Results for the Iris Data Set

Algorithm	Number of errors	Percentage of errors
GLVQ [221]	17	11.3%
FCM [129]	16	10.6%
GFMM [100]	0~7	0~4.7%
Mercer kernel based clustering algorithm [107]	3	2%
SVC [31]	4	2.7%
CDL [82]	6	4%
HC [203]	13~17	8.7~11.3%
RHC [203]	5~6	3.3~4%
FA [23]	6.77~46.4	4.5~30.9%

GLVQ: general learning vector quantization;
GFMM: general fuzzy min-max neural network;
SVC: support vector clustering; FCM: fuzzy c-means;
CDL: cluster detection and labeling network;
HC: hierarchical clustering; RHC: relative hierarchical clustering;
FA: fuzzy adaptive resonance theory.

B. Benchmark Data Sets—Mushroom

Unlike the iris data set, all of the features of the mushroom data set, which can also be accessible at the UCI Machine Learning Repository, are nominal rather than numerical. These 23 species of gilled mushrooms are categorized as either edible or poisonous. The total number of instances is 8 124 with 4 208 being edible and 3 916 poisonous. The 22 features are summarized in Table IV with corresponding possible values. Table V illustrates some experimental results in the literature. As indicated in [117] and [277], traditional clustering strategies, like K-means and hierarchical clustering, work poorly on the data set. The accuracy for K-means is just around 69% [277] and the clusters formed by classical HC are mixed with nearly similar proportion of both edible and poisonous objects [117]. The results reported in the newly developed algorithms, which are specifically used for tackling categorical or mixture data, greatly improve the situation [117], [183]. The algorithm ROCK divides objects into 21 clusters with most of them (except one) consisting of only one category, which increases the accuracy almost to 99%. The algorithm SBAC works on a subset of 200 randomly selected objects, 100 for each category and the general results show the correct partition of 3 clusters (two for edible mushrooms, one for poisonous ones). In both studies, the

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

TABLE IV
FEATURES FOR THE MUSHROOM DATA SET

Features	Potential values
Cap-shape	bell=b, conical=c, convex=x, flat=f, knobbed=k, sunken=s
Cap-surface	fibrous=f, grooves=g, scaly=y, smooth=s
Cap-color	brown=n, buff=b, cinnamon=c, gray=g, green=r, pink=p, purple=u, red=e, white=w, yellow=y
bruises	bruises=t, no=f
odor	almond=a, anise=l, creosote=c, fishy=y, foul=f, musty=m, none=n, pungent=p, spicy=s
Gill-attachment	attached=a, descending=d, free=f, notched=n
Gill-spacing	close=c, crowded=w, distant=d
Gill-size	broad=b, narrow=n
Gill-color	black=k, brown=n, buff=b, chocolate=h, gray=g, green=r, orange=o, pink=p, purple=u, red=e, white=w, yellow=y
Stalk-shape	enlarging=e, tapering=t
Stalk-root	bulbous=b, club=c, cup=u, equal=e, rhizomorphs=z, rooted=r, missing=?
Stalk-surface-above-ring	ibrous=f, scaly=y, silky=k, smooth=s
Stalk-surface-below-ring	ibrous=f, scaly=y, silky=k, smooth=s
Stalk-color-above-ring	brown=n, buff=b, cinnamon=c, gray=g, orange=o, pink=p, red=e, white=w, yellow=y
Stalk-color-below-ring	brown=n, buff=b, cinnamon=c, gray=g, orange=o, pink=p, red=e, white=w, yellow=y
Veil-type	partial=p, universal=u
Veil-color	brown=n, orange=o, white=w, yellow=y
Ring-number	none=n, one=o, two=t
Ring-type	cobwebby=c, evanescent=e, flaring=f, large=l, none=n, pendant=p, sheathing=s, zone=z
Spore-print-color	black=k, brown=n, buff=b, chocolate=h, green=r, orange=o, purple=u, white=w, yellow=y
population	abundant=a, clustered=c, numerous=n, scattered=s, several=v, solitary=y
habitat	grasses=g, leaves=l, meadows=m paths=p, urban=u, waste=w, wcods=d

constitution of each feature for generated clusters is also illustrated and it is observed that some features, like cap-shape and ring-type, represent themselves identically for both categories and, thus, suggest poor performance of traditional approaches. Meanwhile, feature odor shows good discrimination for the different types of mushrooms. Usually, value almond, anise, or none indicates the edibility of mushrooms, while value pungent, foul, or fishy means the high possibility of presence of poisonous contents in the mushrooms.

C. Traveling Salesman Problem

The traveling salesman problem (TSP) is one of the most studied examples in an important class of problems known as 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 each relating two vertices with an associated nonnegative 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 [208]. The divide and conquer paradigm gives the flexibility to hierarchically break large problems into arbitrarily small clusters depending on what tradeoff between accuracy and speed is desired. In addition, the subproblems provide an excellent opportunity to take advantage of parallel systems for further optimization. As the first stage of the algorithm, the ART network is used to sort the cities into clusters. The vigilance parameter is used to set a maximum distance from the current pattern. A vigilance parameter between 0 and 1 is used as a percentage of the global space to determine the vigilance distance. Values were chosen based on the desired number and size of individual clusters. The clusters were then each passed to a version of the Lin-Kernighan (LK) algorithm [187]. The last step combines the subtours back into one complete tour. Tours with good quality for city levels up to 1 000 000 were obtained within 25 minutes on a 2 GHz AMD Athlon MP processor with 512 M of DDR RAM. Fig. 5 shows the visualizing results for 1 000, 10 000, and 1 000 000 cities, respectively.

TABLE V
SOME CLUSTERING RESULTS FOR THE MUSHROOM DATA SET

Algorithm	Percentage of accuracy
K-means [277] *	69%
COP-*K*-means [277]*	96%
Hierarchical clustering [117]	Fail to discriminate the two types of mushroom
ROCK [117]	About 99%
SBAC [183]+	Correctly classify instances into 3 groups
ARTMAP [56]	Over 99% with 1000 training samples
AutoClass [183] +	Correctly classify instances into 4 groups

* A subset with 50 instances and 21 features was used.
\+ A subset with 200 instances was used.
COP: constraint-partitioning;
ROCK: robust clustering using links;
SBAC: similarity-based agglomerative clustering.

It is worthwhile to emphasize the relation between the TSP and very large-scale integrated (VLSI) circuit clustering, which partitions a sophisticated system into smaller and simpler subcircuits to facilitate the circuit design. The object of the partitions is to minimize the number of connections among the components. One strategy for solving the problem is based on geometric representations, either linear or multidimensional [8]. Alpert and Kahng considered a solution to the problem as the "inverse" of the divide-and-conquer TSP method and used a linear tour of the modules to form the subcircuit partitions [7]. They adopted the spacefilling curve heuristic for the TSP to construct the tour so that connected modules are still close in the generated tour. A dynamic programming method was used to generate the resulting partitions. More detailed discussion on VLSI circuit clustering can be found in the survey by Alpert and Kahng [7].

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

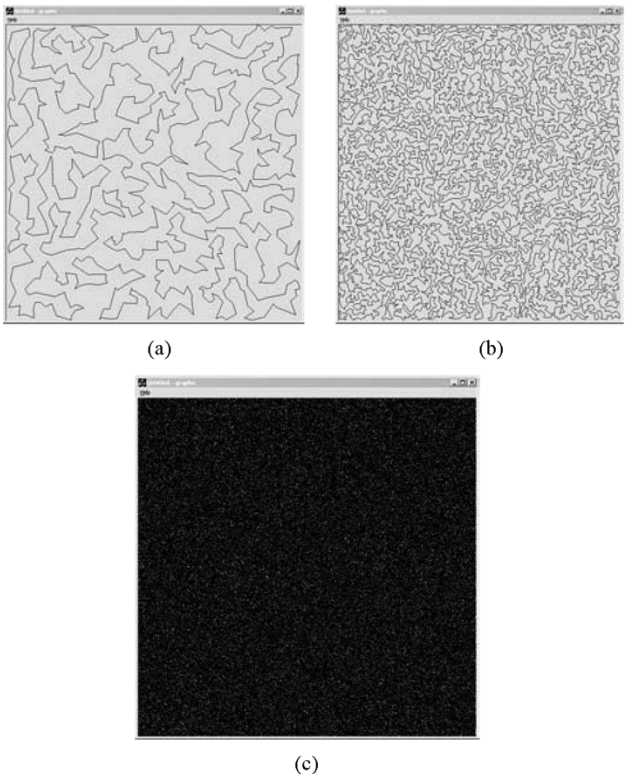


Fig. 5. Clustering divide-and-conquer TSP resulting tours for (a) 1 k, (b) 10 k, (c) 1 M cities. The clustered LK algorithm achieves a significant speedup and shows good scalability.

D. Bioinformatics—Gene Expression Data

Recently, advances in genome sequencing projects and DNA microarray technologies have been achieved. The first draft of the human genome sequence project was completed in 2001, several years earlier than expected [65], [275]. The genomic sequence data for other organizms (e.g., *Drosophila* melanogaster and *Escherichia coli*) are also abundant. DNA microarray technologies provide an effective and efficient way to measure gene expression levels of thousands of genes simultaneously under different conditions and tissues, which makes it possible to investigate gene activities from the angle of the whole genome [79], [188]. With sequences and gene expression data in hand, to investigate the functions of genes and identify their roles in the genetic process become increasingly important. Analyzes under traditional laboratory techniques are time-consuming and expensive. They fall far behind the explosively increasing generation of new data. Among the large number of computational methods used to accelerate the exploration of life science, clustering can reveal the hidden structures of biological data, and is particularly useful for helping biologists investigate and understand the activities of uncharacterized genes and proteins and further, the systematic architecture of the whole genetic network. We demonstrate the applications of clustering algorithms in bioinformatics from two aspects. The first part is based on the analysis of gene expression data generated from DNA microarray technologies. The second part describes clustering processes that directly work on linear DNA or protein sequences. The assumption is that functionally similar genes or proteins usually share similar patterns or primary sequence structures.

DNA microarray technologies generate many gene expression profiles. Currently, there are two major microarray technologies based on the nature of the attached DNA: cDNA with length varying from several hundred to thousand bases, or oligonucleotides containing 20–30 bases. For cDNA technologies, a DNA microarray consists of a solid substrate to which a large amount of cDNA clones are attached according to a certain order [79]. Fluorescently labeled cDNA, obtained from RNA samples of interest through the process of reverse transcription, is hybridized with the array. A reference sample with a different fluorescent label is also needed for comparison. Image analysis techniques are then used to measure the fluorescence of each dye, and the ratio reflects relative levels of gene expression. For a high-density oligonucleotide microarray, oligonucleotides are fixed on a chip through photolithography or solid-phase DNA synthesis [188]. In this case, absolute gene expression levels are obtained. After the normalization of the fluorescence intensities, the gene expression profiles are represented as a matrix $E = (e_{ij})$, where e_{ij} is the expression level of the ith gene in the jth condition, tissue, or experimental stage. Gene expression data analysis consists of a three-level framework based on the complexity, ranging from the investigation of single gene activities to the inference of the entire genetic network [20]. The intermediate level explores the relations and interactions between genes under different conditions, and attracts more attention currently. Generally, cluster analysis of gene expression data is composed of two aspects: clustering genes [80], [206], [260], [268], [283], [288] or clustering tissues or experiments [5], [109], [238].

Results of gene clustering may suggest that genes in the same group have similar functions, or they share the same transcriptional regulation mechanism. 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.* [80]. 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, even under a relatively small set of conditions. Herwig *et al.* developed a variant of K-means algorithm to cluster a set of 2 029 human cDNA clones and adopted mutual information as the similarity measure [230]. Tomayo *et al.* [268] made use of SOFM to cluster gene expression data and its application in hematopoietic differentiation provided new insight for further research. Graph theories based clustering algorithms, like CAST [29] and CLICK [247], showed very promising performances in tackling different types of gene expression data. Since many genes usually display more than one function, fuzzy clustering may be more effective in exposing these relations [73]. Gene expression data is also important to elucidate the genetic regulation mechanism in a cell. By examining the corresponding DNA sequences in the control regions of a cluster of co-expressed genes, we may identify potential short and consensus sequence patterns, known as motifs, and further investigate their interaction with transcriptional binding factors,

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

leading to different gene activities. Spellman *et al.* clustered 800 genes according to their expression during the yeast cell cycle [260]. Analyzes 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 [269]. 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 in the literature. A more comprehensive investigation can be found in [206].

As to another application, clustering tissues or experiments 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 [5], [109], [238]. Golub *et al.* described the restriction of traditional cancer classification methods, which are mostly dependent on morphological appearance of tumors, and divided cancer classification into two challenges: class discovery and class prediction. They utilized SOFM to discriminate two types of human acute leukemias: acute myeloid leukemia (AML) and acute lymphoblastic leukemia (ALL) [109]. According to their results, two subsets of ALL, with different origin of lineage, can be well separated. Alon *et al.* performed a two-way clustering for both tissues and genes and revealed the potential relations, represented as visualizing patterns, among them [6]. Alizadeh *et al.* demonstrated the effectiveness of molecular classification of cancers by their gene expression profiles and successfully distinguished two molecularly distinct subtypes of diffuse large B-cell lymphoma, which cause high percentage failure in clinical treatment [5]. Furthermore, 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 [238]. Other applications of clustering algorithms for tissue classification include: mixtures of multivariate Gaussian distributions [105], ellipsoidal ART [287], and graph theory-based methods [29], [247]. In most of these applications, important genes that are tightly related to the tumor types are identified according to their expression differentiation under different cancerous categories, which are in accord with our prior recognition of roles of these genes, to a large extent [5], [109]. For example, Alon *et al.* found that 5 of 20 statistically significant genes were muscle genes, and the corresponding muscle indices provided an explanation for false classifications [6].

Fig. 7 illustrates an application of hierarchical clustering and SOFM for gene expression data. This data set is on the diagnostic research of small round blue-cell tumors (SRBCT's) of childhood and consists of 83 samples from four categories, known as Burkitt lymphomas (BL), the Ewing family of tumors (EWS), neuroblastoma (NB), and rhabdomyosarcoma (RMS), and 5 non-SRBCT samples [164]. Gene expression levels of 6 567 genes were measured using cDNA microarray for each sample, 2 308 of which passed the filter and were kept for further analyzes. These genes are further ranked according to the scores calculated by some criterion functions [109]. Generally, these criterion functions attempt to seek a subset of genes that contribute most to the discrimination of different cancer types. This can be regarded as a feature selection process. However, problems like how many genes are really required, and whether these genes selected are really biologically meaningful, are still not answered satisfactorily. Hierarchical clustering was performed by the program CLUSTER and the results were visualized by the program TreeView, developed by Eisen in Stanford University. Fig. 7(a) and (b) depicts the clustering results for both the top 100 genes, selected by the Fisher scores, and the samples. Graphic visualization is achieved by associating each data point with a certain color according to the corresponding scale. Some clustering patterns are clearly displayed in the image. Fig. 7(c) depicts a 5-by-5 SOFM topology for all genes, with each cluster represented by the centroid (mean) for each feature (sample). 25 clusters are generated and the number of genes in each cluster is also indicated. The software package GeneCluster, developed by Whitehead Institute/MIT Center for Genome Research (WICGR), was used in this analysis.

Although clustering techniques have already achieved many impressive results in the analysis of gene expression data, there are still many problems that remain open. Gene expression data sets usually are characterized as

1) small set samples with high-dimensional features;
2) high redundancy;
3) inherent noise;
4) sparsity of the data.

Most of the published data sets include usually less than 20 samples for each tumor type, but with as many as thousands of gene measures [80], [109], [238], [268]. This is partly caused by the lag of experimental condition (e.g., sample collection), in contrast to the rapid advancement of microarray and sequencing technologies. In order to evaluate existing algorithms more reasonably and develop more effective new approaches, more data with enough samples or more conditional observations are needed. But from the trend of gene chip technologies, which also follows Moore's law for semiconductor chips [205], the current status will still exist for a long time. This problem is more serious in the application of gene expression data for cancer research, in which clustering algorithms are required to be capable of effectively finding potential patterns under a large number of irrelevant factors, as a result of the introduction of too many genes. At the same time, feature selection, which is also called informative gene selection in the context, also plays a very important role. Without any doubt, clustering algorithms should be feasible in both time and space complexity. Due to the nature of the manufacture process of the microarray chip, noise can be inevitably introduced into the expression data during different stages. Accordingly, clustering algorithms should have noise and outlier detection mechanisms in order to remove their effects. Furthermore, different algorithms usually form different clusters for the same data set, which is a general problem in cluster analysis. How to evaluate the quality of the generated clusters of genes, and how to choose appropriate algorithms for a specified application, are particularly crucial for gene expression data research, because sometimes, even biologists cannot identify the real patterns from the artifacts of the clustering algorithms, due to the limitations of biological

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

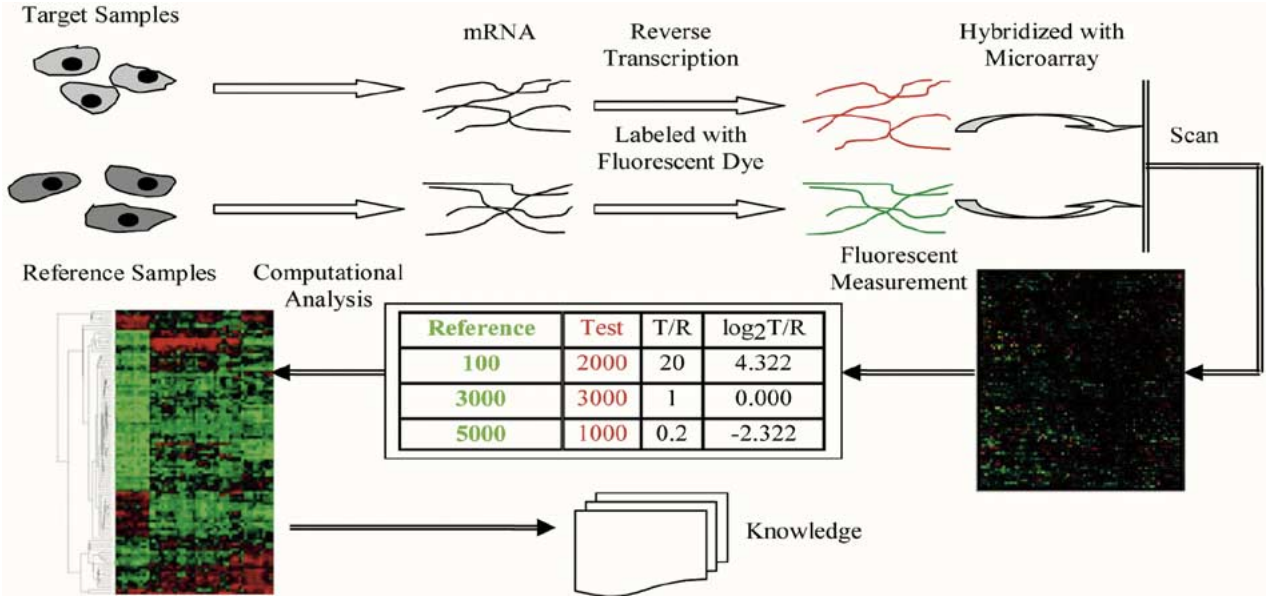


Fig. 6. Basic procedure of cDNA microarray technology [68]. Fluorescently labeled cDNAs, obtained from target and reference samples through reverse transcription, are hybridized with the microarray, which is comprised of a large amount of cDNA clones. Image analysis measures the ratio of the two dyes. Computational methods, e.g., hierarchical clustering, further disclose the relations among genes and corresponding conditions.

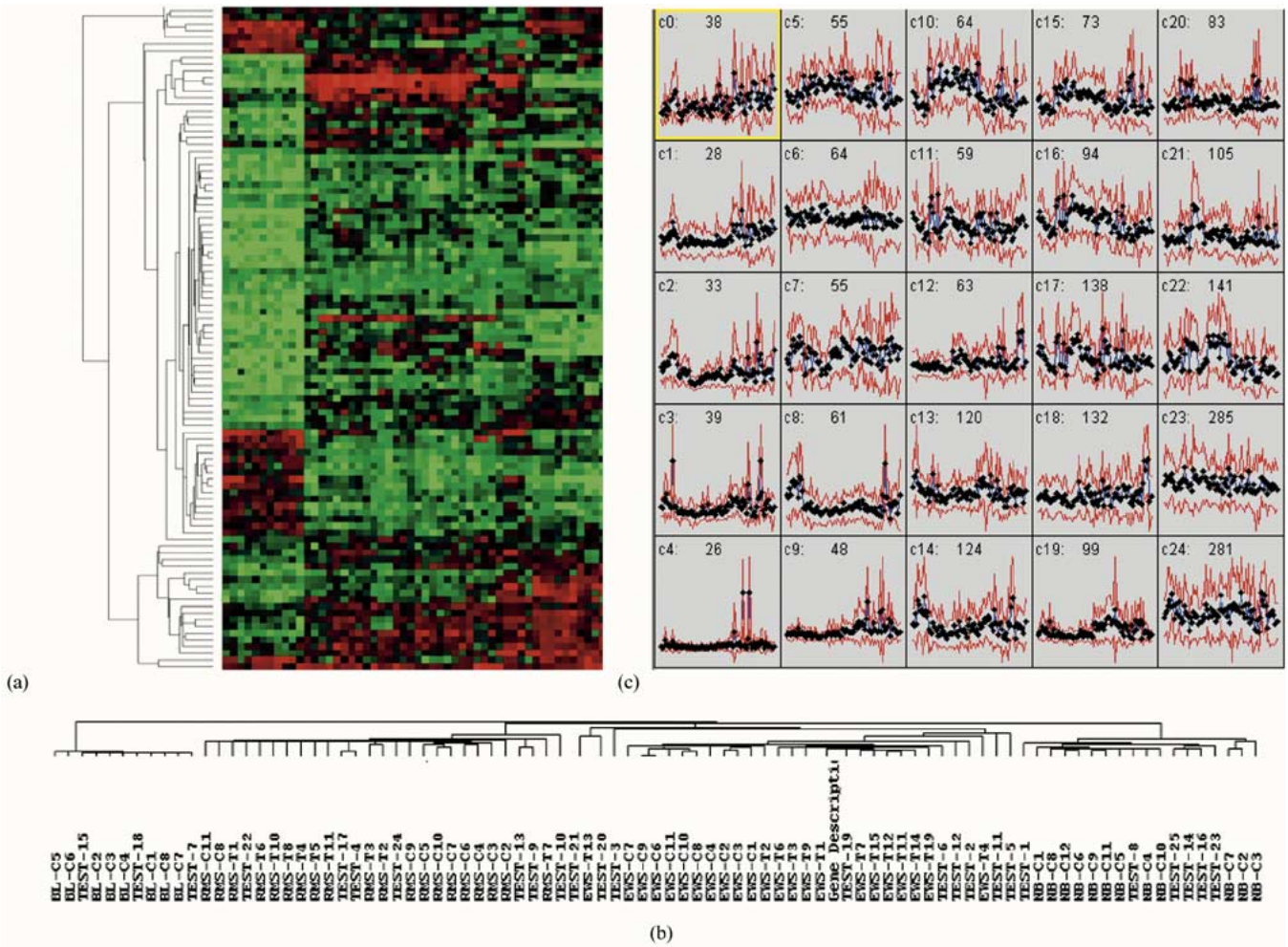


Fig. 7. Hierarchical and SOFM clustering of SRBCT's gene expression data set. (a) Hierarchical clustering result for the 100 selected genes under 83 tissue samples. The gene expression matrix is visualized through a color scale. (b) Hierarchical clustering result for the 83 tissue samples. Here, the dimension is 100 as 100 genes are selected like in (a). (c) SOFM clustering result for the 2308 genes. A 5 × 5 SOFM is used and 25 clusters are formed. Each cluster is represented by the average values.

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

knowledge. Some recent results can be accessed in [29], [247], and [291].

E. Bioinformatics—DNA or Protein Sequences Clustering

DNA (deoxyribonucleic acid) is the hereditary material existing in all living cells. A DNA molecule is a double helix consisting of two strands, each of which is a linear sequence composed of four different nucleotides—adenine, guanine, thymine, and cytosine, abbreviated as the letters A, G, T, and C, respectively. Each letter in a DNA sequence is also called a base. Proteins determine most of cells' structures, functions, properties, and regulatory mechanisms. The primary structure of a protein is also a linear and alphabetic chain with the difference that each unit represents an amino acid, which has twenty types in total. Proteins are encoded by certain segments of DNA sequences through a two-stage process (transcription and translation). These segments are known as genes or coding regions. Investigation of the relations between DNA and proteins, as well as their own functions and properties, is one of the important research directions in both genetics and bioinformatics.

The similarity between newly sequenced genes or proteins and annotated genes or proteins usually offers a cue to identify their functions. Searching corresponding databases for a new DNA or protein sequence has already become routine in genetic research. In contrast to sequence comparison and search, cluster analysis provides a more effective means to discover complicated relations among DNA and protein sequences. We summarize the following clustering applications for DNA and protein sequences:

1) function recognition of uncharacterized genes or proteins [119];
2) structure identification of large-scale DNA or protein databases [237], [257];
3) redundancy decrease of large-scale DNA or protein databases [185];
4) domain identification [83], [115];
5) expressed sequence tag (EST) clustering [49], [200].

As described in Section II-J, classical dynamic programming algorithms for global and local sequence alignment are too intensive in computational complexity. This becomes worse because of the existence of a large volume of nucleic acids and amino acids in the current DNA or protein databases, e.g., bacteria genomes are from 0.5 to 10 Mbp, fungi genomes range from 10 to 50 Mbp, while the human genome is around 3 310 Mbp [18] (Mbp means million base pairs). Thus, conventional dynamic programming algorithms are computationally infeasible. In practice, sequence comparison or proximity measure is achieved via some heuristics. Well-known examples include BLAST and FASTA with many variants [10], [11], [224]. The key idea of these methods is to identify regions that may have potentially high matches, with a list of prespecified high-scoring words, at an early stage. Therefore, further search only needs to focus on these regions with expensive but accurate algorithms. Recognizing the benefit coming from the separation of word matching and sequence alignment to computational burden reduction, Miller, Gurd, and Brass described three algorithms focusing on specific problems [199]. The implementation of the scheme for large database vs. database comparison exhibits an apparent improvement in computation time. Kent and Zahler designed a three-pass algorithm, called wobble aware bulk aligner (WABA) [162], for aligning large-scale genomic sequences of different species, which employs a seven-state pairwise hidden Markov model [78] for more effective alignments. In [201], Miller summarized the current research status of genomic sequence comparison and suggested valuable directions for further research efforts.

Many clustering techniques have been applied to organize DNA or protein sequence data. Some directly operate on a proximity measure; some are based on feature extraction, while others are constructed on statistical models. Somervuo and Kohonen illustrated an application of SOFM to cluster protein sequences in SWISSPROT database [257]. FASTA was used to calculate the sequence similarity. The resulting two-dimensional SOFM provides a visualized representation of the relations within the entire sequence database. Based on the similarity measure of gapped BLAST, Sasson *et al.* utilized an agglomerative hierarchical clustering paradigm to cluster all protein sequences in SWISSPROT [237]. The effects of four merging rules, different from the interpretation of cluster centers, on the resulting protein clusters were examined. The advantages as well as the potential risk of the concept, transitivity, were also elucidated in the paper. According to the transitivity relation, two sequences that do not show high sequence similarity by virtue of direct comparison, may be homologous (having a common ancestor) if there exists an intermediate sequence similar to both of them. This makes it possible to detect remote homologues that can not be observed by similarity comparison. However, unrelated sequences may be clustered together due to the effects of these intermediate sequences [237]. Bolten *et al.* addressed the problem with the construction a directed graph, in which each protein sequence corresponds to a vertex and edges are weighted based on the alignment score between two sequences and self alignment score of each sequence [41]. Clusters were formed through the search of strongly connected components (SCCs), each of which is a maximal subset of vertices and for each pair of vertices u and v in the subset, there exist two directed paths from u to v and vice versa. A minimum normalized cut algorithm for detecting protein families and a minimum spanning tree (MST) application for seeking domain information were presented in [1] and [115], respectively. In contrast with the aforementioned proximity-based methods, Guralnik and Karypis transformed protein or DNA sequences into a new feature space, based on the detected subpatterns working as the sequence features, and clustered with the K-means algorithm [119]. The method is immune from all-against-all expensive sequence comparison and suitable for analyzing large-scale databases. Krogh demonstrated the power of hidden Markov models (HMMs) in biological sequences modeling and clustering of protein families [177]. Fig. 8 depicts a typical structure of HMM, in which match states (abbreviated with letter M), insert states (I) and delete states (D) are represented as rectangles, diamonds, and circles, respectively [78], [177]. These states correspond to substitution, insertion, and deletion in edit operations. For convenience, a begin state and an end state are added to the

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

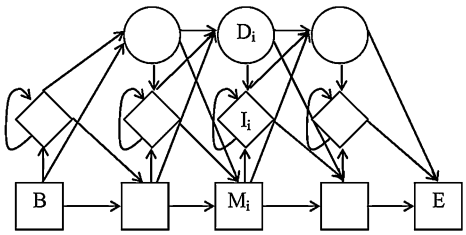


Fig. 8. HMM architecture [177]. There are three different states, match (M), insert (I), and delete (D), corresponding to substitution, insertion, and deletion operation, respectively. A begin (B) and end (E) state are also introduced to represent the start and end of the process. This process goes through a series of states according to the transition probability, and emits either 4-letter nucleotide or 20-letter amino acid alphabet based on the emission probability.

model, denoted by letter B and E. Letters, either from the 4-letter nucleotide alphabet or from 20-letter amino acid alphabet, 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 similar to single HMM case. An example for clustering subfamilies of 628 globins shows the encouraging results. Further discussion can be found in [78] and [145].

IV. Conclusion

As an important tool for data exploration, cluster analysis examines unlabeled data, by either constructing a hierarchical structure, or forming a set of groups according to a prespecified number. This process includes a series of steps, ranging from preprocessing and algorithm development, to solution validity and evaluation. Each of them is tightly related to each other and exerts great challenges to the scientific disciplines. Here, we place the focus on the clustering algorithms and review a wide variety of approaches appearing in the literature. These algorithms evolve from different research communities, aim to solve different problems, and have their own pros and cons. 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. These problems have already attracted and will continue to attract intensive efforts from broad disciplines. We summarize and conclude the survey with listing some important issues and research trends for cluster algorithms.

1) There is no clustering algorithm that can be universally used to solve all problems. Usually, algorithms are designed with certain assumptions and favor some type of biases. In this sense, it is not accurate to say "best" in the context of clustering algorithms, although some comparisons are possible. These comparisons are mostly based on some specific applications, under certain conditions, and the results may become quite different if the conditions change.
2) New technology has generated more complex and challenging tasks, requiring more powerful clustering algorithms. The following properties are important to the efficiency and effectiveness of a novel algorithm.
   I) generate arbitrary shapes of clusters rather than be confined to some particular shape;
   II) handle large volume of data as well as high-dimensional features with acceptable time and storage complexities;
   III) detect and remove possible outliers and noise;
   IV) decrease the reliance of algorithms on users-dependent parameters;
   V) have the capability of dealing with newly occurring data without relearning from the scratch;
   VI) be immune to the effects of order of input patterns;
   VII) provide some insight for the number of potential clusters without prior knowledge;
   VIII) show good data visualization and provide users with results that can simplify further analysis;
   IX) be capable of handling both numerical and nominal data or be easily adaptable to some other data type.

   Of course, some more detailed requirements for specific applications will affect these properties.
3) At the preprocessing and post-processing phase, feature selection/extraction (as well as standardization and normalization) and cluster validation are as important as the clustering algorithms. Choosing appropriate and meaningful features can greatly reduce the burden of subsequent designs and result evaluations reflect the degree of confidence to which we can rely on the generated clusters. Unfortunately, both processes lack universal guidance. Ultimately, the tradeoff among different criteria and methods is still dependent on the applications themselves.

Acknowledgment

The authors 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. They would also like to thank S. Mulder for the part on the traveling salesman problem and also acknowledge extensive comments from the reviewers and the anonymous associate editor.

References

[1] F. Abascal and A. Valencia, "Clustering of proximal sequence space for the identification of protein families," *Bioinformatics*, vol. 18, pp. 908–921, 2002.
[2] C. Aggarwal and P. Yu, "Redefining clustering for high-dimensional applications," *IEEE Trans. Knowl. Data Eng.*, vol. 14, no. 2, pp. 210–225, Feb. 2002.
[3] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, "Automatic subspace clustering of high dimensional data for data mining applications," in *Proc. ACM SIGMOD Int. Conf. Management of Data*, 1998, pp. 94–105.
[4] H. Akaike, "A new look at the statistical model identification," *IEEE Trans. Autom. Control*, vol. AC-19, no. 6, pp. 716–722, Dec. 1974.
[5] A. Alizadeh *et al.*, "Distinct types of diffuse large B-cell Lymphoma identified by gene expression profiling," *Nature*, vol. 403, pp. 503–511, 2000.
[6] U. Alon, N. Barkai, D. Notterman, K. Gish, S. Ybarra, D. Mack, and A. Levine, "Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays," *Proc. Nat. Acad. Sci. USA*, pp. 6745–6750, 1999.

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

[7] C. Alpert and A. Kahng, "Multi-way partitioning via spacefilling curves and dynamic programming," in *Proc. 31st ACM/IEEE Design Automation Conf.*, 1994, pp. 652–657.
[8] ——, "Recent directions in netlist partitioning: A survey," *VLSI J.*, vol. 19, pp. 1–81, 1995.
[9] K. Al-Sultan, "A Tabu search approach to the clustering problem," *Pattern Recognit.*, vol. 28, no. 9, pp. 1443–1451, 1995.
[10] S. Altschul *et al.*, "Gapped BLAST and PSI-BLAST: A new generation of protein database search programs," *Nucleic Acids Res.*, vol. 25, pp. 3389–3402, 1997.
[11] S. Altschul *et al.*, "Basic local alignment search tool," *J. Molec. Biol.*, vol. 215, pp. 403–410, 1990.
[12] G. Anagnostopoulos and M. Georgiopoulos, "Hypersphere ART and ARTMAP for unsupervised and supervised incremental learning," in *Proc. IEEE-INNS-ENNS Int. Joint Conf. Neural Networks (IJCNN'00)*, vol. 6, Como, Italy, pp. 59–64.
[13] ——, "Ellipsoid ART and ARTMAP for incremental unsupervised and supervised learning," in *Proc. IEEE-INNS-ENNS Int. Joint Conf. Neural Networks (IJCNN'01)*, vol. 2, Washington, DC, 2001, pp. 1221–1226.
[14] M. Anderberg, *Cluster Analysis for Applications*. New York: Academic, 1973.
[15] G. Babu and M. Murty, "A near-optimal initial seed value selection in K-means algorithm using a genetic algorithm," *Pattern Recognit. Lett.*, vol. 14, no. 10, pp. 763–769, 1993.
[16] ——, "Clustering with evolution strategies," *Pattern Recognit.*, vol. 27, no. 2, pp. 321–329, 1994.
[17] E. Backer and A. Jain, "A clustering performance measure based on fuzzy set decomposition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-3, no. 1, pp. 66–75, Jan. 1981.
[18] P. Baldi and S. Brunak, *Bioinformatics: The Machine Learning Approach*, 2nd ed. Cambridge, MA: MIT Press, 2001.
[19] P. Baldi and K. Hornik, "Neural networks and principal component analysis: Learning from examples without local minima," *Neural Netw.*, vol. 2, pp. 53–58, 1989.
[20] P. Baldi and A. Long, "A Bayesian framework for the analysis of microarray expression data: Regularized t-test and statistical inferences of gene changes," *Bioinformatics*, vol. 17, pp. 509–519, 2001.
[21] G. Ball and D. Hall, "A clustering technique for summarizing multivariate data," *Behav. Sci.*, vol. 12, pp. 153–155, 1967.
[22] S. Bandyopadhyay and U. Maulik, "Nonparametric genetic clustering: Comparison of validity indices," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 31, no. 1, pp. 120–125, Feb. 2001.
[23] A. Baraldi and E. Alpaydin, "Constructive feedforward ART clustering networks—Part I and II," *IEEE Trans. Neural Netw.*, vol. 13, no. 3, pp. 645–677, May 2002.
[24] A. Baraldi and P. Blonda, "A survey of fuzzy clustering algorithms for pattern recognition—Part I and II," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 29, no. 6, pp. 778–801, Dec. 1999.
[25] A. Baraldi and L. Schenato, "Soft-to-hard model transition in clustering: A review,", Tech. Rep. TR-99-010, 1999.
[26] D. Barbará and P. Chen, "Using the fractal dimension to cluster datasets," in *Proc. 6th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, 2000, pp. 260–264.
[27] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," in *Advances in Neural Information Processing Systems*, T. G. Dietterich, S. Becker, and Z. Ghahramani, Eds. Cambridge, MA: MIT Press, 2002, vol. 14.
[28] R. Bellman, *Adaptive Control Processes: A Guided Tour*. Princeton, NJ: Princeton Univ. Press, 1961.
[29] A. Ben-Dor, R. Shamir, and Z. Yakhini, "Clustering gene expression patterns," *J. Comput. Biol.*, vol. 6, pp. 281–297, 1999.
[30] Y. Bengio, "Markovian models for sequential data," *Neural Comput. Surv.*, vol. 2, pp. 129–162, 1999.
[31] A. Ben-Hur, D. Horn, H. Siegelmann, and V. Vapnik, "Support vector clustering," *J. Mach. Learn. Res.*, vol. 2, pp. 125–137, 2001.
[32] ——, "A support vector clustering method," in *Proc. Int. Conf. Pattern Recognition*, vol. 2, 2000, pp. 2724–2727.
[33] P. Berkhin. (2001) Survey of clustering data mining techniques. [Online]. Available: http://www.accrue.com/products/rp_cluster_review.pdf http://citeseer.nj.nec.com/berkhin02survey.html
[34] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, "When is nearest neighbor meaningful," in *Proc. 7th Int. Conf. Database Theory*, 1999, pp. 217–235.
[35] J. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*. New York: Plenum, 1981.
[36] J. Bezdek and R. Hathaway, "Numerical convergence and interpretation of the fuzzy c-shells clustering algorithms," *IEEE Trans. Neural Netw.*, vol. 3, no. 5, pp. 787–793, Sep. 1992.
[37] J. Bezdek and N. Pal, "Some new indexes of cluster validity," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 28, no. 3, pp. 301–315, Jun. 1998.
[38] C. Bishop, *Neural Networks for Pattern Recognition*. New York: Oxford Univ. Press, 1995.
[39] L. Bobrowski and J. Bezdek, "c-Means clustering with the l_1 and l_∞ norms," *IEEE Trans. Syst., Man, Cybern.*, vol. 21, no. 3, pp. 545–554, May-Jun. 1991.
[40] H. Bock, "Probabilistic models in cluster analysis," *Comput. Statist. Data Anal.*, vol. 23, pp. 5–28, 1996.
[41] E. Bolten, A. Sxhliep, S. Schneckener, D. Schomburg, and R. Schrader, "Clustering protein sequences—Structure prediction by transitive homology," *Bioinformatics*, vol. 17, pp. 935–941, 2001.
[42] N. Boujemaa, "Generalized competitive clustering for image segmentation," in *Proc. 19th Int. Meeting North American Fuzzy Information Processing Soc. (NAFIPS'00)*, Atlanta, GA, 2000, pp. 133–137.
[43] P. Bradley and U. Fayyad, "Refining initial points for K-means clustering," in *Proc. 15th Int. Conf. Machine Learning*, 1998, pp. 91–99.
[44] P. Bradley, U. Fayyad, and C. Reina, "Scaling clustering algorithms to large databases," in *Proc. 4th Int. Conf. Knowledge Discovery and Data Mining (KDD'98)*, 1998, pp. 9–15.
[45] ——, "Clustering very large databases using EM mixture models," in *Proc. 15th Int. Conf. Pattern Recognition*, vol. 2, 2000, pp. 76–80.
[46] ——, "Clustering very large databases using EM mixture models," in *Proc. 15th Int. Conf. Pattern Recognition*, vol. 2, 2000, pp. 76–80.
[47] D. Brown and C. Huntley, "A practical application of simulated annealing to clustering," *Pattern Recognit.*, vol. 25, no. 4, pp. 401–412, 1992.
[48] C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining Knowl. Discov.*, vol. 2, pp. 121–167, 1998.
[49] J. Burke, D. Davison, and W. Hide, "d2_Cluster: A validated method for clustering EST and full-length cDNA sequences," *Genome Res.*, vol. 9, pp. 1135–1142, 1999.
[50] I. Cadez, S. Gaffney, and P. Smyth, "A general probabilistic framework for clustering individuals and objects," in *Proc. 6th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, 2000, pp. 140–149.
[51] G. Carpenter and S. Grossberg, "A massively parallel architecture for a self-organizing neural pattern recognition machine," *Comput. Vis. Graph. Image Process.*, vol. 37, pp. 54–115, 1987.
[52] ——, "ART2: Self-organization of stable category recognition codes for analog input patterns," *Appl. Opt.*, vol. 26, no. 23, pp. 4919–4930, 1987.
[53] ——, "The ART of adaptive pattern recognition by a self-organizing neural network," *IEEE Computer*, vol. 21, no. 3, pp. 77–88, Mar. 1988.
[54] ——, "ART3: Hierarchical search using chemical transmitters in self-organizing pattern recognition architectures," *Neural Netw.*, vol. 3, no. 23, pp. 129–152, 1990.
[55] G. Carpenter, S. Grossberg, N. Markuzon, J. Reynolds, and D. Rosen, "Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps," *IEEE Trans. Neural Netw.*, vol. 3, no. 5, pp. 698–713, 1992.
[56] G. Carpenter, S. Grossberg, and J. Reynolds, "ARTMAP: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network," *Neural Netw.*, vol. 4, no. 5, pp. 169–181, 1991.
[57] G. Carpenter, S. Grossberg, and D. Rosen, "Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system," *Neural Netw.*, vol. 4, pp. 759–771, 1991.
[58] G. Celeux and G. Govaert, "A classification EM algorithm for clustering and two stochastic versions," *Comput. Statist. Data Anal.*, vol. 14, pp. 315–332, 1992.
[59] P. Cheeseman and J. Stutz, "Bayesian classification (AutoClass): Theory and results," in *Advances in Knowledge Discovery and Data Mining*, U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, Eds. Menlo Park, CA: AAAI Press, 1996, pp. 153–180.
[60] V. Cherkassky and F. Mulier, *Learning From Data: Concepts, Theory, and Methods*. New York: Wiley, 1998.
[61] J. Cherng and M. Lo, "A hypergraph based clustering algorithm for spatial data sets," in *Proc. IEEE Int. Conf. Data Mining (ICDM'01)*, 2001, pp. 83–90.
[62] J. Chiang and P. Hao, "A new kernel-based fuzzy clustering approach: Support vector clustering with cell growing," *IEEE Trans. Fuzzy Syst.*, vol. 11, no. 4, pp. 518–527, Aug. 2003.
[63] C. Chinrungrueng and C. Séquin, "Optimal adaptive K-means algorithm with dynamic adjustment of learning rate," *IEEE Trans. Neural Netw.*, vol. 6, no. 1, pp. 157–169, Jan. 1995.

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

[64] S. Chu and J. Roddick, "A clustering algorithm using the Tabu search approach with simulated annealing," in *Data Mining II—Proceedings of Second International Conference on Data Mining Methods and Databases*, N. Ebecken and C. Brebbia, Eds, Cambridge, U.K., 2000, pp. 515–523.

[65] I. H. G. S. Consortium, "Initial sequencing and analysis of the human genome," *Nature*, vol. 409, pp. 860–921, 2001.

[66] J. Corchado and C. Fyfe, "A comparison of kernel methods for instantiating case based reasoning systems," *Comput. Inf. Syst.*, vol. 7, pp. 29–42, 2000.

[67] M. Cowgill, R. Harvey, and L. Watson, "A genetic algorithm approach to cluster analysis," *Comput. Math. Appl.*, vol. 37, pp. 99–108, 1999.

[68] C. Cummings and D. Relman, "Using DNA microarray to study host-microbe interactions," *Genomics*, vol. 6, no. 5, pp. 513–525, 2000.

[69] E. Dahlhaus, "Parallel algorithms for hierarchical clustering and applications to split decomposition and parity graph recognition," *J. Algorithms*, vol. 36, no. 2, pp. 205–240, 2000.

[70] R. Davé, "Adaptive fuzzy c-shells clustering and detection of ellipses," *IEEE Trans. Neural Netw.*, vol. 3, no. 5, pp. 643–662, Sep. 1992.

[71] R. Davé and R. Krishnapuram, "Robust clustering methods: A unified view," *IEEE Trans. Fuzzy Syst.*, vol. 5, no. 2, pp. 270–293, May 1997.

[72] M. Delgado, A. Skármeta, and H. Barberá, "A Tabu search approach to the fuzzy clustering problem," in *Proc. 6th IEEE Int. Conf. Fuzzy Systems*, vol. 1, 1997, pp. 125–130.

[73] D. Dembélé and P. Kastner, "Fuzzy c-means method for clustering microarray data," *Bioinformatics*, vol. 19, no. 8, pp. 973–980, 2003.

[74] *Handbook of Pattern Recognition and Computer Vision*, C. Chen, L. Pau, and P. Wang, Eds., World Scientific, Singapore, 1993, pp. 3–32. R. Dubes, "Cluster analysis and related issue".

[75] R. Duda, P. Hart, and D. Stork, *Pattern Classification*, 2nd ed. New York: Wiley, 2001.

[76] J. Dunn, "A fuzzy relative of the ISODATA process and its use in detecting compact well separated clusters," *J. Cybern.*, vol. 3, no. 3, pp. 32–57, 1974.

[77] B. Duran and P. Odell, *Cluster Analysis: A Survey*. New York: Springer-Verlag, 1974.

[78] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison, *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge, U.K.: Cambridge Univ. Press, 1998.

[79] M. Eisen and P. Brown, "DNA arrays for analysis of gene expression," *Methods Enzymol.*, vol. 303, pp. 179–205, 1999.

[80] M. Eisen, P. Spellman, P. Brown, and D. Botstein, "Cluster analysis and display of genome-wide expression patterns," in *Proc. Nat. Acad. Sci. USA*, vol. 95, 1998, pp. 14 863–14 868.

[81] Y. El-Sonbaty and M. Ismail, "Fuzzy clustering for symbolic data," *IEEE Trans. Fuzzy Syst.*, vol. 6, no. 2, pp. 195–204, May 1998.

[82] T. Eltoft and R. deFigueiredo, "A new neural network for cluster-detection-and-labeling," *IEEE Trans. Neural Netw.*, vol. 9, no. 5, pp. 1021–1035, Sep. 1998.

[83] A. Enright and C. Ouzounis, "GeneRAGE: A robust algorithm for sequence clustering and domain detection," *Bioinformatics*, vol. 16, pp. 451–457, 2000.

[84] S. Eschrich, J. Ke, L. Hall, and D. Goldgof, "Fast accurate fuzzy clustering through data reduction," *IEEE Trans. Fuzzy Syst.*, vol. 11, no. 2, pp. 262–270, Apr. 2003.

[85] M. Ester, H. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. 2nd Int. Conf. Knowledge Discovery and Data Mining (KDD'96)*, 1996, pp. 226–231.

[86] V. Estivill-Castro and I. Lee, "AMOEBA: Hierarchical clustering based on spatial proximity using Delaunay diagram," in *Proc. 9th Int. Symp. Spatial Data Handling (SDH'99)*, Beijing, China, 1999, pp. 7a.26–7a.41.

[87] V. Estivill-Castro and J. Yang, "A fast and robust general purpose clustering algorithm," in *Proc. 6th Pacific Rim Int. Conf. Artificial Intelligence (PRICAI'00)*, R. Mizoguchi and J. Slaney, Eds., Melbourne, Australia, 2000, pp. 208–218.

[88] B. Everitt, S. Landau, and M. Leese, *Cluster Analysis*. London: Arnold, 2001.

[89] D. Fasulo, "An analysis of recent work on clustering algorithms," Dept. Comput. Sci. Eng., Univ. Washington, Seattle, WA, Tech. Rep. 01-03-02, 1999.

[90] M. Figueiredo and A. Jain, "Unsupervised learning of finite mixture models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 3, pp. 381–396, Mar. 2002.

[91] D. Fisher, "Knowledge acquisition via incremental conceptual clustering," *Mach. Learn.*, vol. 2, pp. 139–172, 1987.

[92] R. Fisher, "The use of multiple measurements in taxonomic problems," *Annu. Eugenics*, pt. II, vol. 7, pp. 179–188, 1936.

[93] D. Fogel, "An introduction to simulated evolutionary optimization," *IEEE Trans. Neural Netw.*, vol. 5, no. 1, pp. 3–14, Jan. 1994.

[94] E. Forgy, "Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications," *Biometrics*, vol. 21, pp. 768–780, 1965.

[95] C. Fraley and A. Raftery, "MCLUST: Software for model-based cluster analysis," *J. Classificat.*, vol. 16, pp. 297–306, 1999.

[96] ——, "Model-Based clustering, discriminant analysis, and density estimation," *J. Amer. Statist. Assoc.*, vol. 97, pp. 611–631, 2002.

[97] J. Friedman, "Exploratory projection pursuit," *J. Amer. Statist. Assoc.*, vol. 82, pp. 249–266, 1987.

[98] H. Frigui and R. Krishnapuram, "A robust competitive clustering algorithm with applications in computer vision," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 5, pp. 450–465, May 1999.

[99] B. Fritzke. (1997) Some competitive learning methods. [Online]. Available: http://www.neuroinformatik.ruhr-uni-bochum.de/ini/VDM/research/gsn/JavaPaper

[100] B. Gabrys and A. Bargiela, "General fuzzy min-max neural network for clustering and classification," *IEEE Trans. Neural Netw.*, vol. 11, no. 3, pp. 769–783, May 2000.

[101] V. Ganti, R. Ramakrishnan, J. Gehrke, A. Powell, and J. French, "Clustering large datasets in arbitrary metric spaces," in *Proc. 15th Int. Conf. Data Engineering*, 1999, pp. 502–511.

[102] I. Gath and A. Geva, "Unsupervised optimal fuzzy clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 11, no. 7, pp. 773–781, Jul. 1989.

[103] *GenBank Release Notes 144.0*.

[104] A. Geva, "Hierarchical unsupervised fuzzy clustering," *IEEE Trans. Fuzzy Syst.*, vol. 7, no. 6, pp. 723–733, Dec. 1999.

[105] D. Ghosh and A. Chinnaiyan, "Mixture modeling of gene expression data from microarray experiments," *Bioinformatics*, vol. 18, no. 2, pp. 275–286, 2002.

[106] A. Ghozeil and D. Fogel, "Discovering patterns in spatial data using evolutionary programming," in *Proc. 1st Annu. Conf. Genetic Programming*, 1996, pp. 512–520.

[107] M. Girolami, "Mercer kernel based clustering in feature space," *IEEE Trans. Neural Netw.*, vol. 13, no. 3, pp. 780–784, May 2002.

[108] F. Glover, "Tabu search, part I," *ORSA J. Comput.*, vol. 1, no. 3, pp. 190–206, 1989.

[109] 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*, vol. 286, pp. 531–537, 1999.

[110] A. Gordon, "Cluster validation," in *Data Science, Classification, and Related Methods*, C. Hayashi, N. Ohsumi, K. Yajima, Y. Tanaka, H. Bock, and Y. Bada, Eds. New York: Springer-Verlag, 1998, pp. 22–39.

[111] ——, *Classification*, 2nd ed. London, U.K.: Chapman & Hall, 1999.

[112] J. Gower, "A general coefficient of similarity and some of its properties," *Biometrics*, vol. 27, pp. 857–872, 1971.

[113] S. Grossberg, "Adaptive pattern recognition and universal encoding II: Feedback, expectation, olfaction, and illusions," *Biol. Cybern.*, vol. 23, pp. 187–202, 1976.

[114] P. Grünwald, P. Kontkanen, P. Myllymäki, T. Silander, and H. Tirri, "Minimum encoding approaches for predictive modeling," in *Proc. 14th Int. Conf. Uncertainty in AI (UAI'98)*, 1998, pp. 183–192.

[115] X. Guan and L. Du, "Domain identification by clustering sequence alignments," *Bioinformatics*, vol. 14, pp. 783–788, 1998.

[116] S. Guha, R. Rastogi, and K. Shim, "CURE: An efficient clustering algorithm for large databases," in *Proc. ACM SIGMOD Int. Conf. Management of Data*, 1998, pp. 73–84.

[117] ——, "ROCK: A robust clustering algorithm for categorical attributes," *Inf. Syst.*, vol. 25, no. 5, pp. 345–366, 2000.

[118] S. Gupata, K. Rao, and V. Bhatnagar, "K-means clustering algorithm for categorical attributes," in *Proc. 1st Int. Conf. Data Warehousing and Knowledge Discovery (DaWaK'99)*, Florence, Italy, 1999, pp. 203–208.

[119] V. Guralnik and G. Karypis, "A scalable algorithm for clustering sequential data," in *Proc. 1st IEEE Int. Conf. Data Mining (ICDM'01)*, 2001, pp. 179–186.

[120] D. Gusfield, *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*. Cambridge, U.K.: Cambridge Univ. Press, 1997.

[121] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, "Cluster validity methods: Part I & II," *SIGMOD Record*, vol. 31, no. 2–3, 2002.

[122] L. Hall, I. Özyurt, and J. Bezdek, "Clustering with a genetically optimized approach," *IEEE Trans. Evol. Comput.*, vol. 3, no. 2, pp. 103–112, 1999.

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

[123] R. Hammah and J. Curran, "Validity measures for the fuzzy cluster analysis of orientations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 12, pp. 1467–1472, Dec. 2000.
[124] P. Hansen and B. Jaumard, "Cluster analysis and mathematical programming," *Math. Program.*, vol. 79, pp. 191–215, 1997.
[125] P. Hansen and N. Mladenoviæ, "J-means: A new local search heuristic for minimum sum of squares clustering," *Pattern Recognit.*, vol. 34, pp. 405–413, 2001.
[126] F. Harary, *Graph Theory*. Reading, MA: Addison-Wesley, 1969.
[127] J. Hartigan, *Clustering Algorithms*. New York: Wiley, 1975.
[128] E. Hartuv and R. Shamir, "A clustering algorithm based on graph connectivity," *Inf. Process. Lett.*, vol. 76, pp. 175–181, 2000.
[129] R. Hathaway and J. Bezdek, "Fuzzy c-means clustering of incomplete data," *IEEE Trans. Syst., Man, Cybern.*, vol. 31, no. 5, pp. 735–744, 2001.
[130] R. Hathaway, J. Bezdek, and Y. Hu, "Generalized fuzzy c-means clustering strategies using L_p norm distances," *IEEE Trans. Fuzzy Syst.*, vol. 8, no. 5, pp. 576–582, Oct. 2000.
[131] B. Hay, G. Wets, and K. Vanhoof, "Clustering navigation patterns on a website using a sequence alignment method," in *Proc. Intelligent Techniques for Web Personalization: 17th Int. Joint Conf. Artificial Intelligence*, vol. s.l, 2001, pp. 1–6, 200.
[132] S. Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 1999.
[133] Q. He, "A review of clustering algorithms as applied to IR," Univ. Illinois at Urbana-Champaign, Tech. Rep. UIUCLIS-1999/6+IRG, 1999.
[134] M. Healy, T. Caudell, and S. Smith, "A neural architecture for pattern sequence verification through inferencing," *IEEE Trans. Neural Netw.*, vol. 4, no. 1, pp. 9–20, Jan. 1993.
[135] A. Hinneburg and D. Keim, "An efficient approach to clustering in large multimedia databases with noise," in *Proc. 4th Int. Conf. Knowledge Discovery and Data Mining (KDD'98)*, 1998, pp. 58–65.
[136] ——, "Optimal grid-clustering: Toward breaking the curse of dimensionality in high-dimensional clustering," in *Proc. 25th VLDB Conf.*, 1999, pp. 506–517.
[137] F. Hoeppner, "Fuzzy shell clustering algorithms in image processing: Fuzzy c-rectangular and 2-rectangular shells," *IEEE Trans. Fuzzy Syst.*, vol. 5, no. 4, pp. 599–613, Nov. 1997.
[138] J. Hoey, "Clustering contextual facial display sequences," in *Proc. 5th IEEE Int. Conf. Automatic Face and Gesture Recognition (FGR'02)*, 2002, pp. 354–359.
[139] T. Hofmann and J. Buhmann, "Pairwise data clustering by deterministic annealing," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 1, pp. 1–14, Jan. 1997.
[140] J. Holland, *Adaption in Natural and Artificial Systems*. Ann Arbor, MI: Univ. Michigan Press, 1975.
[141] F. Höppner, F. Klawonn, and R. Kruse, *Fuzzy Cluster Analysis: Methods for Classification, Data Analysis, and Image Recognition*. New York: Wiley, 1999.
[142] Z. Huang, "Extensions to the K-means algorithm for clustering large data sets with categorical values," *Data Mining Knowl. Discov.*, vol. 2, pp. 283–304, 1998.
[143] J. Huang, M. Georgiopoulos, and G. Heileman, "Fuzzy ART properties," *Neural Netw.*, vol. 8, no. 2, pp. 203–213, 1995.
[144] P. Huber, "Projection pursuit," *Ann. Statist.*, vol. 13, no. 2, pp. 435–475, 1985.
[145] R. Hughey and A. Krogh, "Hidden Markov models for sequence analysis: Extension and analysis of the basic method," *CABIOS*, vol. 12, no. 2, pp. 95–107, 1996.
[146] M. Hung and D. Yang, "An efficient fuzzy c-means clustering algorithm," in *Proc. IEEE Int. Conf. Data Mining*, 2001, pp. 225–232.
[147] L. Hunt and J. Jorgensen, "Mixture model clustering using the MULTIMIX program," *Australia and New Zealand J. Statist.*, vol. 41, pp. 153–171, 1999.
[148] J. Hwang, J. Vlontzos, and S. Kung, "A systolic neural network architecture for hidden Markov models," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 37, no. 12, pp. 1967–1979, Dec. 1989.
[149] A. Hyvärinen, "Survey of independent component analysis," *Neural Comput. Surv.*, vol. 2, pp. 94–128, 1999.
[150] A. Jain and R. Dubes, *Algorithms for Clustering Data*. Englewood Cliffs, NJ: Prentice-Hall, 1988.
[151] A. Jain, R. Duin, and J. Mao, "Statistical pattern recognition: A review," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 1, pp. 4–37, 2000.
[152] A. Jain, M. Murty, and P. Flynn, "Data clustering: A review," *ACM Comput. Surv.*, vol. 31, no. 3, pp. 264–323, 1999.
[153] D. Jiang, C. Tang, and A. Zhang, "Cluster analysis for gene expression data: A survey," *IEEE Trans. Knowl. Data Eng.*, vol. 16, no. 11, pp. 1370–1386, Nov. 2004.
[154] C. Jutten and J. Herault, "Blind separation of sources, Part I: An adaptive algorithms based on neuromimetic architecture," *Signal Process.*, vol. 24, no. 1, pp. 1–10, 1991.
[155] T. Kanungo, D. Mount, N. Netanyahu, C. Piatko, R. Silverman, and A. Wu, "An efficient K-means clustering algorithm: Analysis and implementation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 881–892, Jul. 2000.
[156] N. Karayiannis, "A methodology for construction fuzzy algorithms for learning vector quantization," *IEEE Trans. Neural Netw.*, vol. 8, no. 3, pp. 505–518, May 1997.
[157] N. Karayiannis, J. Bezdek, N. Pal, R. Hathaway, and P. Pai, "Repairs to GLVQ: A new family of competitive learning schemes," *IEEE Trans. Neural Netw.*, vol. 7, no. 5, pp. 1062–1071, Sep. 1996.
[158] J. Karhunen, E. Oja, L. Wang, R. Vigário, and J. Joutsensalo, "A class of neural networks for independent component analysis," *IEEE Trans. Neural Netw.*, vol. 8, no. 3, pp. 486–504, May 1997.
[159] G. Karypis, E. Han, and V. Kumar, "Chameleon: Hierarchical clustering using dynamic modeling," *IEEE Computer*, vol. 32, no. 8, pp. 68–75, Aug. 1999.
[160] R. Kathari and D. Pitts, "On finding the number of clusters," *Pattern Recognit. Lett.*, vol. 20, pp. 405–416, 1999.
[161] L. Kaufman and P. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*: Wiley, 1990.
[162] W. Kent and A. Zahler, "Conservation, regulation, synteny, and introns in a large-scale C. Briggsae—C. elegans genomic alignment," *Genome Res.*, vol. 10, pp. 1115–1125, 2000.
[163] P. Kersten, "Implementation issues in the fuzzy c-medians clustering algorithm," in *Proc. 6th IEEE Int. Conf. Fuzzy Systems*, vol. 2, 1997, pp. 957–962.
[164] 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 Med.*, vol. 7, no. 6, pp. 673–679, 2001.
[165] S. Kirkpatrick, C. Gelatt, and M. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.
[166] J. Kleinberg, "An impossibility theorem for clustering," in *Proc. 2002 Conf. Advances in Neural Information Processing Systems*, vol. 15, 2002, pp. 463–470.
[167] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Proc. 14th Int. Joint Conf. Artificial Intelligence*, 1995, pp. 338–345.
[168] T. Kohonen, "The self-organizing map," *Proc. IEEE*, vol. 78, no. 9, pp. 1464–1480, Sep. 1990.
[169] ——, *Self-Organizing Maps*, 3rd ed. New York: Springer-Verlag, 2001.
[170] T. Kohonen, S. Kaski, K. Lagus, J. Salojärvi, J. Honkela, V. Paatero, and A. Saarela, "Self organization of a massive document collection," *IEEE Trans. Neural Netw.*, vol. 11, no. 3, pp. 574–585, May 2000.
[171] E. Kolatch. (2001) Clustering algorithms for spatial databases: A Survey. [Online]. Available: http://citeseer.nj.nec.com/436 843.html
[172] J. Kolen and T. Hutcheson, "Reducing the time complexity of the fuzzy c-means algorithm," *IEEE Trans. Fuzzy Syst.*, vol. 10, no. 2, pp. 263–267, Apr. 2002.
[173] K. Krishna and M. Murty, "Genetic K-means algorithm," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 29, no. 3, pp. 433–439, Jun. 1999.
[174] R. Krishnapuram, H. Frigui, and O. Nasraoui, "Fuzzy and possiblistic shell clustering algorithms and their application to boundary detection and surface approximation—Part I and II," *IEEE Trans. Fuzzy Syst.*, vol. 3, no. 1, pp. 29–60, Feb. 1995.
[175] R. Krishnapuram and J. Keller, "A possibilistic approach to clustering," *IEEE Trans. Fuzzy Syst.*, vol. 1, no. 2, pp. 98–110, Apr. 1993.
[176] R. Krishnapuram, O. Nasraoui, and H. Frigui, "The fuzzy c spherical shells algorithm: A new approach," *IEEE Trans. Neural Netw.*, vol. 3, no. 5, pp. 663–671, Sep. 1992.
[177] A. Krogh, M. Brown, I. Mian, K. Sjölander, and D. Haussler, "Hidden Markov models in computational biology: Applications to protein modeling," *J. Molec. Biol.*, vol. 235, pp. 1501–1531, 1994.
[178] G. Lance and W. Williams, "A general theory of classification sorting strategies: 1. Hierarchical systems," *Comput. J.*, vol. 9, pp. 373–380, 1967.
[179] M. Law and J. Kwok, "Rival penalized competitive learning for model-based sequence clustering," in *Proc. 15th Int. Conf. Pattern Recognition*, vol. 2, 2000, pp. 195–198.

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

[180] Y. Leung, J. Zhang, and Z. Xu, "Clustering by scale-space filtering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 12, pp. 1396–1410, Dec. 2000.

[181] E. Levine and E. Domany, "Resampling method for unsupervised estimation of cluster validity," *Neural Comput.*, vol. 13, pp. 2573–2593, 2001.

[182] C. Li and G. Biswas, "Temporal pattern generation using hidden Markov model based unsupervised classification," in *Advances in Intelligent Data Analysis*. ser. Lecture Notes in Computer Science, D. Hand, K. Kok, and M. Berthold, Eds. New York: Springer-Verlag, 1999, vol. 1642.

[183] ——, "Unsupervised learning with mixed numeric and nominal data," *IEEE Trans. Knowl. Data Eng.*, vol. 14, no. 4, pp. 673–690, Jul.-Aug. 2002.

[184] C. Li, H. Garcia-Molina, and G. Wiederhold, "Clustering for approximate similarity search in high-dimensional spaces," *IEEE Trans. Knowl. Data Eng.*, vol. 14, no. 4, pp. 792–808, Jul.-Aug. 2002.

[185] W. Li, L. Jaroszewski, and A. Godzik, "Clustering of highly homologous sequences to reduce the size of large protein databases," *Bioinformatics*, vol. 17, pp. 282–283, 2001.

[186] A. Likas, N. Vlassis, and J. Verbeek, "The global K-means clustering algorithm," *Pattern Recognit.*, vol. 36, no. 2, pp. 451–461, 2003.

[187] S. Lin and B. Kernighan, "An effective heuristic algorithm for the traveling salesman problem," *Operat. Res.*, vol. 21, pp. 498–516, 1973.

[188] R. Lipshutz, S. Fodor, T. Gingeras, and D. Lockhart, "High density synthetic oligonucleotide arrays," *Nature Genetics*, vol. 21, pp. 20–24, 1999.

[189] G. Liu, *Introduction to Combinatorial Mathematics*. New York: McGraw-Hill, 1968.

[190] J. Lozano and P. Larrañaga, "Applying genetic algorithms to search for the best hierarchical clustering of a dataset," *Pattern Recognit. Lett.*, vol. 20, pp. 911–918, 1999.

[191] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp.*, vol. 1, 1967, pp. 281–297.

[192] S. C. Madeira and A. L. Oliveira, "Biclustering algorithms for biological data analysis: A survey," *IEEE/ACM Trans. Computat. Biol. Bioinformatics*, vol. 1, no. 1, pp. 24–45, Jan. 2004.

[193] Y. Man and I. Gath, "Detection and separation of ring-shaped clusters using fuzzy clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, no. 8, pp. 855–861, Aug. 1994.

[194] J. Mao and A. Jain, "A self-organizing network for hyperellipsoidal clustering (HEC)," *IEEE Trans. Neural Netw.*, vol. 7, no. 1, pp. 16–29, Jan. 1996.

[195] U. Maulik and S. Bandyopadhyay, "Genetic algorithm-based clustering technique," *Pattern Recognit.*, vol. 33, pp. 1455–1465, 2000.

[196] G. McLachlan and T. Krishnan, *The EM Algorithm and Extensions*. New York: Wiley, 1997.

[197] G. McLachlan and D. Peel, *Finite Mixture Models*. New York: Wiley, 2000.

[198] G. McLachlan, D. Peel, K. Basford, and P. Adams, "The EMMIX software for the fitting of mixtures of normal and t-components," *J. Statist. Software*, vol. 4, 1999.

[199] C. Miller, J. Gurd, and A. Brass, "A RAPID algorithm for sequence database comparisons: Application to the identification of vector contamination in the EMBL databases," *Bioinformatics*, vol. 15, pp. 111–121, 1999.

[200] R. Miller *et al.*, "A comprehensive approach to clustering of expressed human gene sequence: The sequence tag alignment and consensus knowledge base," *Genome Res.*, vol. 9, pp. 1143–1155, 1999.

[201] W. Miller, "Comparison of genomic DNA sequences: Solved and unsolved problems," *Bioinformatics*, vol. 17, pp. 391–397, 2001.

[202] G. Milligan and M. Cooper, "An examination of procedures for determining the number of clusters in a data set," *Psychometrika*, vol. 50, pp. 159–179, 1985.

[203] R. Mollineda and E. Vidal, "A relative approach to hierarchical clustering," in *Pattern Recognition and Applications, Frontiers in Artificial Intelligence and Applications*, M. Torres and A. Sanfeliu, Eds. Amsterdam, The Netherlands: IOS Press, 2000, vol. 56, pp. 19–28.

[204] B. Moore, "ART1 and pattern clustering," in *Proc. 1988 Connectionist Models Summer School*, 1989, pp. 174–185.

[205] S. Moore, "Making chips to probe genes," *IEEE Spectr.*, vol. 38, no. 3, pp. 54–60, Mar. 2001.

[206] Y. Moreau, F. Smet, G. Thijs, K. Marchal, and B. Moor, "Functional bioinformatics of microarray data: From expression to regulation," *Proc. IEEE*, vol. 90, no. 11, pp. 1722–1743, Nov. 2002.

[207] T. Morzy, M. Wojciechowski, and M. Zakrzewicz, "Pattern-oriented hierarchical clustering," in *Proc. 3rd East Eur. Conf. Advances in Databases and Information Systems*, 1999, pp. 179–190.

[208] S. Mulder and D. Wunsch, "Million city traveling salesman problem solution by divide and conquer clustering with adaptive resonance neural networks," *Neural Netw.*, vol. 16, pp. 827–832, 2003.

[209] K. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf, "An introduction to kernel-based learning algorithms," *IEEE Trans. Neural Netw.*, vol. 12, no. 2, pp. 181–201, Mar. 2001.

[210] F. Murtagh, "A survey of recent advances in hierarchical clustering algorithms," *Comput. J.*, vol. 26, no. 4, pp. 354–359, 1983.

[211] F. Murtagh and M. Berry, "Overcoming the curse of dimensionality in clustering by means of the wavelet transform," *Comput. J.*, vol. 43, no. 2, pp. 107–120, 2000.

[212] S. Needleman and C. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *J. Molec. Biol.*, vol. 48, pp. 443–453, 1970.

[213] R. Ng and J. Han, "CLARANS: A method for clustering objects for spatial data mining," *IEEE Trans. Knowl. Data Eng.*, vol. 14, no. 5, pp. 1003–1016, Sep.-Oct. 2002.

[214] T. Oates, L. Firoiu, and P. Cohen, "Using dynamic time warping to bootstrap HMM-based clustering of time series," in *Sequence Learning*. ser. LNAI 1828, R. Sun and C. Giles, Eds. Berlin, Germany: Springer-Verlag, 2000, pp. 35–52.

[215] E. Oja, "Principal components minor components, and linear neural networks," *Neural Netw.*, vol. 5, pp. 927–935, 1992.

[216] J. Oliver, R. Baxter, and C. Wallace, "Unsupervised learning using MML," in *Proc. 13th Int. Conf. Machine Learning (ICML'96)*, Lorenza, Saitta, 1996, pp. 364–372.

[217] C. Olson, "Parallel algorithms for hierarchical clustering," *Parallel Comput.*, vol. 21, pp. 1313–1325, 1995.

[218] C. Ordonez and E. Omiecinski, "Efficient disk-based K-means clustering for relational databases," *IEEE Trans. Knowl. Data Eng.*, vol. 16, no. 8, pp. 909–921, Aug. 2004.

[219] L. Owsley, L. Atlas, and G. Bernard, "Self-organizing feature maps and hidden Markov models for machine-tool monitoring," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2787–2798, Nov. 1997.

[220] N. Pal and J. Bezdek, "On cluster validity for the fuzzy c-means model," *IEEE Trans. Fuzzy Syst.*, vol. 3, no. 3, pp. 370–379, Aug. 1995.

[221] N. Pal, J. Bezdek, and E. Tsao, "Generalized clustering networks and Kohonen's self-organizing scheme," *IEEE Trans. Neural Netw.*, vol. 4, no. 4, pp. 549–557, Jul. 1993.

[222] G. Patanè and M. Russo, "The enhanced-LBG algorithm," *Neural Netw.*, vol. 14, no. 9, pp. 1219–1237, 2001.

[223] ——, "Fully automatic clustering system," *IEEE Trans. Neural Netw.*, vol. 13, no. 6, pp. 1285–1298, Nov. 2002.

[224] W. Pearson, "Improved tools for biological sequence comparison," *Proc. Nat. Acad. Sci.*, vol. 85, pp. 2444–2448, 1988.

[225] D. Peel and G. McLachlan, "Robust mixture modeling using the t-distribution," *Statist. Comput.*, vol. 10, pp. 339–348, 2000.

[226] D. Pelleg and A. Moore, "X-means: Extending K-means with efficient estimation of the number of clusters," in *Proc. 17th Int. Conf. Machine Learning (ICML'00)*, 2000, pp. 727–734.

[227] J. Peña, J. Lozano, and P. Larrañaga, "An empirical comparison of four initialization methods for the K-means algorithm," *Pattern Recognit. Lett.*, vol. 20, pp. 1027–1040, 1999.

[228] C. Pizzuti and D. Talia, "P-AutoClass: Scalable parallel clustering for mining large data sets," *IEEE Trans. Knowl. Data Eng.*, vol. 15, no. 3, pp. 629–641, May-Jun. 2003.

[229] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, Feb. 1989.

[230] Ralf-Herwig, A. Poustka, C. Müller, C. Bull, H. Lehrach, and J. O'Brien, "Large-scale clustering of cDNA-fingerprinting data," *Genome Res.*, pp. 1093–1105, 1999.

[231] A. Rauber, J. Paralic, and E. Pampalk, "Empirical evaluation of clustering algorithms," *J. Inf. Org. Sci.*, vol. 24, no. 2, pp. 195–209, 2000.

[232] S. Ridella, S. Rovetta, and R. Zunino, "Plastic algorithm for adaptive vector quantization," *Neural Comput. Appl.*, vol. 7, pp. 37–51, 1998.

[233] J. Rissanen, "Fisher information and stochastic complexity," *IEEE Trans. Inf. Theory*, vol. 42, no. 1, pp. 40–47, Jan. 1996.

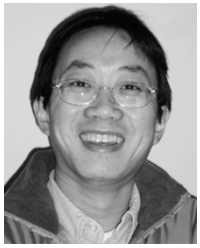
[234] K. Rose, "Deterministic annealing for clustering, compression, classification, regression, and related optimization problems," *Proc. IEEE*, vol. 86, no. 11, pp. 2210–2239, Nov. 1998.

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

[235] S. Roweis and L. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.
[236] D. Sankoff and J. Kruskal, *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*. Stanford, CA: CSLI Publications, 1999.
[237] O. Sasson, N. Linial, and M. Linial, "The metric space of proteins—Comparative study of clustering algorithms," *Bioinformatics*, vol. 18, pp. s14–s21, 2002.
[238] 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*, vol. 24, no. 3, pp. 236–244, 2000.
[239] P. Scheunders, "A comparison of clustering algorithms applied to color image quantization," *Pattern Recognit. Lett.*, vol. 18, pp. 1379–1384, 1997.
[240] B. Schölkopf and A. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA: MIT Press, 2002.
[241] B. Schölkopf, A. Smola, and K. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Computat.*, vol. 10, no. 5, pp. 1299–1319, 1998.
[242] G. Schwarz, "Estimating the dimension of a model," *Ann. Statist.*, vol. 6, no. 2, pp. 461–464, 1978.
[243] G. Scott, D. Clark, and T. Pham, "A genetic clustering algorithm guided by a descent algorithm," in *Proc. Congr. Evolutionary Computation*, vol. 2, Piscataway, NJ, 2001, pp. 734–740.
[244] P. Sebastiani, M. Ramoni, and P. Cohen, "Sequence learning via Bayesian clustering by dynamics," in *Sequence Learning*. ser. LNAI 1828, R. Sun and C. Giles, Eds. Berlin, Germany: Springer-Verlag, 2000, pp. 11–34.
[245] S. Selim and K. Alsultan, "A simulated annealing algorithm for the clustering problems," *Pattern Recognit.*, vol. 24, no. 10, pp. 1003–1008, 1991.
[246] R. Shamir and R. Sharan, "Algorithmic approaches to clustering gene expression data," in *Current Topics in Computational Molecular Biology*, T. Jiang, T. Smith, Y. Xu, and M. Zhang, Eds. Cambridge, MA: MIT Press, 2002, pp. 269–300.
[247] R. Sharan and R. Shamir, "CLICK: A clustering algorithm with applications to gene expression analysis," in *Proc. 8th Int. Conf. Intelligent Systems for Molecular Biology*, 2000, pp. 307–316.
[248] G. Sheikholeslami, S. Chatterjee, and A. Zhang, "WaveCluster: A multiresolution clustering approach for very large spatial databases," in *Proc. 24th VLDB Conf.*, 1998, pp. 428–439.
[249] P. Simpson, "Fuzzy min-max neural networks—Part 2: Clustering," *IEEE Trans. Fuzzy Syst.*, vol. 1, no. 1, pp. 32–45, Feb. 1993.
[250] *Handbook of Pattern Recognition and Computer Vision*, C. Chen, L. Pau, and P. Wang, Eds., World Scientific, Singapore, 1993, pp. 61–124. J. Sklansky and W. Siedlecki, "Large-scale feature selection".
[251] T. Smith and M. Waterman, "New stratigraphic correlation techniques," *J. Geology*, vol. 88, pp. 451–457, 1980.
[252] P. Smyth, "Clustering using Monte Carlo cross-validation," in *Proc. 2nd Int. Conf. Knowledge Discovery and Data Mining*, 1996, pp. 126–133.
[253] ——, "Clustering sequences with hidden Markov models," in *Advances in Neural Information Processing*, M. Mozer, M. Jordan, and T. Petsche, Eds. Cambridge, MA: MIT Press, 1997, vol. 9, pp. 648–654.
[254] ——, "Model selection for probabilistic clustering using cross validated likelihood," *Statist. Comput.*, vol. 10, pp. 63–72, 1998.
[255] ——, "Probabilistic model-based clustering of multivariate and sequential data," in *Proc. 7th Int. Workshop on Artificial Intelligence and Statistics*, 1999, pp. 299–304.
[256] P. Sneath, "The application of computers to taxonomy," *J. Gen. Microbiol.*, vol. 17, pp. 201–226, 1957.
[257] P. Somervuo and T. Kohonen, "Clustering and visualization of large protein sequence databases by means of an extension of the self-organizing map," in *LNAI 1967*, 2000, pp. 76–85.
[258] T. Sorensen, "A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its application to analyzes of the vegetation on Danish commons," *Biologiske Skrifter*, vol. 5, pp. 1–34, 1948.
[259] H. Späth, *Cluster Analysis Algorithms*. Chichester, U.K.: Ellis Horwood, 1980.
[260] 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*, vol. 9, pp. 3273–3297, 1998.
[261] "Tech. Rep. 00–034," Univ. Minnesota, Minneapolis, 2000.
[262] K. Stoffel and A. Belkoniene, "Parallel K-means clustering for large data sets," in *Proc. EuroPar'99 Parallel Processing*, 1999, pp. 1451–1454.
[263] M. Su and H. Chang, "Fast self-organizing feature map algorithm," *IEEE Trans. Neural Netw.*, vol. 11, no. 3, pp. 721–733, May 2000.
[264] M. Su and C. Chou, "A modified version of the K-means algorithm with a distance based on cluster symmetry," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 6, pp. 674–680, Jun. 2001.
[265] R. Sun and C. Giles, "Sequence learning: Paradigms, algorithms, and applications," in *LNAI 1828*, . Berlin, Germany, 2000.
[266] C. Sung and H. Jin, "A Tabu-search-based heuristic for clustering," *Pattern Recognit.*, vol. 33, pp. 849–858, 2000.
[267] *SWISS-PROT Protein Knowledgebase Release 45.0 Statistics*.
[268] 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," *Proc. Nat. Acad. Sci.*, pp. 2907–2912, 1999.
[269] S. Tavazoie, J. Hughes, M. Campbell, R. Cho, and G. Church, "Systematic determination of genetic network architecture," *Nature Genetics*, vol. 22, pp. 281–285, 1999.
[270] J. Tenenbaum, V. Silva, and J. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, pp. 2319–2323, 2000.
[271] R. Tibshirani, T. Hastie, M. Eisen, D. Ross, D. Botstein, and P. Brown, "Clustering methods for the analysis of DNA microarray data," Dept. Statist., Stanford Univ., Stanford, CA, Tech. Rep..
[272] R. Tibshirani and K. Knight, "The covariance inflation criterion for adaptive model selection," *J. Roy. Statist. Soc. B*, vol. 61, pp. 529–546, 1999.
[273] L. Tseng and S. Yang, "A genetic approach to the automatic clustering problem," *Pattern Recognit.*, vol. 34, pp. 415–424, 2001.
[274] V. Vapnik, *Statistical Learning Theory*. New York: Wiley, 1998.
[275] J. Venter *et al.*, "The sequence of the human genome," *Science*, vol. 291, pp. 1304–1351, 2001.
[276] J. Vesanto and E. Alhoniemi, "Clustering of the self-organizing map," *IEEE Trans. Neural Netw.*, vol. 11, no. 3, pp. 586–600, May 2000.
[277] K. Wagstaff, S. Rogers, and S. Schroedl, "Constrained K-means clustering with background knowledge," in *Proc. 8th Int. Conf. Machine Learning*, 2001, pp. 577–584.
[278] C. Wallace and D. Dowe, "Intrinsic classification by MML—The SNOB program," in *Proc. 7th Australian Joint Conf. Artificial Intelligence*, 1994, pp. 37–44.
[279] H. Wang, W. Wang, J. Yang, and P. Yu, "Clustering by pattern similarity in large data sets," in *Proc. ACM SIGMOD Int. Conf. Management of Data*, 2002, pp. 394–405.
[280] C. Wei, Y. Lee, and C. Hsu, "Empirical comparison of fast clustering algorithms for large data sets," in *Proc. 33rd Hawaii Int. Conf. System Sciences*, Maui, HI, 2000, pp. 1–10.
[281] J. Williamson, "Gaussian ARTMAP: A neural network for fast incremental learning of noisy multidimensional maps," *Neural Netw.*, vol. 9, no. 5, pp. 881–897, 1996.
[282] M. Windham and A. Culter, "Information ratios for validating mixture analysis," *J. Amer. Statist. Assoc.*, vol. 87, pp. 1188–1192, 1992.
[283] S. Wu, A. W.-C. Liew, H. Yan, and M. Yang, "Cluster analysis of gene expression data based on self-splitting and merging competitive learning," *IEEE Trans. Inf. Technol. Biomed.*, vol. 8, no. 1, pp. 5–15, Jan. 2004.
[284] D. Wunsch, "An optoelectronic learning machine: Invention, experimentation, analysis of first hardware implementation of the ART1 neural network," Ph.D. dissertation, Univ. Washington, Seattle, WA, 1991.
[285] D. Wunsch, T. Caudell, C. Capps, R. Marks, and R. Falk, "An optoelectronic implementation of the adaptive resonance neural network," *IEEE Trans. Neural Netw.*, vol. 4, no. 4, pp. 673–684, Jul. 1993.
[286] Y. Xiong and D. Yeung, "Mixtures of ARMA models for model-based time series clustering," in *Proc. IEEE Int. Conf. Data Mining*, 2002, pp. 717–720.
[287] R. Xu, G. Anagnostopoulos, and D. Wunsch, "Tissue classification through analysis of gene expression data using a new family of ART architectures," in *Proc. Int. Joint Conf. Neural Networks (IJCNN'02)*, vol. 1, 2002, pp. 300–304.
[288] Y. Xu, V. Olman, and D. Xu, "Clustering gene expression data using graph-theoretic approach: An application of minimum spanning trees," *Bioinformatics*, vol. 18, no. 4, pp. 536–545, 2002.

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.

[289] R. Yager, "Intelligent control of the hierarchical agglomerative clustering process," *IEEE Trans. Syst., Man, Cybern.*, vol. 30, no. 6, pp. 835–845, 2000.
[290] R. Yager and D. Filev, "Approximate clustering via the mountain method," *IEEE Trans. Syst., Man, Cybern.*, vol. 24, no. 8, pp. 1279–1284, 1994.
[291] K. Yeung, D. Haynor, and W. Ruzzo, "Validating clustering for gene expression data," *Bioinformatics*, vol. 17, no. 4, pp. 309–318, 2001.
[292] F. Young and R. Hamer, *Multidimensional Scaling: History, Theory, and Applications*. Hillsdale, NJ: Lawrence Erlbaum, 1987.
[293] L. Zadeh, "Fuzzy sets," *Inf. Control*, vol. 8, pp. 338–353, 1965.
[294] J. Zhang and Y. Leung, "Improved possibilistic C-means clustering algorithms," *IEEE Trans. Fuzzy Syst.*, vol. 12, no. 2, pp. 209–217, Apr. 2004.
[295] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: An efficient data clustering method for very large databases," in *Proc. ACM SIGMOD Conf. Management of Data*, 1996, pp. 103–114.
[296] Y. Zhang and Z. Liu, "Self-splitting competitive learning: A new on-line clustering paradigm," *IEEE Trans. Neural Networks*, vol. 13, no. 2, pp. 369–380, Mar. 2002.
[297] X. Zhuang, Y. Huang, K. Palaniappan, and Y. Zhao, "Gaussian mixture density modeling, decomposition, and applications," *IEEE Trans. Image Process.*, vol. 5, no. 9, pp. 1293–1302, Sep. 1996.



Rui Xu (S'00) received the B.E. degree in electrical engineering from Huazhong University of Science and Technology, Wuhan, Hubei, China, in 1997, and the M.E. degree in electrical engineering from Sichuan University, Chengdu, Sichuan, in 2000. He is currently pursuing the Ph.D. degree in the Department of Electrical and Computer Engineering, University of Missouri-Rolla.

His research interests include machine learning, neural networks, pattern classification and clustering, and bioinformatics.

Mr. Xu is a Student Member of the IEEE Computational Intelligence Society, Engineering in Medicine and Biology Society, and the International Society for Computational Biology.

Donald C. Wunsch II (S'87–M'92–SM'94–F'05) received the B.S. degree in applied mathematics from the University of New Mexico, Albuquerque, and the M.S. degree in applied mathematics and the Ph.D. degree in electrical engineering from the University of Washington, Seattle.

Heis the Mary K. Finley Missouri Distinguished Professor of Computer Engineering, University of Missouri-Rolla, where he has been since 1999. His prior positions were Associate Professor and Director of the Applied Computational Intelligence Laboratory, Texas Tech University, Lubbock; Senior Principal Scientist, Boeing; Consultant, Rockwell International; and Technician, International Laser Systems. He has well over 200 publications, and has attracted over $5 million in research funding. He has produced eight Ph.D. recipients—four in electrical engineering, three in computer engineering, and one in computer science.

Dr. Wunsch has received the Halliburton Award for Excellence in Teaching and Research, and the National Science Foundation CAREER Award. He served as a Voting Member of the IEEE Neural Networks Council, Technical Program Co-Chair for IJCNN'02, General Chair for IJCNN'03, International Neural Networks Society Board of Governors Member, and is now President of the International Neural Networks Society.

Authorized licensed use limited to: Penn State University. Downloaded on March 02,2026 at 21:22:43 UTC from IEEE Xplore. Restrictions apply.