Pattern Recognition Letters 31 (2010) 651–666

Contents lists available at ScienceDirect

Pattern Recognition Letters

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

Data clustering: 50 years beyond K-means ☆

Anil K. Jain *

Department of Computer Science and Engineering, Michigan State University, East Lansing, Michigan 48824, USA
Department of Brain and Cognitive Engineering, Korea University, Anam-dong, Seoul, 136-713, Korea

ARTICLE INFO

Article history:
Available online 9 September 2009

Keywords:
Data clustering
User's dilemma
Historical developments
Perspectives on clustering
King-Sun Fu prize

ABSTRACT

Organizing data into sensible groupings is one of the most fundamental modes of understanding and learning. As an example, a common scheme of scientific classification puts organisms into a system of ranked taxa: domain, kingdom, phylum, class, etc. Cluster analysis is the formal study of methods and algorithms for grouping, or clustering, objects according to measured or perceived intrinsic characteristics or similarity. Cluster analysis does not use category labels that tag objects with prior identifiers, i.e., class labels. The absence of category information distinguishes data clustering (unsupervised learning) from classification or discriminant analysis (supervised learning). The aim of clustering is to find structure in data and is therefore exploratory in nature. Clustering has a long and rich history in a variety of scientific fields. One of the most popular and simple clustering algorithms, K-means, was first published in 1955. In spite of the fact that K-means was proposed over 50 years ago and thousands of clustering algorithms have been published since then, K-means is still widely used. This speaks to the difficulty in designing a general purpose clustering algorithm and the ill-posed problem of clustering. We provide a brief overview of clustering, summarize well known clustering methods, discuss the major challenges and key issues in designing clustering algorithms, and point out some of the emerging and useful research directions, including semi-supervised clustering, ensemble clustering, simultaneous feature selection during data clustering, and large scale data clustering.

1. Introduction

Advances in sensing and storage technology and dramatic growth in applications such as Internet search, digital imaging, and video surveillance have created many high-volume, high-dimensional data sets. It is estimated that the digital universe consumed approximately 281 exabytes in 2007, and it is projected to be 10 times that size by 2011 (1 exabyte is $\sim 10^{18}$ bytes or 1,000,000 terabytes) (Gantz, 2008). Most of the data is stored digitally in electronic media, thus providing huge potential for the development of automatic data analysis, classification, and retrieval techniques. In addition to the growth in the amount of data, the variety of available data (text, image, and video) has also increased. Inexpensive digital and video cameras have made available huge archives of images and videos. The prevalence of RFID tags or transponders due to their low cost and small size has resulted in the deployment of millions of sensors that transmit data regularly. E-mails, blogs, transaction data, and billions of Web pages create terabytes of new data every day. Many of these data streams are unstructured, adding to the difficulty in analyzing them.

The increase in both the volume and the variety of data requires advances in methodology to automatically understand, process, and summarize the data. Data analysis techniques can be broadly classified into two major types (Tukey, 1977): (i) *exploratory* or descriptive, meaning that the investigator does not have pre-specified models or hypotheses but wants to understand the general characteristics or structure of the high-dimensional data, and (ii) *confirmatory* or inferential, meaning that the investigator wants to confirm the validity of a hypothesis/model or a set of assumptions given the available data. Many statistical techniques have been proposed to analyze the data, such as analysis of variance, linear regression, discriminant analysis, canonical correlation analysis, multi-dimensional scaling, factor analysis, principal component analysis, and cluster analysis to name a few. A useful overview is given in (Tabachnick and Fidell, 2007).

In pattern recognition, data analysis is concerned with predictive modeling: given some training data, we want to predict the behavior of the unseen test data. This task is also referred to as *learning*. Often, a clear distinction is made between learning problems that are (i) supervised (classification) or (ii) unsupervised (clustering), the first involving only *labeled data* (training patterns with known category labels) while the latter involving only *unlabeled data* (Duda et al., 2001). Clustering is a more difficult and

☆ This paper is based on the King-Sun Fu Prize lecture delivered at the 19th International Conference on Pattern Recognition (ICPR), Tempa, FL, December 8, 2008.

* Tel.: +1 517 355 9282; fax: +1 517 432 1061.
E-mail address: jain@cse.msu.edu

0167-8655/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.patrec.2009.09.011

challenging problem than classification. There is a growing interest in a hybrid setting, called *semi-supervised learning* (Chapelle et al., 2006); in semi-supervised classification, the labels of only a small portion of the training data set are available. The unlabeled data, instead of being discarded, are also used in the learning process. In semi-supervised clustering, instead of specifying the class labels, pair-wise constraints are specified, which is a *weaker* way of encoding the prior knowledge. A pair-wise *must-link* constraint corresponds to the requirement that two objects should be assigned the same cluster label, whereas the cluster labels of two objects participating in a *cannot-link* constraint should be different. Constraints can be particularly beneficial in data clustering (Lange et al., 2005; Basu et al., 2008), where precise definitions of underlying clusters are absent. In the search for good models, one would like to include all the available information, no matter whether it is unlabeled data, data with constraints, or labeled data. Fig. 1 illustrates this spectrum of different types of learning problems of interest in pattern recognition and machine learning.

2. Data clustering

The goal of data clustering, also known as cluster analysis, is to discover the *natural* grouping(s) of a set of patterns, points, or objects. Webster (Merriam-Webster Online Dictionary, 2008) defines cluster analysis as "a statistical classification technique for discovering whether the individuals of a population fall into different groups by making quantitative comparisons of multiple characteristics." An example of clustering is shown in Fig. 2. The objective is to develop an automatic algorithm that will discover the natural groupings (Fig. 2b) in the unlabeled data (Fig. 2a).

An operational definition of clustering can be stated as follows: Given a *representation* of n objects, find K groups based on a measure of *similarity* such that the similarities between objects in the same group are high while the similarities between objects in different groups are low. But, what is the notion of similarity? What is the definition of a cluster? Fig. 2 shows that clusters can differ in terms of their *shape*, *size*, and *density*. The presence of noise in the data makes the detection of the clusters even more difficult. An ideal cluster can be defined as a set of points that is *compact* and *isolated*. In reality, a cluster is a subjective entity that is in the eye of the beholder and whose significance and interpretation requires domain knowledge. But, while humans are excellent cluster seekers in two and possibly three dimensions, we need automatic algorithms for high-dimensional data. It is this challenge along with the unknown number of clusters for the given data that has resulted in thousands of clustering algorithms that have been published and that continue to appear.

2.1. Why clustering?

Cluster analysis is prevalent in any discipline that involves analysis of multivariate data. A search via Google Scholar (2009) found 1660 entries with the words *data clustering* that appeared in 2007

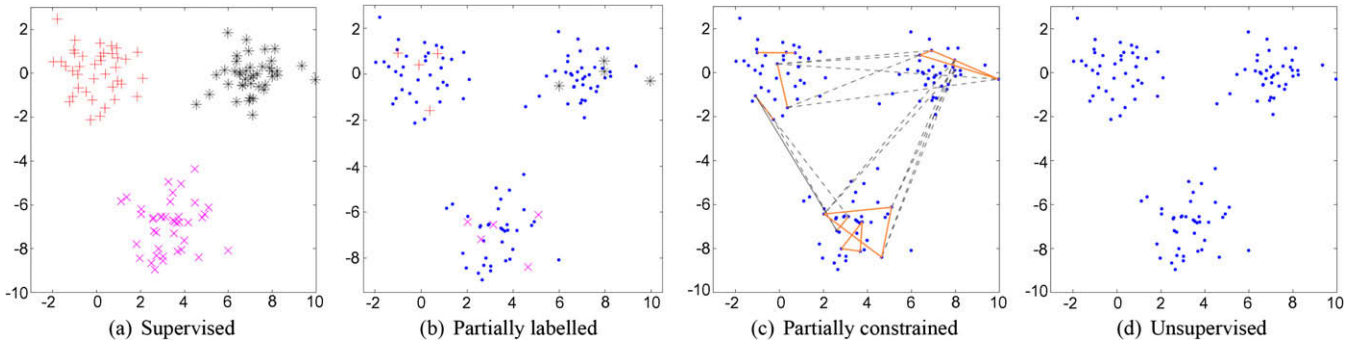


Fig. 1. Learning problems: dots correspond to points without any labels. Points with labels are denoted by plus signs, asterisks, and crosses. In (c), the must-link and cannot-link constraints are denoted by solid and dashed lines, respectively (figure taken from Lange et al. (2005).

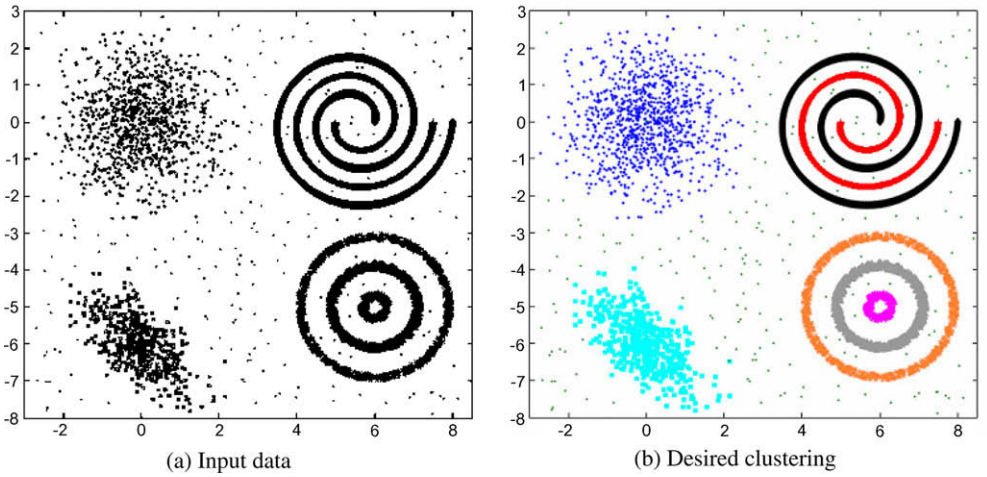


Fig. 2. Diversity of clusters. The seven clusters in (a) (denoted by seven different colors in 1(b)) differ in shape, size, and density. Although these clusters are apparent to a data analyst, none of the available clustering algorithms can detect all these clusters.

alone. This vast literature speaks to the importance of clustering in data analysis. It is difficult to exhaustively list the numerous scientific fields and applications that have utilized clustering techniques as well as the thousands of published algorithms. Image segmentation, an important problem in computer vision, can be formulated as a clustering problem (Jain and Flynn, 1996; Frigui and Krishnapuram, 1999; Shi and Malik, 2000). Documents can be clustered (Iwayama and Tokunaga, 1995) to generate topical hierarchies for efficient information access (Sahami, 1998) or retrieval (Bhatia and Deogun, 1998). Clustering is also used to group customers into different types for efficient marketing (Arabie and Hubert, 1994), to group services delivery engagements for workforce management and planning (Hu et al., 2007) as well as to study genome data (Baldi and Hatfield, 2002) in biology.

Data clustering has been used for the following three main purposes.

- *Underlying structure:* to gain insight into data, generate hypotheses, detect anomalies, and identify salient features.
- *Natural classification:* to identify the degree of similarity among forms or organisms (phylogenetic relationship).
- *Compression:* as a method for organizing the data and summarizing it through cluster prototypes.

An example of class discovery is shown in Fig. 3. Here, clustering was used to discover subclasses in an online handwritten character recognition application (Connell and Jain, 2002). Different users write the same digits in different ways, thereby increasing the within-class variance. Clustering the training patterns from a class can discover new subclasses, called the lexemes in handwritten characters. Instead of using a single model for each character, multiple models based on the number of subclasses are used to improve the recognition accuracy (see Fig. 3).

Given the large number of Web pages on the Internet, most search queries typically result in an extremely large number of hits. This creates the need for search results to be organized. Search engines like Clusty (www.clusty.org) cluster the search results and present them in a more organized way to the user.

2.2. Historical developments

The development of clustering methodology has been a truly interdisciplinary endeavor. Taxonomists, social scientists, psychologists, biologists, statisticians, mathematicians, engineers, computer scientists, medical researchers, and others who collect and process real data have all contributed to clustering methodology. According to JSTOR (2009), *data clustering* first appeared in the title of a 1954 article dealing with anthropological data. Data clustering is also known as Q-analysis, typology, clumping, and taxonomy (Jain and Dubes, 1988) depending on the field where it is applied. There are several books published on data clustering; classic ones are by Sokal and Sneath (1963), Anderberg (1973), Hartigan (1975), Jain and Dubes (1988), and Duda et al. (2001). Clustering algorithms have also been extensively studied in data mining (see books by Han and Kamber (2000) and Tan et al. (2005) and machine learning (Bishop, 2006).

Clustering algorithms can be broadly divided into two groups: *hierarchical* and *partitional*. Hierarchical clustering algorithms recursively find nested clusters either in agglomerative mode (starting with each data point in its own cluster and merging the most similar pair of clusters successively to form a cluster hierarchy) or in divisive (top-down) mode (starting with all the data points in one cluster and recursively dividing each cluster into smaller clusters). Compared to hierarchical clustering algorithms, partitional clustering algorithms find all the clusters simultaneously as a partition of the data and do not impose a hierarchical structure. Input to a hierarchical algorithm is an $n \times n$ similarity matrix, where n is the number of objects to be clustered. On the other hand, a partitional algorithm can use either an $n \times d$ pattern matrix, where n objects are embedded in a d-dimensional feature space, or an $n \times n$ similarity matrix. Note that a similarity matrix can be easily derived from a pattern matrix, but ordination methods such as multi-dimensional scaling (MDS) are needed to derive a pattern matrix from a similarity matrix.

The most well-known hierarchical algorithms are single-link and complete-link; the most popular and the simplest partitional algorithm is K-means. Since partitional algorithms are preferred in pattern recognition due to the nature of available data, our coverage here is focused on these algorithms. K-means has a rich and diverse history as it was independently discovered in different scientific fields by Steinhaus (1956), Lloyd (proposed in 1957, published in 1982), Ball and Hall (1965), and MacQueen (1967). Even though K-means was first proposed over 50 years ago, it is still one of the most widely used algorithms for clustering. Ease of implementation, simplicity, efficiency, and empirical success are the main reasons for its popularity. Below we will first summarize the development in K-means, and then discuss the major approaches that have been developed for data clustering.

2.3. K-means algorithm

Let $X = \{x_i\}$, $i = 1, \ldots, n$ be the set of n d-dimensional points to be clustered into a set of K clusters, $C = \{c_k, k = 1, \ldots, K\}$. K-means algorithm finds a partition such that the squared error between the empirical mean of a cluster and the points in the cluster is minimized. Let μ_k be the mean of cluster c_k. The squared error between μ_k and the points in cluster c_k is defined as

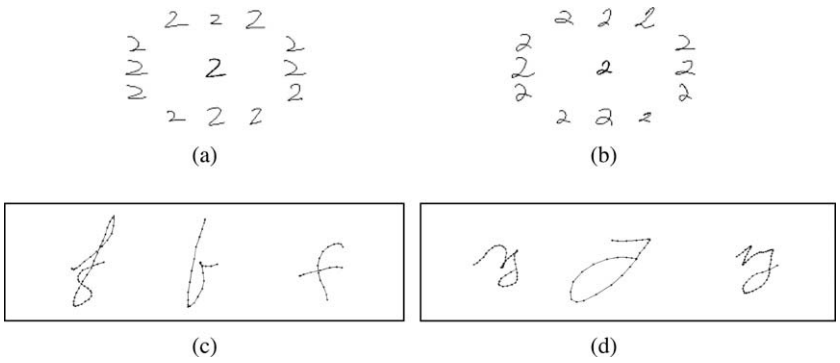


Fig. 3. Finding subclasses using data clustering. (a) and (b) show two different ways of writing the digit 2; (c) three different subclasses for the character 'f'; (d) three different subclasses for the letter 'y'.

$$J(c_k) = \sum_{x_i \in c_k} ||x_i - \mu_k||^2.$$

The goal of K-means is to minimize the sum of the squared error over all K clusters,

$$J(C) = \sum_{k=1}^{K} \sum_{x_i \in c_k} ||x_i - \mu_k||^2.$$

Minimizing this objective function is known to be an NP-hard problem (even for $K = 2$) (Drineas et al., 1999). Thus K-means, which is a greedy algorithm, can only converge to a local minimum, even though recent study has shown with a large probability K-means could converge to the global optimum when clusters are well separated (Meila, 2006). K-means starts with an initial partition with K clusters and assign patterns to clusters so as to reduce the squared error. Since the squared error always decreases with an increase in the number of clusters K (with $J(C) = 0$ when $K = n$), it can be minimized only for a fixed number of clusters. The main steps of K-means algorithm are as follows (Jain and Dubes, 1988):

1. Select an initial partition with K clusters; repeat steps 2 and 3 until cluster membership stabilizes.
2. Generate a new partition by assigning each pattern to its closest cluster center.
3. Compute new cluster centers.

Fig. 4 shows an illustration of the K-means algorithm on a 2-dimensional dataset with three clusters.

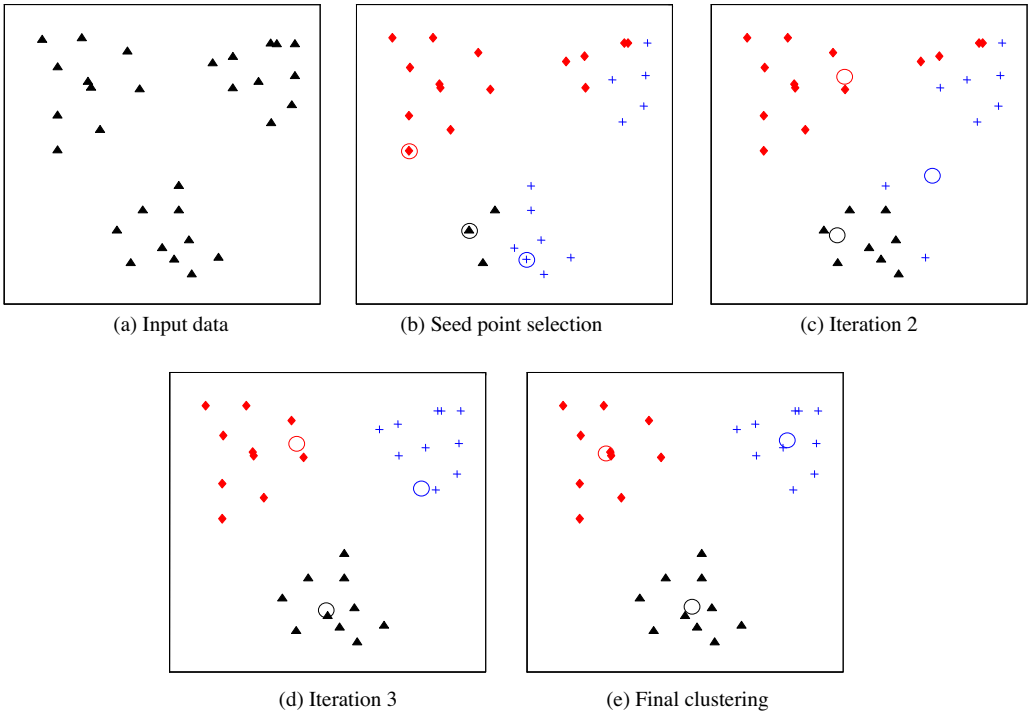
2.4. Parameters of K-means

The K-means algorithm requires three user-specified parameters: number of clusters K, cluster initialization, and distance metric. The most critical choice is K. While no perfect mathematical criterion exists, a number of heuristics (see (Tibshirani et al., 2001), and discussion therein) are available for choosing K. Typically, K-means is run independently for different values of K and the partition that appears the most meaningful to the domain expert is selected. Different initializations can lead to different final clustering because K-means only converges to local minima. One way to overcome the local minima is to run the K-means algorithm, for a given K, with multiple different initial partitions and choose the partition with the smallest squared error.

K-means is typically used with the Euclidean metric for computing the distance between points and cluster centers. As a result, K-means finds spherical or ball-shaped clusters in data. K-means with Mahalanobis distance metric has been used to detect hyperellipsoidal clusters (Mao and Jain, 1996), but this comes at the expense of higher computational cost. A variant of K-means using the Itakura–Saito distance has been used for vector quantization in speech processing (Linde et al., 1980) and K-means with L_1 distance was proposed in (Kashima et al., 2008). Banerjee et al. (2004) exploits the family of Bregman distances for K-means.

2.5. Extensions of K-means

The basic K-means algorithm has been extended in many different ways. Some of these extensions deal with additional heuristics involving the minimum cluster size and merging and splitting clusters. Two well-known variants of K-means in pattern recognition literature are ISODATA Ball and Hall (1965) and FORGY Forgy (1965). In K-means, each data point is assigned to a single cluster (called *hard assignment*). *Fuzzy c-means*, proposed by Dunn (1973) and later improved by Bezdek (1981), is an extension of K-means where each data point can be a member of multiple clusters with a membership value (*soft assignment*). A good overview of fuzzy set based clustering is available in (Backer, 1978). Data reduction by replacing group examples with their centroids before clustering them was used to speed up K-means and fuzzy C-means in (Eschrich et al., 2003). Some of the other significant modifica-



(a) Input data (b) Seed point selection (c) Iteration 2

(d) Iteration 3 (e) Final clustering

Fig. 4. Illustration of K-means algorithm. (a) Two-dimensional input data with three clusters; (b) three seed points selected as cluster centers and initial assignment of the data points to clusters; (c) and (d) intermediate iterations updating cluster labels and their centers; (e) final clustering obtained by K-means algorithm at convergence.

tions are summarized below. Steinbach et al. (2000) proposed a hierarchical divisive version of K-means, called *bisecting K-means*, that recursively partitions the data into two clusters at each step. In (Pelleg and Moore, 1999), *kd*-tree is used to efficiently identify the closest cluster centers for all the data points, a key step in K-means. Bradley et al. (1998) presented a fast scalable and single-pass version of K-means that does not require all the data to be fit in the memory at the same time. *X-means* (Pelleg and Moore, 2000) automatically finds K by optimizing a criterion such as Akaike Information Criterion (AIC) or Bayesian Information Criterion (BIC). In *K-medoid* (Kaufman and Rousseeuw, 2005), clusters are represented using the median of the data instead of the mean. *Kernel K-means* (Scholkopf et al., 1998) was proposed to detect arbitrary shaped clusters, with an appropriate choice of the kernel similarity function. Note that all these extensions introduce some additional algorithmic parameters that must be specified by the user.

2.6. Major approaches to clustering

As mentioned before, thousands of clustering algorithms have been proposed in the literature in many different scientific disciplines. This makes it extremely difficult to review all the published approaches. Nevertheless, clustering methods differ in the choice of the objective function, probabilistic generative models, and heuristics. We will briefly review some of the major approaches.

Clusters can be defined as high density regions in the feature space separated by low density regions. Algorithms following this notion of clusters directly search for connected dense regions in the feature space. Different algorithms use different definitions of connectedness. The Jarvis–Patrick algorithm defines the similarity between a pair of points as the number of common neighbors they share, where neighbors are the points present in a region of prespecified radius around the point (Frank and Todeschini, 1994). Ester et al. (1996) proposed the DBSCAN clustering algorithm, which is similar to the Jarvis–Patrick algorithm. It directly searches for connected dense regions in the feature space by estimating the density using the Parzen window method. The performance of the Jarvis–Patrick algorithm and DBSCAN depend on two parameters: neighborhood size in terms of distance, and the minimum number of points in a neighborhood for its inclusion in a cluster. In addition, a number of probabilistic models have been developed for data clustering that model the density function by a probabilistic mixture model. These approaches assume that the data is generated from a mixture distribution, where each cluster is described by one or more mixture components (McLachlan and Basford, 1987). The EM algorithm (Dempster et al., 1977) is often used to infer the parameters in mixture models. Several Bayesian approaches have been developed to improve the mixture models for data clustering, including Latent Dirichlet Allocation (LDA) (Blei et al., 2003), Pachinko Allocation model (Li and McCallum, 2006) and undirected graphical model for data clustering (Welling et al., 2005).

While the density based methods, particularly the non-parametric density based approaches, are attractive because of their inherent ability to deal with arbitrary shaped clusters, they have limitations in handling high-dimensional data. When the data is high-dimensional, the feature space is usually sparse, making it difficult to distinguish high-density regions from low-density regions. Subspace clustering algorithms overcome this limitation by finding clusters embedded in low-dimensional subspaces of the given high-dimensional data. CLIQUE (Agrawal et al., 1998) is a scalable clustering algorithm designed to find subspaces in the data with high-density clusters. Because it estimates the density only in a low dimensional subspace, CLIQUE does not suffer from the problem of high dimensionality.

Graph theoretic clustering, sometimes referred to as spectral clustering, represents the data points as nodes in a weighted graph. The edges connecting the nodes are weighted by their pair-wise similarity. The central idea is to partition the nodes into two subsets A and B such that the cut size, i.e., the sum of the weights assigned to the edges connecting between nodes in A and B, is minimized. Initial algorithms solved this problem using the minimum cut algorithm, which often results in clusters of imbalanced sizes. A cluster size (number of data points in a cluster) constraint was later adopted by the ratio cut algorithm (Hagen and Kahng, 1992). An efficient approximate graph-cut based clustering algorithm with cluster size (volume of the cluster, or sum of edge weights within a cluster) constraint, called Normalized Cut, was first proposed by Shi and Malik (2000). Its multiclass version was proposed by Yu and Shi (2003). Meila and Shi (2001) presented a Markov Random Walk view of spectral clustering and proposed the Modified Normalized Cut (MNCut) algorithm that can handle an arbitrary number of clusters. Another variant of spectral clustering algorithm was proposed by Ng et al. (2001), where a new data representation is derived from the normalized eigenvectors of a kernel matrix. Laplacian Eigenmap (Belkin and Niyogi, 2002) is another spectral clustering method that derives the data representation based on the eigenvectors of the graph Laplacian. Hofmann and Buhmann (1997) proposed a deterministic annealing algorithm for clustering data represtented using proximity measures between the data objects. Pavan and Pelillo (2007) formulate the pair-wise clustering problem by relating clusters to maximal *dominant sets* (Motzkin and Straus, 1965), which are a continuous generalization of cliques in a graph.

Several clustering algorithms have an information theoretic formulation. For example, the minimum entropy method presented in Roberts et al. (2001) assumes that the data is generated using a mixture model and each cluster is modeled using a semi-parametric probability density. The parameters are estimated by maximizing the KL-divergence between the unconditional density and the conditional density of a data points conditioned over the cluster. This minimizes the overlap between the conditional and unconditional densities, thereby separating the clusters from each other. In other words, this formulation results in an approach that minimizes the expected entropy of the partitions over the observed data. The information bottleneck method (Tishby et al., 1999) was proposed as a generalization to the rate-distortion theory and adopts a lossy data compression view. In simple words, given a joint distribution over two random variables, information bottleneck compresses one of the variables while retaining the maximum amount of mutual information with respect to the other variable. An application of this to document clustering is shown in (Slonim and Tishby, 2000) where the two random variables are words and documents. The words are clustered first, such that the mutual information with respect to documents is maximally retained, and using the clustered words, the documents are clustered such that the mutual information between clustered words and clustered documents is maximally retained.

3. User's dilemma

In spite of the prevalence of such a large number of clustering algorithms, and their success in a number of different application domains, clustering remains a difficult problem. This can be attributed to the inherent vagueness in the definition of a cluster, and the difficulty in defining an appropriate similarity measure and objective function.

The following fundamental challenges associated with clustering were highlighted in (Jain and Dubes, 1988), which are relevant even to this day.

(a) What is a cluster?
(b) What features should be used?
(c) Should the data be normalized?
(d) Does the data contain any outliers?
(e) How do we define the pair-wise similarity?
(f) How many clusters are present in the data?
(g) Which clustering method should be used?
(h) Does the data have any clustering tendency?
(i) Are the discovered clusters and partition valid?

We will highlight and illustrate some of these challenges below.

3.1. Data representation

Data representation is one of the most important factors that influence the performance of the clustering algorithm. If the representation (choice of features) is good, the clusters are likely to be compact and isolated and even a simple clustering algorithm such as K-means will find them. Unfortunately, there is no universally good representation; the choice of representation must be guided by the domain knowledge. Fig. 5a shows a dataset where K-means fails to partition it into the two "natural" clusters. The partition obtained by K-means is shown by a dashed line in Fig. 5a. However, when the same data points in (a) are represented using the top two eigenvectors of the RBF similarity matrix computed from the data in Fig. 5b, they become well separated, making it trivial for K-means to cluster the data (Ng et al., 2001).

3.2. Purpose of grouping

The representation of the data is closely tied with the purpose of grouping. The representation must go hand in hand with the end goal of the user. An example dataset of 16 animals represented using 13 Boolean features was used in (Pampalk et al., 2003) to demonstrate how the representation affects the grouping. The animals are represented using 13 Boolean features related to their appearance and activity. When a large weight is placed on the appearance features compared to the activity features, the animals were clustered into *mammals vs. birds*. On the other hand, a large weight on the activity features clustered the dataset into *predators vs. non-predators*. Both these partitionings shown in Fig. 6 are equally valid, and they uncover meaningful structures in the data. It is up to the user to carefully choose his representation to obtain a desired clustering.

3.3. Number of clusters

Automatically determining the number of clusters has been one of the most difficult problems in data clustering. Most methods for automatically determining the number of clusters cast it into the problem of model selection. Usually, clustering algorithms are run with different values of K; the best value of K is then chosen based on a predefined criterion. Figueiredo and Jain (2002) used the minimum message length (MML) criteria (Wallace and Boulton, 1968; Wallace and Freeman, 1987) in conjunction with the Gaussian mixture model (GMM) to estimate K. Their approach starts with a large number of clusters, and gradually merges the clusters if this leads to a decrease in the MML criterion. A related approach but using the principle of Minimum Description Length (MDL) was used in (Hansen and Yu, 2001) for selecting the number of clusters. The other criteria for selecting the number of clusters are the Bayes Information Criterion (BIC) and Akiake Information Criterion (AIC). Gap statistics (Tibshirani et al., 2001) is another commonly used approach for deciding the number of clusters. The key assumption is that when dividing data into an optimal number of clusters, the resulting partition is most resilient to the random perturbations. The Dirichlet Process (DP) (Ferguson, 1973; Rasmussen, 2000) introduces a non-parametric prior for the number of clusters. It is often used by probabilistic models to derive a posterior distribution for the number of clusters, from which the most likely number of clusters can be computed. In spite of these objective criteria, it is not easy to decide which value of K leads to more meaningful clusters. Fig. 7a shows a two-dimensional synthetic dataset generated from a mixture of six Gaussian components. The true labels of the points are shown in Fig. 7e. When a mixture of Gaussians is fit to the data with 2, 5, and 6 components, shown in Fig. 7b–d, respectively, each one of them seems to be a reasonable fit.

3.4. Cluster validity

Clustering algorithms tend to find clusters in the data irrespective of whether or not any clusters are present. Fig. 8a shows a dataset with no *natural* clustering; the points here were generated

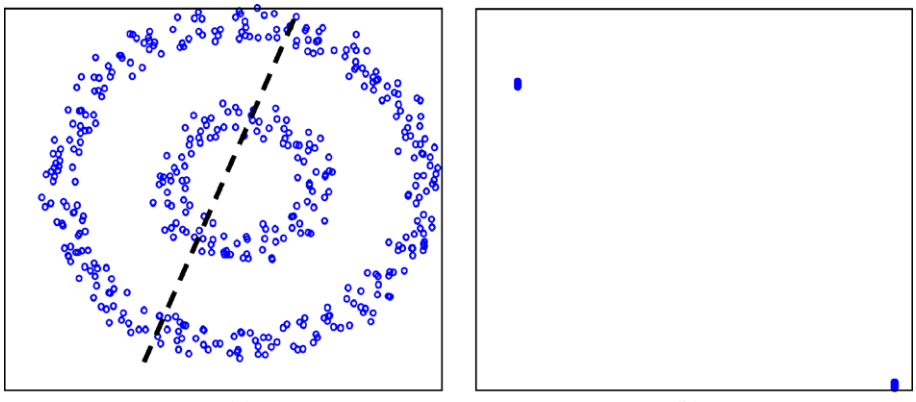
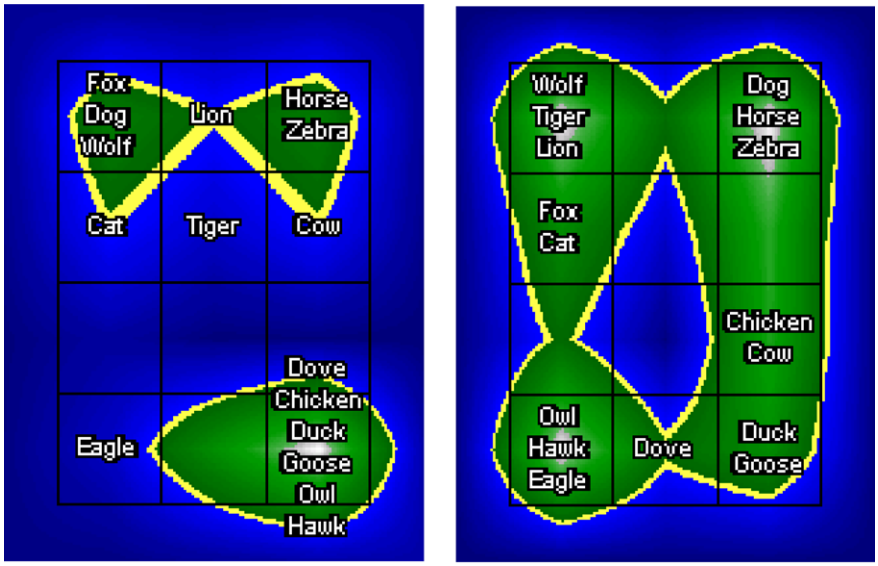


Fig. 5. Importance of a good representation. (a) "Two rings" dataset where K-means fails to find the two "natural" clusters; the dashed line shows the linear cluster separation boundary obtained by running K-means with $K = 2$. (b) a new representation of the data in (a) based on the top 2 eigenvectors of the graph Laplacian of the data, computed using an RBF kernel; K-means now can easily detect the two clusters.

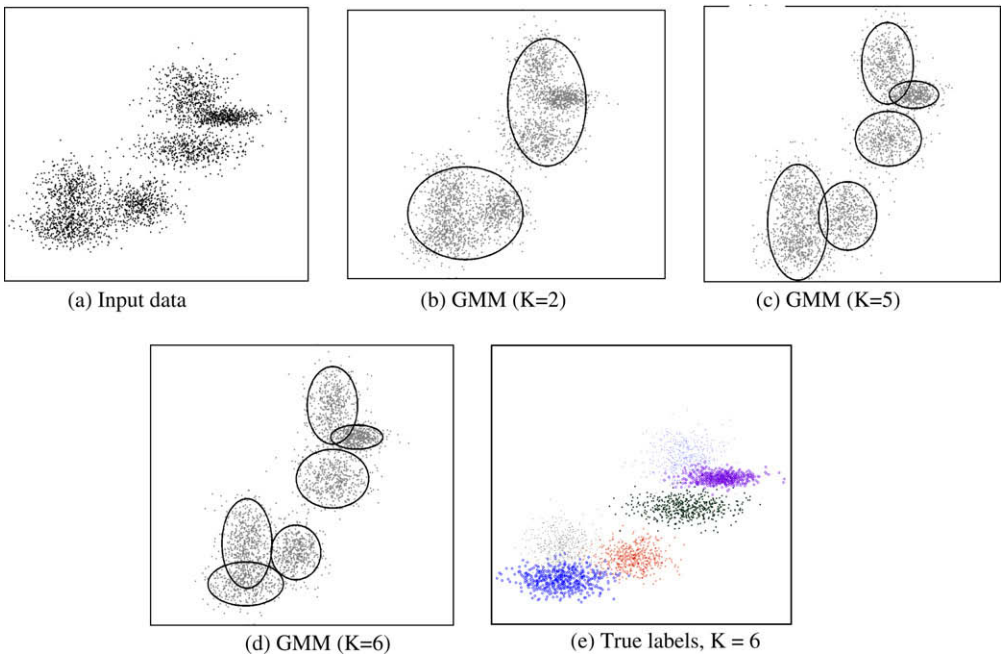


(a) (b)

Fig. 6. Different weights on features result in different partitioning of the data. Sixteen animals are represented based on 13 Boolean features related to appearance and activity. (a) Partitioning with large weights assigned to the appearance based features; (b) a partitioning with large weights assigned to the activity features. The figures in (a) and (b) are excerpted from Pampalk et al. (2003), and are known as "heat maps" where the colors represent the density of samples at a location; the warmer the color, the larger the density.

uniformly in a unit square. However, when the K-means algorithm is run on this data with $K = 3$, three clusters are identified as shown in Fig. 8b! *Cluster validity* refers to formal procedures that evaluate the results of cluster analysis in a quantitative and objective fashion (Jain and Dubes, 1988). In fact, even before a clustering algorithm is applied to the data, the user should determine if the data even has a *clustering tendency* (Smith and Jain, 1984).

Cluster validity indices can be defined based on three different criteria: *internal*, *relative*, and *external* (Jain and Dubes, 1988). Indices based on *internal criteria* assess the fit between the structure imposed by the clustering algorithm (clustering) and the data using the data alone. Indices based on *relative criteria* compare multiple structures (generated by different algorithms, for example) and decide which of them is better in some sense. *External indices* measure the performance by matching cluster structure to the a priori information, namely the "true" class labels (often referred to as ground truth). Typically, clustering results are evaluated using the external criterion, but if the true labels are available, why even bother with clustering? The notion of *cluster stability* (Lange et al., 2004) is appealing as an internal stability measure. Cluster stability is measured as the amount of variation in the clustering solution over different subsamples drawn from



(a) Input data (b) GMM (K=2) (c) GMM (K=5)

(d) GMM (K=6) (e) True labels, K = 6

Fig. 7. Automatic selection of number of clusters, *K*. (a) Input data generated from a mixture of six Gaussian distributions; (b)–(d) Gaussian mixture model (GMM) fit to the data with 2, 5, and 6 components, respectively; and (e) true labels of the data.

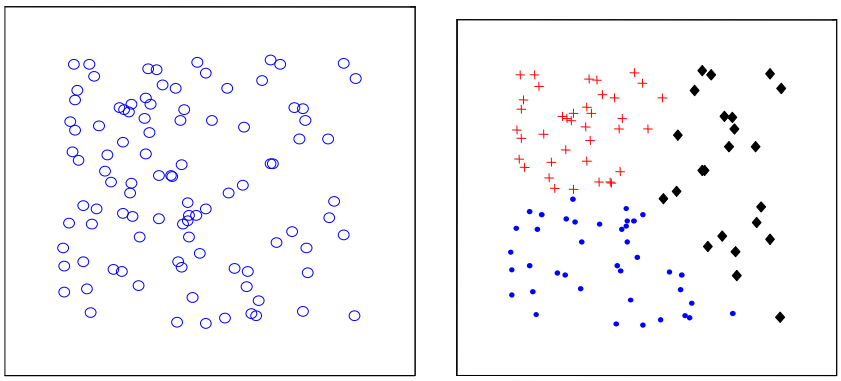


Fig. 8. Cluster validity. (a) A dataset with no "natural" clustering; (b) K-means partition with $K = 3$.

the input data. Different measures of variation can be used to obtain different stability measures. In (Lange et al., 2004), a supervised classifier is trained on one of the subsamples of the data, by using the cluster labels obtained by clustering the subsample, as the *true* labels. The performance of this classifier on the testing subset(s) indicates the stability of the clustering algorithm. In model based algorithms (e.g., centroid based representation of clusters in K-means, or Gaussian mixture models), the distance between the models found for different subsamples can be used to measure the stability (von Luxburg and David, 2005). Shamir and Tishby (2008) define stability as the generalization ability of a clustering algorithm (in PAC-Bayesian sense). They argue that since many algorithms can be shown to be asymptotically stable, the *rate* at which the asymptotic stability is reached with respect to the number of samples is a more useful measure of cluster stability. Cross-validation is a widely used evaluation method in supervised learning. It has been adapted to unsupervised learning by replacing the notation of "prediction accuracy" with a different validity measure. For example, given the mixture models obtained from the data in one fold, the likelihood of the data in the other folds serves as an indication of the algorithm's performance, and can be used to determine the number of clusters K.

3.5. Comparing clustering algorithms

Different clustering algorithms often result in entirely different partitions even on the same data. In Fig. 9, seven different algorithms were applied to cluster the 15 two-dimensional points. FORGY, ISODATA, CLUSTER, and WISH are partitional algorithms that minimize the squared error criterion (they are variants of the basic K-means algorithm). Of the remaining three algorithms, MST (minimum spanning tree) can be viewed as a single-link hierarchical algorithm, and JP is a nearest neighbor clustering algorithm. Note that a hierarchical algorithm can be used to generate a partition by specifying a threshold on the similarity. It is evident that none of the clustering is superior to the other, but some are similar to the other.

An interesting question is to identify algorithms that generate similar partitions irrespective of the data. In other words, can we cluster the clustering algorithms? Jain et al. (2004) clustered 35 different clustering algorithms into 5 groups based on their partitions on 12 different datasets. The similarity between the clustering algorithms is measured as the averaged similarity between the partitions obtained on the 12 datasets. The similarity between a pair of partitions is measured using the Adjusted Rand Index (ARI). A hierarchical clustering of the 35 clustering algorithms is

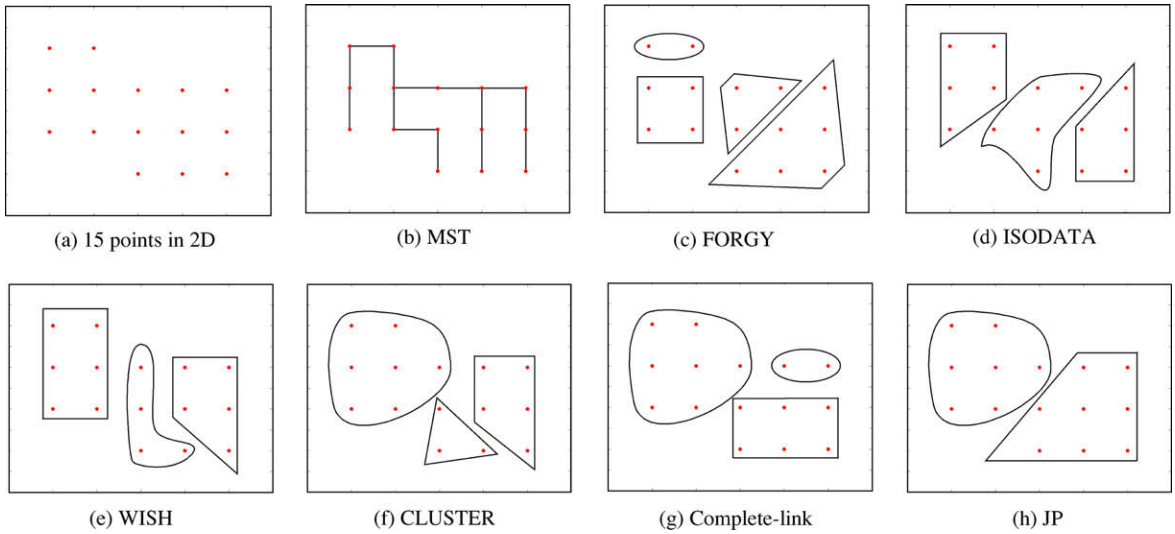


Fig. 9. Several clusterings of fifteen patterns in two dimensions: (a) fifteen patterns; (b) minimum spanning tree of the fifteen patterns; (c) clusters from FORGY; (d) clusters from ISODATA; (e) clusters from WISH; (f) clusters from CLUSTER; (g) clusters from complete-link hierarchical clustering; and (h) clusters from Jarvis-Patrick clustering algorithm. (Figure reproduced from Dubes and Jain (1976).)

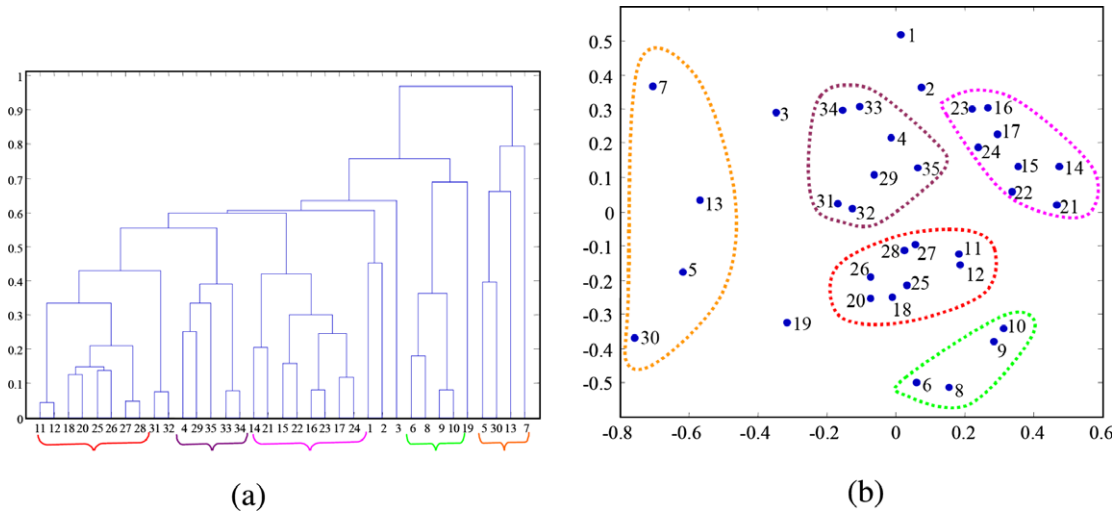


Fig. 10. Clustering of clustering algorithms. (a) Hierarchical clustering of 35 different algorithms; (b) Sammon's mapping of the 35 algorithms into a two-dimensional space, with the clusters highlighted for visualization. The algorithms in the group (4, 29, 31–35) correspond to K-means, spectral clustering, Gaussian mixture models, and Ward's linkage. The algorithms in group (6, 8–10) correspond to CHAMELEON algorithm with different objective functions.

shown in Fig. 10a. It is not surprising to see that the related algorithms are clustered together. For a visualization of the similarity between the algorithms, the 35 algorithms are also embedded in a two-dimensional space; this is achieved by applying the Sammon's projection algorithm (Sammon, 1969) to the 35×35 similarity matrix. Fig. 10b shows that all the CHAMELEON variations (6, 8–10) are clustered into a single cluster. This plot suggests that the clustering algorithms following the same clustering strategy result in similar clustering in spite of minor variations in the parameters or objective functions involved. In (Meila, 2003), a different metric in the space of clusterings, termed Variation of Information, was proposed. It measures the similarity between two clustering algorithms by the amount of information lost or gained when choosing one clustering over the other.

Clustering algorithms can also be compared at the theoretical level based on their objective functions. In order to perform such a comparison, a distinction should be made between a *clustering method* and a *clustering algorithm* (Jain and Dubes, 1988). A clustering method is a general strategy employed to solve a clustering problem. A clustering algorithm, on the other hand, is simply an instance of a method. For instance, minimizing the squared error is a clustering method, and there are many different clustering algorithms, including K-means, that implement the minimum squared error method. Some equivalence relationships even between different clustering methods have been shown. For example, Dhillon et al. (2004) show that spectral methods and kernel K-means are equivalent; for a choice of kernel in spectral clustering, there exists a kernel for which the objective functions of kernel K-means and spectral clustering are the same. The equivalence between non-negative matrix factorization for clustering and kernel K-means algorithm is shown in (Ding et al., 2005). All these methods are directly related to the analysis of eigenvectors of the similarity matrix.

The above discussion underscores one of the important facts about clustering; *there is no best clustering algorithm*. Each clustering algorithm imposes a structure on the data either explicitly or implicitly. When there is a good match between the model and the data, good partitions are obtained. Since the structure of the data is not known a priori, one needs to try competing and diverse approaches to determine an appropriate algorithm for the clustering task at hand. This idea of no best clustering algorithm is partially captured by the impossibility theorem (Kleinberg, 2002), which states that no single clustering algorithm simultaneously satisfies a set of basic axioms of data clustering.

3.6. Admissibility analysis of clustering algorithms

Fisher and vanNess (1971) formally analyzed clustering algorithms with the objective of comparing them and providing guidance in choosing a clustering procedure. They defined a set of *admissibility criteria* for clustering algorithms that test the sensitivity of clustering algorithms with respect to the changes that do not alter the essential structure of the data. A clustering is called *A-admissible* if it satisfies criterion *A*. Example criteria include *convex, point and cluster proportion, cluster omission*, and *monotone*. They are briefly described below.

- *Convex:* A clustering algorithm is *convex-admissible* if it results in a clustering where the convex hulls of clusters do not intersect.
- *Cluster proportion:* A clustering algorithm is *cluster-proportion admissible* if the cluster boundaries do not alter even if some of the clusters are duplicated an arbitrary number of times.
- *Cluster omission:* A clustering algorithm is *omission-admissible* if by removing one of the clusters from the data and re-running the algorithm, the clustering on the remaining $K-1$ clusters is identical to the one obtained on them with K clusters.
- *Monotone:* A clustering algorithm is *monotone-admissible* if the clustering results do not change when a monotone transformation is applied to the elements of the similarity matrix.

Fisher and Van Ness proved that one cannot construct algorithms that satisfy certain admissibility criteria. For example, if an algorithm is monotone-admissible, it cannot be a hierarchical clustering algorithm.

Kleinberg (2002) addressed a similar problem, where he defined three criteria:

- *Scale invariance:* An arbitrary scaling of the similarity metric must not change the clustering results.
- *Richness:* The clustering algorithm must be able to achieve all possible partitions on the data.
- *Consistency:* By shrinking within-cluster distances and stretching between-cluster distances, the clustering results must not change.

Kleinberg also provides results similar to that of (Fisher and vanNess, 1971), showing that it is impossible to construct an algorithm that satisfies all these properties, hence the title of his paper

"An Impossibility Theorem for Clustering". Further discussions in (Kleinberg, 2002) reveal that a clustering algorithm can indeed be designed by relaxing the definition of *satisfying* a criterion to *nearly-satisfying* the criterion. While the set of axioms defined here are reasonable to a large extent, they are in no way the only possible set of axioms, and hence the results must be interpreted accordingly (Ben-David and Ackerman, 2008).

4. Trends in data clustering

Information explosion is not only creating large amounts of data but also a diverse set of data, both *structured* and *unstructured*. *Unstructured data* is a collection of objects that do not follow a specific format. For example, images, text, audio, video, etc. On the other hand, in *structured data,* there are semantic relationships within each object that are important. Most clustering approaches ignore the structure in the objects to be clustered and use a feature vector based representation for both structured and unstructured data. The traditional view of data partitioning based on vector-based feature representation does not always serve as an adequate framework. Examples include objects represented using sets of points (Lowe, 2004), consumer purchase records (Guha et al., 2000), data collected from questionnaires and rankings (Critchlow, 1985), social networks (Wasserman and Faust, 1994), and data streams (Guha et al., 2003b). Models and algorithms are being developed to process huge volumes of heterogeneous data. A brief summary of some of the recent trends in data clustering is presented below.

4.1. Clustering ensembles

The success of ensemble methods for supervised learning has motivated the development of ensemble methods for unsupervised learning (Fred and Jain, 2002). The basic idea is that by taking *multiple looks* at the same data, one can generate multiple partitions (*clustering ensemble*) of the same data. By combining the resulting partitions, it is possible to obtain a good data partitioning even when the clusters are not compact and well separated. Fred and Jain used this approach by taking an ensemble of partitions obtained by K-means; the ensemble was obtained by changing the value of K and using random cluster initializations. These partitions were then combined using a co-occurrence matrix that resulted in a good separation of the clusters. An example of a clustering ensemble is shown in Fig. 11 where a "two-spiral" dataset is used to demonstrate its effectiveness. K-means is run multiple, say N, times with varying values of the number of clusters *K*. The new similarity between a pair of points is defined as the number of times the two points co-occur in the same cluster in N runs of K-means. The final clustering is obtained by clustering the data based on the new pair-wise similarity. Strehl and Ghosh (2003) proposed several probabilistic models for integrating multiple partitions. More recent work on cluster ensembles can be found in (Hore et al., 2009a).

There are many different ways of generating a clustering ensemble and then combining the partitions. For example, multiple data partitions can be generated by: (i) applying different clustering algorithms, (ii) applying the same clustering algorithm with different values of parameters or initializations, and (iii) combining of different data representations (feature spaces) and clustering algorithms. The evidence accumulation step that combines the information provided by the different partitions can be viewed as learning the similarity measure among the data points.

4.2. Semi-supervised clustering

Clustering is inherently an ill-posed problem where the goal is to partition the data into some unknown number of clusters based on intrinsic information alone. The data-driven nature of clustering makes it very difficult to design clustering algorithms that will correctly find clusters in the given data. Any external or *side information* available along with the $n \times d$ pattern matrix or the $n \times n$ similarity matrix can be extremely useful in finding a good partition of data. Clustering algorithms that utilize such side information are said to be operating in a *semi-supervised mode* (Chapelle et al., 2006). There are two open questions: (i) how should the side information be specified? and (ii) how is it obtained in practice? One of the most common methods of specifying the side information is in the form of pair-wise constraints. A *must-link constraint* specifies that the point pair connected by the constraint belong to the same cluster. On the other hand, a *cannot-link constraint* specifies that the point pair connected by the constraint do not belong to the same cluster. It is generally assumed that the con-

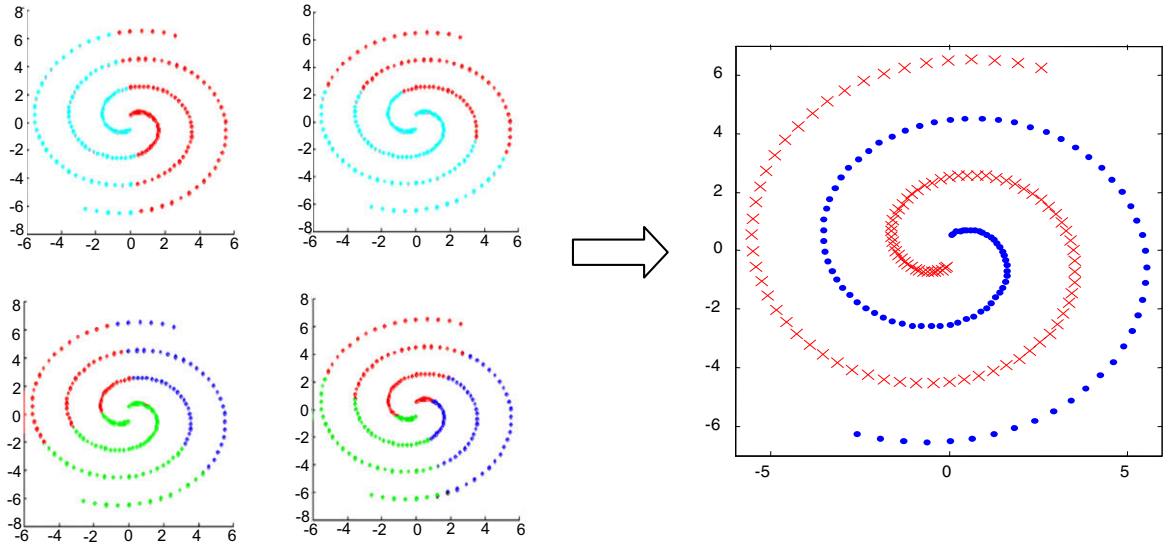


Fig. 11. Clustering ensembles. Multiple runs of K-means are used to learn the pair-wise similarity using the "co-occurrence" of points in clusters. This similarity can be used to detect arbitrary shaped clusters.

straints are provided by the domain expert. There is limited work on automatically deriving constraints from the data. Some attempts to derive constraints from domain ontology and other external sources into clustering algorithms include the usage of WordNet ontology, gene ontology, Wikipedia, etc. to guide clustering solutions. However, these are mostly feature constraints and not constraints on the instances (Hotho et al., 2003; Liu et al., 2004; Banerjee et al., 2007b). Other approaches for including side information include (i) "seeding", where some labeled data is used along with large amount of unlabeled data for better clustering (Basu et al., 2002) and (ii) methods that allow encouraging or discouraging some links (Law et al., 2005; Figueiredo et al., 2006).

Fig. 12 illustrates the semi-supervised learning in an image segmentation application (Lange et al., 2005). The textured image to be segmented (clustered) is shown in Fig. 12a. In addition to the image, a set of user-specified pair-wise constraints on the pixel labels are also provided. Fig. 12b shows the clustering obtained when no constraints are used, while Fig. 12c shows improved clustering with the use of constraints. In both the cases, the number of clusters was assumed to be known ($K = 5$).

Most approaches (Bar-Hillel et al., 2003; Basu et al., 2004; Chapelle et al., 2006; Lu and Leen, 2007) to semi-supervised clustering modify the objective function of existing clustering algorithms to incorporate the pair-wise constraints. It is desirable to have an approach to semi-supervised clustering that can improve the performance of an already existing clustering algorithm without modifying it. *BoostCluster* (Liu et al., 2007) adopts this philosophy and follows a boosting framework to improve the performance of any given clustering algorithm using pair-wise constraints. It iteratively modifies the input to the clustering algorithm by generating new data representations (transforming the $n \times n$ similarity matrix) such that the pair-wise constraints are satisfied while also maintaining the integrity of the clustering output. Fig. 13 shows the performance of BoostCluster evaluated on handwritten digit database in the UCI repository (Blake, 1998) with 4000 points in 256-dimensional feature space. BoostCluster is able to improve the performance of all the three commonly used clustering algorithms, K-means, single-link, and Spectral clustering as pair-wise constraints are added to the data. Only must-link constraints are specified here and the number of true clusters is assumed to be known ($K = 10$).

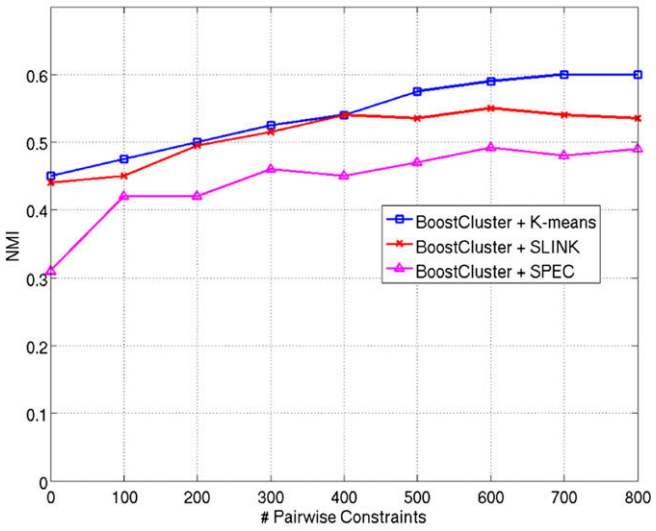


Fig. 13. Performance of BoostCluster (measured using Normalized Mutual Information (NMI)) as the number of pair-wise constraints is increased. The three plots correspond to boosted performance of K-means, Single-Link (SLINK), and Spectral clustering (SPEC).

4.3. Large-scale clustering

Large-scale data clustering addresses the challenge of clustering millions of data points that are represented in thousands of features. Table 1 shows a few examples of real-world applications for large-scale data clustering. Below, we review the application of large-scale data clustering to content-based image retrieval.

The goal of Content Based Image Retrieval (CBIR) is to retrieve visually similar images to a given query image. Although the topic has been studied for the past 15 years or so, there has been only limited success. Most early work on CBIR was based on computing *color*, *shape*, and *texture* based features and using them to define a similarity between the images. A 2008 survey on CBIR highlights the different approaches used for CBIR through time (Datta et al., 2008). Recent approaches for CBIR use key point based features. For example, SIFT (Lowe, 2004) descriptors can be used to represent the images (see Fig. 14). However, once the size of the image database increases (~10 million), and assuming 10 ms to compute the matching score between an image pair, a linear search would take approximately 30 h to answer one query. This clearly is unacceptable.

On the other hand, text retrieval applications are much faster. It takes about one-tenth of a second to search 10 billion documents in Google. A novel approach for image retrieval is to convert the problem into a text retrieval problem. The key points from all the images are first clustered into a large number of clusters (which is usually much less than the number of key points

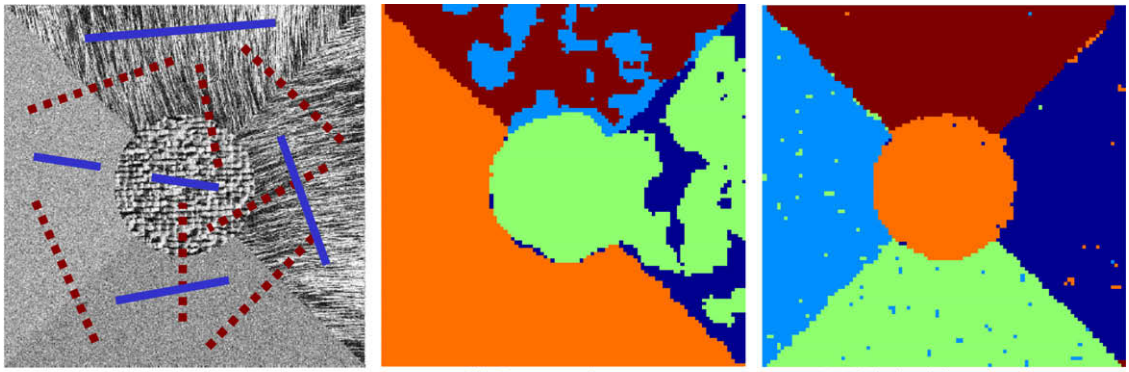


Fig. 12. Semi-supervised learning. (a) Input image consisting of five homogeneous textured regions; examples of must-link (solid blue lines) and must not link (broken red lines) constraints between pixels to be clustered are specified. (b) 5-Cluster solution (segmentation) without constraints. (c) Improved clustering (with five clusters) with 10% of the data points included in the pair-wise constraints (Lange et al., 2005).

Table 1
Example applications of large-scale data clustering.

Application	Description	# Objects	# Features
Document clustering	Group documents of similar topics (Andrews et al., 2007)	10^6	10^4
Gene clustering	Group genes with similar expression levels (Lukashin et al., 2003)	10^5	10^2
Content-based image retrieval	Quantize low-level image features (Philbin et al., 2007)	10^9	10^2
Clustering of earth science data	Derive climate indices (Steinbach et al., 2003)	10^5	10^2

themselves). These are called *visual words*. An image is then represented by a histogram of visual words, i.e., the number of key-points from the image that are in each word or each cluster. By representing each image by a histogram of visual words, we can then cast the problem of image search into a problem of text retrieval and exploit text search engines for efficient image retrieval. One of the major challenges in quantizing key points is the number of objects to be clustered. For a collection of 1000 images with an average of 1000 key points and target number of 5000 visual words, it requires clustering 10^6 objects into 5000 clusters.

A large number of clustering algorithms have been developed to efficiently handle large-size data sets. Most of these studies can be classified into four categories:

- *Efficient Nearest Neighbor (NN) Search:* One of the basic operations in any data clustering algorithm is to decide the cluster membership of each data point, which requires NN search. Algorithms for efficient NN search are either tree-based (e.g. kd-tree (Moore, 1998; Muja and Lowe, 2009)) or random projection based (e.g., Locality Sensitive Hash (Buhler, 2001)).
- *Data summarization:* The objective here is to improve the clustering efficiency by first summarizing a large data set into a relatively small subset, and then applying the clustering algorithms to the summarized data set. Example algorithms include BIRCH (Zhang et al., 1996), divide-and-conquer (Steinbach et al., 2000), coreset K-means (Har-peled and Mazumdar, 2004), and coarsening methods (Karypis and Kumar, 1995).
- *Distributed computing:* Approaches in this category (Dhillon and Modha, 1999) divide each step of a data clustering algorithm into a number of procedures that can be computed independently. These independent computational procedures will then be carried out in parallel by different processors to reduce the overall computation time.
- *Incremental clustering:* These algorithms, for example (Bradley et al., 1998) are designed to operate in a single pass over data points to improve the efficiency of data clustering. This is in contrast to most clustering algorithms that require multiple passes over data points before identifying the cluster centers. COBWEB is a popular hierarchical clustering algorithm that does a single pass through the available data and arranges it into a classification tree incrementally (Fisher, 1987).
- *Sampling-based methods:* Algorithms like CURE (Guha et al., 1998; Kollios et al., 2003) subsample a large dataset selectively, and perform clustering over the smaller set, which is later transferred to the larger dataset.

4.4. Multi-way clustering

Objects or entities to be clustered are often formed by a combination of *related* heterogeneous components. For example, a document is made of words, title, authors, citations, etc. While objects can be converted into a pooled feature vector of its components prior to clustering, it is not a natural representation of the objects and may result in poor clustering performance.

Co-clustering (Hartigan, 1972; Mirkin, 1996) aims to cluster both features and instances of the data (or both rows and columns of the $n \times d$ pattern matrix) simultaneously to identify the subset of features where the resulting clusters are meaningful according to certain evaluation criterion. This problem was first studied under the name *direct clustering* by Hartigan (1972). It is also called *bi-dimensional clustering* (Cheng et al., 2000), *double clustering*,

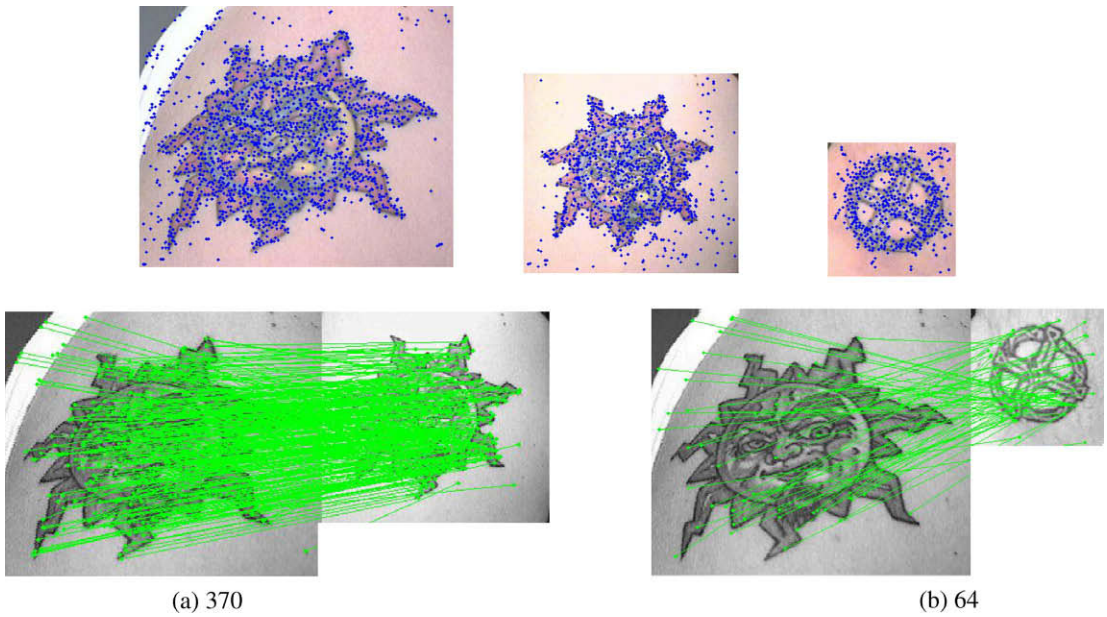


Fig. 14. Three tattoo images represented using SIFT key points. (a) A pair of similar images has 370 matching key points; (b) a pair of different images has 64 matching key points. The green lines show the matching key-points between the images (Lee et al., 2008).

coupled clustering, or *bimodal clustering*. This notion is also related to subspace clustering where all the clusters are identified in a common subspace. Co-clustering is most popular in the field of bioinformatics, especially in gene clustering, and has also been successfully applied to document clustering (Slonim and Tishby, 2000; Dhillon et al., 2003).

The co-clustering framework was extended to *multi-way clustering* in (Bekkerman et al., 2005) to cluster a set of objects by simultaneously clustering their heterogeneous components. Indeed, the problem is much more challenging because different pairs of components may participate in different types of similarity relationships. In addition, some relations may involve more than two components. Banerjee et al. (2007a) present a family of multi-way clustering schemes that is applicable to a class of loss functions known as Bregman divergences. Sindhwani et al. (2008) apply semi-supervised learning in the co-clustering framework.

4.5. Heterogeneous data

In traditional pattern recognition settings, a feature vector consists of measurements of different properties of an object. This representation of objects is not a natural representation for several types of data. *Heterogeneous data* refers to the data where the objects may not be *naturally* represented using a fixed length feature vector.

Rank data: Consider a dataset generated by ranking of a set of n movies by different people; only some of the n objects are ranked. The task is to cluster the users whose rankings are similar and also to identify the 'representative rankings' of each group (Mallows, 1957; Critchlow, 1985; Busse et al., 2007).

Dynamic data: Dynamic data, as opposed to static data, can change over the course of time e.g., blogs, Web pages, etc. As the data gets modified, clustering must be updated accordingly. A *data stream* is a kind of dynamic data that is transient in nature, and cannot be stored on a disk. Examples include network packets received by a router and stock market, retail chain, or credit card transaction streams. Characteristics of the data streams include their high volume and potentially unbounded size, sequential access, and dynamically evolving nature. This imposes additional requirements to traditional clustering algorithms to rapidly process and summarize the massive amount of continuously arriving data. It also requires the ability to adapt to changes in the data distribution, the ability to detect emerging clusters and distinguish them from outliers in the data, and the ability to merge old clusters or discard expired ones. All of these requirements make data stream clustering a significant challenge since they are expected to be single-pass algorithms (Guha et al., 2003b). Because of the high-speed processing requirements, many of the data stream clustering methods (Guha et al., 2003a; Aggarwal et al., 2003; Cao et al., 2006; Hore et al., 2009b) are extensions of simple algorithms such as K-means, K-medoid, fuzzy c-means, or density-based clustering, modified to work in a data stream environment setting.

Graph data: Several objects, such as chemical compounds, protein structures, etc. can be represented most naturally as graphs. Many of the initial efforts in graph clustering focused on extracting graph features to allow existing clustering algorithms to be applied to the graph feature vectors (Tsuda and Kudo, 2006). The features can be extracted based on patterns such as frequent subgraphs, shortest paths, cycles, and tree-based patterns. With the emergence of kernel learning, there have been growing efforts to develop kernel functions that are more suited for graph-based data (Kashima et al., 2003). One way to determine the similarity between graphs is by aligning their corresponding adjacency matrix representations (Umeyama, 1988).

Relational data: Another area that has attracted considerable interest is clustering relational (network) data. Unlike the clustering of graph data, where the objective is to partition a collection of graphs into disjoint groups, the task here is to partition a large graph (i.e., network) into cohesive subgraphs based on their link structure and node attributes. The problem becomes even more complicated when the links (which represent relations between objects) are allowed to have diverse types. One of the key issues is to define an appropriate clustering criterion for relational data. A general probabilistic model for relational data was first proposed in (Taskar et al., 2001), where different related entities are modeled as distributions conditioned on each other. Newman's modularity function (Newman and Girvan, 2004; Newman, 2006) is a widely-used criterion for finding community structures in networks, but the measure considers only the link structure and ignores attribute similarities. A spectral relaxation to Newman and Girvan's objective function (Newman and Girvan, 2004) for network graph clustering is presented in (White and Smyth, 2005). Since real networks are often dynamic, another issue is to model the evolutionary behavior of networks, taking into account changes in the group membership and other characteristic features (Backstrom et al., 2006).

5. Summary

Organizing data into sensible groupings arises naturally in many scientific fields. It is, therefore, not surprising to see the continued popularity of data clustering. It is important to remember that cluster analysis is an exploratory tool; the output of clustering algorithms only suggest hypotheses. While numerous clustering algorithms have been published and new ones continue to appear, there is no single clustering algorithm that has been shown to dominate other algorithms across all application domains. Most algorithms, including the simple K-means, are admissible algorithms. With the emergence of new applications, it has become increasingly clear that the task of seeking the best clustering principle might indeed be futile. As an example, consider the application domain of enterprise knowledge management. Given the same set of document corpus, different user groups (e.g., legal, marketing, management, etc.) may be interested in generating partitions of documents based on their respective needs. A clustering method that satisfies the requirements for one group of users may not satisfy the requirements of another. As mentioned earlier, "clustering is in the eye of the beholder" – so indeed data clustering must involve the user or application needs.

Clustering has numerous success stories in data analysis. In spite of this, machine learning and pattern recognition communities need to address a number of issues to improve our understanding of data clustering. Below is a list of problems and research directions that are worth focusing in this regard.

(a) There needs to be a suite of benchmark data (with ground truth) available for the research community to test and evaluate clustering methods. The benchmark should include data sets from various domains (documents, images, time series, customer transactions, biological sequences, social networks, etc.). Benchmark should also include both static and dynamic data (the latter would be useful in analyzing clusters that change over time), quantitative and/or qualitative attributes, linked and non-linked objects, etc. Though the idea of providing a benchmark data is not new (e.g., UCI ML and KDD repository), current benchmarks are limited to small, static data sets.

(b) We need to achieve a tighter integration between clustering algorithms and the application needs. For example, some applications may require generating only a few cohesive clusters (less cohesive clusters can be ignored), while others

may require the best partition of the entire data. In most applications, it may not necessarily be the best clustering algorithm that really matters. Rather, it is more crucial to choose the right feature extraction method that identifies the underlying clustering structure of the data.

(c) Regardless of the principle (or objective), most clustering methods are eventually cast into combinatorial optimization problems that aim to find the partitioning of data that optimizes the objective. As a result, computational issue becomes critical when the application involves large-scale data. For instance, finding the global optimal solution for K-means is NP-hard. Hence, it is important to choose clustering principles that lead to computationally efficient solutions.

(d) A fundamental issue related to clustering is its stability or consistency. A good clustering principle should result in a data partitioning that is stable with respect to perturbations in the data. We need to develop clustering methods that lead to stable solutions.

(e) Choose clustering principles according to their satisfiability of the stated axioms. Despite Kleinberg's impossibility theorem, several studies have shown that it can be overcome by relaxing some of the axioms. Thus, maybe one way to evaluate a clustering principle is to determine to what degree it satisfies the axioms.

(f) Given the inherent difficulty of clustering, it makes more sense to develop semi-supervised clustering techniques in which the labeled data and (user specified) pair-wise constraints can be used to decide both (i) data representation and (ii) appropriate objective function for data clustering.

Acknowledgements

I would like to acknowledge the National Science Foundation and the Office of Naval research for supporting my research in data clustering, dimensionality reduction, classification, and semi-supervised learning. I am grateful to Rong Jin, Pang-Ning Tan, and Pavan Mallapragada for helping me prepare the King-Sun Fu lecture as well as this manuscript. I have learned much from and enjoyed my fruitful collaborations in data clustering with Eric Backer, Joachim Buhmann, (late) Richard Dubes, Mario Figueiredo, Patrick Flynn, Ana Fred, Martin Law, J.C. Mao, M. Narasimha Murty, Steve Smith, and Alexander Topchy. Joydeep Ghosh, Larry Hall, Jianying Hu, Mario Figueredo, and Ana Fred provided many useful suggestions to improve the quality of this paper.

References

Aggarwal, Charu C., Han, Jiawei, Wang, Jianyong, Yu, Philip S., 2003. A framework for clustering evolving data streams. In: Proc. 29th Internat. Conf. on Very Large Data Bases, pp. 81–92.

Agrawal, Rakesh, Gehrke, Johannes, Gunopulos, Dimitrios, Raghavan, Prabhakar, 1998. Automatic subspace clustering of high dimensional data for data mining applications. In: Proc. ACM SIGMOD, pp. 94–105.

Anderberg, M.R., 1973. Cluster Analysis for Applications. Academic Press.

Andrews, Nicholas, O., Fox, Edward A., 2007. Recent developments in document clustering. Technical report TR-07-35. Deparment of Computer Science, Virginia Tech.

Arabie, P., Hubert, L., 1994. Cluster analysis in marketing research. In: Advanced Methods in Marketing Research. Blackwell, Oxford, pp. 160–189.

Backer, Eric, 1978. Cluster Analysis by Optimal Decomposition of Induced Fuzzy Sets. Delft University Press.

Backstrom, L., Huttenlocher, D., Kleinberg, J., Lan, X., 2006. Group formation in large social networks: Membership, growth, and evolution. In: Proc. 12th KDD.

Baldi, P., Hatfield, G., 2002. DNA Microarrays and Gene Expression. Cambridge University Press.

Ball, G., Hall, D., 1965. ISODATA, a novel method of data anlysis and pattern classification. Technical report NTIS AD 699616. Stanford Research Institute, Stanford, CA.

Banerjee, Arindam, Merugu, Srujana, Dhillon, Inderjit, Ghosh, Joydeep. 2004. Clustering with bregman divergences. J. Machine Learn. Res., 234–245.

Banerjee, Arindam, Basu, Sugato, Merugu, Srujana, 2007a. Multi-way clustering on relation graphs. In: Proc. 7th SIAM Internat. Conf. on Data Mining.

Banerjee, S., Ramanathan, K., Gupta, A., 2007b. Clustering short texts using Wikipedia. In: Proc. SIGIR.

Bar-Hillel, Aaron, Hertz, T., Shental, Noam, Weinshall, Daphna, 2003. Learning distance functions using equivalence relations. In: Proc. 20th Internat. Conf. on Machine Learning, pp. 11–18.

Basu, Sugato, Banerjee, Arindam, Mooney, Raymond, 2002. Semi-supervised clustering by seeding. In: Proc. 19th Internat. Conf. on Machine Learning.

Basu, Sugato, Bilenko, Mikhail, Mooney, Raymond J., 2004. A probabilistic framework for semi-supervised clustering. In: Proc. 10th KDD, pp. 59–68.

Basu, Sugato, Davidson, Ian, Wagstaff, Kiri (Eds.), 2008. Constrained Clustering: Advances in Algorithms, Theory and Applications. Data Mining and Knowledge Discovery, vol. 3, Chapman & Hall/CRC.

Bekkerman, Ron, El-Yaniv, Ran, McCallum, Andrew, 2005. Multi-way distributional clustering via pairwise interactions. In: Proc. 22nd Internat. Conf. Machine Learning, pp. 41–48.

Belkin, Mikhail, Niyogi, Partha, 2002. Laplacian eigenmaps and spectral techniques for embedding and clustering. Advances in Neural Information Processing Systems, vol. 14. pp. 585–591.

Ben-David, S., Ackerman, M., 2008. Measures of clustering quality: A working set of axioms for clustering. Advances in Neural Information Processing Systems.

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

Bhatia, S., Deogun, J., 1998. Conceputal clustering in information retrieval. IEEE Trans. Systems Man Cybernet. 28 (B), 427–436.

Bishop, Christopher M., 2006. Pattern Recognition and Machine Learning. Springer.

Blake, Merz C.J., 1998. UCI repository of machine learning databases.

Blei, D.M., Ng, A.Y., Jordan, M.I., 2003. Latent dirichlet allocation. J. Machine Learn. Res. 3, 993–1022.

Bradley, P.S., Fayyad, U., Reina, C., 1998. Scaling clustering algorithms to large databases. In: Proc. 4th KDD.

Buhler, J., 2001. Efficient large-scale sequence comparison by locality-sensitive hashing. Bioinformatics 17 (5), 419–428.

Busse, Ludwig M., Orbanz, Peter, Buhmann, Joachim M., 2007. Cluster analysis of heterogeneous rank data. In: Proc. 24th Internat. Conf. on Machine Learning, pp. 113–120.

Cao, F., Ester, M., Qian, W., Zhou, A., 2006. Density-based clustering over an evolving data stream with noise. In: Proc. SIAM Conf. Data Mining.

Chapelle, O., Schölkopf, B., Zien, A. (Eds.), 2006. Semi-Supervised Learning. MIT Press, Cambridge, MA.

Cheng, Yizong, Church, George M., 2000. Biclustering of expression data. In: Proc. Eighth Internat. Conf. on Intelligent Systems for Molecular Biology, AAAI Press, pp. 93–103.

Connell, S.D., Jain, A.K., 2002. Writer adaptation for online handwriting recognition. IEEE Trans. Pattern Anal. Machine Intell. 24 (3), 329–346.

Critchlow, D., 1985. Metric Methods for Analyzing Partially Ranked Data. Springer.

Datta, Ritendra, Joshi, Dhiraj, Li, Jia, Wang, James Z., 2008. Image retrieval: Ideas, influences, and trends of the new age. ACM Computing Surveys 40 (2) (Article 5).

Dempster, A.P., Laird, N.M., Rubin, D.B., 1977. Maximum likelihood from incomplete data via the EM algorithm. J. Roy. Statist. Soc. 39, 1–38.

Dhillon, I., Modha, D., 1999. A data-clustering algorithm on distributed memory multiprocessors. In: Proc. KDD'99 Workshop on High Performance Knowledge Discovery, pp. 245–260.

Dhillon, Inderjit S., Mallela, Subramanyam, Guyon, Isabelle, Elisseeff, André, 2003. A divisive information-theoretic feature clustering algorithm for text classification. J. Machine Learn. Res. 3, 2003.

Dhillon, Inderjit S., Guan, Yuqiang, Kulis, Brian, 2004. Kernel *k*-means: Spectral clustering and normalized cuts. In: Proc. 10th KDD, pp. 551–556.

Ding, Chris, He, Xiaofeng, Simon, Horst D., 2005. On the equivalence of nonnegative matrix factorization and spectral clustering. In: Proc. SIAM Internat. Conf. on Data Mining, pp. 606–610.

Drineas, P., Frieze, A., Kannan, R., Vempala, S., Vinay, V., 1999. Clustering large graphs via the singular value decomposition. Machine Learn. 56 (1–3), 9–33.

Dubes, Richard C., Jain, Anil K., 1976. Clustering techniques: User's dilemma. Pattern Recognition, 247–260.

Duda, R., Hart, P., Stork, D., 2001. Pattern Classification, second ed. John Wiley and Sons, New York.

Dunn, J.C., 1973. A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. J. Cybernet. 3, 32–57.

Eschrich, S., Ke, Jingwei, Hall, L.O., Goldgof, D.B., 2003. Fast accurate fuzzy clustering through data reduction. IEEE Trans. Fuzzy Systems 11 (2), 262–270.

Ester, Martin, Peter Kriegel, Hans, S., Jörg, Xu, Xiaowei, 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proc. 2nd KDD, AAAI Press.

Ferguson, Thomas S., 1973. A Bayesian analysis of some nonparametric problems. Ann. Statist. 1, 209–230.

Figueiredo, Mario, Jain, Anil K., 2002. Unsupervised learning of finite mixture models. IEEE Trans. Pattern Anal. Machine Intell. 24 (3), 381–396.

Figueiredo, M.A.T., Chang, D.S., Murino, V., 2006. Clustering under prior knowledge with application to image segmentation. Adv. Neural Inform. Process. Systems 19, 401–408.

Fisher, Douglas H., 1987. Knowledge acquisition via incremental conceptual clustering. Machine Learn., 139–172.
Fisher, L., vanNess, J., 1971. Admissible clustering procedures. Biometrika.
Forgy, E.W., 1965. Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications. Biometrics 21, 768–769.
Frank, Ildiko E., Todeschini, Roberto, 1994. Data Analysis Handbook. Elsevier Science Inc., pp. 227–228.
Fred, A., Jain, A.K., 2002. Data clustering using evidence accumulation. In: Proc. Internat. Conf. Pattern Recognition (ICPR).
Frigui, H., Krishnapuram, R., 1999. A robust competitive clustering algorithm with applications in computer vision. IEEE Trans. Pattern Anal. Machine Intell. 21, 450–465.
Gantz, John F., 2008. The diverse and exploding digital universe. Available online at: <http://www.emc.com/collateral/analyst-reports/diverse-exploding-digital-universe.pdf>.
Google Scholar, 2009 (February). Google Scholar. <http://scholar.google.com>.
Guha, Sudipto, Rastogi, Rajeev, Shim, Kyuseok, 1998. CURE: An efficient clustering algorithm for large databases. In: Proc. ICDM., pp. 73–84.
Guha, Sudipto, Rastogi, Rajeev, Shim, Kyuseok, 2000. Rock: A robust clustering algorithm for categorical attributes. Inform. Systems 25 (5), 345–366.
Guha, Sudipto, Meyerson, A., Mishra, Nina, Motwani, Rajeev, O'Callaghan, L., 2003a. Clustering data streams: Theory and practice. Trans. Knowledge Discovery Eng.
Guha, Sudipto, Mishra, Nina, Motwani, Rajeev, 2003b. Clustering data streams. IEEE Trans. Knowledge Data Eng. 15 (3), 515–528.
Hagen, L., Kahng, A.B., 1992. New spectral methods for ratio cut partitioning and clustering. IEEE Trans. Comput.-Aid. Des. Integrated Circuits Systems 11 (9), 1074–1085.
Han, Jiawei, Kamber, Micheline, 2000. Data Mining: Concepts and Techniques. Morgan Kaufmann.
Hansen, Mark H., Yu, Bin, 2001. Model selection and the principle of minimum description length. J. Amer. Statist. Assoc. 96 (454), 746–774.
Har-peled, Sariel, Mazumdar, Soham, 2004. Coresets for *k*-means and *k*-median clustering and their applications. In: Proc. 36th Annu. ACM Sympos. Theory Comput., pp. 291–300.
Hartigan, J.A., 1972. Direct clustering of a data matrix. J. Amer. Statist. Assoc. 67 (337), 123–132.
Hartigan, J.A., 1975. Clustering Algorithms. John Wiley and Sons.
Hofmann, T., Buhmann, J.M., 1997. Pairwise data clustering by deterministic annealing. IEEE Trans. Pattern Anal. Machine Intell. 19 (1), 1–14.
Hore, Prodip, Hall, Lawrence O., Goldgof, Dmitry B., 2009a. A scalable framework for cluster ensembles. Pattern Recognition 42 (5), 676–688.
Hore, Prodip, Hall, Lawrence O., Goldgof, Dmitry B., Gu, Yuhua, Maudsley, Andrew A., Darkazanli, Ammar, 2009b. A scalable framework for segmenting magnetic resonance images. J. Signal Process. Systems 54 (1–3), 183–203.
Hotho, A., Staab, S., Stumme, G., 2003. Ontologies to improve text document clustering. In: Proc. of the ICDM.
Hu, J., Ray, B.K., Singh, M., 2007. Statistical methods for automated generation of service engagement staffing plans. IBM J. Res. Dev. 51 (3), 281–293.
Iwayama, M., Tokunaga, T., 1995. Cluster-based text categorization: A comparison of category search strategies. In: Proc. 18th ACM Internat. Conf. on Research and Development in Information Retrieval, pp. 273–281.
Jain, Anil K., Dubes, Richard C., 1988. Algorithms for Clustering Data. Prentice Hall.
Jain, Anil K., Flynn, P., 1996. Image segmentation using clustering. In: Advances in Image Understanding. IEEE Computer Society Press, pp. 65–83.
Jain, A.K., Topchy, A., Law, M.H.C., Buhmann, J.M., 2004. Landscape of clustering algorithms. In: Proc. Internat. Conf. on Pattern Recognition, vol. 1, pp. 260–263.
JSTOR, 2009. JSTOR. <http://www.jstor.org>.
Karypis, George, Kumar, Vipin, 1995. A fast and high quality multilevel scheme for partitioning irregular graphs. In: Proc. Internat. Conf. on Parallel Processing, pp. 113–122.
Kashima, H., Tsuda, K., Inokuchi, A., 2003. Marginalized Kernels between labeled graphs. In: Proc. 20th Internat. Conf. on Machine Learning, pp. 321–328.
Kashima, H., Hu, J., Ray, B., Singh, M., 2008. K-means clustering of proportional data using L1 distance. In: Proc. Internat. Conf. on Pattern Recognition, pp. 1–4.
Kaufman, Leonard, Rousseeuw, Peter J., 2005. Finding groups in data: An introduction to cluster analysis. Wiley series in Probability and Statistics.
Kleinberg, Jon, 2002. An impossibility theorem for clustering. In: NIPS 15. pp. 463–470.
Kollios, G., Gunopulos, D., Koudas, N., Berchtold, S., 2003. Efficient biased sampling for approximate clustering and outlier detection in large data sets. IEEE Trans. Knowledge Data Eng. 15 (5), 1170–1187.
Lange, Tilman, Roth, Volker, Braun, Mikio L., Buhmann, Joachim M., 2004. Stability-based validation of clustering solutions. Neural Comput. 16 (6), 1299–1323.
Lange, T., Law, M.H., Jain, A.K., Buhmann, J., 2005. Learning with constrained and unlabelled data. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognition 1, 730–737.
Law, Martin, Topchy, Alexander, Jain, A.K., 2005. Model-based clustering with probabilistic constraints.In: Proc. SIAM Conf. on Data Mining, pp. 641–645.
Lee, Jung-Eun, Jain, Anil K., Jin, Rong, 2008. Scars, marks and tattoos (SMT): Soft biometric for suspect and victim identification. In: Proceedings of the Biometric Symposium.
Li, W., McCallum, A., 2006. Pachinko allocation: Dag-structured mixture models of topic correlations. In: Proc. 23rd Internat. Conf. on Machine Learning, pp. 577–584.
Linde, Y., Buzo, A., Gray, R., 1980. An algorithm for vector quantizer design. IEEE Trans. Comm. 28, 84–94.
Liu, J., Wang, W., Yang, J., 2004. A framework for ontology-driven subspace clustering. In: Proc. KDD.
Liu, Yi, Jin, Rong, Jain, A.K., 2007. Boostcluster: Boosting clustering by pairwise constraints. In: Proc. 13th KDD, pp. 450–459.
Lloyd, S., 1982. Least squares quantization in PCM. IEEE Trans. Inform. Theory 28, 129–137. Originally as an unpublished Bell laboratories Technical Note (1957).
Lowe, David G., 2004. Distinctive image features from scale-invariant keypoints. Internat. J. Comput. Vision 60 (2), 91–110.
Lu, Zhengdong, Leen, Todd K., 2007. Penalized probabilistic clustering. Neural Comput. 19 (6), 1528–1567.
Lukashin, A.V., Lukashev, M.E., Fuchs, R., 2003. Topology of gene expression networks as revealed by data mining and modeling. Bioinformatics 19 (15), 1909–1916.
MacQueen, J., 1967. Some methods for classification and analysis of multivariate observations. In: Fifth Berkeley Symposium on Mathematics. Statistics and Probability. University of California Press, pp. 281–297.
Mallows, C.L., 1957. Non-null ranking models. Biometricka 44, 114–130.
Mao, J., Jain, A.K., 1996. A self-organizing network for hyper-ellipsoidal clustering (HEC). IEEE Trans. Neural Networks 7 (January), 16–29.
McLachlan, G.L., Basford, K.E., 1987. Mixture Models: Inference and Applications to Clustering. Marcel Dekker.
Meila, Marina, 2003. Comparing clusterings by the variation of information. In: COLT, pp. 173–187.
Meila, Marina, 2006. The uniqueness of a good optimum for *k*-means. In: Proc. 23rd Internat. Conf. Machine Learning, pp. 625–632.
Meila, Marina, Shi, Jianbo, 2001. A random walks view of spectral segmentation. In: Proc. AISTATAS.
Merriam-Webster Online Dictionary, 2008. Cluster analysis. <http://www.merriam-webster-online.com>.
Mirkin, Boris, 1996. Mathematical Classification and Clustering. Kluwer Academic Publishers.
Moore, Andrew W., 1998. Very fast EM-based mixture model clustering using multiresolution kd-trees. In: NIPS, pp. 543–549.
Motzkin, T.S., Straus, E.G., 1965. Maxima for graphs and a new proof of a theorem of Turan. Canadian J. Math. 17, 533–540.
Muja, M., Lowe, D.G., 2009. Fast approximate nearest neighbors with automatic algorithm configuration. In: Proc. Internat. Conf. on Computer Vision Theory and Applications (VISAPP'09).
Newman, M.E.J., 2006. Modularity and community structure in networks. In: Proc. National Academy of Sciences, USA.
Newman, M., Girvan, M., 2004. Finding and evaluating community structure in networks. Phys. Rev. E 69 (026113), 3.
Ng, Andrew Y., Jordan, Michael I., Weiss, Yair, 2001. On spectral clustering: Analysis and an algorithm. Adv. Neural Inform. Process. Systems, vol. 14. MIT Press, pp. 849–856.
Pampalk, Elias, Dixon, Simon, Widmer, Gerhard, 2003. On the evaluation of perceptual similarity measures for music. In: Proc. Sixth Internat. Conf. on Digital Audio Effects (DAFx-03). pp. 7–12.
Pavan, Massimilliano, Pelillo, Marcello, 2007. Dominant sets and pairwise clustering. IEEE Trans. Pattern Anal. Machine Intell. 29 (1), 167–172.
Pelleg Dan, Moore Andrew, 1999. Accelerating exact k-means algorithms with geometric reasoning. In: Chaudhuri Surajit, Madigan David (Eds.), Proc. Fifth Internat. Conf. on Knowledge Discovery in Databases, AAAI Press, pp. 277–281.
Pelleg, Dan, Moore, Andrew, 2000. X-means: Extending *k*-means with efficient estimation of the number of clusters. In: Proc. Seventeenth Internat. Conf. on Machine Learning. pp. 727–734.
Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A., 2007. Object retrieval with large vocabularies and fast spatial matching. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition.
Rasmussen, Carl, 2000. The infinite gaussian mixture model. Adv. Neural Inform. Process. Systems 12, 554–560.
Roberts Stephen J., Holmes, Christopher, Denison, Dave, 2001. Minimum-entropy data clustering using reversible jump Markov chain Monte Carlo. In: Proc. Internat. Conf. Artificial Neural Networks. pp. 103–110.
Sahami, Mehran, 1998. Using Machine Learning to Improve Information Access. Ph.D. Thesis, Computer Science Department, Stanford University.
Sammon Jr., J.W., 1969. A nonlinear mapping for data structure analysis. IEEE Trans. Comput. 18, 401–409.
Scholkopf, Bernhard, Smola, Alexander, Muller, Klaus-Robert, 1998. Nonlinear component analysis as a kernel eigenvalue problem. Neural Comput. 10 (5), 1299–1319.
Shamir, Ohad, Tishby, Naftali, 2008. Cluster stability for finite samples. Adv. Neural Inform. Process. Systems 20, 1297–1304.
Shi, Jianbo, Malik, Jitendra, 2000. Normalized cuts and image segmentation. IEEE Trans. Pattern Anal. Machine Intell. 22, 888–905.
Sindhwani, V., Hu, J., Mojsilovic, A., 2008. Regularized co-clustering with dual supervision. In: Advances in Neural Information Processing Systems.
Slonim, Noam, Tishby, Naftali, 2000. Document clustering using word clusters via the information bottleneck method. In: ACM SIGIR 2000, pp. 208–215.
Smith, Stephen P., Jain, Anil K., 1984. Testing for uniformity in multidimensional data. IEEE Trans. Pattern Anal. Machine Intell. 6 (1), 73–81.
Sokal, Robert R., Sneath, Peter H.A., 1963. Principles of Numerical Taxonomy. W.H. Freeman, San Francisco.
Steinbach, M., Karypis, G., Kumar, V., 2000. A comparison of document clustering techniques. In: KDD Workshop on Text Mining.

Steinbach, Michael, Tan, Pang-Ning, Kumar, Vipin, Klooster, Steve, Potter, Christopher, 2003. Discovery of climate indices using clustering. In: Proc. Ninth ACM SIGKDD Internat. Conf. on Knowledge Discovery and Data Mining.

Steinhaus, H., 1956. Sur la division des corp materiels en parties. Bull. Acad. Polon. Sci. IV (C1.III), 801–804.

Strehl, Alexander, Ghosh, Joydeep, 2003. Cluster ensembles – A knowledge reuse framework for combining multiple partitions. J. Machine Learn. Res. 3, 583–617.

Tabachnick, B.G., Fidell, L.S., 2007. Using Multivariate Statistics, fifth ed. Allyn and Bacon, Boston.

Tan, Pang-Ning, Steinbach, Michael, Kumar, Vipin, 2005. Introduction to Data Mining, first ed. Addison-Wesley Longman Publishing Co. Inc., Boston, MA, USA.

Taskar, B., Segal, E., Koller, D., 2001. Probabilistic clustering in relational data. In: Proc. Seventeenth Internat. Joint Conf. on Artificial Intelligence (IJCAI), pp. 870–887.

Tibshirani, R., Walther, G., Hastie, T., 2001. Estimating the number of clusters in a data set via the gap statistic. J. Roy. Statist. Soc. B, 411–423.

Tishby, Naftali, Pereira, Fernando C., Bialek, William, 1999. The information bottleneck method. In: Proc. 37th Allerton Conf. on Communication, Control and Computing, pp. 368–377.

Tsuda, Koji, Kudo, Taku, 2006. Clustering graphs by weighted substructure mining. In: Proc. 23rd Internat. Conf. on Machine Learning. pp. 953–960.

Tukey, John Wilder, 1977. Exploratory Data Analysis. Addison-Wesley.

Umeyama, S., 1988. An eigen decomposition approach to weighted graph matching problems. IEEE Trans. Pattern Anal. Machine Intell. 10 (5), 695–703.

von Luxburg, U., David, Ben S., 2005. Towards a statistical theory of clustering. In: Pascal Workshop on Statistics and Optimization of Clustering.

Wallace, C.S., Boulton, D.M., 1968. An information measure for classification. Comput. J. 11, 185–195.

Wallace, C.S., Freeman, P.R., 1987. Estimation and inference by compact coding (with discussions). JRSSB 49, 240–251.

Wasserman, S., Faust, K., 1994. Social Network Analysis: Methods and Applications. Cambridge University Press.

Welling, M., Rosen-Zvi, M., Hinton, G., 2005. Exponential family harmoniums with an application to information retrieval. Adv. Neural Inform. Process. Systems 17, 1481–1488.

White, Scott, Smyth, Padhraic, 2005. A spectral clustering approach to finding communities in graph. In: Proc. SIAM Data Mining.

Yu, Stella X., Shi, Jianbo, 2003. Multiclass spectral clustering. In: Proc. Internat. Conf. on Computer Vision, pp. 313–319.

Zhang, Tian, Ramakrishnan, Raghu, Livny, Miron. 1996. BIRCH: An efficient data clustering method for very large databases. In: Proc.1996 ACM SIGMOD Internat. Conf. on Management of data, vol. 25, pp. 103–114.