Data Clustering: A User's Dilemma*

Anil K. Jain and Martin H.C. Law

Department of Computer Science and Engineering, Michigan State University,
East Lansing, MI 48823, USA
{jain, lawhiu}@cse.msu.edu

Abstract. Cluster analysis deals with the automatic discovery of the grouping of a set of patterns. Despite more than 40 years of research, there are still many challenges in data clustering from both theoretical and practical viewpoints. In this paper, we describe several recent advances in data clustering: clustering ensemble, feature selection, and clustering with constraints.

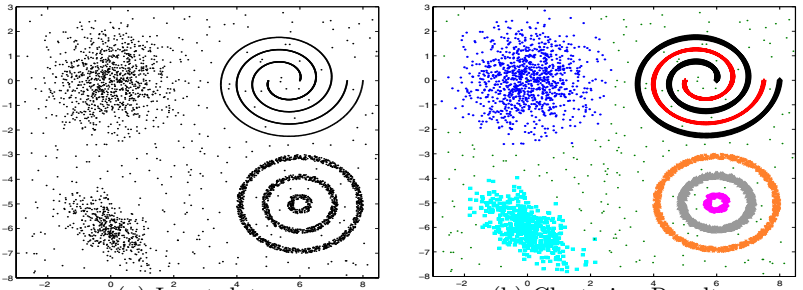
1 Introduction

The goal of data clustering [1], also known as cluster analysis, is to discover the "natural" grouping(s) of a set of patterns, points, or objects. Cluster analysis is prevalent in any discipline that involves analysis or processing of multivariate data. Image segmentation, an important problem in computer vision, is often formulated as a clustering problem [2,3]. Clustering has also been used to discover subclasses in a supervised setting to reduce intra-class variability. Different writing styles were automatically discovered by clustering in [4] to facilitate online handwriting recognition. Contours of MR brain images are clustered into different subclasses in [5]. Documents can be clustered to generate topical hierarchies for information access [6] or retrieval. The study of genome data [7] in biology often involves clustering – either on the subjects, the genes, or both.

Many clustering algorithms have been proposed in different application scenarios (see [8] for a survey). Important partitional clustering algorithms in the pattern recognition community include the k-means algorithm [9], the EM algorithm [10], different types of linkage methods (see [11]), the mean-shift algorithm [12], algorithms that minimize some graph-cut criteria (such as [13]), path-based clustering [14] and different flavors of spectral clustering [3,15]. However, a universal clustering algorithm remains an elusive goal. The fundamental reason for this is the intrinsic ambiguity of the notion of natural grouping. Another difficulty is the diversity of clusters: clusters can be of different shapes, different densities, different sizes, and are often overlapping (Figure 1). The problem is even more challenging when the data is high-dimensional, because the presence of irrelevant features can obscure the cluster structure. Above all, even for data without any cluster structure, most clustering algorithms still generate spurious clusters (see [16,17] for a discussion)! All these issues make clustering a dilemma [18] for the user.

* Research supported by the U.S. ONR grant no. N000140410183.

S.K. Pal et al. (Eds.): PReMI 2005, LNCS 3776, pp. 1–10, 2005.



(a) Input data (b) Clustering Results

Fig. 1. Diversity of clusters. The clusters in this data set, though easily identified by a human, are difficult to be detected automatically. The clusters are of different shapes, sizes and densities. Background noise makes the clustering task even more difficult.

In the rest of this paper, we shall describe some recent advances in clustering that alleviate these limitations for partitional clustering. In Section 2, we examine how different data partitions in a clustering ensemble can be combined to discover clusters with high diversity. Feature selection can be performed when high dimensional data is to be clustered (Section 3). The arbitrariness of clustering can be reduced by introducing side-information – notably constraints on the cluster labels (Section 4). Finally, we conclude in Section 5.

2 Clustering Ensemble

In supervised learning, it is often beneficial to combine the outputs of multiple classifiers in order to improve the classification accuracy. The goal of *clustering ensemble* is similar: we seek a combination of multiple partitions that provides improved overall clustering of the data set. Clustering ensembles are beneficial in several aspects. It can lead to better average performance across different domains and data sets. Combining multiple partitions can lead to a solution unattainable by any single clustering algorithm. We can also perform parallel clustering of the data and combine the results subsequently, thereby improving scalability. Solutions from multiple distributed sources of data or attributes (features) can also be integrated.

We shall examine several issues related to clustering ensemble in this section. Consider a set of n objects $\mathcal{X} = \{x_1, \ldots, x_n\}$. The clustering ensemble consists of N different partitions of the data set $\mathcal{X}$. The k-th partition is represented by the function π_k, where $\pi_k(x_i)$ denotes the cluster label of the object x_i in the k-th partition, which consists of m_k clusters.

2.1 Diversity

How can we generate each of the partitions in the ensemble? While the optimal strategy is probably dependent on the problem domain and the goal for

performing clustering, there are also some general procedures. One can apply different algorithms (like k-means, complete-link, spectral clustering) to create different partitions of the data. Some clustering algorithms like k-means require initialization of parameters. Different initializations can lead to different clustering results. The parameters of a clustering algorithm, such as the number of clusters, can be altered to create different data partitions. Different "versions" of the data can also be used as the input to the clustering algorithm, leading to different partitions. For example, the data set can be perturbed by re-sampling with replacement or without replacement. Different feature subsets or projection to different subspaces can also be used.

Note that these strategies can be combined. For example, the k-means algorithm with different number of clusters and different initializations is used to create the clustering ensemble in [19].

2.2 Consensus Function

A consensus function is a mapping from the set of N partitions $\{\pi_1, \ldots, \pi_N\}$ in the clustering ensemble to a consensus partition π^*. Different consensus functions have been proposed in the literature (Table 1). In this paper, we shall present two examples of consensus functions: consensus by co-association matrix [19], and consensus by quadratic mutual information [20].

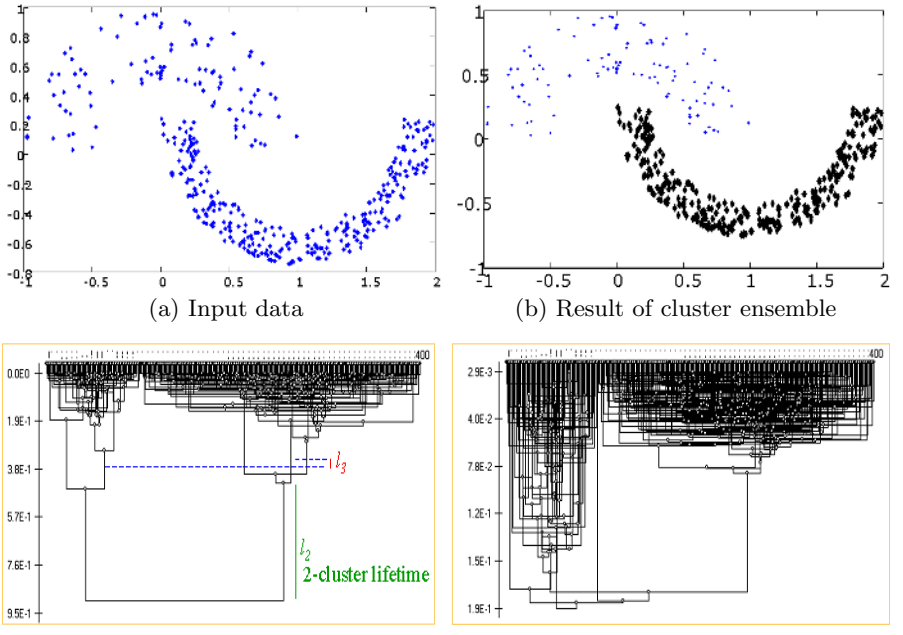
In consensus by co-association, we compute the $n \times n$ co-association matrix $\mathbf{C}$, whose (i, j)-th entry is given by

$$c_{ij} = \frac{1}{N} \sum_{k=1}^{N} I\big(\pi_k(x_i) = \pi_k(x_j)\big); \quad i, j = 1, \ldots, n. \tag{1}$$

Here, $I(.)$ is the indicator function, which is 1 if the argument is true and 0 otherwise. Intuitively, c_{ij} measures how many times x_i and x_j are put in the same cluster in the clustering ensemble. The co-association matrix can be viewed as a new similarity matrix, which is superior than the original distance matrix based on Euclidean distance. The consensus partition is found by clustering with $\mathbf{C}$

Table 1. Different consensus functions in the literature

Method	Key ideas
Co-association matrix [19]	Similarity between patterns is estimated by co-association; single-link with max life-time is used for finding the consensus partition
Mutual Information [20]	Maximize the quadratic mutual information between the individual partitions and the consensus partition
Hyper-graph methods [21]	Clusters in different partitions are represented by hyper-edges; consensus partition is found by a k-way min-cut of the hyper-graph
Finite mixture model [20]	Maximum likelihood solution to latent class analysis problem in the space of cluster labels via EM
Re-labeling and voting [22]	Assuming the label correspondence problem is solved, a voting procedure is used to combine the partitions



(a) Input data (b) Result of cluster ensemble

(c) Dendrogram based on co association matrix (d) Dendrogram based on Euclidean distances computed in (a)

Fig. 2. Example of consensus partition by co-association matrix in [19]. The number of clusters (2 in this case) is determined by the lifetime criteria. Note that the co-association values lead to a much more clear-cut dendrogram than Euclidean distances.

as the similarity matrix. The single-link algorithm is used, because the "chaining" behavior in $\mathbf{C}$ is often desirable. The number of clusters is determined by maximizing the *lifetime*. The lifetime of a partition with K clusters is defined as the range of threshold values on the dendrogram that leads to the identification of the partition. The longer the lifetime, the more stable the partition upon data perturbation. A simple illustration of this consensus function can be seen in Figure 2. Additional experimental results can be found in [19].

One drawback of the co-association consensus function is its $O(n^2)$ memory requirement. A consensus function that uses only $O(n)$ memory was proposed in [20]. It is based on the notion of *median partition*, which maximizes the average utility of the partitions in the ensemble with respect to the consensus partition. Formally, we seek π^* that maximizes $J(\pi^*) = \sum_{k=1}^{N} U(\pi^*, \pi_k)/N$, where $U(\pi^*, \pi_k)$ denotes a utility function. The utility function based on mutual information is advocated in [20]. Define a random variable X_t that corresponds to the cluster labels of the objects in π_t. The joint probability $p(X_t = i, X_s = j)$ is proportional to the number of objects that are in the i-th cluster for the partition π_t and in the j-th cluster for π_s. The similarity/utility between π_t and π_s can be quantified by Shannon mutual information $I(X_t, X_s)$: it takes the largest value when π_t and π_s are identical, and the smallest value when π_t is

"independent" of π_s. However, maximizing $I(X_t, X_s)$ is difficult. Instead, the qualitatively similar generalized mutual information is used:

$$I^{\alpha}(X_t, X_s) = H^{\alpha}(X_t) - H^{\alpha}(X_t|X_s)$$
$$H^{\alpha}(X_t) = (2^{1-\alpha} - 1)^{-1} \left(\sum_{i=1}^{m_t} P(X_t = i)^{\alpha} - 1 \right) \quad \alpha > 0, \alpha \neq 1 \tag{2}$$

Note that $H^{\alpha}(X_t)$ is the Renyi entropy with order α. When $\alpha = 2$, the above quantity is known as the "quadratic mutual information", and it is proportional to the category utility function proposed in [23]. Because of this equivalence, the consensus partition that maximizes the sum of the quadratic mutual information can be found by the k-means algorithm in a new feature space. Specifically, let $\mathbf{y}_{ik}$ be a vector of length m_k, where the j-th component of $\mathbf{y}_{ik}$ is 1 if $\pi_k(x_i) = j$, and 0 otherwise. A new feature vector for the object x_i is obtained by concatenating $\mathbf{y}_{ik}$ for different k. The new pattern matrix is then standardized to have zero column and row means. Running k-means on this new pattern matrix gives us the consensus partition that maximizes the quadratic mutual information. Note that the number of clusters in the consensus partition π^* is assumed to be known for this consensus function. Empirical study in [20] showed that this consensus function outperformed other consensus functions over a variety of data sets.

2.3 Strength of Components

Combining multiple partitions can lead to an improved partition. Is the improvement possible only when the individual partitions are "good" and close to the target partition? Another way to phrase this question is: will the clustering ensemble still work if the clustering algorithms used to generate the partitions in the ensemble are "weak", meaning that they give only a slightly better than random partition? In [20], two different types of "weak" algorithms were considered. The first is to project the data randomly on a 1D space and run k-means on the resulting 1D data set. The second is to split the data in the space of given features into two clusters by a random hyperplane. Note that computing a partition by these weak algorithms is very fast, and this allows one to create a large clustering ensemble. It turns out that, despite the poor quality of each of the partitions in the ensemble, the consensus partition is meaningful and can even outperform the results of combining strong partitions [20]. A similar idea of combining the clustering results by projecting to random subspaces was explored in [24].

2.4 Convergence

Is the success of clustering ensemble purely empirical? In [25], a theoretical analysis of the utility of clustering ensemble was performed. The analysis is based on the premise that each partition in the ensemble is a "corrupted" version of the true, but unknown, partition $\tilde{\pi}$. Two different partition generation models were considered. In both cases, the consensus partition π^* is more likely to be

equal to $\tilde{\pi}$ than individual partitions in the ensemble, and π^* converges to $\tilde{\pi}$ as the number of partitions in the ensemble increases.

In the first model, a noise process $F(.)$ first corrupts the labels of $\tilde{\pi}$ in an identical and independent manner, followed by a process $T(.)$ that permutes the labels, in order to generate π_k. The consensus function is plurality voting [22], where the Hungarian algorithm is first applied to inverse $T(.)$. The label of each object in the consensus partition π^* is found by voting on the cluster labels in the ensemble. It can be shown that, as N goes to infinity, the probability that π^* is the same as $\tilde{\pi}$ goes to one, despite the fact that (i) plurality voting is used instead of majority voting, and (ii) the Hungarian algorithm may undo the effect of $T(.)$ erroneously.

In the second model, a distance function $d(\pi_t, \pi_s)$ on two partitions in the space of possible partitions $\mathbb{P}$ is assumed. Note that $d(\pi_t, \pi_s)$ needs not satisfy the triangle inequality. Some example distance functions for two partitions can be found in [26]. The observed partition π_k is a corrupted version of $\tilde{\pi}$, which is generated according to a probability distribution $p(\pi_k|\tilde{\pi})$. Under some mild assumptions on $p(\pi_k|\tilde{\pi})$, the consensus by median partition, $\pi^* = \arg\min_\pi \sum_{k=1}^N d(\pi, \pi_k)$, converges to $\tilde{\pi}$ at an exponential rate when N goes to infinity.

3 Feature Selection in Clustering

Clustering, similar to supervised classification and regression, can be improved by using a good subset of the available features. However, the issue of feature selection in unsupervised learning is rarely addressed. This is in sharp contrast with supervised learning, where many feature selection techniques have been proposed (see [27,28,29] and references therein). One important reason is that it is not at all clear how to assess the relevance of a subset of features without resorting to class labels. The problem is made even more challenging when the number of clusters is unknown, since the optimal number of clusters and the optimal feature subset are inter-related. Recently, several feature selection/weighting algorithms [30,31,32,33] for clustering have been proposed. We shall describe the algorithm in [33], which estimates both the importance of different features and the number of clusters automatically.

Consider model-based clustering using a Gaussian mixture. The algorithm in [33] begins by assuming that the features are conditionally independent given the (hidden) cluster label. This assumption is made regularly for models involving high-dimensional data, such as naive Bayes classifiers and the emission densities of continuous hidden Markov models. Formally, the density of the data $\mathbf{y}$ is assumed to take the form $p(\mathbf{y}) = \sum_{j=1}^k \alpha_j \prod_{l=1}^d p(y_l|\theta_{jl})$, where α_j is the weight of the j-th component/cluster and θ_{jl} is the parameter of the j-th component for the l-th feature. The j-th cluster is modeled by the distributions $p(y_l|\theta_{jl})$ with different l, and they are typically assumed to be Gaussians.

The l-th feature is irrelevant if its distribution is independent of the class labels. In other words, it follows a common density $q(y_l|\lambda_l)$ which is parameterized

by λ_l, and is the same irrespective of the value of j. The form of $q(y_l|\lambda_l)$ reflects our prior knowledge about the distribution of the non-salient features, and it can be assumed to be a Gaussian. Denote the saliency of the l-th feature by ρ_l. The higher the saliency, the more important a feature is. With the introduction of ρ_l and $q(y_l|\lambda_l)$, the density of $\mathbf{y}$ can be written as

$$p(\mathbf{y}) = \sum_{j=1}^{k} \alpha_j \prod_{l=1}^{d} \big(\rho_l p(y_l|\theta_{jl}) + (1-\rho_l) q(y_l|\lambda_l)\big), \tag{3}$$

where there are k clusters for the d-dimensional vector $\mathbf{y}$. The saliencies of different features, together with the cluster parameters, can be estimated by maximizing the log-likelihood. Because of the need to estimate k automatically, the minimum message length (MML) criterion [34] is adopted. The optimal parameters $\{\alpha_j, \rho_l, \theta_{jl}, \lambda_l\}$ can be found by an EM algorithm [33].

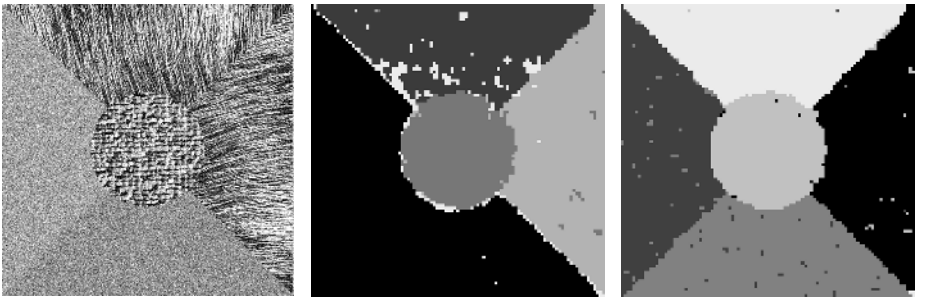
One key property of that EM algorithm is its pruning behavior, which can force some of the α_j to go to zero and some of the ρ_l to go to zero or one during parameter estimation. Therefore, the algorithm is initialized to have a large number (greater than the true number) of clusters. The redundant clusters will be pruned by the algorithm automatically. This initialization strategy (first proposed in [35]) can also alleviate the problem of poor local minimum. Experimental results and further details on the algorithm can be found in [33].

4 Clustering with Constraints

In many applications of clustering, there is a preference for certain clustering solutions. This preference or extrinsic information is often referred to as *side-information*. Examples include alternative metrics between objects, orthogonality to a known partition, additional labels or attributes, relevance of different features and ranks of the objects. The most natural type of side-information in clustering is a set of *constraints*, which specifies the relationship between cluster labels of different objects. A pairwise must-link (must-not-link) constraint corresponds to the requirement that two objects should be placed in the same (different) cluster. Constraints are naturally available in many clustering applications. For instance, in image segmentation one can have partial grouping cues for some regions of the image to assist in the overall clustering [36]. Clustering of customers in market-basket database can have multiple records pertaining to the same person. In video retrieval tasks, different users may provide alternative annotations of images in small subsets of a large database; such groupings may be used for semi-supervised clustering of the entire database.

There is a growing literature on clustering with constraints (see [37] and the references therein). We shall describe the algorithm in [37], which tackles the problem of model-based clustering with constraints. Let $\mathcal{Y}^c$ and $\mathcal{Y}^u$ be the set of data points with and without constraints, respectively. Clustering with constraints is performed by seeking the cluster parameter Θ that explains both the constrained and unconstrained data well. This is done by maximizing

$$J(\Theta) = (1-\gamma)\mathcal{L}(\mathcal{Y}^c; \Theta) + \gamma \mathcal{L}(\mathcal{Y}^u; \Theta), \tag{4}$$



(a) Input Image (b) 1% and 5% sites in constraints (c) 10% sites in constraints

Fig. 3. Results of image segmentation. (a): input image. (b): segmentation result with 1% and 5% of sites in constraints. (c): segmentation result with 10% sites in constraints.

where $\mathcal{L}(\mathcal{Y}^c;\Theta)$ and $\mathcal{L}(\mathcal{Y}^u;\Theta)$ are the log-likelihood of the data points with and without constraints, respectively, and γ is the tradeoff parameter. The term $\mathcal{L}(\mathcal{Y}^u;\Theta)$ is defined as in the standard mixture distribution. To define $\mathcal{L}(\mathcal{Y}^c;\Theta)$, a more elaborate distribution that considers the constraints is needed.

Let z_i be the cluster label of the data $\mathbf{y}_i$. Let $\mathcal{S}$ and $\mathcal{D}$ be the set of must-link and must-not-link constraints, respectively. The number of violations of must-link and must-not-link constraints can be written as $\sum_{(i,j)\in\mathcal{S}} I(z_i \neq z_j)$ and $\sum_{(i,j)\in\mathcal{D}} I(z_i = z_j)$, respectively. By using the maximum entropy principle with λ^+ and λ^- as the Lagrange parameters, one can arrive at a prior distribution for the cluster labels $z_1,\ldots,z_m$ that participate in the constraints:

$$p(z_1,\ldots,z_m) \propto \exp\Big(-\lambda^+ \sum_{(i,j)\in\mathcal{S}} I(z_i \neq z_j) - \lambda^- \sum_{(i,j)\in\mathcal{D}} I(z_i = z_j)\Big). \qquad (5)$$

This prior distribution on z_i, together with the component density $p(\mathbf{y}_i|z_i)$, yields the log-likelihood $\mathcal{L}(\mathcal{Y}^c;\Theta)$. A mean-field approximation is applied to the posterior distribution of the cluster labels to keep the computation tractable. An EM algorithm can be derived to find the parameter Θ that maximizes $J(\Theta)$. The algorithm is fairly efficient, and is of similar computational complexity to the standard EM algorithm for estimating a mixture distribution. Figure 3 shows the results of applying this algorithm to an image segmentation task.

5 Conclusion

Data clustering is an important unsupervised learning problem with applications in different domains. In this paper, we have reviewed some of the recent advances in cluster analysis. Combination of data partitions in a clustering ensemble can produce a superior partition when compared with any of the individual partitions. Clustering of high-dimensional data sets can be improved by estimating

the saliencies of different features. Side-information, in particular constraints on cluster labels, can lead to a more desirable data partition. The research topics presented here give a glimpse of the state-of-the-art in cluster analysis, and we hope that this will stimulate the readers to investigate other problems in the area of data clustering.

References

1. Jain, A., Dubes, R.: Algorithms for Clustering Data. Prentice Hall (1988)
2. Jain, A., Flynn, P.: Image segmentation using clustering. In: Advances in Image Understanding, IEEE Computer Society Press (1996) 65–83
3. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence **22** (2000) 888–905
4. Connell, S., Jain, A.: Writer adaptation for online handwriting recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence **24** (2002) 329–346
5. Duta, N., Jain, A., Dubuisson-Jolly, M.P.: Automatic construction of 2D shape models. IEEE Transactions on Pattern Analysis and Machine Intelligence **23** (2001) 433–446
6. Sahami, M.: Using Machine Learning to Improve Information Access. PhD thesis, Computer Science Department, Stanford University (1998)
7. Baldi, P., Hatfield, G.: DNA Microarrays and Gene Expression. Cambridge University Press (2002)
8. Jain, A., Murty, M., Flynn, P.: Data clustering: A review. ACM Computing Surveys **31** (1999) 264–323
9. MacQueen, J.: Some methods for classification and analysis of multivariate observations. In: Proc. Fifth Berkeley Symposium on Math. Stat. and Prob., University of California Press (1967) 281–297
10. McLachlan, G., Peel, D.: Finite Mixture Models. John Wiley & Sons, New York (2000)
11. Duda, R., Hart, P., Stork, D.: Pattern Classification. 2nd edn. John Wiley & Sons, New York (2001)
12. Comaniciu, D., Meer, P.: Mean shift: A robust approach toward feature space analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence **24** (2002) 603–619
13. Wu, Z., Leahy, R.: An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence **15** (1993) 1101–1113
14. Fischer, B., Buhmann, J.: Path-based clustering for grouping smooth curves and texture segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence **25** (2003) 513–518
15. Verma, D., Meila, M.: A comparison of spectral clustering algorithms. Technical Report 03-05-01, CSE Department, University of Washington (2003)
16. Smith, S., Jain, A.: Testing for uniformity in multidimensional data. IEEE Transactions on Pattern Analysis and Machine Intelligence **6** (1984) 73–81
17. Jain, A., Xu, X., Ho, T., Xiao, F.: Uniformity testing using minimum spanning tree. In: Proc. the 16th International Conference on Pattern Recognition. (2002) IV:281–284
18. Dubes, R., Jain, A.: Clustering techniques: The user's dilemma. Pattern Recognition **8** (1976) 247–260

19. Fred, A., Jain, A.: Evidence accumulation clustering. To appear in IEEE Transactions on Pattern Analysis and Machine Intelligence (2005)
20. Topchy, A., Jain, A., Punch, W.: Clustering ensembles: Models of consensus and weak partitions. To appear in IEEE Transactions on Pattern Analysis and Machine Intelligence (2005)
21. Strehl, A., Ghosh, J.: Cluster ensembles – a knowledge reuse framework for combining multiple partitions. Journal of Machine Learning Research **3** (2002) 583–617
22. Fridlyand, J., Dudoit, S.: Applications of resampling methods to estimate the number of clusters and to improve the accuracy of clustering method. Technical report, University of California, Berkeley (2001)
23. Mirkin, B.: Reinterpreting the category utility function. Machine Learning **45** (2001) 219–228
24. Fern, X., Brodley, C.: Random projection for high dimensional data clustering: A cluster ensemble approach. In: Proc. the 20th International Conference on Machine Learning, AAAI press (2003) 186–193
25. Topchy, A., Law, M.H., Jain, A.K., Fred, A.: Analysis of consensus partition in cluster ensemble. In: Proc. the 5th IEEE International Conference on Data Mining. (2004) 225–232
26. Meila, M.: Comparing clusterings by the variation of information. In: Proc. The 16th Annual Conference on Learning Theory, Springer (2003) 173–187
27. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. Journal of Machine Learning Research **3** (2003) 1157–1182
28. Blum, A., Langley, P.: Selection of relevant features and examples in machine learning. Artificial Intelligence **97** (1997) 245–271
29. Jain, A., Zongker, D.: Feature selection: Evaluation, application, and small sample performance. IEEE Transactions on Pattern Analysis and Machine Intelligence **19** (1997) 153–157
30. Dy, J., Brodley, C.: Feature selection for unsupervised learning. Journal of Machine Learning Research **5** (2004) 845–889
31. Roth, V., Lange, T.: Feature selection in clustering problems. In: Advances in Neural Information Processing Systems 16. MIT Press, Cambridge, MA (2004)
32. Modha, D., Scott-Spangler, W.: Feature weighting in k-means clustering. Machine Learning **52** (2003) 217–237
33. Law, M., Figueiredo, M., Jain, A.: Simultaneous feature selection and clustering using mixture models. IEEE Transactions on Pattern Analysis and Machine Intelligence **26** (2004) 1154–1166
34. Wallace, C., Freeman, P.: Estimation and inference via compact coding. Journal of the Royal Statistical Society. Series B (Methodological) **49** (1987) 241–252
35. Figueiredo, M., Jain, A.: Unsupervised learning of finite mixture models. IEEE Transactions on Pattern Analysis and Machine Intelligence **24** (2002) 381–396
36. Yu, S., Shi, J.: Segmentation given partial grouping constraints. IEEE Transactions on Pattern Analysis and Machine Intelligence **26** (2004) 173–183
37. Lange, T., Law, M., Jain, A., Buhmann, J.: Learning with constrained and unlabelled data. In: Proc. the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. (2005)