Consensus Clusterings

Nam Nguyen, Rich Caruana
Department of Computer Science, Cornell University
Ithaca, New York 14853
{nhnguyen,caruana}@cs.cornell.edu

Abstract

In this paper we address the problem of combining multiple clusterings without access to the underlying features of the data. This process is known in the literature as clustering ensembles, clustering aggregation, or consensus clustering. Consensus clustering yields a stable and robust final clustering that is in agreement with multiple clusterings. We find that an iterative EM-like method is remarkably effective for this problem. We present an iterative algorithm and its variations for finding clustering consensus. An extensive empirical study compares our proposed algorithms with eleven other consensus clustering methods on four data sets using three different clustering performance metrics. The experimental results show that the new ensemble clustering methods produce clusterings that are as good as, and often better than, these other methods.

1. Introduction

Clustering often is a first step in data analysis. Many different clustering methods have been developed [9, 19] such as hierarchical agglomerative clustering, mixture densities, graph partitioning, and spectral clustering. Most clustering methods focus on finding a single optimal or near-optimal clustering according to some specific clustering criterion. Consensus clustering can provide benefits beyond what a single clustering algorithm can achieve. Consensus clustering algorithms often: generate better clusterings; find a combined clustering unattainable by any single clustering algorithm; are less sensitive to noise, outliers or sample variations; and are able to integrate solutions from multiple distributed sources of data or attributes.

In addition to the benefits outlined above, consensus clustering can be useful in a variety of domains. For example, clustering categorical data (where it is more difficult to define useful distance metrics) can be considered as a consensus clustering problem where each discrete feature is viewed as a simple clustering of the data. Then the consensus clustering algorithm can be applied to the ensemble of all clusterings produced by discrete features of the data set. An example of categorical data is a *Movie* database where some of discrete attributes are *Director, Actor, Actress, Genre, Year,* etc. As another example, consensus clustering can be employed in "privacy-preserving" scenarios where it is not possible to centrally collect all of the underlying features for all data points, but only how the data points are grouped together. Such a situation might arise when different companies or governmental agencies have information about individuals that they can not share, but still need to find high-quality clusters of individuals. For such cases, the consensus clustering algorithm offers a natural model for clustering the data maintained in separate sites in a privacy-preserving manner, that is, without the need for different sites to fully-reveal their data to each other, and without the need for relying on a trusted authority.

In this paper, we propose an EM-like consensus clustering algorithm and its variations which utilize a feature map constructed from the set of base level clusterings. Our experiments show that the proposed algorithms produce results as good as or better than other existing consensus clustering algorithms. The paper is structured as follows: in section 2 we introduce the consensus clustering framework; in section 3 we give a brief description of related works in this area; in section 4 we discuss drawbacks of current consensus clustering algorithms and advantages of EM-like approach; in section 5 we describe in detail our proposed algorithms; in section 6 and 7 we present the evaluation criteria and data sets; and the experimental results, analysis and conclusion are given in section 8, 9 and 10.

2. Consensus Clustering Framework

We are given a set of N data points $\mathbf{X} = \{x_1, x_2, ..., x_N\}$ and a set of C clusterings $\mathbf{\Pi} = \{\pi_1, \pi_2, ..., \pi_C\}$ of the data points in $\mathbf{X}$. Each clustering π_i is a mapping from $\mathbf{X}$ to $\{1, ..., n_{\pi_i}\}$ where n_{π_i} is the number of clusters in π_i. The problem of clustering consensus is to find a new clustering π^* of the data $\mathbf{X}$ that best summarizes the clustering ensemble $\mathbf{\Pi}$.

Similar to the approach of [18], our proposed algorithms utilize the space of constructed features induced by the ensemble of clusterings $\mathbf{\Pi}$. For each data point x, we construct

a corresponding C-dimensional feature vector y, where the i^{th} feature is simply the cluster label from the clustering π_i. The set of vectors $\mathbf{Y} = \{y_1, y_2, ..., y_N\}$ where $y_i = \langle \pi_1(x_i), \pi_2(x_i), ..., \pi_C(x_i) \rangle$, is exploited in our proposed algorithms to find the consensus clustering.

3. Related Work

Consensus clustering has recently attracted the interest of a number of researchers in the machine learning community. This section provides a brief summary of the work in this area. (This is by no means a complete survey of consensus clusterings.)

3.1. Pairwise Similarity Approach

In this approach, a measure of similarity between a pair of data points can be estimated as the ratio of the number of clusterings in which the two data points are in the same clusters to the total number of clusterings in the ensemble. More precisely, the similarity between two data points x_i and x_j is defined as,

$$S_{ij} = S(x_i, x_j) = \frac{1}{C} \sum_{c=1}^{C} \mathcal{I}(\pi_c(x_i) = \pi_c(x_j)), \quad (1)$$

where $\mathcal{I}$ is the indicator function. Thus, any similarity-based clustering algorithm can be applied to the similarity matrix S to find a consensus clustering of the ensemble.

We experiment with two similarity-based clustering algorithms: Furthest Consensus (FC) [7] and Hierarchical Agglomerative Clustering Consensus (HAC) [5, 6, 12]. In both of these algorithms, the matrix S is used as the similarity measure.

Furthest Consensus (FC): The goal of the algorithm is to find K cluster centers that are furthest from each other. The algorithm starts with finding a pair of data points that are furthest apart, and assigning them as cluster centers. Then it repeatedly finds a next cluster center that is furthest apart from previous found centers. At the end, all data points are assigned to its closest center.

Hierarchical Agglomerative Clustering Consensus (HAC): HAC algorithm is a standard bottom-up algorithm for the correlation clustering problem. It starts by placing all data points into singleton clusters. Then it repeatedly merges two clusters that have the largest averaged similarity measure which is given in the similarity matrix S. The algorithm stops when there are K remaining clusters.

3.2. Graph-based Approach

In [16] the authors propose three consensus clustering algorithms: Cluster-based Similarity Partitioning Algorithm (CSPA), HyperGraph Partitioning Algorithm (HGPA), and Meta-Clustering Algorithm (MCLA). All algorithms first transform the ensemble of clusterings into a graph representation.

CSPA: First, the similarity matrix is computed as in eq. 1. Then, a induced similarity graph, where vertices correspond to data points and edges' weights to similarity measures, is partitioned into K clusters using METIS [11].

HGPA: The hypergraph is construct using the ensemble of C clusterings, $\mathbf{\Pi} = \{\pi_1, \pi_2, ..., \pi_C\}$, where vertices correspond to data points and each hyperedge is represented by a cluster from one of the clusterings in the ensemble. All hyperedges have the same weight. This algorithm looks for a hyperedge separator that partitions the hypergraph into K unconnected components of approximately the same size. The hypergraph partitioning package HMETIS [10] is used to perform the partitioning.

MCLA: This algorithm is based on clustering clusters where each cluster is also represented as a hyperedge. The algorithm groups and collapses related hyperedges into K clusters; and then assigns each data point to the collapsed hyperedge in which it participates most strongly.

3.3. Mutual Information Approach

In [17], an objective function is formulated as the mutual information between the target consensus clustering and the clustering ensemble. Then the quadratic mutual information is maximized via an EM algorithm in the space of constructed features as define in Section 2 to find the target consensus function. This algorithm usually requires multiple restarts in order to avoid convergence to low quality local minima.

3.4. Mixture Model Approach

In [18], the authors propose a probabilistic model of consensus using a finite mixture of multinomial distributions in a space of constructed features as defined in Section 2. A combined clustering is found as a solution to the corresponding maximum likelihood problem using the EM algorithm.

3.5. Cluster Correspondence Approaches

In [2], the authors first solve the problem of finding the correspondence between clusters of different clusterings in the ensemble either by constrained and unconstrained search via optimizing a linear programming formulation (CCC and CCU), or with Singular Value Decomposition (CCSVD). Then a simple voting procedure is applied to assign data points into clusters.

3.6. Objective Dependence Approach

The representative example of this approach is the **BEST** algorithm [7]. Given an objective function, BEST chooses among all clusterings in the ensemble the one that maximizes the objective function.

In our experiments, we compare the proposed new algorithms with these eleven algorithms: Furthest Consensus (FC), Hierarchical Agglomerative Clustering Consensus (HAC), Cluster-based Similarity Partitioning Algorithm (CSPA), HyperGraph Partitioning Algorithm (HGPA), Meta-Clustering Algorithm (MCLA), Quadratic Mutual Information Algorithm (QMI), EM Mixture Model Algorithm(EM), BEST, Cluster Correspondence Constrained Search (CCC), Cluster Correspondence Unconstrained Search (CCU), and Cluster Correspondence SVD (CCSVD).

4. Motivations

In this section, we point out some drawbacks of consensus clusterings algorithms and advantages of EM-like approach to consensus clustering problem. In the consensus clustering setting, pairwise similarity often does not reflect a good measure of similarity between data points, especially when the number of base-clusterings are limited. Consider a 5-items data set, $\{x_i\}_{i=1}^{5}$, where the ground-truth partition of the data is $\{(x_1, x_2, x_3), (x_4, x_5, x_6)\}$. The four clusterings of the data set and the similarity matrix are shown in Figure 1. We notice that similarity-based clustering algorithms will not be able recover the ground-truth partition of the data set since the similarities of the pairs $\{(x_1, x_6), (x_3, x_6)\}$ have the highest values. However, an EM-like approach (described in the next section) can recover the ground-truth partition since it is a local minimum in the EM process.

Clustering	x_1	x_2	x_3	x_4	x_5	x_6
I	1	1	1	2	2	1
II	2	1	1	2	2	2
III	2	1	2	1	2	2
IV	1	1	2	1	2	2

	x_1	x_2	x_3	x_4	x_5	x_6
x_1		1/2	1/2	1/2	1/2	3/4
x_2			1/2	1/2	0	1/4
x_3				0	1/2	3/4
x_4					1/2	1/4
x_5						3/4
x_6						

Figure 1. Example of Consensus Clustering

Most of the current consensus clustering algorithms return a single consensus clustering as the final result. EM-like approaches have an advantage over these because they can generate multiple consensus clusterings with different restarts, and the best consensus clustering with respect the evaluation criteria can be selected.

5. Consensus Clustering Algorithms

In this section we present in detail our EM-like consensus clustering algorithms. Our proposed algorithms can be applied to any group of clusterings, including clusterings in which each individual clustering may have different numbers of clusters. In addition, our new algorithms conform to two constraints which most other consensus clustering algorithms follow. The first constraint is that the number of clusters K in the target clustering π^* is given. We are not trying to solve the difficult problem of determining the correct number of clusters. The second constraint is to only use the information provided by the ensemble of clusterings, i.e. $\mathbf{\Pi} = \{\pi_1, \pi_2, ..., \pi_C\}$, without any access to the underlying features of data points in $\mathbf{X}$.

In the following sections, we describe in details the iterative consensus clustering algorithm and its variations.

5.1. Iterative Voting Consensus (IVC)

The algorithm is an iterative process which utilizes the constructed feature-vectors $\mathbf{Y} = \{y_1, y_2, ..., y_N\}$ where $y_i = \langle \pi_1(x_i), \pi_2(x_i), ..., \pi_C(x_i) \rangle$ induced by the ensemble of clusterings $\mathbf{\Pi}$. Each cluster in the target consensus clustering has a cluster center which is also a C-dimensional vector.

Each iteration of the algorithm involves two steps: the first step computes the cluster center of each cluster in the target consensus clustering, and the second step reassigns each data point to its closest cluster center. In the first step, the i^{th} feature of the cluster center vector is just the majority values of all i^{th} features of data points belonging to the considered cluster. This is possible since all i^{th} features of data points are coming from the same clustering π_i in the clustering ensemble. In the second step, we assign each data point to its closest center by finding the center of the minimum distance to the considered data point. The distance of a point to a cluster center is just the Hamming distance between the two vector representations. The pseudo-code of the IVC algorithm is described in Algorithm 1.

5.2. Variations of IVC

We have experimented with two variations of IVC algorithm, Iterative Probabilistic Voting Consensus (IPVC) and Iterative Pairwise Consensus (IPC). The main difference between IVC and its variations is how to compute the distance between a point and a cluster. Due to lack of space, the detailed description of IPVC and IPC is omitted.[1]

6. Evaluation Criteria

Evaluating the quality of a clustering is a nontrivial and ill-posed task [15]. In supervised learning, model performance is assessed by comparing model predictions to targets. In clustering we do not have targets and usually do not

[1] See the complete version at http: \\www.cs.cornell.edu\ ∼nhnguyen \consensus_longversion.pdf

Algorithm 1 Iterative Voting Consensus

Input: a set of N data points $\mathbf{X} = \{x_1, x_2, ..., x_N\}$
a set of C clusterings $\mathbf{\Pi} = \{\pi_1, \pi_2, ..., \pi_C\}$
K is a desired number of clusters

Output: a consensus clustering π^* with K clusters

Initialize π^*
repeat
Let $P_i = \{y \mid \pi^*(y) = i\}$ be the i^{th} cluster

Compute the representation of each cluster: $y_{P_i} = \langle \textit{majority}\{(P_i)_1\}, ..., \textit{majority}\{(P_i)_C\}\rangle$, where $(P_i)_j$ is the set of the j^{th} features of all data points in P_i

for y **in** $\mathbf{Y}$ **do**
Re-assign $\pi^*(y) \leftarrow \textit{argmin}_i D(y, y_{P_i})$, where $D(y, y_{P_i}) = \sum_{j=1}^{C} \mathcal{I}((y)_j \neq (y_{P_i})_j)$
end for
until π^* does not change

know *a priori* what groupings of the data are best. This hinders discerning when one clustering is better than another, or when one clustering algorithm outperforms another. In general, there are two main approaches to evaluate consensus clusterings: consensus criteria measure how the target consensus clustering is in agreement with all clusterings in the ensemble, and clustering criteria measure how well the target consensus clustering performs on the underlying features of the data points $\mathbf{X}$, possibly with respect to hidden true labels on these points.

6.1. Consensus Criteria

Given an ensemble of clusterings $\mathbf{\Pi} = \{\pi_1, \pi_2, ..., \pi_C\}$ and a target consensus clustering π^*, the consensus performance is computed as

$$Perf(\pi^*, \mathbf{\Pi}) = \frac{1}{C}\sum_{i=1}^{C} \textit{Rand}(\pi^*, \pi_i), \tag{2}$$

where $\textit{Rand}(\pi^*, \pi_i)$ is a measure of how dissimilar the two clusterings are.

The Rand distance is based on counting pairs of points on which two clusterings agree or disagree. The (adjusted) Rand distance was introduced by [8] of Rand's [14] criterion,

$$f(\pi, \pi') = \frac{N_{10} + N_{01}}{N_{11} + N_{10} + N_{01} + N_{00}}, \tag{3}$$

where N_{11} and N_{00} are the number of point pairs that are in the same cluster, in the different clusters respectively under both π and π'; N_{10} and N_{01} are the number of point pairs that are in the same cluster under π but not under π', and the same cluster under π' but not under π respectively.

6.2. Clustering Criteria

In addition to evaluate a clustering based on how well it combines multiple clusterings in the ensemble, there are also two additional clustering criteria to measure how well a clustering partitions the data into its natural groupings. In our experiment, we examine two clustering criteria: compactness and accuracy. Compactness measures the average pairwise distances between points in the same cluster:

$$Compactness(\pi) = \frac{1}{N}\sum_{k=1}^{K} n_k\left(\frac{\sum_{x_i, x_j \in C_k} d(x_i, x_j)}{n_k(n_k - 1)/2}\right), \tag{4}$$

where $d(x_i, x_j)$ is the distance between x_i and x_j.

All of our test data sets have external true labels which are not used by the clustering process. The second clustering criterion is accuracy, which measures how well the target clustering perform in comparison to the external true labels of the data points:

$$Accuracy(\pi) = \frac{\sum_{k=1}^{K} majority(C_k|L_k)}{N}, \tag{5}$$

where $majority(C_k|L_k)$ is the number of points with the plurality label in the C_k cluster (if label l appeared in cluster k more often than any other label, then $majority(C_k|L_k)$ is the number of points in C_k with label l).

7. Data Sets

Data Set	features	cases	classes	clusters
Australia	17	245	10	10
Bergmark	254	1000	25	25
Covertype	49	1000	7	15
Letters	617	514	7	10

Table 1. Description of Data Sets

We evaluate our consensus clustering algorithms on four data sets: Australia, Bergmark, Covertype, and Letter. The Australia Coastal data is a subset of the data available from the Biogeoinformatics of Hexacorals environmental database [1]. The Bergmark data was collected using 25 focused web crawls, each with a different keyword. The variables are counts in a bags-of-words model describing the web pages. The Covertype data is from the UCI Machine Learning Repository [13]. It contains cartographic variables sampled at 30×30 meter grid cells in four wilderness areas in Roosevelt National Forest in northern Colorado. The letters data is a subset of the isolet spoken letter data set from the UCI Machine Learning Repository [13].

In our experiments, different clusterings of each data set have the same number of clusters, and the same number of clusters is chosen for the target consensus clusterings as well. The clusterings that will be combined are generated for the

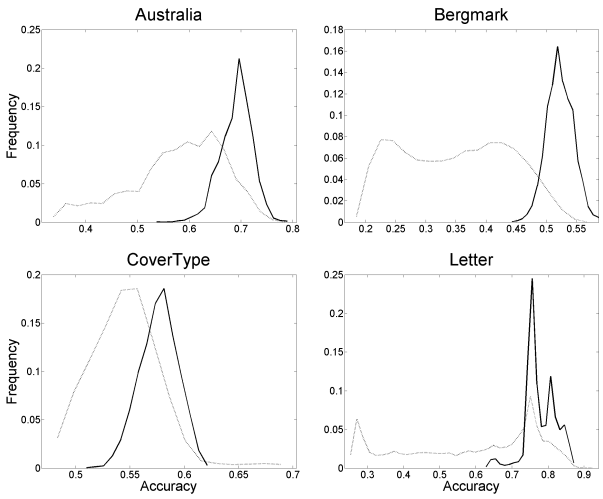

Figure 2. Two different accuracy distributions (dotted line: Feature Weighting K-means, solid line: K-means) of four data sets.

data sets using two of the methods described in [3]: feature weighting k-means and k-means with different random restarts. The accuracy distributions of these two methods are quite different as shown in Figure 2. The clusterings of the feature weighting k-means category are generated by applying a Zipf feature weighting method and tend to spread out from low accuracy to high accuracy. The clusterings of the k-means category are generated by applying the k-means algorithm with different random initializations and usually concentrate on the high range of accuracy.

8. Experiments

Similar to k-means our proposed algorithms are sensitive to the initial partition of the data. Hence, for each ensemble of clusterings our algorithms are repeated with 100 different initializations. The initial partition is randomly selected as one of the clusterings in the ensemble. (We tried initializing the new algorithms both with random partitions of the data as well as with clusterings from the base level clusterings. We observe that better initializations usually lead to better results, so the random partition initialization does not work as well as initializing with a base-level clustering.)

In the experiment, the number of clusterings in the ensemble is 200. All consensus clustering algorithms are evaluated with respect to three different performance criteria as described in Section 6. For the Rand distance and compactness, lower values indicate better performance. For accuracy, higher values indicate better performance. Since the results under the two ensemble distributions (i.e. k-means and feature weighting k-means) are very similar, we simplify the presentation of results by showing the averaged scores.

Figure 3 shows the performance of 14 consensus clustering methods on the three metrics averaged across the four problems and two ensemble distributions. The three new algorithms perform comparably to each other. Across the three different performance criteria, the new algorithms produce results that usually are equal to, and often better than, other algorithms. Of the existing methods, the graph-based approach tends to have the worst performance compared to others, especially the HyperGraph Partitioning Algorithm (HGPA). In addition, we notice that the HAC and CCSVD algorithms sometimes are competitive in comparison to the new algorithms on some performance metrics with some data sets. However, none of the existing algorithms consistently produce results as good as the new methods across the different performance metrics and different data sets.

9. Complexity Analysis

Similar to k-means, the space complexity of IVC and IPVC is $O(NC)$ where N is the number of data points and C is the number of base-level clusterings. The space complexity of IPC is $O(N^2)$ since it needs to keep track of all pair-wise distances.

Since our proposed algorithms are EM algorithms, the number of iterations that the algorithms update the target clustering before getting to a local minima is varied and depends on the data sets. In our experiments, we observe that the algorithms always terminate with less than 40 iterations. There are many speed-up techniques (i.e. using inequality triangle to accelerate k-means [4]) developed for k-means which may also be applied to our algorithms to improve running time.

10. Conclusion

We present three EM-like algorithms for the consensus clustering problem. An extensive empirical study shows that the proposed algorithms yield results as good as, and usually better than, eleven other clustering consensus methods. Our proposed algorithms are in essence variations of k-means algorithm using different distance measures applied to the vector of base-level clusterings. The results indicate that this approach works well for consensus clustering. It is interesting that this simple, and somewhat *obvious*, approach performs better than other more complicated methods in the literature.

In future work, we intend to enhance the new algorithms by relaxing the two constraints of the new algorithms mentioned in Section 5. First, we want to use consensus to determine the best number of clusters in the consensus clustering. Second, we will develop methods to leverage the underlying features of the original data.

Acknowledgments

This work was supported by NSF CAREER Grant # 0347318.

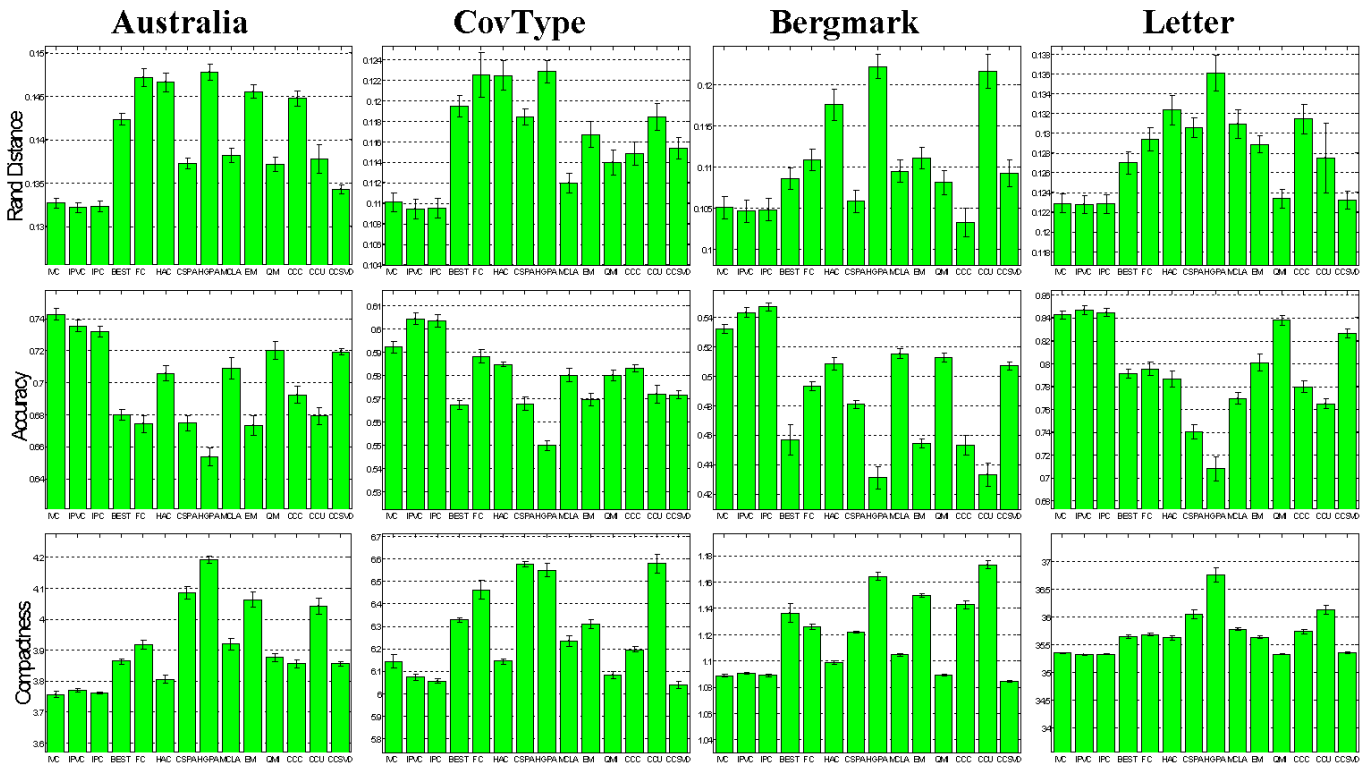

Figure 3. Consensus clustering results for different evaluation criteria.

References

[1] Biogeoinformatics of Hexacorals. Environmental database, http://www.kgs.ku.edu/hexacoral/, 1998.

[2] C. Boulis and M. Ostendorf. Combining multiple clustering systems. In *The 8th European conference on Principles and Practice of Knowledge Discovery in Databases(PKDD), LNAI 3202*, pages 63–74, 2004.

[3] R. Caruana, M. Elhawary, N. Nguyen, and C. Smith. Meta clustering. In *Proceedings IEEE International Conference on Data Mining*, 2006.

[4] C. Elkan. Using the triangle inequality to accelerate k-means. In *Proceedings of the 20th International Conference on Machine Learning*, 2003.

[5] X. Z. Fern and C. E. Brodley. Random projection for high dimensional data clustering: A cluster ensemble approach. In *Proceedings of the 20th ICML*, 2003.

[6] A. L. Fred and A. K. Jain. Data clustering using evidence accumulation. In *Proceedings of the 16th International Conference on Pattern Recognition*, 2002.

[7] A. Gionis, H. Mannila, and P. Tsaparas. Clustering aggregation. In *Proceedings of the 21st ICDM*, 2005.

[8] L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2:193–218, 1985.

[9] A. K. Jain, M. N. Murty, and F. P. J. Data clustering: a review. *ACM Computing Surveys*, 31:264–323, 1999.

[10] G. Kharypis, R. Aggarwal, V. Kumar, and S. Shekhar. Multilevel hypergraph partitioning: Applications in vlsi domain. In *Proceedings of the Design and Automation Conference*, 1997.

[11] G. Kharypis and V. Kumar. Multiple k-way partitioning scheme for irregular graphs. *SIAM Journal on Scientific Computing*, 20(1):359–392, 1998.

[12] G. N. Lance and W. T. Williams. A general theory of classificatory sorting strategies. i. hierarchical systems. *Computer Journal*, 9:373–380, 1967.

[13] D. J. Newman, S. Hettich, C. L. Blake, and C. J. Merz. UCI repository of machine learning databases, http://www.ics.uci.edu /∼mlearn/mlrepository.html, 1998.

[14] W. M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66:846–850, 1971.

[15] N. Slonim and N. Tishby. Agglomerative information bottleneck. In *Proceedings of Neural Information Processing Systems*, 1999.

[16] A. Strehl and J. Ghosh. Cluster ensembles - a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 3:583–617, 2002.

[17] A. Topchy, A. K. Jain, and W. Punch. Combining multiple weak clusterings. In *Proceedings IEEE International Conference on Data Mining*, 2003.

[18] A. Topchy, A. K. Jain, and W. Punch. A mixture model for clustering ensembles. In *Proceedings SIAM International Conference on Data Mining*, 2004.

[19] D. Verma and M. Meila. A comparison of spectral clustering algorithms. Technical Report submitted, Department of Computer Science, University of Washington, 2001.