Combining Multiple Weak Clusterings

Alexander Topchy, Anil K. Jain, and William Punch
Computer Science Department, Michigan State University,
East Lansing, MI, 48824, USA
{topchyal, jain, punch}@cse.msu.edu

Abstract

A data set can be clustered in many ways depending on the clustering algorithm employed, parameter settings used and other factors. Can multiple clusterings be combined so that the final partitioning of data provides better clustering? The answer depends on the quality of clusterings to be combined as well as the properties of the fusion method. First, we introduce a unified representation for multiple clusterings and formulate the corresponding categorical clustering problem. As a result, we show that the consensus function is related to the classical intra-class variance criterion using the generalized mutual information definition. Second, we show the efficacy of combining partitions generated by weak clustering algorithms that use data projections and random data splits. A simple explanatory model is offered for the behavior of combinations of such weak clustering components. We analyze the combination accuracy as a function of parameters controlling the power and resolution of component partitions as well as the learning dynamics vs. the number of clusterings involved. Finally, some empirical studies compare the effectiveness of several consensus functions.

1. Introduction

In contrast to supervised classification, clustering is an inherently ill-posed problem, whose solution violates at least one of the common assumptions about scale-invariance, richness, and cluster consistency [1]. Exploratory nature of the problem forces us to seek generic and robust clustering algorithms when explicit model-based approaches prove to be ineffective.

One of the methods used to increase the robustness of the clustering solution is to combine outputs of several clustering algorithms. Combination of clusterings using multiple sources of data or features is important in distributed data mining. Several recent studies on clustering combination [2,3,4] have pioneered a new area in the conventional taxonomy of clustering algorithms [5,6]. The problem of clustering fusion can be defined generally as follows: given multiple clusterings of the data set, find a combined clustering with better quality. We offer a representation of multiple clusterings as a set of new attributes characterizing the data items. Such a view directly leads to a formulation of the combination problem as a categorical clustering problem in the space of these attributes, or, in other terms, a median partition problem. We show how median partition is related to the classical intra-class variance criterion when generalized mutual information is used as the evaluation function.

While the problem of clustering combination bears some traits of a classical clustering, it also has three major issues which are specific to combination design:

1. Consensus function: How to combine different clusterings? How to resolve the label correspondence problem? How to ensure symmetrical and unbiased consensus with respect to all the component partitions?

2. Diversity of clustering: How to generate different partitions? What is the source of diversity in the components?

3. Strength of constituents/components: How "weak" could each input partition be? What is the minimal complexity of component clusterings to ensure a successful combination?

Similar questions have already been addressed in the framework of multiple classifier systems [7]. However, it is not possible to mechanically apply the combination algorithms from classification (supervised) to clustering (unsupervised). Indeed, no labeled training data is available in clustering; therefore the ground truth feedback necessary for boosting the overall accuracy cannot be used. In addition, different clusterings may produce incompatible data labelings, resulting in intractable correspondence problems, especially when the numbers of clusters are different.

From the supervised case we also learn that the proper combination of weak classifiers [8,9] may achieve arbitrarily low error rates on training data, as well as reduce the predictive error. One of the goals of our work is to adopt weak clustering algorithms and combine their outputs. Vaguely defined, a weak clustering algorithm produces a partition, which is only slightly better than a random one. We propose two different weak clustering algorithms as the components of the combination:

1. Clustering of random 1-dimensional projections of multidimensional data. This can be generalized to clustering in any random subspace of the original data space.

2. Clustering by splitting the data using a number of random hyperplanes. For example, if only one hyperplane is used then data is split into two groups.

One can expect that using many simple, but computationally inexpensive components will be preferred to combining clusterings obtained by sophisticated, but computationally involved algorithms.

The second goal of this paper is to compare the performance of different consensus functions. A consensus function maps multiple clusterings to a final partitioning of the data. We study a family of consensus functions based on categorical clustering including the co-association based hierarchical methods [4], hypergraph algorithms [3] and a new simple centroid-based heuristic consensus function. Combination accuracy is analyzed as a function of the number and the resolution of the clustering components.

Previous research on clustering ensembles has addressed both how the component clusterings are obtained as well as method by which they are combined. Consensus functions using co-association values were explored in [2,4] with multiple *k*-means partitions. Hypergraph algorithms for consensus were analyzed in [3]. Other related work can be found in [10,11,12,13]

2. Problem of consensus clustering

Let X be a set of N data points (objects) in d-dimensional space. No assumptions are needed at the moment about the data input: it could be represented in a non-metric space or as an $N \times N$ dissimilarity matrix. Suppose we are given a set of H partitions $\Pi = \{\pi_1,\ldots, \pi_H\}$ of objects in X. Each component partition in Π is a set of disjoint, exhaustive and nonempty clusters $\pi_i = \{L^i_1, L^i_2, \ldots, L^i_{K(i)}\}$, $X = L^i_1 \cup \ldots \cup L^i_{K(i)}$, $\forall \pi_i$, and $K(i)$ is the number of clusters in the i-th partition. The problem of consensus clustering is to find a new partition $\sigma = \{C_1,\ldots,C_K\}$ of data X given the partitions in Π, such that the objects in a cluster of σ are more similar to each other than to objects in different clusters of σ. This statement of the problem is virtually the same as for a conventional clustering except that it uses information contained in already existing partitions $\{\pi_1,\ldots, \pi_H\}$. Other variants of this definition could be obtained by putting some extra requirements on the target partition σ, such as fixing the number of clusters in σ, or allowing fuzzy membership values for data points. In general, one could use information from two available sources: the partitions in Π and/or the original attributes of objects in X.

It is convenient to characterize consensus clustering as clustering in a space of new features induced by the set Π. Indeed, each component partition π_i represents a feature with categorical values. The values assumed by the i-th new feature are simply the cluster labels from partition π_i. Therefore, membership of each object in different partitions is treated as a new feature vector, an H-tuple, given H different partitions in Π. Combined clustering becomes equivalent to a problem of clustering of H-tuples if we ignore the original d attributes.

3. Consensus functions

A consensus function maps a given set of partitions $\Pi = \{\pi_1,\ldots,\pi_H\}$ to a target partition σ. A family of hierarchical clustering consensus functions immediately follows from the similarity between two objects x and y:

$$s(x, y) = \sum_{i=1}^{H} \delta(\pi_i(x), \pi_i(y)), \quad \delta(a,b) = \begin{cases} 1, & \text{if } a = b \\ 0, & \text{if } a \neq b \end{cases} \qquad (1)$$

Similarity between a pair of objects simply counts the number of clusters shared by these objects in the partitions $\{\pi_1,\ldots, \pi_H\}$. This is the same as the co-association value introduced in [2]. One can use a co-association matrix for subsequent clustering by single-link algorithm [4] or any other type of agglomerative procedure to obtain a target consensus clustering σ.

Another candidate consensus function is based on the notion of median partition. A median partition σ is the best summary of existing partitions in Π. In contrast to the co-association approach, median partition is derived from estimates of similarities between attributes (i.e., partitions in Π), rather then from similarities between objects. A well-known example of this approach is implemented in the COBWEB algorithm in the context of conceptual clustering [14]. The category utility function $U(\sigma, \pi_i)$ evaluates the quality of a candidate median partition $\sigma = \{C_1,\ldots,C_K\}$ against some other partition $\pi_i = \{L^i_1,\ldots, L^i_{K(i)}\}$, with labels L^i_j for j-th cluster [15]:

$$U(\sigma, \pi_i) = \sum_{r=1}^{K} p(C_r) \sum_{j=1}^{K(i)} p(L^i_j \mid C_r)^2 - \sum_{j=1}^{K(i)} p(L^i_j)^2 , \qquad (2)$$

with the following notations: $p(C_r) = |C_r|/N$, $p(L^i_j) = |L^i_j|/N$, and $p(L^i_j \mid C_r) = |L^i_j \cap C_r| / |C_r|$.

The function $U(\sigma, \pi_i)$ assesses the agreement between two partitions as the difference between the expected number of labels of partition π_i that can be correctly predicted both with the knowledge of clustering σ and without it. The overall utility of the partition σ with respect to all the partitions in Π can be measured as the sum of pair-wise agreements:

$$U(\sigma, \Pi) = \sum_{i=1}^{H} U(\sigma, \pi_i) \qquad (3)$$

Therefore, the best median partition should maximize the value of overall utility:

$$\sigma_{best} = \arg\max_{\sigma} U(\sigma, \Pi) \qquad (4)$$

Mirkin [16] has proved that maximization of partition utility (3) is equivalent to minimization of the square-error clustering criterion if the number of clusters K in target partition σ is fixed. This is somewhat suprising in that the partition utility function (4) uses only the between-attribute similarity measure (2), while the square-error criterion makes use of distances between objects and prototypes. Simple standardization of categorical labels in $\{\pi_1,\ldots,\pi_H\}$ effectively transforms them to quantitative features [16]. This transformation replaces the i-th partition π_i assuming $K(i)$ values by $K(i)$ binary features, and standardizes each binary feature to a zero mean. In other words, for each object x we can compute the values of the new features $y_{ij}(x)$, for j=1…$K(i)$, i=1…H, as following:

$$y_{ij}(x) = \delta(L_j^i, \pi_i(x)) - p(L_j^i), \tag{5}$$

Hence, the solution of median partition problem (4) can be approached by the k-means clustering algorithm operating in the space of features y_{ij} if the number of target clusters is predetermined. We use this heuristic as a part of empirical study of consensus functions.

In information-theoretic framework, the quality of the consensus partition σ is determined by the amount of information, $I(\sigma,\Pi)$, it shares with the given partitions in Π. Strehl and Ghosh [3] suggest an objective function that is based on the classical mutual information:

$$\sigma_{best} = \arg\max_{\sigma} I(\sigma,\Pi), \tag{6}$$

$$\text{where } I(\sigma,\Pi) = \sum_{i=1}^{H} I(\sigma,\pi_i),$$

$$I(\sigma,\pi_i) = \sum_{r=1}^{K}\sum_{j=1}^{K(i)} p(C_r, L_j^i)\log\left(\frac{p(C_r, L_j^i)}{p(C_r)\,p(L_j^i)}\right) \tag{7}$$

Again, an optimal median partition can be found by solving this optimization problem. However, it is not clear how to use these equations to search for consensus.

We show that another information-theoretic definition of entropy will reduce the mutual information criterion to the category utility function discussed before. We proceed from the generalized entropy of degree s for a discrete probability distribution P=(p_1,…,p_n) :

$$H^s(P) = (2^{1-s}-1)^{-1}\left(\sum_{i=1}^{n} p_i^s - 1\right), \quad s>0, \; s\neq 1 \tag{8}$$

Shannon's entropy is the limit form of (8) when s→1. Generalized mutual information between σ and π can be defined as:

$$I^s(\sigma,\pi) = H^s(\pi) - H^s(\pi \mid \sigma) \tag{9}$$

Quadratic entropy (s=2) is of particular interest, since it is known to be closely related to classification error. When s=2, generalized mutual information $I^s(\sigma, \pi_i)$ becomes:

$$\begin{aligned} I^2(\sigma,\pi_i) &= -2\left(\sum_{j=1}^{K(i)} p(L_j^i)^2 - 1\right) + \\ &+ 2\sum_{r=1}^{K} p(C_r)\left(\sum_{j=1}^{K(i)} p(L_j^i \mid C_r)^2 - 1\right) = 2U(\sigma,\pi_i) \end{aligned} \tag{10}$$

Therefore, generalized mutual information gives the same consensus clustering criterion as category utility function (3). The gini-index measure for attribute selection used by Breiman et al. [17] also follows from (2) and (10). In light of Mirkin's result, all these criteria are equivalent to within-cluster variance minimization, after simple label transformation.

Other interesting consensus functions can be obtained as solutions to the hypergraph minimum cut problem. The details can be found in [3], where three different hypergraph algorithms were investigated. These algorithms are also used in our comparative empirical study.

4. Combination of weak clusterings

We now turn to the issue of generating different clusterings for the combination. Do we use the partitions produced by numerous existing clustering algorithms? We argue that it is possible to generate the partitions using weak, but less expensive, clustering algorithms and still achieve comparable or better performance. Certainly, the key motivation is that the synergy of many such components will compensate for their weaknesses. We consider two simple clustering algorithms:

1. Clustering of the data projected to a random subspace of lower dimension. In the simplest case, the data is projected on 1-dimensional subspace, a random line. The k-means algorithm clusters the projected data and gives a partition for the combination.
2. Random splitting of data by hyperplanes. For example, a single random hyperplane would create rather trivial clustering of d-dimensional data by cutting the hypervolume into two regions.

4.1. Splitting by random hyperplanes

Direct clustering by use of a random hyperplane illustrates how a reliable consensus emerges from low-informative components. The random splits approach pushes the notion of weak clustering almost to an extreme. The data set is cut by random hyperplanes dissecting the original volume of d-dimensional space containing the points. Points separated by the hyperplanes are declared to be in different clusters. In this situation, a co-association consensus function is appropriate since the only information needed is whether the patterns are in the same cluster or not. Thus the contribution of a hyperplane

partition to the co-association value for any pair of objects can be either 0 or 1. Finer resolutions of distance are possible by counting the number of hyperplanes separating the objects, but for simplicity we do not use it here. Consider a random line dissecting the classic 2-spiral data shown in Fig. 1(a). While any single partition does little to reveal the true underlying clusters, analysis of the hyperplane generating mechanism shows how multiple partitions can discover the true clusters.

Consider first the case of one-dimensional data. Splitting objects in 1-dimensional space is done by a random threshold in R^1. In general, if r points are randomly selected, then (r+1) clusters are formed. It is easy to derive that in 1-dimensional space the probability of separating two points whose inter-point distance is x is exactly:

$$P(\text{split}) = 1-(1-x/L)^r, \qquad (11)$$

where L is the length of the interval containing the objects, and r points are drawn at random from uniform distribution on this interval. Fig. 1(b) illustrates the dependence for L=1 and r =1,2,3,4. If a co-association matrix is used to combine H different partitions, then the expected value of co-association between two objects is $H(1-P(\text{split}))$, that follows from the binomial distribution of the number of splits in H attempts. Therefore, the co-association values found after combining many random split partitions are generally expected to be a non-linear and a monotonic function of respective distances. The situation is similar for multidimensional data, however, the generation of random hyperplanes is a bit more complex. To generate a random hyperplane in d dimensions, we should first draw a random point in the multidimensional region that will serve as a point of origin. Then we randomly choose a unit normal vector $\boldsymbol{u}$ that defines the hyperplane. The two objects characterized by vectors $\boldsymbol{p}$ and $\boldsymbol{q}$ will be in the same cluster if ($\boldsymbol{up}$)($\boldsymbol{uq}$)>0 and will be separated otherwise (here $\boldsymbol{ab}$ denotes a scalar product). If r hyperplanes are generated, then the total probability that two objects remain in the same cluster is just the product of probabilities that each of hyperplanes does not split the objects. Thus we can expect that the law governing the co-association values is close to what is obtained in one-dimensional space in (11).

Let us compare the actual dependence of co-association values with the function in (11). Fig. 2 shows the results of experiments with 1000 different partitions by random splits of the Iris data set. The Iris data is 4-dimensional and contains 150 points. There are (150·149)/2 pair-wise distances between the data items. For all the possible pairs of points, each plot in fig. 2 shows the number of times a pair was split. The observed dependence of the inter-point "distances" derived from the co-association values vs. the true Euclidean distance, indeed, can be described by the function in (11).

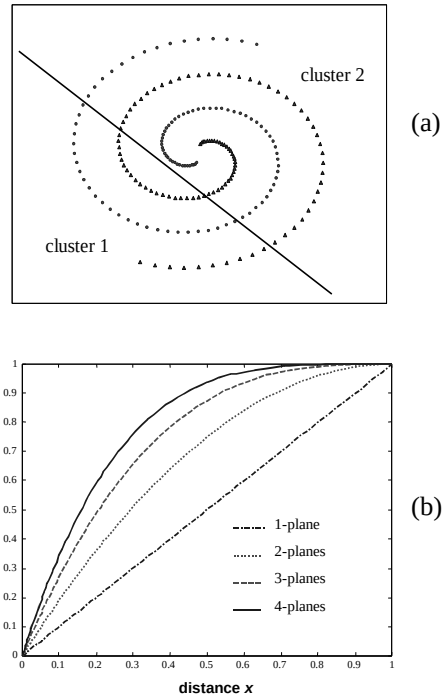

Figure 1. (a) An example of splitting 2-spiral data set by a random line. Points on the same side of the line are in the same cluster. (b) Probability of splitting two objects for different number of random thresholds as a function of distance between objects.

Clearly, the inter-point distances dictate the behavior of respective co-association values. The probability of a cut between any two given objects does not depend on the other objects in the data set. Therefore, we can conclude that any clustering algorithm that works well with the original inter-point distances is also expected to work well with co-association values obtained from a combination of multiple partitions by random splits. However, this result is more of theoretical value when true distances are available, since they can be used directly instead of co-association values. It illustrates the main idea of the approach, namely that the synergy of multiple weak clusterings can be very effective.

4.2. Combination in random subspaces

The weak projections approach combines multiple views of sample data. Each projection is much weaker, contains less information, then the data in the original space. However, combined partitions of projections become at least as powerful as clustering using the original data representation and may help to reveal data structure unattainable by any single clustering algorithm. Subspaces are not necessarily obtained by taking some of

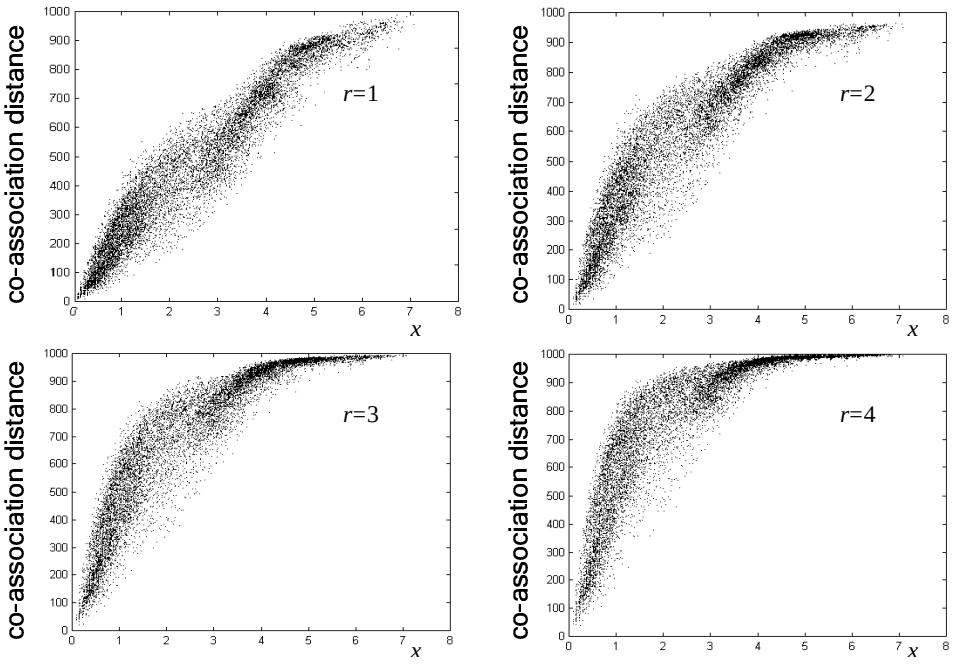

Figure 2. Dependence of distances derived from the co-association values vs. the actual Euclidean distance x for each possible pair of objects in Iris data. Co-association matrices were computed for several number of hyper planes r =1,2,3,4.

the original features (dimensions) as a whole, but could be created by projections, even random ones. Random subspaces are an excellent source of clustering diversity that provides different views of the data.

Each random subspace can be of very low dimension and it is by itself not very informative. On the other hand, clustering in 1-dimensional space is computationally cheap and can be effectively performed by *k*-means algorithm. The main subroutine of *k*-means algorithm – distance computation – becomes d times faster in 1-dimensional space. The cost of projection is linear with respect to the sample size and number of dimensions $O(Nd)$, and is less than the cost of one *k*-means iteration.

The main idea of our approach is to generate multiple partitions by projecting the data on a random line. A fast and simple algorithm such as *k*-means clusters the projected data, and the resulting partition becomes a component in the combination. Afterwards, a chosen consensus function is applied to the components. We discuss and compare several consensus functions in the experimental section.

It is instructive to consider a simple 2-dimensional data and one of its projections, as illustrated in Fig. 3(a). There are two natural clusters in the data. This data looks the same in any 1-dimensional projection, but the actual distribution of points is different in different clusters in the projected subspace. For example, Fig. 3(b) shows one possible histogram distribution of points in 1-dimensional projection of this data. There are three identifiable modes, each having a clear majority of points from one of the classes. One can expect that clustering by *k*-means algorithm will reliably separate at least a portion of the points from class 2. It is easy to imagine that projection of the data in Fig. 3(a) on another random line would result in different distribution of points and different label assignment, but for this particular data set it will always appear as a mixture of three bell-shaped components. Most probably, these modes will be identified as clusters by *k*-means algorithm. Thus each new 1-dimensional view correctly helps to group some data points. Accumulation of multiple views eventually should result in a correct combined clustering.

The important parameter is the number of clusters in the component partition π_i returned by *k*-means algorithm at each iteration, i.e. the value of k. If the value of k is too large then the partitions $\{\pi_i\}$ will overfit the data set which in turn may cause unreliable co-association values. Small number of clusters in $\{\pi_i\}$ may not be sufficient to capture the true structure of data set. In addition, if the number of clusterings in the combination is too small then the effective sample size for the estimates of distances from co-association values is also insufficient, resulting in a larger variance of the estimates. That is why the consensus functions based on the co-association values are more sensitive to the number of partitions in the combination (value of H) than consensus functions based on hypergraph algorithms.

5. Experimental Results and Discussion

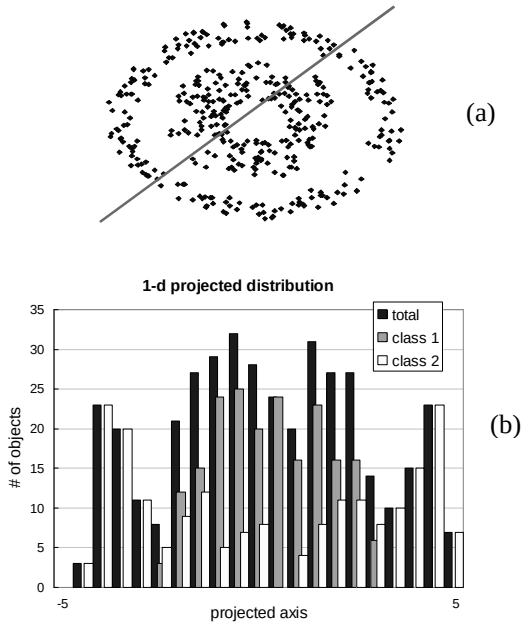

Figure 3. (a) A sample data with two identifiable natural clusters and a line randomly selected for projection. (b) Histogram of the distribution of points that resulted from data projection onto a random line.

The experiments were performed with several data sets, including three classic problems, the "2 spirals" and "half-rings", the Iris dataset from UCI benchmarks repository and large real-world dataset of galaxies and stars described in [18]. All consensus functions were provided with the true known number of clusters in the data. By providing this information, we can use the misassignment rate (error) of the consensus partition as a measure of performance of clustering combination. Thus the optimal solution of the correspondence problem between the labels of known and derived clusters is easily found using Hungarian method for minimal weight bipartite matching problem

We study the consensus accuracy as a function of the resolution of partitions (value of k) as well as its dependence on the number of components for seven consensus functions:

1. Consensus functions operated on the co-association matrix, but with three different hierarchical clustering algorithms for obtaining the final partition, namely single-linkage, average-linkage, and complete-linkage.
2. Consensus function based on k-means clustering in the space of standardized features defined in (5), that is equivalent to maximization of partition utility criterion in (3).
3. Three consensus functions based on hypergraph algorithms [3]. We used a set of programs 'ClusterEnsemble' implemented by Strehl and available at http://www.strehl.com/.

Three fundamental parameters affect the quality of the target consensus partition: H – the number of combined clusterings that directly influences the reliability and resolution of the co-association values; k – the number of clusters in the component clusterings $\{\pi_1,\ldots,\pi_H\}$ produced by k-means algorithm on one-dimensional projections; r – the number of hyperplanes used for obtaining clusterings $\{\pi_1,\ldots, \pi_H\}$ by random splitting algorithm. The value of k was varied in the interval [2,10], r in [2,5] and H in [5,1000]. Note that we report the average error rate for 20 independent runs. We omit the detailed tables due to space limitations and refer the readers to complete experimental reports at http://www.cse.msu.edu/prip/. Some characteristics of the datasets are:

	Number of features	Number of classes	Number of points/class	Total number of patterns
Iris	4	3	50-50-50	150
Galaxy	14	2	2082-2110	4192
2-spirals	2	2	100-100	200
Half-rings	2	2	100-300	400

Let us start by demonstrating how the combination of clusterings in projected 1-dimensional subspaces outperforms the combination of clusterings in the original multidimensional space. Fig. 4(a) shows the learning dynamics for Iris data and k=4, using average-link consensus function based on co-association values. Note that the number of clusters in each of the components $\{\pi_1,\ldots, \pi_H\}$ is set to k=4, and is different from the true number of clusters (=3). Clearly, each individual clustering in full multidimensional space is much stronger than any 1-dim partition, and therefore with only a small number of partitions (H<50) the combination of weaker partitions is no yet effective. However, for larger numbers of combined partitions (H>50), 1-dimensional projections taken together better reveal the true structure of the data. It is quite unexpected, since the k-means algorithm with k=3 makes, on average, 19 mistakes in original 4-dimensional space and 25 mistakes in 1-dimensional random subspace. Moreover, clustering in the projected subspace is d times faster than in multidimensional space. Although, the cost of computing a consensus partition σ is the same in both cases.

The results regarding the impact of value of k are reported in Fig. 4(b), which shows that there is a critical value of k for the Iris data set. This occurs when the average-linkage of co-association distances is used as a consensus function. In this case the value k=2 is not sufficient to separate the true clusters.

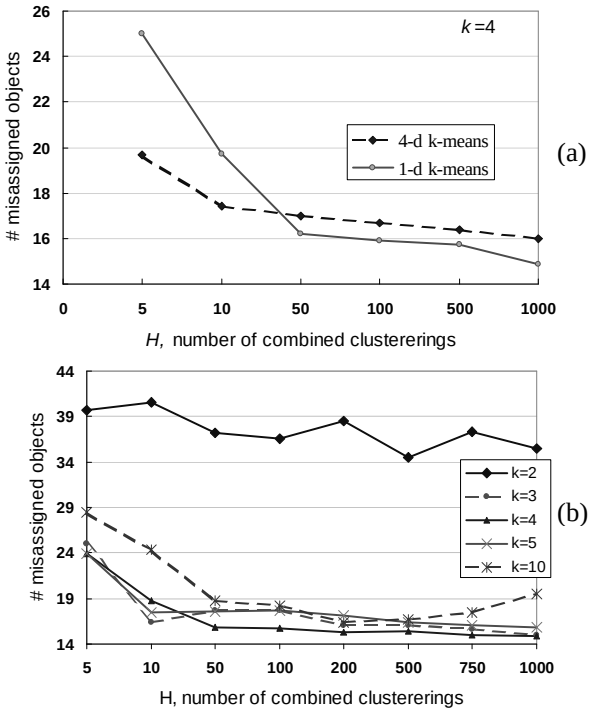

Figure 4. Performance of random subspaces algorithm on Iris data. (a) Number of errors by the combination of *k*-means partitions (*k*=4) in multidimensional space and projected 1-d subspaces. Average-link consensus function was used. (b) Error of projection algorithm as a function of the number of components and the number of clusters k in each component.

The role of the consensus function is illustrated in Fig.5. Three consensus functions are compared on the Iris data set. They all use similarities from the co-association matrix but cluster the objects using three different criterion functions, namely, single link, average link and complete link. It is clear that the combination using single-link performs significantly worse than the other two consensus functions. This is expected since the three classes in Iris data have hyperellipsoidal shape.

More results were obtained on two data sets, which are traditionally difficult for any partitional centroid-based algorithm: "half-rings" data set and "2 spirals" data in Fig. 1(a). The single-link consensus function performed the best and was able to identify both the 'half-rings' clusters as well as spirals. In contrast to the results for Iris data, average-link and complete-link consensus were not suitable for these data sets. In general, one can expect that average-link (single-link) consensus will be appropriate if standard average-link (single-link) agglomerative clustering works well for the data and vice versa. Moreover, none of the three hypergraph consensus functions could find a correct combined partition. This is somewhat surprising given that the hypergraph algorithms performed well on the Iris data. However, the Iris data is far less problematic because one of the clusters is linearly separable, and the other classes are well described as a

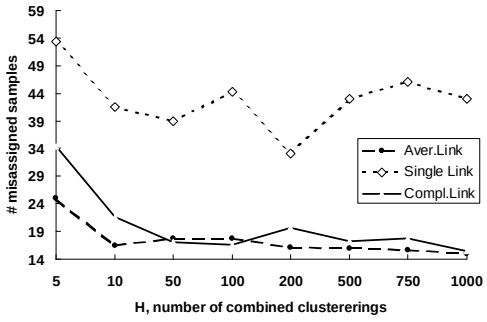

Figure 5. Dependence of performance of the projection algorithm on the type of consensus function for Iris data set. *k*=3.

mixture of two multivariate normal distributions. Perfect separation of natural clusters was achieved with a large number of partitions in clustering combination ($H > 200$) and for values of $k > 3$ for "half-rings" and "2 spirals". It indicates that for each problem there is a critical value of resolution of component partitions that guarantees good clustering combination. This further supports the work of Fred and Jain [4] who showed that a random number of clusters in each partition ensures a greater diversity of components. We see that the minimal required value of resolution for the Iris data is *k*=3, for "half-rings" it is *k*=2 and for "2 spirals" it is *k*=4. In general, the value of *k* should be larger than the true number of clusters.

The number of partitions affects the relative performance of the consensus functions. With large values of *H* (>100), co-association consensus becomes stronger, while with small values of *H* it is preferable to use hypergraph algorithms or *k*-means median partition algorithm.

It is interesting to compare the combined clustering accuracy with the accuracy of some of the classical clustering algorithms. For example, for Iris data the EM algorithm has the best average error rate of 6.17%. In our experiments, the best performers for Iris data were the hypergraph methods, with an error as low as 3%, with $H > 200$ and $k > 5$. For the "half-rings" data, the best standard result is 5.25% error by the average-link algorithm, while the combined clustering using the single-link co-association algorithm achieved a 0% error with $H > 200$. Also, for the "2 spirals" data the clustering combination achieves 0% error, the same as by regular single-link clustering. Hence, with an appropriate choice of consensus function, clustering combination outperforms many standard algorithms. However, the choice of good consensus function is similar to the problem of choice of proper conventional clustering algorithm. Perhaps the good alternative to guessing the right consensus function, is simply to run all the available consensus functions and then pick the final consensus partition according to the partition utility criteria in (4) or (6).

Table 1. Summary of the best results

Dataset	Best consensus function(s)	Lowest error obtained	Prefered values of parameters
Galaxy	Median partition k-means	< 13%	$H > 10, k > 3$
Iris	Hypergraph methods	< 3%	$H >100, k > 4$
2 spirals	Co-association SL	0%	$H >200, k > 3$
Half-rings	Co-association SL	0%	$H >200, k > 4$

Quadratic computational complexity effectively prohibits co-association based consensus functions from being used on large data sets, since $O(N^2)$ factor arises when co-association matrix is built for all the N objects. Even though computation of component partitions is d times faster due to projecting, the overall computational effort can be dominated by the complexity of computing the consensus partition. Therefore for large datasets it is problematic to use three hierarchical agglomerative methods as well as the CSPA hypergraph algorithm [3]. The k-means algorithm for median partition is most attractive in terms of speed with the complexity $O(kNH)$. For "galaxy" data we limited the number of components in combination to H=20 because of large data size. The results show that k-means algorithm for median partition has the best performance. On "galaxy" data HGPA did not work well due to its bias toward balanced cluster sizes, as it also happened in the case of the "half-rings" data set. Again the accuracy improved when the number of partitions and the number of clusters increases.

The same set of experiments was also performed with clustering combination via splits by random hyperplanes. The results in many details are close to what has been obtained by using random subspaces, with a slightly worse performance.

6. Conclusion

This study extended previous research on clustering ensembles in several respects. First, we have introduced a unified representation for multiple clusterings and formulated the corresponding categorical clustering problem. It is shown that the consensus function is related to classical intra-class variance criterion using the generalized mutual information definition. Second, we have considered combining weak clustering algorithms that use data projections and random data splits. A simple explanatory model is offered for the behavior of combination of such weak components. We have analyzed combination accuracy as a function of parameters, which control the power and resolution of component partitions. Empirical study compared effectiveness of several consensus functions.

Acknowledgements. This research was supported in part by ONR grant # N00014-01-1-0266 (A.K.J.) and by research award from Center for Biological Modeling at Michigan State University (A.T.)

7. References

[1] J. Kleinberg. "An Impossibility Theorem for Clustering", *Proc. of Adv. in Neural Information Processing Sys.* (NIPS 15), 2002.

[2] A.L.N. Fred, "Finding Consistent Clusters in Data Partitions". In *Proc. 3d Int. Workshop on Multiple Classifier Systems*. Eds. F. Roli, J. Kittler, LNCS 2364,2002, pp. 309-318.

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

[4] A.L.N. Fred and A.K. Jain, "Data Clustering Using Evidence Accumulation", In *Proc. of the 16th International Conference on Pattern Recognition*, ICPR 2002 ,Quebec City, 2002, pp. 276-280.

[5] A. Jain, M. N. Murty, and P. Flynn, "Data clustering: A review. *ACM Computing Surveys"*, 31(3), 1999, pp. 264–323.

[6] A.K. Jain and R.C. Dubes, *Algorithms for Clustering Data*. Prentice-Hall Inc., New Jersey, 1988.

[7] J. R. Quinlan, "Bagging, boosting, and C4.5", In *Proc. of the 13th AAAI Conference on Artificial Intelligence*, AAAI Press, Menlo Park, CA, 1996, pp. 725-30.

[8] E.M. Kleinberg, "Stochastic Discrimination", *Annals of Mathematics and Artificial Intelligence*, 1, 1990, pp. 207-239.

[9] Y. Freund, R.E. Schapire, "Experiments with a New Boosting Algorithm", in *Proc. of the Thirteenth International Conference on Machine Learning*, Morgan Kaufmann, 1996, pp. 148-156.

[10] P. Kellam, X. Liu, N.J. Martin, C. Orengo, S. Swift and A. Tucker, "Comparing, contrasting and combining clusters in viral gene expression data", Proc. of 6th Workshop on Intelligent Data Analysis in Medicine and Pharmocology, 2001, pp. 56-62.

[11] F. Leisch, "Bagged clustering", Working Papers SFB "Adaptive Information Systems and Modelling in Economics and Management Science", 51, Institut für Informationsverarbeitung, Abt. Produktionsmanagement, Wien, 1999.

[12] E. Dimitriadou, A. Weingessel and K. Hornik, "Voting-merging: An ensemble method for clustering", In *Proc. Int. Conf. on Artificial Neural Networks*, Vienna, 2001, pp. 217-224

[13] E. Johnson and H. Kargupta, "Collective, hierarchical clustering from distributed, heterogeneous data", In *Large-Scale Parallel KDD Systems*, Eds. Zaki M. and Ho C., LNCS 1759, Springer-Verlag, 1999, pp. 221–244.

[14] D. H. Fisher, "Knowledge acquisition via incremental conceptual clustering", *Machine Learning*, 2, 1987, pp. 139-172.

[15] M.A. Gluck and J.E. Corter, "Information, uncertainty, and the utility of categories", In *Proc. of the Seventh Annual Conference of the Cognitive Science Society*, Hillsdale, NJ: Lawrence Erlbaum, 1985, pp. 283-287.

[16] B. Mirkin, "Reinterpreting the Category Utility Function", *Machine Learning*, 45(2), 2001, pp. 219-228.

[17] L. Breiman, J. Friedman, R. Olshen, C. Stone, *Classification and Regression Trees*. Wadsworth, Monterrey, Ca, 1984.

[18] S.C. Odewahn, E.B. Stockwell, R.L. Pennington, R.M. Humphreys and W.A. Zumach, "Automated Star/Galaxy Discrimination with Neural Networks", *Astronomical Journal*, 103: 1992, pp. 308-331.