Weighted Clustering Ensembles

Muna Al-Razgan* Carlotta Domeniconi*

Abstract

Cluster ensembles offer a solution to challenges inherent to clustering arising from its ill-posed nature. Cluster ensembles can provide robust and stable solutions by leveraging the consensus across multiple clustering results, while averaging out emergent spurious structures that arise due to the various biases to which each participating algorithm is tuned. In this paper, we address the problem of combining multiple *weighted clusters* which belong to different subspaces of the input space. We leverage the diversity of the input clusterings in order to generate a consensus partition that is superior to the participating ones. Since we are dealing with weighted clusters, our consensus function makes use of the weight vectors associated with the clusters. The experimental results show that our ensemble technique is capable of producing a partition that is as good as or better than the best individual clustering.

1 Introduction

In an effort to achieve improved classifier accuracy, extensive research has been conducted in classifier ensembles. Very recently, cluster ensembles have emerged. It is well known that off-the-shelf clustering methods may discover very different structures in a given set of data. This is because each clustering algorithm has its own bias resulting from the optimization of different criteria. Furthermore, there is no ground truth against which the clustering result can be validated. Thus, no cross-validation technique can be carried out to tune input parameters involved in the clustering process. As a consequence, the user is not equipped with any guidelines for choosing the proper clustering method for a given dataset.

Cluster ensembles offer a solution to challenges inherent to clustering arising from its ill-posed nature. Cluster ensembles can provide more robust and stable solutions by leveraging the consensus across multiple clustering results, while averaging out emergent spurious structures that arise due to the various biases to which each participating algorithm is tuned.

*Department of Information and Software Engineering, George Mason University

In this paper, we introduce the problem of combining multiple *weighted clusters*, discovered by a locally adaptive algorithm [5] which detects clusters in different subspaces of the input space. We believe this paper is the first attempt to design a cluster ensemble for subspace clustering.

Recently, many different subspace clustering methods have been proposed [14]. They all attempt to dodge the curse of dimensionality which affects any algorithm in high dimensional spaces. In high dimensional spaces, it is highly likely that, for any given pair of points within the same cluster, there exist at least a few dimensions on which the points are far apart from each other. As a consequence, distance functions that equally use all input features may not be effective.

Furthermore, several clusters may exist in different subspaces comprised of different combinations of features. In many real-world problems, some points are correlated with respect to a given set of dimensions, while others are correlated with respect to different dimensions. Each dimension could be relevant to at least one of the clusters.

Global dimensionality reduction techniques are unable to capture local correlations of data. Thus, a proper feature selection procedure should operate locally in input space. Local feature selection allows one to embed different distance measures in different regions of the input space; such distance metrics reflect local correlations of data. In [5] we proposed a *soft* feature selection procedure (called LAC) that assigns weights to features according to the local correlations of data along each dimension. Dimensions along which data are loosely correlated receive a small weight, which has the effect of elongating distances along that dimension. Features along which data are strongly correlated receive a large weight, which has the effect of constricting distances along that dimension. Thus the learned weights perform a directional local reshaping of distances which allows a better separation of clusters, and therefore the discovery of different patterns in different subspaces of the original input space.

The clustering result of LAC depends on two input parameters. The first one is common to all clustering algorithms: the number of clusters k to be discovered in the data. The second one (called h) controls the

strength of the incentive to cluster on more features (more details are provided in Section 3). The setting of h is particularly difficult, since no domain knowledge for its tuning is likely to be available. Thus, it would be convenient if the clustering process automatically determined the relevant subspaces.

In this paper we design two cluster ensemble techniques for the LAC algorithm. Our cluster ensemble methods can be easily extended for any subspace clustering algorithm. We focus on setting the parameter h and assume that the number of clusters k is fixed. (The problem of finding k in an automated fashion through a cluster ensemble will be addressed in future work.) We leverage the diversity of the clusterings produced by LAC when different values of h are used, in order to generate a consensus clustering that is superior to the participating ones. The major challenge we face is to find a consensus partition from the outputs of the LAC algorithm to achieve an "improved" overall clustering of the data. Since we are dealing with weighted clusters, we need to design a proper consensus function that makes use of the weight vectors associated with the clusters. Our techniques leverage such weights to define a similarity measure which is associated to the edges of a graph. The problem of finding a consensus function is then mapped to a graph partitioning problem.

2 Related work

In many domains it has been shown that a classifier ensemble is often more accurate than any of the single components. This result has recently initiated further investigation in ensemble methods for clustering. In [8] the authors combine different clusterings obtained via the k-means algorithm. The clusterings produced by k-means are mapped into a co-association matrix, which measures the similarity between the samples. Kuncheva et al. [13] extend the work in [8] by choosing at random the number of clusters for each ensemble member. The authors in [16] introduce a meta-clustering procedure: first, each clustering is mapped into a distance matrix; second, the multiple distance matrices are combined, and a hierarchical clustering method is introduced to compute a consensus clustering. In [11] the authors propose a similar approach, where a graph-based partitioning algorithm is used to generate the combined clustering. Ayad et al. [1] propose a graph approach where data points correspond to vertices, and an edge exists between two vertices when the associated points share a specific number of nearest neighbors. In [6] the authors combine random projection with a cluster ensemble. EM is used as clustering algorithm, and an agglomerative approach is utilized to produce the final clustering. Greene et al. [10] apply an ensemble technique to medical diagnostic datasets. The authors focus on different generation and integration techniques for input clusterings to the ensemble. k-means, k-medoids and fast *weak clustering* are used as generation strategies. The diverse clusterings are aggregated into a co-occurrence matrix. Hierarchical schemes are then applied to compute the consensus clustering. Greene's approach follows closely Fred and Jain's approach [8]. However, they differ in the generation strategies. Similarly, in [2] the association between different clusterings produced by various algorithms is investigated. Techniques based on constrained and unconstrained clustering and on SVD are considered. Gionis et al.'s [9] approach finds an ensemble clustering that agrees as much as possible with the given clusterings. The proposed technique does not require the number of clusters as an input parameter, and handles missing data.

In [15] the authors propose a consensus function aimed at maximizing the normalized mutual information of the combined clustering with the input ones. Three heuristics are introduced: Cluster-based Similarity Partitioning Algorithm (CSPA), HyperGraph Partitioning Algorithm (HGPA), and Meta-Clustering Algorithm (MCLA). All three algorithms first transform the set of clusterings into a hypergraph representation.

In CSPA, a binary similarity matrix is constructed for each input clustering. Each column corresponds to a cluster: an entry has a value of 1 if the corresponding two points belong to the cluster, 0 otherwise. An entry-wise average of all the matrices gives an overall similarity matrix S. S is utilized to recluster the data using a graph-partitioning based approach. The induced similarity graph, where vertices correspond to data and edges' weights to similarities is partitioned using METIS [12].

HGPA seeks a partitioning of the hypergraph by cutting a minimal number of hyperedges. (Each hyperedge represents a cluster of an input clustering.) 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. It makes use of the package HMETIS.

MCLA is based on the clustering of clusters. It provides object-wise confidence estimates of cluster membership. Hyperedges are grouped, and each data point is assigned to the collapsed hyperedge in which it participates most strongly.

Since our weighted clustering ensemble approaches also map the problem of finding a consensus partition to a graph partitioning problem, it is natural to compare our techniques with the three algorithms CSPA, MCLA, and HGPA. The results of these experiments are presented in Section 6.

3 Locally Adaptive Clustering

Let us consider a set of n points in some space of dimensionality D. A *weighted cluster* C is a subset of data points, together with a vector of weights $\mathbf{w} = (w_1, \ldots, w_D)^t$, such that the points in C are closely clustered according to the L_2 norm distance weighted using $\mathbf{w}$. The component w_j measures the degree of correlation of points in C along feature j. The problem is how to estimate the weight vector $\mathbf{w}$ for each cluster in the dataset.

In traditional clustering, the partition of a set of points is induced by a set of *representative* vectors, also called *centroids* or *centers*. The partition induced by discovering weighted clusters is formally defined as follows.

Definition: Given a set S of n points $\mathbf{x} \in \Re^D$, a set of k centers $\{\mathbf{c}_1, \ldots, \mathbf{c}_k\}$, $\mathbf{c}_j \in \Re^D$, $j = 1, \ldots, k$, coupled with a set of corresponding weight vectors $\{\mathbf{w}_1, \ldots, \mathbf{w}_k\}$, $\mathbf{w}_j \in \Re^D$, $j = 1, \ldots, k$, partition S into k sets:

$$(3.1)\quad \begin{aligned} S_j &= \{\mathbf{x} | (\sum_{i=1}^{D} w_{ji}(x_i - c_{ji})^2)^{1/2} \\ &< (\sum_{i=1}^{D} w_{li}(x_i - c_{li})^2)^{1/2}, \forall l \neq j\}, j = 1, \ldots, k \end{aligned}$$

where w_{ji} and c_{ji} represent the ith components of vectors $\mathbf{w}_j$ and $\mathbf{c}_j$ respectively (ties are broken randomly).

The set of centers and weights is *optimal* with respect to the Euclidean norm, if they minimize the error measure:

$$(3.2)\quad E_1(C, W) = \sum_{j=1}^{k} \sum_{i=1}^{D} (w_{ji} \frac{1}{|S_j|} \sum_{\mathbf{x} \in S_j} (c_{ji} - x_i)^2)$$

subject to the constraints $\forall j \ \sum_i w_{ji} = 1$. C and W are $(D \times k)$ matrices whose columns are $\mathbf{c}_j$ and $\mathbf{w}_j$ respectively, i.e. $C = [\mathbf{c}_1 \ldots \mathbf{c}_k]$ and $W = [\mathbf{w}_1 \ldots \mathbf{w}_k]$. For notational brevity, we set $X_{ji} = \frac{1}{|S_j|} \sum_{\mathbf{x} \in S_j} (c_{ji} - x_i)^2$, where $|S_j|$ is the cardinality of set S_j. X_{ji} represents the average distance from the centroid $\mathbf{c}_j$ of points in cluster j along dimension i. The solution

$$(C^*, W^*) = \arg\min_{(C,W)} E_1(C, W)$$

will discover one-dimensional clusters: it will put maximal (unit) weight on the feature with smallest dispersion X_{ji} within each cluster j, and zero weight on all other features. Our objective, instead, is to find weighted multidimensional clusters, where the unit weight gets distributed among all features according to the respective dispersion of data within each cluster. One way to achieve this goal is to add the regularization term $\sum_{i=1}^{D} w_{ji} log w_{ji}$, which represents the negative entropy of the weight distribution for each cluster. It penalizes solutions with maximal weight on the single feature with smallest dispersion within each cluster. The resulting error function is

$$(3.3)\quad E_2(C, W) = \sum_{j=1}^{k} \sum_{i=1}^{D} (w_{ji} X_{ji} + h w_{ji} \log w_{ji})$$

subject to the same constraints $\forall j \ \sum_i w_{ji} = 1$. The coefficient $h \geq 0$ is a parameter of the procedure; it controls the strength of the incentive for clustering on more features. Increasing (decreasing) its value will encourage clusters on more (less) features. This constrained optimization problem can be solved by introducing the Lagrange multipliers. It gives the solution [5]:

$$(3.4)\quad w_{ji}^* = \frac{\exp(-X_{ji}/h)}{\sum_{i=1}^{D} \exp(-X_{ji}/h)}$$

$$(3.5)\quad c_{ji}^* = \frac{1}{|S_j|} \sum_{\mathbf{x} \in S_j} x_i$$

Solution (3.4) puts increased weights on features along which the dispersion X_{ji} is smaller, within each cluster. The degree of this increase is controlled by the value h. Setting $h = 0$ places all weight on the feature i with smallest X_{ji}, whereas setting $h = \infty$ forces all features to be given equal weight for each cluster j.

We need to provide a search strategy to find a partition P that identifies the solution clusters. We propose an approach that progressively improves the quality of initial centroids and weights, by investigating the space near the centers to estimate the dimensions that matter the most. We start with *well-scattered* points in S as the k centroids. We initially set all weights to $1/D$. Given the initial centroids $\mathbf{c}_j$, for $j = 1, \ldots, k$, we compute the corresponding sets S_j as previously defined. We then compute the average distance X_{ji} along each dimension from the points in S_j to $\mathbf{c}_j$. The smaller X_{ji}, the larger the correlation of points along dimension i. We use the value X_{ji} in an exponential weighting scheme to credit weights to features (and to clusters), as given in equation (3.4). The computed weights are used to update the sets S_j, and therefore the centroids' coordinates as given in equation (3.5). The procedure is iterated until convergence is reached.

We point out that LAC has shown a highly competitive performance with respect to other state-of-the-art subspace clustering algorithms [5]. Therefore, improving upon LAC performance is a desirable achievement.

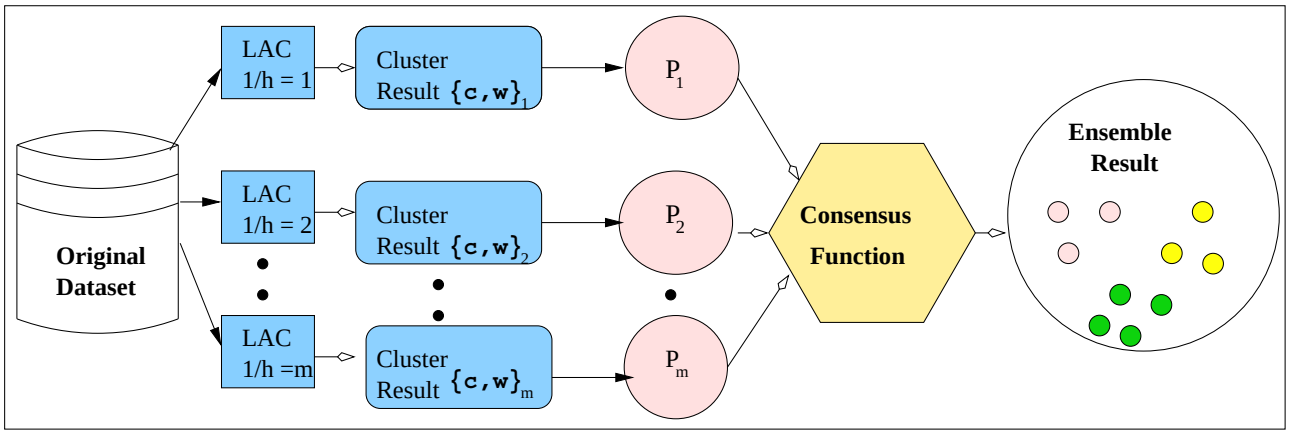

Figure 1: The clustering ensemble process

4 Weighted Clustering Ensembles

In the following we introduce two consensus functions to identify an emergent clustering that arises from multiple clustering results. We reduce the problem of defining a consensus function to a graph partitioning problem. This approach has shown good results in the literature [4, 15, 7]. Moreover, the *weighted clusters* computed by the LAC algorithm offer a natural way to define a similarity measure to be integrated in the weights associated to the edges of a graph. The overall clustering ensemble process is illustrated in Figure 1.

4.1 Weighted Similarity Partitioning Algorithm (WSPA) LAC outputs a partition of the data, identified by the two sets $\{\mathbf{c}_1, \ldots, \mathbf{c}_k\}$ and $\{\mathbf{w}_1, \ldots, \mathbf{w}_k\}$. Our aim here is to generate robust and stable solutions via a consensus clustering method. We can generate contributing clusterings by changing the parameter h (as illustrated in Figure 1). The objective is then to find a consensus partition from the output partitions of the contributing clusterings, so that an "improved" overall clustering of the data is obtained. Since LAC produces *weighted clusters*, we need to design a consensus function that makes use of the weight vectors associated with the clusters. The details of our approach are as follows.

For each data point $\mathbf{x}_i$, the weighted distance from cluster C_l is given by

$$d_{il} = \sqrt{\sum_{s=1}^{D} w_{ls}(x_{is} - c_{ls})^2}$$

Let $D_i = \max_l\{d_{il}\}$ be the largest distance of $\mathbf{x}_i$ from any cluster. We want to define the probability associated with cluster C_l given that we have observed $\mathbf{x}_i$. At a given point $\mathbf{x}_i$, the cluster label C_l is assumed to be a random variable from a distribution with probabilities $\{P(C_l|\mathbf{x}_i)\}_{l=1}^{k}$. We provide a nonparametric estimation of such probabilities based on the data and on the clustering result. We do not make any assumption about the specific form (e.g., Gaussian) of the underlying data distributions, thereby avoiding parameter estimations of models, which are problematic in high dimensions when the available data are limited.

In order to embed the clustering result in our probability estimations, the smaller the distance d_{il} is, the larger the corresponding probability credited to C_l should be. Thus, we can define $P(C_l|\mathbf{x}_i)$ as follows:

$$P(C_l|\mathbf{x}_i) = \frac{D_i - d_{il} + 1}{kD_i + k - \sum_l d_{il}} \tag{4.6}$$

where the denominator serves as a normalization factor to guarantee $\sum_{l=1}^{k} P(C_l|\mathbf{x}_i) = 1$. We observe that $\forall l = 1, \ldots, k$ and $\forall i = 1, \ldots, n$ $P(C_l|\mathbf{x}_i) > 0$. In particular, the added value of 1 in (4.6) allows for a non-zero probability $P(C_L|\mathbf{x}_i)$ when $L = \arg\max_l\{d_{il}\}$. In this last case $P(C_l|\mathbf{x}_i)$ assumes its minimum value $P(C_L|\mathbf{x}_i) = 1/(kD_i + k + \sum_l d_{il})$. For smaller distance values d_{il}, $P(C_l|\mathbf{x}_i)$ increases proportionally to the difference $D_i - d_{il}$: the larger the deviation of d_{il} from D_i, the larger the increase. As a consequence, the corresponding cluster C_l becomes more likely, as it is reasonable to expect based on the information provided by the clustering process. Thus, equation (4.6) provides a nonparametric estimation of the posterior probability associated to each cluster C_l.

We can now construct the vector P_i of posterior probabilities associated with $\mathbf{x}_i$:

$$P_i = (P(C_1|\mathbf{x}_i), P(C_2|\mathbf{x}_i), \ldots, P(C_k|\mathbf{x}_i))^t \tag{4.7}$$

where t denotes the transpose of a vector. The transformation $\mathbf{x}_i \rightarrow P_i$ maps the D dimensional data points $\mathbf{x}_i$

onto a new space of *relative coordinates* with respect to cluster centroids, where each dimension corresponds to one cluster. This new representation embeds information from both the original input data and the clustering result.

We then define the similarity between $\mathbf{x}_i$ and $\mathbf{x}_j$ as the cosine similarity between the corresponding probability vectors:

$$s(\mathbf{x}_i, \mathbf{x}_j) = \frac{P_i^t P_j}{\|P_i\|\|P_j\|} \tag{4.8}$$

We combine all pairwise similarities (4.8) into an $(n \times n)$ similarity matrix S, where $S_{ij} = s(\mathbf{x}_i, \mathbf{x}_j)$. We observe that, in general, each clustering may provide a different number of clusters, with different sizes and boundaries. The size of the similarity matrix S is independent of the clustering approach, thus providing a way to align the different clustering results onto the same space, with no need to solve a label correspondence problem.

After running the LAC algorithm m times for different values of the h parameter, we obtain the m similarity matrices $S_1, S_2, \ldots, S_m$. The combined similarity matrix Ψ defines a *consensus function* that can guide the computation of a consensus partition:

$$\Psi = \frac{1}{m} \sum_{l=1}^{m} S_l \tag{4.9}$$

Ψ_{ij} reflects the average similarity between $\mathbf{x}_i$ and $\mathbf{x}_j$ (through P_i and P_j) across the m contributing clusterings.

We now map the problem of finding a consensus partition to a graph partitioning problem. We construct a complete graph $G = (V, E)$, where $|V| = n$ and the vertex V_i identifies $\mathbf{x}_i$. The edge E_{ij} connecting the vertices V_i and V_j is assigned the weight value Ψ_{ij}. We run METIS [12] on the resulting graph to compute a k-way partitioning of the n vertices that minimizes the edge weight-cut. This gives the consensus clustering we seek. The size of the resulting graph partitioning problem is n^2. The steps of the algorithm, which we call WSPA (Weighted Similarity Partitioning Algorithm), are summarized in the following.

Input: n points $\mathbf{x} \in R^D$, and k.

1. Run LAC m times with different h values. Obtain the m partitions: $\{\mathbf{c}_1^\nu, \ldots, \mathbf{c}_k^\nu\}$,$\{\mathbf{w}_1^\nu, \ldots, \mathbf{w}_k^\nu\}$, $\nu = 1, \ldots, m$.
2. For each partition $\nu = 1, \ldots, m$:
   (a) Compute $d_{il}^\nu = \sqrt{\sum_{s=1}^{D} w_{ls}^\nu (x_{is} - c_{ls}^\nu)^2}$.
   (b) Set $D_i^\nu = \max_l\{d_{il}^\nu\}$.
   (c) Compute $P(C_l^\nu|\mathbf{x}_i) = \frac{D_i^\nu - d_{il}^\nu + 1}{kD_i^\nu + k - \sum_l d_{il}^\nu}$.
   (d) Set $P_i^\nu = (P(C_1^\nu|\mathbf{x}_i), P(C_2^\nu|\mathbf{x}_i), \ldots, P(C_k^\nu|\mathbf{x}_i))^t$.
   (e) Compute the similarity
   $$s^\nu(\mathbf{x}_i, \mathbf{x}_j) = \frac{P_i^\nu P_j^\nu}{\|P_i^\nu\|\|P_j^\nu\|}, \forall i, j$$
   (f) Construct the matrix S^ν where $S_{ij}^\nu = s^\nu(\mathbf{x}_i, \mathbf{x}_j)$.
3. Build the *consensus function* $\Psi = \frac{1}{m}\sum_{\nu=1}^{m} S^\nu$.
4. Construct the complete graph $G = (V, E)$, where $|V| = n$ and $V_i \equiv \mathbf{x}_i$. Assign Ψ_{ij} as the weight value of the edge E_{ij} connecting the vertices V_i and V_j.
5. Run METIS on the resulting graph G. Output the resulting k-way partition of the n vertices as the consensus clustering.

4.2 Weighted Bipartite Partitioning Algorithm (WBPA) Our second approach maps the problem of finding a consensus partition to a bipartite graph partitioning problem. This mapping was first introduced in [7]. In [7], however, 0/1 weight values are used. Here we extend the range of weight values to [0,1].

In this context, the graph models both instances (e.g., data points) and clusters, and the graph edges can only connect an instance vertex to a cluster vertex, thus forming a bipartite graph. In detail, we proceed as follows for the construction of the graph.

Suppose, again, that we run the LAC algorithm m times for different values of the h parameter. For each instance $\mathbf{x}_i$, and for each clustering $\nu = 1, \ldots, m$ we then can compute the vector of posterior probabilities P_i^ν, as defined in equations (4.7) and (4.6). Using the P vectors, we construct the following matrix A:

$$A = \begin{pmatrix} (P_1^1)^t & (P_1^2)^t & \ldots & (P_1^m)^t \\ (P_2^1)^t & (P_2^2)^t & \ldots & (P_2^m)^t \\ \vdots & \vdots & & \vdots \\ (P_n^1)^t & (P_n^2)^t & \ldots & (P_n^m)^t \end{pmatrix}$$

Note that the $(P_i^\nu)^t$s are row vectors (t denotes the transpose). The dimensionality of A is therefore $n \times km$, under the assumption that each of the m clusterings produces k clusters. (We observe that the definition of A can be easily generalized to the case where each clustering may discover a different number of clusters.)

Based on A we can now define a bipartite graph to which our consensus partition problem maps. Consider the graph $G = (V, E)$ with V and E constructed as follows. $V = V^C \cup V^I$, where V^C contains km vertices, each representing a cluster of the ensemble, and V^I contains n vertices, each representing an input data point. Thus $|V| = km + n$. The edge E_{ij} connecting the vertices V_i and V_j is assigned a weight value defined as follows. If the vertices V_i and V_j represent both clusters or both instances, then $E(i, j) = 0$; otherwise, if vertex

Table 1: Characteristics of the datasets

dataset	k	D	n (points-per-clsss)
Two-Gaussian	2	2	600 (300-300)
Three-Gaussian	3	2	900 (300-300-300)
Iris	3	4	150 (50-50-50)
WDBC	2	31	424 (212-212)
Breast	2	9	478 (239-239)
Modis-4	4	112	1989 (497-490-503-499)
Letter(A,B)	2	16	1555 (789-766)
SatImage	2	36	2110 (1072-1038)

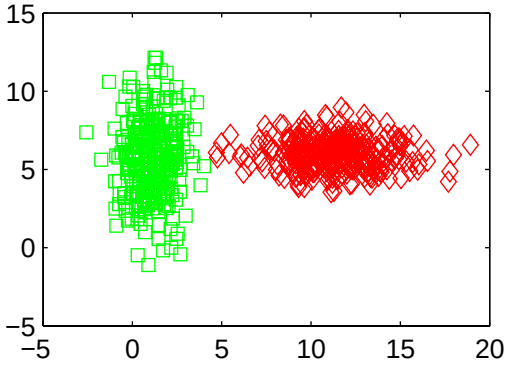

Figure 2: Two-Gaussian data

V_i represents an instance $\mathbf{x}_i$ and vertex V_j represents a cluster C_j^ν (or vice versa) then the corresponding entry of E is $A(i, k(\nu-1)+j)$. More formally:

- $E(i,j) = 0$ when $((1 \le i \le km)$ and $(1 \le j \le km))$ or $((km+1 \le i \le km+n)$ and $(km+1 \le j \le km+n))$ (This is the case in which V_i and V_j are both clusters or both instances.)
- $E(i,j) = A(i-km, j)$ when $(km+1 \le i \le km+n)$ and $(1 \le j \le km)$ (This is the case in which V_i is an instance and V_j is a cluster.)
- $E(i,j) = E(j,i)$ when $(1 \le i \le km)$ and $(km+1 \le j \le km+n)$ (This is the case in which V_i is a cluster and V_j is an instance.)

Note that the dimensionality of E is $(km+n)\times(km+n)$, and E can be written as follows:

$$E = \begin{pmatrix} 0 & A^t \\ A & 0 \end{pmatrix}$$

A partition of the bipartite graph G partitions the cluster vertices and the instance vertices simultaneously. The partition of the instances can then be output as the final clustering. Due to the special structure of the graph G (sparse graph), the size of the resulting bipartite graph partitioning problem is kmn. Assuming that $(km) << n$, this complexity is much smaller than the size n^2 of WSPA.

We again run METIS on the resulting bipartite graph to compute a k-way partitioning that minimizes the edge weight-cut. We call the resulting algorithm WBPA (Weighted Bipartite Partitioning Algorithm).

5 An Illustrative Example

We have designed one simulated dataset with two clusters distributed as bivariate Gaussians (Figure 2). The mean and standard deviation vectors for each cluster are as follows: $\mathbf{m}_1 = (0.5, 5)$, $\mathbf{s}_1 = (1, 9)$; $\mathbf{m}_2 = (12, 5)$, $\mathbf{s}_2 = (6, 2)$. Each cluster has 300 points. We ran the LAC algorithm on the Two-Gaussian dataset for two values of the $1/h$ parameter (7 and 12). For $(1/h) = 7$, LAC provides a perfect separation (the error rate is 0.0%); the corresponding weight vectors associated to each cluster are $\mathbf{w}_1^{(7)} = (0.81, 0.19)$, $\mathbf{w}_2^{(7)} = (0.18, 0.82)$. For $(1/h) = 12$, the error rate of LAC is 5.3%; the weight vectors in this case are $\mathbf{w}_1^{(12)} = (0.99, 0.01)$, $\mathbf{w}_2^{(12)} = (0.0002, 0.9998)$.

For the purpose of plotting the two-dimensional posterior probability vectors associated with each point $\mathbf{x}$, we consider a random sample of 100 points from each cluster (as shown in Figure 3). The probability vectors (computed as in equations (4.7) and (4.6)) of such sample points are plotted in Figures 4 and 6, respectively for $(1/h) = 7$ and $(1/h) = 12$. We observe that in Figure 4 ($(1/h) = 7$) for points $\mathbf{x}$ of cluster 1 (green points square-shaped) $P(C_1|\mathbf{x}) > P(C_2|\mathbf{x})$, and for points $\mathbf{x}$ of cluster 2 (red points diamond-shaped) $P(C_2|\mathbf{x}) > P(C_1|\mathbf{x})$. Thus, there is no overlapping (in relative coordinate space) between points of the two clusters, and LAC achieves a perfect separation (the error rate is 0.0%). On the other hand, Figure 6 ($(1/h) = 12$) demonstrates that for a few points $\mathbf{x}$ of cluster 1 (green points square-shaped) $P(C_1|\mathbf{x}) < P(C_2|\mathbf{x})$ (overlapping region in Figure 6). LAC misclassifies these points as members of cluster 2, which results in an error rate of 5.3%.

Thus, the relative coordinates $P(C|\mathbf{x})$ provide a suitable representation to compute the pairwise similarity measure in our clustering ensemble approaches. By combining the clustering results in the relative coordinate space obtained by different runs of LAC, we aim at leveraging the consensus across multiple clusterings, while averaging out emergent spurious structures. The results of the ensembles for this dataset are provided in Table 3 and in Figure 5. We observe that our two clus-

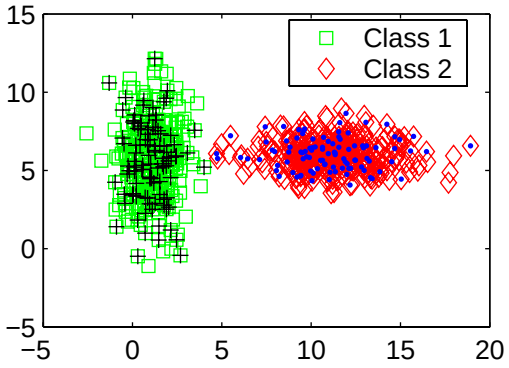

Figure 3: Random sampling of 100 points (crosses and dots) from each cluster.

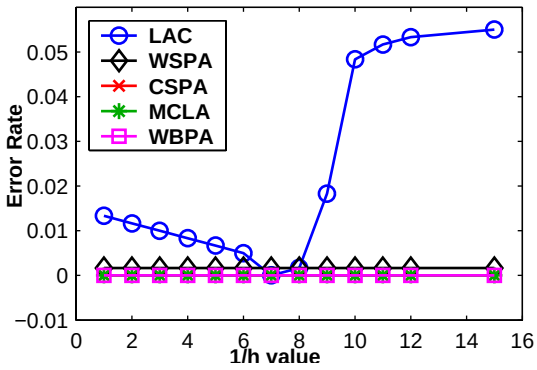

Figure 5: Results on Two-Gaussian data

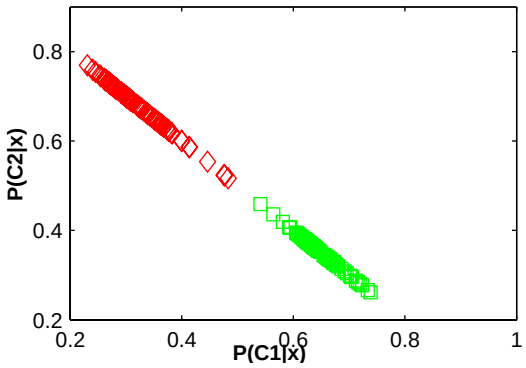

Figure 4: Two dimensional probability vectors $P = (P(C_1|\mathbf{x}), P(C_2|\mathbf{x}))^t$, $(1/h) = 7$. LAC error rate is 0.0%.

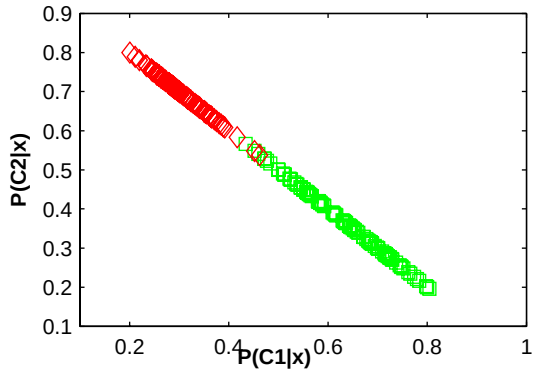

Figure 6: Two dimensional probability vectors $P = (P(C_1|\mathbf{x}), P(C_2|\mathbf{x}))^t$, $(1/h) = 12$. LAC error rate is 5.3%.

tering ensemble methods WSPA and WBPA achieved 0.17% and 0.0% error rates. Thus, they successfully separated the two clusters, as the best input clustering provided by LAC did.

6 Experimental Results

We have designed two simulated datasets with two and three clusters, respectively, distributed as bivariate Gaussians (Figures 2 and 7). We also tested our technique on six real datasets. The characteristics of all datasets are given in Table 1. Iris, Breast, Letter, and SatImage are from the UCI Machine Learning Repository. WDBC is the Wisconsin Diagnostic Breast Cancer dataset [3]. The Modis-4 dataset (land-cover classification #2) was downloaded from http://mow.ecn.purdue.edu/∼xz/ (the first four classes, which are balanced, were used in our experiments). Since METIS requires balanced datasets, we performed random sampling on the WDBC and Breast datasets. In each case, we sub-sampled the most populated class: from 357 to 212 for WDBC, and from 444 to 239 for Breast. For the Letter dataset, we used the classes "A" and "B" (balanced), and for the SatImage classes 1 and 7 (again balanced).

We compare our weighted clustering ensemble techniques (WSPA and WBPA) with the three methods CSPA, HGPA, and MCLA [15]. These three techniques also transform the problem of finding a consensus clustering to a graph partitioning problem, and make use of METIS. Thus, it was a natural choice for us to compare our methods with these approaches. In this paper we report the accuracy achieved by CSPA and MCLA, as HGPA was consistently the worst. The ClusterPack Matlab Toolbox was used (available at: www.lans.ece.utexas.edu/∼strehl/).

Evaluating the quality of clustering is in general a difficult task. Since class labels are available for the datasets used here, we evaluate the results by computing

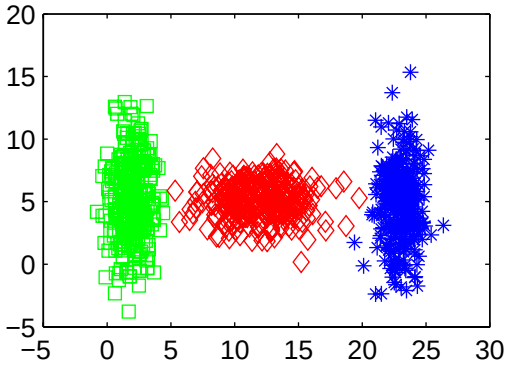

Figure 7: Three-Gaussian data

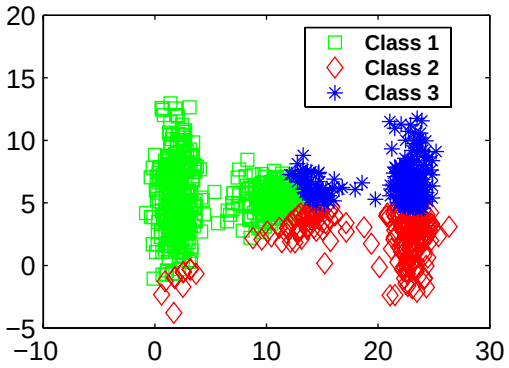

Figure 8: LAC: Clustering results for Three-Gaussian data, $(1/h) = 1$. The error rate is 34.6%

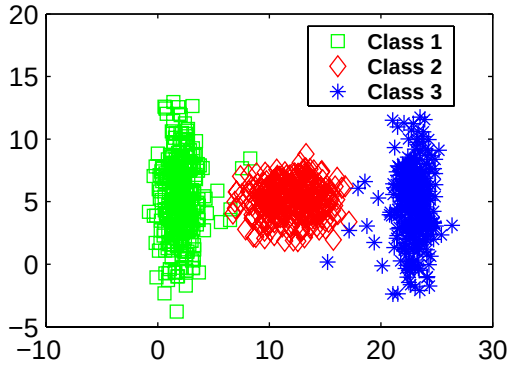

Figure 9: LAC: Clustering results for Three-Gaussian data, $(1/h) = 4$. The error rate is 1.3%

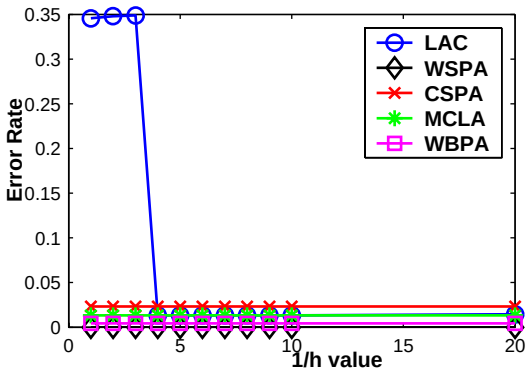

Figure 10: Results on Three-Gaussian data

the error rate and the normalized mutual information (NMI). The error rate is computed according to the confusion matrix. The NMI provides a measure that is impartial with respect to the number of clusters [15]. It reaches its maximum value of one only when the result completely matches the original labels. The NMI is computed according to the average mutual information between every pair of cluster and class [15]:

$$NMI = \frac{\sum_{i=1}^{k}\sum_{j=1}^{k} n_{i,j} \log \frac{n_{i,j} n}{n_i n_j}}{\sqrt{\sum_{i=1}^{k} n_i \log \frac{n_i}{n} \sum_{j=1}^{k} n_j \log \frac{n_j}{n}}}$$

where $n_{i,j}$ is the number of agreement between cluster i and class j, n_i is the number of data in cluster i, n_j is the number of data in class j, and n is the total number of points.

6.1 Analysis of the Results For each dataset, we run the LAC algorithm for several values of the input parameter h. The clustering results of LAC are then given in input to the consensus clustering techniques being compared. (As the value of k, we input both LAC and the ensemble algorithms with the actual number of classes in the data.) Figures 5 and 10-16 plot the error rate (%) achieved by LAC as a function of the $1/h$ parameter, for each dataset considered. The corresponding error rates of our weighted clustering ensemble methods (WSPA and WBPA) and of the CSPA and MCLA techniques are also reported. Each figure clearly shows the sensitivity of the LAC algorithm to the value of h. The trend of the error rate clearly depends on the data distribution.

We further illustrate the sensitivity of the LAC algorithm to the value of h for the Three-Gaussian data (Figure 7). Figures 8 and 9 depict the clustering results of LAC for $(1/h) = 1$ and $(1/h) = 4$, respectively. Figure 8 clearly shows that for $(1/h) = 1$, LAC is unable to discover the structure of the three clusters, and gives an error rate of 34.6%. On the other hand, LAC achieves a nearly perfect seperation for $(1/h) = 4$,

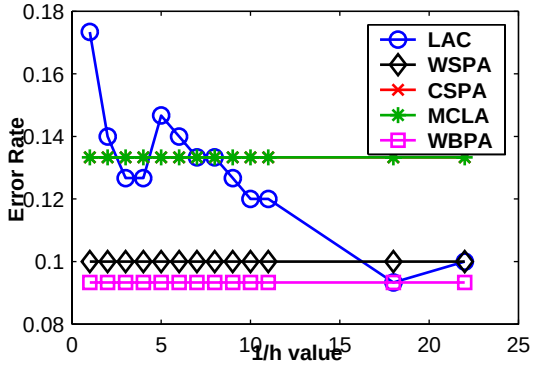

Figure 11: Results on Iris data

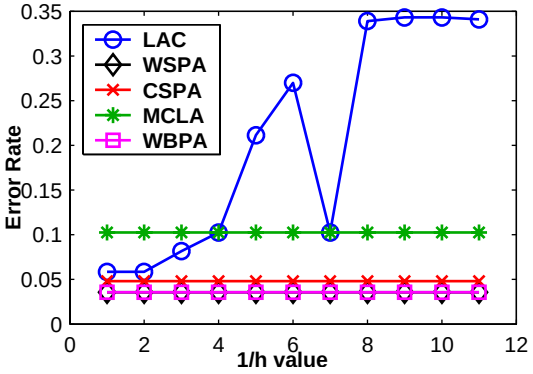

Figure 13: Results on Breast data

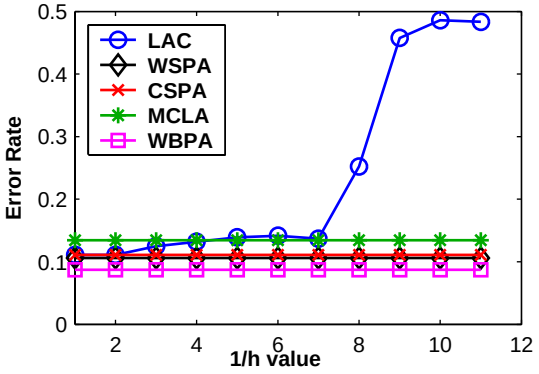

Figure 12: Results on WDBC data

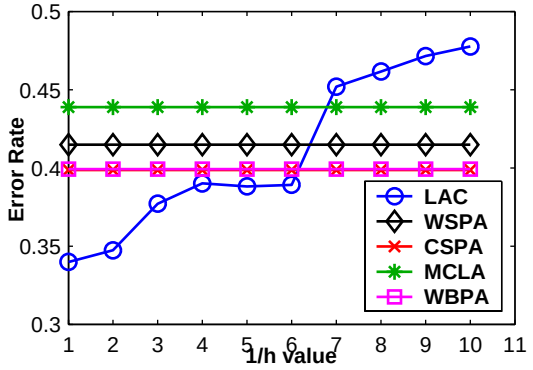

Figure 14: Results on Modis data

as shown in Figure 9. The error rate in this case is 1.3%, which is also the minimum achieved in all the runs of the algorithm. Results for the ensemble techniques on the Three-Gaussian data are given in Figure 10 and in Table 4. We observe that the WSPA technique perfectly separates the data (0.0% error), and WBPA gives a 0.44% error rate. In both cases, the error achieved is lower than the minimum error rate among the input clusterings (1.3%). Thus, the spurious structure identified by some runs of LAC was successfully filtered out by our ensemble techniques, which also perform better than any single run of LAC.

Detailed results for all data are provided in Tables 3-10, where we report the NMI and error rate (ER) of the ensembles, and the maximum, minimum, and average NMI and error rate values for the input clusterings. Each ensemble method is given as input the complete set of clusterings obtained by LAC.

We observe that for all six real datasets either WBPA or WSPA provides the lowest error rate among the four methods being compared. For the Iris, WDBC, Breast, and SatImage (four out of six total) datasets the error rate provided by the WBPA technique is as good or better than the best individual input clustering. For the Modis-4 and Letter(A,B) datasets, the error rate of WBPA is still below the average error rate of the input clusterings. Moreover, for each real dataset the WSPA technique provides the second best error rate, with a tie for the best error rate on the Breast data. For the WDBC, Breast, and SatImage datasets, the error rate provided by WSPA is also better than the best individual input clustering. For the Iris and Letter(A,B) datasets the error rate of WSPA is below the average error rate of the input clusterings (and still close to the average for the Modis-4 data).

Clearly, our weighted clustering ensemble techniques are capable of achieving superior accuracy results with respect to the CSPA and MCLA techniques on the tested datasets. This result is summarized in Table 2, where we report the average NMI and average error rate on all real datasets. We also report the average values for the LAC algorithm to emphasize the

Table 2: Average NMIs and error rates

	Avg-NMI	Avg-Error
WSPA	**0.557**	**14.9**
CSPA	0.517	16.3
MCLA	0.502	17.9
WBPA	**0.575**	**14.1**
LAC	0.454	21.2

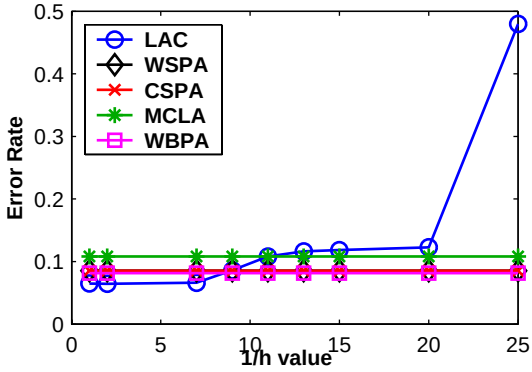

Figure 15: Results on Letter (A,B) dataset

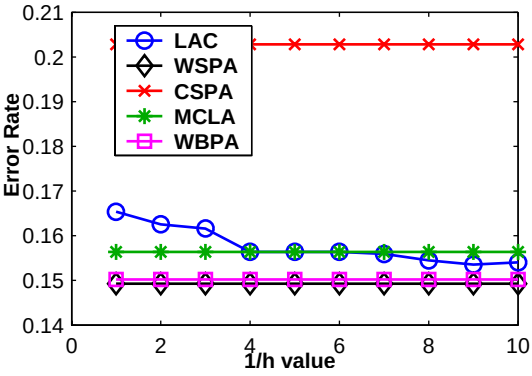

Figure 16: Results on SatImage dataset

large improvements obtained by the ensembles across the real datasets. Given the competitive behavior shown by LAC in the literature [5], this is a significant result.

We observe that the consensus function Ψ defined in (4.9) measures the similarity of points in terms of how close the "patterns" captured by the corresponding probability vectors are. As a consequence, Ψ (as well as the matrix A for the WBPA technique) takes into account not only how often the points are grouped together across the various input clusterings, but also the degree of confidence of the groupings. On the other hand, the CSPA and MCLA approaches take as input the partitions provided by each contributing clustering algorithm. That is, $\forall \nu$ and $\forall i$, $P(C_l^\nu|\mathbf{x}_i) = 1$ for a given l, and 0 otherwise. Thus, the information concerning the degree of confidence associated with the clusterings is lost. This is likely the reason for the superior performance achieved by our weighted clustering ensemble algorithms.

In several cases, the WBPA technique gives a lower error rate compared to the WSPA technique. These results may be due to a conceptual advantage of WBPA with respect to WSPA. We observe that the consensus function ψ used in WSPA measures pairwise similarities which are solely instance-based. On the other hand, the bipartite graph partitioning problem, to which the WBPA technique reduces, partitions both cluster vertices and instance vertices simultaneously. Thus, it also accounts for similarities between clusters. Consider, for example, four instances $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$, and $\mathbf{x}_4$. Suppose that $\mathbf{x}_1$ and $\mathbf{x}_2$ are never clustered together in the input clusterings, and the same holds for $\mathbf{x}_3$ and $\mathbf{x}_4$. However, the groups to which $\mathbf{x}_1$ and $\mathbf{x}_2$ belong often share the same instances, but this is not the case for the groups $\mathbf{x}_3$ and $\mathbf{x}_4$ belong to. Intuitively, we would consider $\mathbf{x}_1$ and $\mathbf{x}_2$ more similar to each other than $\mathbf{x}_3$ and $\mathbf{x}_4$. But WSPA is unable to distinguish these two cases, and may assign low similarity values to both pairs. On the other hand, WBPA is able to differenciate the two cases by modeling both instance-based and cluster-based similarities.

7 Conclusions and Future Work

We have introduced two cluster ensemble techniques for the LAC algorithm. The experimental results show that our weighted clustering ensembles can provide solutions that are as good as or better than the best individual clustering, provided that the input clusterings are diverse.

In our future work we will consider utilizing our consensus function as a similarity matrix for hierarchical and spectral clustering. This approach will eliminate the requirement for balanced clusters. We will extend our approach to be used with any subspace clustering technique. In addition, we aim at designing an ensemble that preserves a subspace clustering structure. One possibility is to leverage the weight vectors associated with the input clustering that shares the highest NMI with the clustering produced by the ensemble (this can be performed using the RAND statistic). Another possibility is to infer a set of dimensions for each cluster from the clustering result of the ensemble.

The diversity-accuracy requirements of the individ-

Table 3: Results on Two-Gaussian data

Methods	Ens-NMI	Ens-ER	Max-NMI	Min-NMI	Avg-NMI	Max-ER	Min-ER	Avg-ER
WSPA	0.984	0.17	1	0.75	0.88	5.5	0	2.2
CSPA	1	**0**	1	0.75	0.88	5.5	0	2.2
MCLA	1	**0**	1	0.75	0.88	5.5	0	2.2
WBPA	1	**0**	1	0.75	0.88	5.5	0	2.2

Table 4: Results on Three Gaussian data

Methods	Ens-NMI	Ens-ER	Max-NMI	Min-NMI	Avg-NMI	Max-ER	Min-ER	Avg-ER
WSPA	1	**0**	0.940	0.376	0.789	34.9	1.3	10.5
CSPA	0.893	2.3	0.940	0.376	0.789	34.9	1.3	10.5
MCLA	0.940	1.3	0.940	0.376	0.789	34.9	1.3	10.5
WBPA	0.976	**0.44**	0.940	0.376	0.789	34.9	1.3	10.5

ual clusterings, in order for the ensemble to be effective, will be also investigated. It is expected that the accuracy of the ensemble improves when a larger number of input clusterings is given, provided that the contributing clusterings are diverse.

Acknowledgments

This work was in part supported by NSF CAREER Award IIS-0447814.

References

[1] H. Ayad and M. Kamel. Finding natural clusters using multi-clusterer combiner based on shared nearest neighbors. *Multiple Classifier Systems, Fourth International Workshop*, 2003.

[2] C. Boulis and M. Ostendorf. Combining multiple clustering systems. *Principles and Practice of Knowledge Discovery in Databases*, 2004.

[3] J. Demsar, B. Zupan, G. Leban. Orange: From Experimental Machine Learning to Interactive Data Mining. White Paper (www.ailab.si/orange), Faculty of Computer and Information Science, University of Ljubljana, 2004.

[4] I. S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. *ACM International Conference on Knowledge Discovery and Data Mining*, 2001.

[5] C. Domeniconi, D. Papadopoulos, D. Gunopulos, and S. Ma. Subspace clustering of high dimensional data. *SIAM International Conference on Data Mining*, 2004.

[6] X. Z. Fern and C. E.Brodley. Random projection for high dimensional data clustering: A cluster ensemble approach. *International Conference on Machine Learning*, 2003.

[7] X. Z. Fern and C. E.Brodley. Solving Cluster Ensemble Problems by Bipartite Graph Partitioning. *International Conference on Machine Learning*, 2004.

[8] A. Fred and A. Jain. Data clustering using evidence accumulation. *International Conference on Pattern Recognition*, 2002.

[9] A. Gionis, H. Mannila, and P. Tsaparas. Clustering aggregation. *International Conference on Data Engineering*, 2005.

[10] D. Greene, A. Tsymbal, N. Bolshakova, and P. Cunningham. Ensemble clustering in medical diagnostics. *IEEE Symposium on Computer-Based Medical Systems*, 2004.

[11] X. Hu. Integration of cluster ensemble and text summarization for gene expression analysis. *Symposium on Bioinformatics and Bioengineering*, 2004.

[12] G. Kharypis and V. Kumar. Multilevel k-way partitioning scheme for irregular graphs. Technical report, University of Minnesota Department of Computer Science and Army HPC Research Center, 1995.

[13] L. Kuncheva and S. Hadjitodorov. Using diversity in cluster ensembles. *International Conference on Systems, Man and Cybernetics*, 2004.

[14] L. Parsons, E. Haque, and H. Liu. Subspace clustering for high dimensional data: a review. *SIGKDD Explor. Newsl.*, 6(1):90–105, 2004.

[15] A. Strehl and J. Ghosh. Cluster ensembles — a knowledge reuse framework for combining multiple partitions. *J. Mach. Learn. Res.*, 3:583–617, 2003.

[16] Y. Zeng, J. Tang, J. Garcia-Frias, and G. R. Gao. An adaptive meta-clustering approach: Combining the information from different clustering results. *Computational Systems Bioinformatics Conference*, 2002.

Table 5: Results on Iris data

Methods	Ens-NMI	Ens-ER	Max-NMI	Min-NMI	Avg-NMI	Max-ER	Min-ER	Avg-ER
WSPA	0.744	**10.0**	0.758	0.657	0.709	17.3	9.3	12.9
CSPA	0.677	13.3	0.758	0.657	0.709	17.3	9.3	12.9
MCLA	0.708	13.3	0.758	0.657	0.709	17.3	9.3	12.9
WBPA	0.754	**9.3**	0.758	0.657	0.709	17.3	9.3	12.9

Table 6: Results on WDBC data

Methods	Ens-NMI	Ens-ER	Max-NMI	Min-NMI	Avg-NMI	Max-ER	Min-ER	Avg-ER
WSPA	0.512	**10.6**	0.524	0.009	0.329	48.5	11.1	23.4
CSPA	0.498	11.1	0.524	0.009	0.329	48.5	11.1	23.4
MCLA	0.457	13.4	0.524	0.009	0.329	48.5	11.1	23.4
WBPA	0.573	**8.7**	0.524	0.009	0.329	48.5	11.1	23.4

Table 7: Results on Breast data

Methods	Ens-NMI	Ens-ER	Max-NMI	Min-NMI	Avg-NMI	Max-ER	Min-ER	Avg-ER
WSPA	0.779	**3.6**	0.700	0.197	0.422	34.1	5.9	20.5
CSPA	0.722	4.8	0.700	0.197	0.422	34.1	5.9	20.5
MCLA	0.575	10.3	0.700	0.197	0.422	34.1	5.9	20.5
WBPA	0.779	**3.6**	0.700	0.197	0.422	34.1	5.9	20.5

Table 8: Results on Modis-4 data

Methods	Ens-NMI	Ens-ER	Max-NMI	Min-NMI	Avg-NMI	Max-ER	Min-ER	Avg-ER
WSPA	0.336	41.5	0.448	0.229	0.326	47.8	33.9	40.9
CSPA	0.355	**39.9**	0.448	0.229	0.326	47.8	33.9	40.9
MCLA	0.335	43.9	0.448	0.229	0.326	47.8	33.9	40.9
WBPA	0.359	**39.9**	0.448	0.229	0.326	47.8	33.9	40.9

Table 9: Results on Letter(A,B) data

Methods	Ens-NMI	Ens-ER	Max-NMI	Min-NMI	Avg-NMI	Max-ER	Min-ER	Avg-ER
WSPA	0.579	**8.6**	0.707	0.001	0.514	47.9	6.4	13.6
CSPA	0.579	**8.6**	0.707	0.001	0.514	47.9	6.4	13.6
MCLA	0.512	10.8	0.707	0.001	0.514	47.9	6.4	13.6
WBPA	0.592	**8.2**	0.707	0.001	0.514	47.9	6.4	13.6

Table 10: Results on SatImage data

Methods	Ens-NMI	Ens-ER	Max-NMI	Min-NMI	Avg-NMI	Max-ER	Min-ER	Avg-ER
WSPA	0.392	**14.9**	0.433	0.400	0.423	16.5	15.4	15.8
CSPA	0.273	20.3	0.433	0.400	0.423	16.5	15.4	15.8
MCLA	0.427	15.6	0.433	0.400	0.423	16.5	15.4	15.8
WBPA	0.389	**15.0**	0.433	0.400	0.423	16.5	15.4	15.8