Adaptive Evidence Accumulation Clustering Using the Confidence of the Objects' Assignments

João M.M. Duarte[1,2], Ana L.N. Fred[2], and F. Jorge F. Duarte[1]

[1] GECAD - Knowledge Engineering and Decision Support Group,
Institute of Engineering, Polytechnic of Porto (ISEP/IPP),
Porto, Portugal
{jod,fjd}@isep.ipp.pt

[2] Instituto de Telecomunicações,
Instituto Superior Técnico,
Lisboa, Portugal
afred@lx.it.pt

Abstract. Ensemble methods are known to increase the performance of learning algorithms, both on supervised and unsupervised learning. Boosting algorithms are quite successful in supervised ensemble methods. These algorithms build incrementally an ensemble of classifiers by focusing on objects previously misclassified while training the current classifier. In this paper we propose an extension to the Evidence Accumulation Clustering method inspired by the Boosting algorithms. While on supervised learning the identification of misclassified objects is a trivial task because the labels for each object are known, on unsupervised learning these are unknown, making it difficult to identify the objects on which the clustering algorithm should focus. The proposed approach uses the information contained in the co-association matrix to identify degrees of confidence of the assignments of each object to its cluster. The degree of confidence is then used to select which objects should be emphasized in the learning process of the clustering algorithm. New consensus partition validity measures, based on the notion of degree of confidence, are also proposed. In order to evaluate the performance of our approaches, experiments on several artificial and real data sets were performed and shown the adaptive clustering ensemble method and the consensus partition validity measure help to improve the quality of data clustering.

1 Introduction

The general goal of data clustering is to find structure in data. Specifically, clustering consists of grouping a set of objects into clusters, such that similar objects are assigned to the same cluster and distinct objects are assigned to different clusters, according to some notion of similarity between data. A large number of clustering algorithms have been proposed over time. However, none of the clustering algorithms can alone discover all sorts of shapes and structures of clusters.

In the last decade, several clustering ensemble methods were proposed stimulated by the effectiveness of classifier ensemble methods. These methods combine multiple data partitions to improve data clustering robustness and quality [9], reuse single-run clustering algorithms solutions [18], cluster data distributively, speed-up clustering process and cluster data with heterogeneous features.

Boosting algorithms have been very successful in supervised learning. These algorithms combine *weak* classifiers iteratively, such that, objects misclassified in previous iterations have greater importance in the current learning iteration [10]. By focusing on regions containing objects more difficult to classify it is expected the combination of these *weak* learners lead to a *strong* classifier. On unsupervised learning the class of each training object is unknown making the identification of the misclassified objects very difficult. Topchy et al. [20] proposed a clustering ensemble construction method following the boosting principles by checking the consistency of the objects' assignments on the previous iterations. At each iteration a new data set is subsampled. An object consistency index is computed as the fraction of the maximal number of times an object was grouped is a certain cluster over the current number of data partitions. The probability of an object being selected is the weighted sum of the object consistency index plus the probability of the object in the previous iteration. Zhai et al. [23] proposed a fuzzy clustering ensemble method based on dual boosting. Fuzzy partitions produced from subsamples of the original data are iteratively mapped into a co-association matrix and the probability distribution of an object being selected is computed so that objects easy and hard to cluster have great importance in the clustering process. Saffari and Bischof [15] introduced an unified and generic boosting framework which builds the clustering ensemble using any model-based clustering algorithm.

We propose an adaptive clustering ensemble construction method for Evidence Accumulation Clustering [7]. The clustering ensemble is build iteratively using an object weight clustering algorithm which focuses the learning process on the objects with more weight. After building each partition the co-association matrix is updated and the object weights are computed given the degrees of confidence of assigning each object to its cluster. The degrees of confidence are estimated using the similarity space induced from the co-association matrix and are viewed as indicators of how good/bad the objects are clustered. Comparing with the boosting methods mentioned before, our approach does not rely on subsampling techniques or a model-based clustering algorithm, but rather on an object-weighted clustering algorithm. Also, in order to build the clustering ensemble, the number of clusters for each data partition is not required to be the *natural* number of clusters, which makes it possible to use the Evidence Accumulation Clustering's split-and-merge strategy [8]. In this paper, we study the effect of focusing on the objects hard to cluster, on the objects easy to cluster, and on a mix of the previous. We also use the notion of degree of confidence to assess the quality of the produced consensus data partitions.

The rest of this paper is organized as follows. In Section 2, the clustering combination problem is introduced. An adaptive clustering ensemble approach and an object-weighted clustering algorithm are proposed in Section 3. The consensus clustering validity is addressed in Section 4. In Section 5, the experimental setup and results are discussed. Section 6 concludes this paper.

2 Clustering Combination

2.1 Problem Definition

Let $\mathcal{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$ be a data set with n objects and $\mathbf{X} = \left[\mathbf{x}_1^T, \cdots, \mathbf{x}_n^T\right] \in \mathbb{R}^{n \times d}$ its matricial representation s.t. $\mathbf{x}_i = [x_{i1}, \cdots, x_{id}]^T \in \mathbb{R}^d$ is a vector containing the values for d attributes that describes $\mathbf{x}_i$. A clustering ensemble, $\mathcal{P}$, is defined as a set of N data partitions of $\mathcal{X}$:

$$\mathcal{P} = \{P^1, \cdots, P^N\}, \ \ P^c = \{C_1^c, \cdots, C_{K^c}^c\}, \tag{1}$$

where C_k^c is the k^{th} cluster in data partition P^c, which contains K^c clusters. Different partitions capture different views of the structure of the data. Clustering ensemble methods use a consensus function f which maps a clustering ensemble $\mathcal{P}$ into a consensus partition $P^* = f(\mathcal{P})$.

2.2 Related Work

Clustering ensemble approaches may be categorized according to the way data partitions belonging to clustering ensemble are produced – the *clustering generation step* – and to the combination scheme of them – the *consensus step*. Figure 1 shows an overview on multiple data clustering combination. The main approaches for the clustering generation and consensus steps are presented next.

Clustering Generation Step - The process of building the clustering ensemble defines how the data partitions which are going to be combined are generated. In this step, it is important to create diversity among the clustering ensemble in order to produce consensus partitions of superior quality [11]. In the clustering ensemble step the following options may be used separately or in combination.

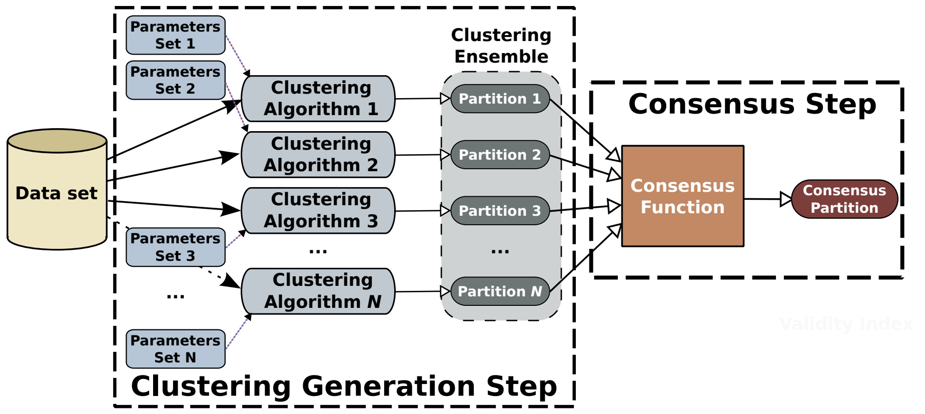


Fig. 1. Clustering ensemble steps

- Clustering algorithms - The data partitions may be produced using only one clustering algorithm or using several clustering algorithms [3]. In this case, diversity is created by optimizing distinct objective functions.
- Parameters and initializations - Even if only one clustering algorithm is used, diversity may be obtained by using different parameters and/or initializations. For instance, the k-means algorithm may be applied for each data partition using different number of clusters and initializations of centroids [7].
- Subsets of data objects - Each data partition may be produced using different sets of data objects. In real-life scenarios the data may be spread in different physical locations. Instead of concentrate all the data in one location, one may produce data clusterings at every locations, centralize only these clusterings, and then obtain the consensus clustering. Even if all the data is centralized, it may be advantageous to use different subsets of data. The use of resampling techniques may increase stability, robustness and quality in consensus clustering [14,20], and the use of subsampling techniques [3] can also speed-up the clustering generation step.
- Subsets of data features - The data partitions may be generated using all the features of the data set or by selecting distinct subsets of data features for each data partition [1]. Each subset of features can be considered as a partial view of the data, thereby, the clustering combination may be thought as an aggregation of distinct views of the data. Using subsets of data features also enables clustering data distributively, reduces memory usage, and enables the clustering of heterogeneous data.
- Projecting to subspaces - To prevent the use of noisy or irrelevant features, and to avoid the problem of the "curse of dimensionality" in high dimensional data, some clustering ensemble construction methods project the original data space into a lower dimensional data space before building the clustering ensemble. Fern and Brodley proposed the use of the random projection technique to build the clustering ensemble [5]. In this method, the original data features are linearly combined using random weights. Topchy et al. proposed to build ensembles of *weak* clusterings by projecting the feature space into only one dimension or by splitting the data by random hyperplanes [19].

Consensus Step - This step defines how the multiple data partitions are combined into consensus partitions. The most popular approaches are presented below.

- Majority voting - The majority voting approaches are the most commonly used in supervised classifier ensembles. Each classifier "votes" for the class of the object $\mathbf{x}_i$, and then $\mathbf{x}_i$ is given the class with more votes. The problem is more complex in unsupervised learning because the labels of the objects do not represent the objects classes, i.e., an object having the same label on different clusterings do not mean that the object was assign to the same class twice. Therefore, the cluster correspondence problem need to be solved to perform majority voting [22,4].
- Co-associations between pairs of objects - These methods store in a $n \times n$ matrix the frequency in which each pair of objects was grouped in the same cluster in all the partitions belonging to the cluster ensemble. This matrix may be viewed as a similarity matrix between objects, so a clustering algorithm can be used to produce the consensus partition [7].

- Searching for the median partition - Some approaches define the consensus clustering as finding the partition P^* that maximizes the average similarity between P^* and all the partitions belonging to the cluster ensemble. Topchy et al. proposed to maximize the Average Normalized Mutual Information by applying the k-means algorithm to a particular representation of the clustering ensemble [19]. Jouve and Nicoloyannis [12] proposed to represent the clustering ensemble as a categorical data set and search for the median partition using a categorical data clustering algorithm.
- Mapping the clustering ensemble into graph or hypergraph problems - Some approaches capture the relations between objects and transform them into graph problems. The CSPA [18] and IBGF [6] methods are some examples. Other approaches map the relations between the clusters in the clustering ensemble into graph problems (e.g. the CBGF [6] and WSPA [2] methods), or hypergraph problems (e.g. the HGPA and MCLA methods [18]). Another approaches, such as HBGF [6] and WBPA [2], represent both objects and clusters as vertices of a graph and map the object-to-cluster relations as edges.

For more informations on this topic, the interested reader may check the survey by Vega-Pons and Ruiz-Shulcloper [21].

2.3 Evidence Accumulation Clustering

The Evidence Accumulation Clustering method (EAC) [7] considers each data partition $P^c \in \mathcal{P}$ as an independent evidence of data organization. The underlying assumption of EAC is that two objects belonging to the same “natural” cluster will be frequently grouped together. A vote is given to a pair of objects every time they co-occur in the same cluster. Pairwise votes are stored in a $n \times n$ co-association matrix, $\mathbf{C}$, normalized by the total number of combined data partitions:

$$\mathbf{C}_{ij} = \frac{\sum_{l=1}^{N} vote_{ij}^{c}}{N}, \tag{2}$$

where $vote_{ij}^{c} = 1$ if $\mathbf{x}_i$ and $\mathbf{x}_j$ co-occur in a cluster of data partition P^c; otherwise $vote_{ij}^{c} = 0$. The consensus partition is obtained by applying some clustering algorithm over the co-association matrix, $\mathbf{C}$.

3 Proposed Combination Method

3.1 Adaptive Clustering Ensembles

In this section, an extension to the Evidence Accumulation Clustering method is proposed. It is inspired by the supervised learning Boosting algorithms, where a different weight is assigned to each object depending on its hardness to be well classified. We will refer to the proposed algorithm as Adaptive Evidence Accumulation Clustering (AdaEAC).

Our method relies on estimating the degree of confidence of assigning an object $\mathbf{x}_i$ to its cluster C_k, using the information contained in the co-association matrix $\mathbf{C}$. The idea is simple: if the average similarity of $\mathbf{x}_i$ with respect to the other objects belonging to the same cluster ($\{\mathbf{x}_j : \mathbf{x}_j \in C_k\}$) is higher than the average similarity to the objects belonging to the closest cluster (excluding C_k), then $\mathbf{x}_i$ probably was well assigned. Otherwise, the confidence of the assignment is low and $\mathbf{x}_i$ probably should have been assigned to the other cluster. The degree of confidence of assigning an object $\mathbf{x}_i$ to its cluster C_{P_i} is computed as

$$\mathrm{conf}(\mathbf{x}_i) = \left(\frac{1}{|C_{P_i}| - 1} \sum_{j:\mathbf{x}_j \in \{C_{P_i}\} \backslash \mathbf{x}_i} C_{ij} \right) - \left(\max_{1 \le k \le K, k \ne P_i} \frac{1}{|C_k|} \sum_{j:\mathbf{x}_j \in C_k} C_{ij} \right) \quad (3)$$

where $|\cdot|$ is the cardinality of a set.

While in the EAC approach all N data partitions belonging to the clustering ensemble are assumed to already exist, in AdaEAC the clusterings are produced in F folds. In each fold, an object-weighted clustering algorithm uses as input the object weights obtained in the previous fold to bias the production of L data partitions which are used to update the co-association matrix $\mathbf{C}$. After the co-association matrix $\mathbf{C}$ is updated, a consensus clustering algorithm is applied to $\mathbf{C}$ to obtain the current consensus partition P^* and the degree of confidence for each object is computed as described in Eq. 3. Finally, the weights of the objects for the next iteration can be update considering the degrees of confidence for the assignments of all objects. The proposed approach is summarized in Algorithm 1. In this paper, we assume the clustering ensemble $\mathcal{P}$ is built using the split-and-merge strategy. In this setting, the data partitions belonging to the clustering ensemble have an higher number of clusters than the *real* number of clusters, so, the clusters are smaller but more dense. The clustering ensemble is constructed by generating each data partition P^c with K^c clusters, where K^c is a random integer (different for each P^c) belonging to the set $\{K_{\min}, K_{\min}+1, \cdots, K_{\max}-1, K_{\max}\}$. $K_{\min}$ and $K_{\max}$ are parameters defined by the user.

We studied three distinct ways to compute the weights of the objects:

1. Emphasizing objects with **low degree of confidence:** The idea is to focus on the objects which have weak similarities with remaining objects of their group, according to the co-association matrix. As an example, if there are two touching clusters, the weak objects should be the ones that are positioned near the region the clusters touch. Concentrating the object-weighted clustering algorithm on this region should help the definition of the clusters borders. Equation 4 expresses this idea and this version of AdaEAC will be referred as *AdaEAC L*

$$w_i = \frac{\left[\max_{m=1,\cdots,n} \mathrm{conf}(\mathbf{x}_m)\right] - \mathrm{conf}(\mathbf{x}_i)}{\sum_{j=1}^{n} \left[\max_{m=1,\cdots,n} \mathrm{conf}(\mathbf{x}_m)\right] - \mathrm{conf}(\mathbf{x}_j)}. \quad (4)$$

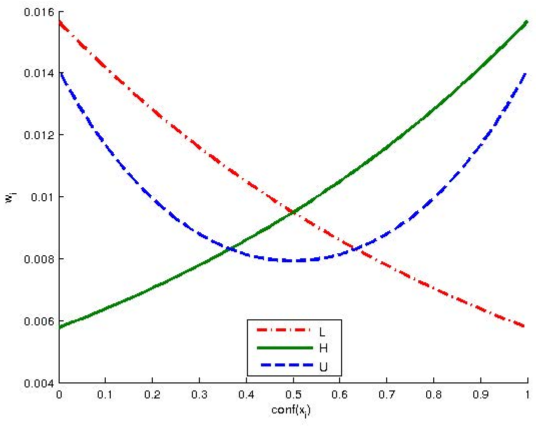


Fig. 2. Object weights w_i against degree of confidence $\mathrm{conf}(\mathbf{x}_i)$

2. Emphasizing objects with **high degree of confidence:** Focusing the objects with high degree of confidence, the ones more similar to the other objects of the same cluster, should reduce the problem of noisy points. This is expected because these noisy points will have low impact on the decisions taken by the object-weighted clustering algorithm. This idea if reflected in equation 5 and originates the version *AdaEAC H*

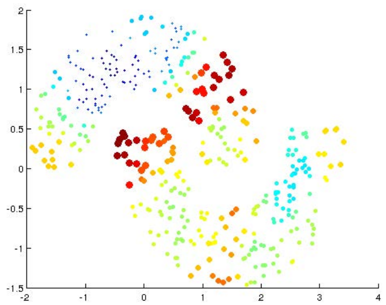
$$w_i = \frac{\mathrm{conf}(\mathbf{x}_i)}{\sum_{j=1}^{n} \mathrm{conf}(\mathbf{x}_j)}. \tag{5}$$

3. Emphasizing objects with **low and high degree of confidence:** With the combination of both previous ideas we expect the object-weighted clustering algorithm to focus both the clustering borders and well defined regions of the clusters. In order to compute the objects weights, the degree of confidences are first stretched to the $[0; 1]$ interval (Eq. 6) and then the *AdaEAC U* is derived from equation 7:

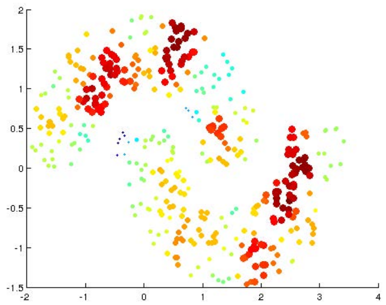
$$q_i = \frac{\mathrm{conf}(\mathbf{x}_i) - \min_{m=1,\cdots,n} \mathrm{conf}(\mathbf{x}_m)}{\max_{m=1,\cdots,n} \mathrm{conf}(\mathbf{x}_m) - \min_{m=1,\cdots,n} \mathrm{conf}(\mathbf{x}_m)}, \tag{6}$$

$$w_i = \frac{[1 - q_i(1-q_i)]^2}{\sum_{j=1}^{n} [1 - q_j(1-q_j)]^2}. \tag{7}$$

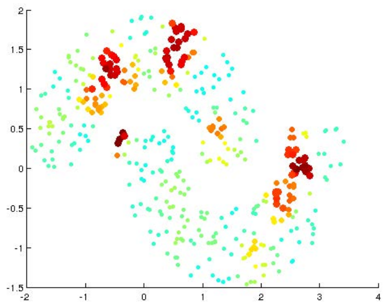
Figure 1 presents the behavior of object weights w_i against the confidence of the assignments $\mathrm{conf}(\mathbf{x}_i)$ and Figure 3 illustrates the corresponding weights in an artificial data set. Big (and red) points correspond to objects with high weights and small (and blue) points to objects with low weights.



(a) AdaEAC L



(b) AdaEAC H



(c) AdaEAC U

Fig. 3. Example: weights on each object for an artificial data set, according to Equations 4 to 7

Algorithm 1. Adaptive cluster ensembles using an object-weighted clustering algorithm

```
Input: Data set matrix $\mathbf{X}$; Number of folds $F$; Number of clusterings for each fold $L$; Minimum
    and maximum number of clusters $K_{\min}$, $K_{\max}$; and the natural number of clusters $K^*$.
 1. $\mathbf{C} \leftarrow \mathbf{0}_{n,n}$ // Initialize co-association matrix
 2. $\mathbf{W}^1 \leftarrow \left[w_1^1, \cdots, w_n^1\right]^T$ ,[2] $w_i^1 = \frac{1}{n}$ // Initialize object selection probabilities
 3. $c \leftarrow 0$
 4. for $f \leftarrow 1$ to $F$ do
 5.   for $l \leftarrow 1$ to $L$ do
 6.     $c \leftarrow c + 1$
        //Produce data partition using the distribution $\mathbf{W}^c$
 7.     $K \leftarrow \mathrm{RandomInteger}(K_{\min}, K_{\max})$;
 8.     $P^c \leftarrow \mathrm{ObjectWeightedClusterer}(\mathbf{X}, \mathbf{W}^c, K)$
        //Update co-association matrix
 9.     for all $C_k^c \in P^c$ do
10.       for all $(\mathbf{x}_i, \mathbf{x}_j) \in C_k^c$ do
11.         $\mathbf{C}_{ij} \leftarrow \mathbf{C}_{ij} + 1$
12.       end for
13.     end for
        //Produce consensus partition
14.     $P^* \leftarrow \mathrm{ConsensusClusterer}(\mathbf{C}, K^*)$
        //Update object confidence
15.     for all $(\mathbf{x}_i) \in \mathcal{X}$ do
16.       Compute $\mathrm{conf}(\mathbf{x}_i)$ as in equation 3.
17.     end for
        //Update object weights
18.     for all $(\mathbf{x}_i) \in X$ do
19.       Update $w_i^{c+1}$ using equations 4-7.
20.     end for
21.     $\mathbf{W}^{c+1} \leftarrow \left[w_1^{c+1}, \cdots, w_n^{c+1}\right]^T$
22.   end for
23. end for
24. return $P^*$
```

3.2 Object-Weighted k-Means

In this subsection, an object-weighted clustering algorithm is proposed. To incorporate distinct weights for different objects, a modification to the well-know k-means clustering algorithm [13] is presented. Given the desired number of clusters K, k-means algorithm proceeds by alternating between the assignment and update steps. During the assignment step, each object $\mathbf{x}_i$ is grouped in the cluster C_k with the closest center $\overline{\mathbf{x}}_k$. In the update step, the center of each cluster $\overline{\mathbf{x}}_l . \forall l \in \{1, \cdots, K\}$ is computed as the mean of the objects belonging C_l.

The proposed modification consists on modifying the update set in order to shift the center of the groups towards the objects with more weight. Thus, the centers of the clusters will be moved to more important regions, according to the object weights $\mathbf{W} = [w_1, \cdots, w_n]^T$. Algorithm 2 describes the proposed object-weighted clustering algorithm.

Algorithm 2. Object-weighted k-means

```
Input: Data set matrix $\mathbf{X}$; Object weights $\mathbf{W} = [w_1, \cdots, w_n]^T$; and the number of clusters $K$.
 1. Randomly initialize clusters centroids $\overline{\mathbf{x}}_k, \forall k \in \{1, \cdots, K\}$.
 2. repeat
 3.     //Assign each object to the cluster of the closest centroid
 4.     for $i \leftarrow 1$ to $n$ do
 5.         $C_{k^*} = C_{k^*} \bigcup \{\mathbf{x}_i\}$, s.t., $k^* = \arg\min_k ||\mathbf{x}_i - \overline{\mathbf{x}}_k||^2$
 6.     end for
        //Compute new cluster centroids
 7.     for $k \leftarrow 1$ to $K$ do
 8.         $\overline{\mathbf{x}}_k \leftarrow \frac{1}{\sum_{j:\mathbf{x}_j \in C_k} w_j} \sum_{\mathbf{x}_i \in C_k} w_i \mathbf{x}_i$
 9.     end for
10. until Objects do not change cluster assignments
11. return $P = \{C_1, \cdots, C_K\}$
```

4 Consensus Partition Validation

After the consensus partition is generated, it may be useful to assess its quality, especially if one wants to choose the best partition among several consensus partitions. Given the definition of the degree of confidence of assigning an object to a cluster (subsection 3.1), a straightforward way to validate a consensus partition is the Average Confidence of assignment of the objects to its clusters:

$$AC(P^*) = \frac{1}{n} \sum_{i=1}^{n} \text{conf}(\mathbf{x}_\text{i}) \tag{8}$$

$$= \frac{1}{n} \sum_{i=1}^{n} \left(\frac{\sum\limits_{j:\mathbf{x}_j \in \{C_{P_i}\} \backslash \mathbf{x}_i} C_{ij}}{|C_{P_i}| - 1} - \max_{1 \le k \le K, k \ne P_i} \frac{\sum\limits_{j:\mathbf{x}_j \in C_k} C_{ij}}{|C_k|} \right). \tag{9}$$

The value of $AC(P^*)$ is defined in the interval $[-1, 1]$. In the best-case scenario, where the co-associations of all objects with the objects belonging to the same cluster is 1 and the co-associations with objects belonging to the other clusters is 0, $AC(P^*)$ takes value 1. In the worst case scenario, where the co-associations between objects on the same cluster are 0 and belonging to different clusters are 1, $AC(P^*)$ takes value -1.

Figure 4 shows an example of a co-association matrix obtained using the split-and-merge strategy for the Iris data set. The objects are sorted by cluster: objects 1 to 50 belong to the first cluster, objects 51 to 100 to the second cluster, and the remaining objects to the third cluster. The similarities between objects (frequencies of co-associations) are represented in a gray scale. The co-association entries of highly similar objects ($C_{ij} = 1$) are shown in black, while the entries of very dissimilar objects ($C_{ij} = 0$) are shown in white. In a perfect-case scenario, the co-associations between objects in the

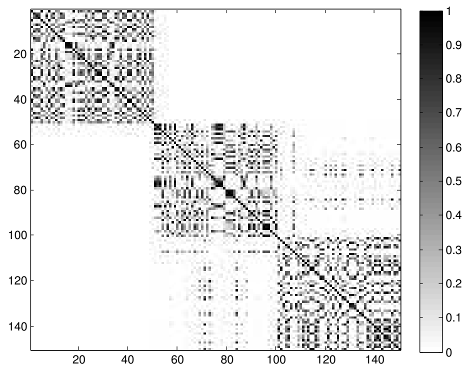


Fig. 4. Example of a co-association matrix for the Iris data set

same cluster should be 1 and the co-associations between objects belonging to different clusters should be 0, resulting in a figure with three 50×50 black squares. It can be seen that it did not occur for the given example (figure 4). One reason is due to some objects belonging to the second and third *natural* clusters being erroneously clustered together. Another reason is related to the use of the split-and-merge strategy: the number of clusters for each partition in the clustering ensemble is higher than the *natural* number of clusters, therefore some objects belonging to the same *natural* cluster have never been placed in the same cluster while building clustering ensemble. In these situations, where the intra-cluster co-associations are sparse, it may be helpful to assess the confidence of the assignments only on the neighborhood of each object. To do so, only the m^{th} nearest neighbors of each cluster should be considered while computing the average confidence. Let $V(\mathbf{x}_i, C_k, m)$ be the set of the m^{th} most similar objects of the cluster C_k to $\mathbf{x}_i$, according to the co-association matrix $\mathbf{C}$. The Average Neighborhood Confidence (ANC) of assigning the objects to its clusters is computed as

$$ANC(P^*, m) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{\sum\limits_{j:\mathbf{x}_j \in V(\mathbf{x}_i, C_{P_i}, m)} C_{ij}}{|V(\mathbf{x}_i, C_{P_i}, m)|} - \max_{1 \leq k \leq K, k \neq P_i} \frac{\sum\limits_{j:\mathbf{x}_j \in V(\mathbf{x}_i, C_k, m)} C_{ij}}{|V(\mathbf{x}_i, C_{P_k}, m)|}\right). \quad (10)$$

Figure 5 shows the sum of the co-associations related to each individual object $\mathbf{x}_i$ for the matrix shown in figure 4, i.e. $\sum_j C_{ij}$. Each of these values is related to the average number of objects that were placed in the same cluster of each object. We observed the values are not constant for all the objects. In our example, the values vary from 6 to 17. This may be easily explained: the central objects of each cluster should be co-clustered with more objects than the peripheral objects. Another factor that may contribute for such variations are data sets with unbalanced size of clusters. Considering this fact, we propose an alternative version of ANC, where the neighborhood of each object $V_i(\mathbf{x}_i, C_{P_i}, m_i)$ has a dynamic size m_i. This alternative will be referred as

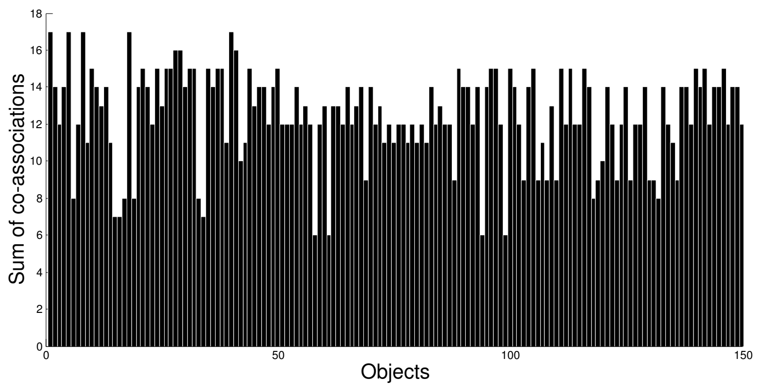


Fig. 5. Sum of the co-associations for each object of the Iris data set

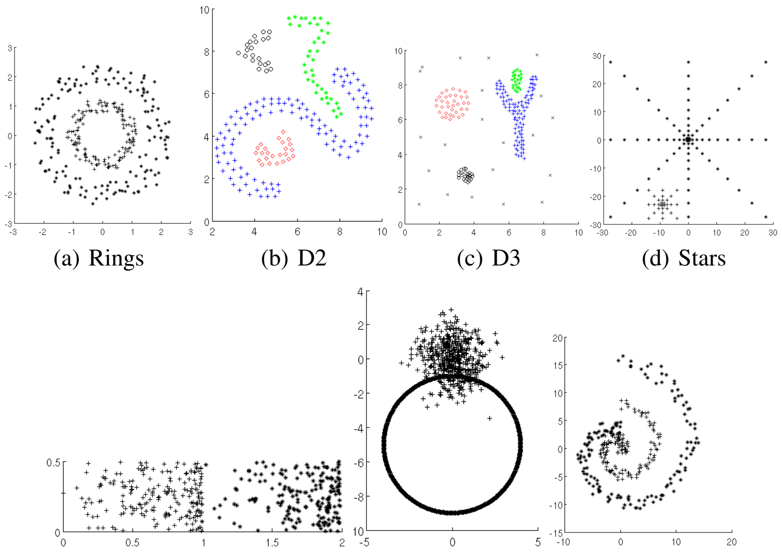
Average Dynamic Neighborhood Confidence (ADNC). The size of the neighborhood for each object m_i should be proportional to the sum of its co-associations. In this paper, we compute m_i as:

$$m_i = \left\lceil \alpha \sum_{j \in \{1,\cdots,n\} \setminus i} C_{ij} \right\rceil , \tag{11}$$

where $\alpha > 0$ is a parameter specified by the user.

5 Experimental Setup and Results

7 synthetic and 7 real data sets were used to assess the performance of the proposed approach on a wide variety of situations, such as data sets with different cardinality and dimensionality, arbitrary shaped clusters, well separated and touching clusters and distinct cluster densities. Table 1 presents the summary (number of objects n, number of dimensions d and the number of objects for each cluster) of all data sets used in our experiments and Figure 6 illustrates the 2-dimensional synthetic data sets used in our experiments. A brief description for each real data set is given next. The Iris data set consists of 50 objects from each of three species of Iris flowers (setosa, virginica and versicolor) characterized by four features. One of the clusters is well separated from the other two overlapping clusters. The Breast Cancer data set is composed of 683 objects characterized by nine features and divided into two clusters: benign and malignant. The Yeast Cell data set consists of 384 objects described by 17 attributes, split into five clusters concerning five phases of the cell cycle. There are two versions of this dataset, the first one is called Log Yeast and uses the logarithm of the expression level and the other is called Std Yeast and is a “standardized” version of the same data set, with mean 0 and variance 1. The Optdigits is a subset of Handwritten Digits data set containing only the first 100 objects of each digit, from a total of 3823 objects characterized by 64 attributes. The House Votes data set is composed of two clusters of votes for each of


(a) Rings (b) D2 (c) D3 (d) Stars

(e) Bars (f) Image2 (g) Spiral

Fig. 6. Synthetic data sets

the U.S. House of Representatives Congressmen on the 16 key votes identified by the The Wine data set consists of the results of a chemical analysis of wines grown in the same region in Italy divided into three clusters with 59, 71 and 48 objects described by 13 features.

To build the clustering ensembles we used the object-weighted k-means, proposed in Section 2, for both EAC and AdaEAC approaches. For EAC the object weights were set to $\frac{1}{n}$, making it equivalent to the standard k-means and the number of data partitions of the clustering ensemble was defined as $N = 200$. For AdaEAC, the number of folds was defined as $F = 10$ and the number of clusterings for each fold as $L = 20$ such that the number of partitions in both approaches were the same. The minimum and maximum number of clusters were defined as $K_{\text{min}} = \left\lfloor \min\left[\frac{2n}{20}, \max\left(\frac{2n}{50}, \sqrt{n}\right)\right]\right\rfloor$ and $K_{\text{max}} = \left\lceil \min\left[K_{\text{min}} + \max\left(\frac{2n}{50}, 2\sqrt{n}\right), \frac{n}{5}\right]\right\rceil$, respectively. Figure 7 shows the minimum and maximum number of clusters for $n = 1$ to 1000.

To extract the consensus partition from the co-association matrix the Average-link [17] and the Single-link [16] algorithms were applied and the number of clusters K^* was defined as the *real* number of clusters K^0 for each data set. Each clustering combination method was applied 30 times for each data set.

The Consistency index (Ci) [9] was used to assess the quality of the consensus partitions P^*. Ci measures the fraction of shared objects in matching clusters of the consensus partition (P^*) and the *natural* data partition (P^0) obtained from known labeling of data. The Consistency index is defined as $Ci(P^*, P^0) = \frac{1}{n}\sum_{k=1}^{\min(K^*,K^0)} |C_k^* \cap C_k^0|$, where it is assumed that consensus cluster C_k^* matches with the real cluster C_k^0.

Table 1. Data sets overview

Data sets	n	d	K	**Cluster Distribution**
Bars	400	2	2	$200 + 200$
D2	200	2	4	$116 + 39 + 21 + 24$
D3	200	2	5	$98 + 23 + 23 + 35 + 21$
Stars	114	2	2	$33 + 81$
Rings	300	3	2	2×150
Image2	1000	2	2	2×500
Spiral	300	2	2	2×150
Wine	178	13	3	$59 + 71 + 48$
Yeast	384	17	5	$67 + 135 + 75 + 52 + 55$
Optdigits	1000	64	10	10×100
Iris	150	4	3	3×50
House Votes	232	16	2	$124 + 108$
Breast Cancer	683	9	2	$444 + 239$

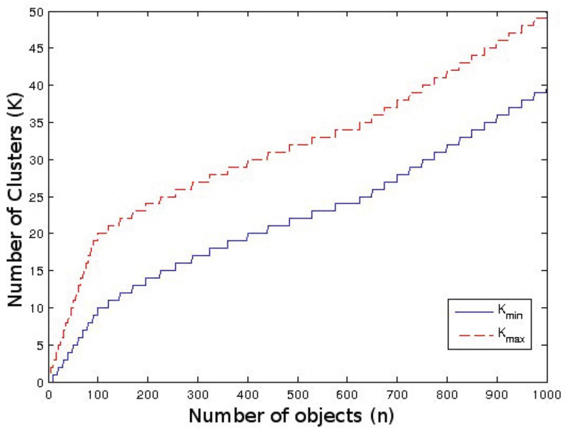


Fig. 7. Minimum and maximum number of clusters

Table 2 presents the average $Ci(P^*, P^0) \times 100$ values for the clustering ensemble methods using both Average-link (AL) and Single-link (SL) algorithms for extracting the consensus data partition (columns 2-9). Lines 3 to 9 show the results for the artificial data sets while lines 10 to 16 show the results for the real data sets. The clustering combination methods that achieved the best results in each data set are highlighted in bold. For the Bars data set, the best result was produced by AdaEAC L using both the average-link and single-link algorithms with 99.5%. For the D2 data set, the AdaEAC L and U using the single-link algorithm achieved the best results with 100%. The EAC outperform the adaptive approaches only in D3 data set using the single-link algorithm and in Std Yeast and Log Yeast data sets using the average-link algorithm. The best results for Stars and Wine data sets were obtained by AdaEAC U in combination with the

Table 2. Average $Ci(P^*, P^0) \times 100$ values for the clustering combination methods

Comb. Method	**EAC**		**AdaEAC L**		**AdaEAC H**		**AdaEAC U**	
Extraction Alg.	**AL**	**SL**	**AL**	**SL**	**AL**	**SL**	**AL**	**SL**
Bars	99.43	92.04	**99.5**	**99.5**	99.08	82.58	99.19	88.73
D2	73.55	98.3	57.28	**100**	73.28	99.15	74.03	**100**
D3	71.62	**90.55**	64.72	77.43	73.67	80.92	72.85	77.67
Stars	92.75	67.6	93.13	68.19	92.51	67.54	**93.57**	67.54
Rings	99.67	91.89	**99.8**	79.6	99.67	97.12	99.67	95.02
Image2	91.06	52.17	**91.4**	50.13	90.15	51.06	89.44	50.84
Spiral	80.2	**85**	77.26	83.82	80.79	**85**	81.7	**85**
Wine	72.21	72.19	72.23	71.05	72.17	62.4	**72.36**	64.72
Std Yeast	**68.35**	47.46	67.14	36.55	68.3	36.54	68.04	36.1
Optdigits	85.27	61.13	**88.03**	30.76	83.74	35.24	83.81	32.61
Log Yeast	**42.01**	36.52	38.99	36.4	41.23	36.73	41.4	36.76
Iris	89.93	74.67	**95.33**	85.07	90.16	74.76	90.18	75.73
House Votes	89.25	69.08	88.32	53.05	**89.71**	53.05	89.25	53.02
Breast Cancer	96.97	63.01	**97.06**	62.82	96.96	64.52	97.05	63.84

average-link algorithm with 93.57% and 72.36%, respectively. The AdaEAC L using the average-link algorithm, remarkably, achieved the best results for Rings, Image2, Optdigits, Iris, Breast Cancer and Bars data sets. We highlight the 95.33% result obtained in the Iris data set, which was superior to all the other methods by a margin higher than 5%. For the Spiral data sets the best result was 85% and was obtained by EAC, AdaEAC H and U using single-link algorithm. In summary, the AdaEAC L approach achieved the best result in 4 out of 7 synthetic data sets while the AdaEAC U approach obtained the best result in 3 out of 7 synthetic data sets, the EAC method in 2 out of 7, and the AdaEAC H in 1 out of 7 data sets. For the real data sets, the AdaEAC L approach obtained again the best result in 4 out of 7 data sets, the EAC in 2 and both AdaEAC H and U only in 1 data set. These results suggest that the AdaEAC L approach is a good option for combining multiple data partitions.

Table 3 shows the average $Ci(P^*, P^0) \times 100$ values of all the consensus partitions produced for a given data set (column 2) and the average $Ci(P^*, P^0) \times 100$ of the partitions resulting of picking the best partition among the EAC and AdaEAC approaches (for a single run) according to the consensus clustering validity measures $AC(P^*)$ (column 3), $ANC(P^*, m)$ (columns 4 to 7), and $ADNC(P^*, m_i)$ (columns 8 to 10). Line 2 indicates the parameters used by the consensus measures. Four different values for the number of neighbors, $m = \{5, 10, 20, 40\}$, for the ANC measure were tested. For the ADNC consensus validity measure, we defined the value α as 0.5, 1 and 2. When the average quality of the partitions selected by the consensus validity measures outperform the average consensus results, the corresponding results are highlighted in bold. The highest average result for each data set is also underlined.

Table 3. Average $Ci(P^*, P^0) \times 100$ values of all consensus partitions and average $Ci(P^*, P^0) \times 100$ values for the consensus partitions selected by the consensus validity measures

Val. Measure	**Consensus**	**AC**	**ANC**				**ADNC**		
Parameters	**Average**		$\mathbf{m = 5}$	$\mathbf{m = 10}$	$\mathbf{m = 20}$	$\mathbf{m = 40}$	$\alpha = \mathbf{0.5}$	$\alpha = \mathbf{1}$	$\alpha = \mathbf{2}$
Bars	95.01	**99.5**	**99.20**	**99.10**	**99.15**	**98.1**	**99.38**	**99.18**	82.88
D2	84.45	73.78	73.90	73.82	73.07	72.53	74.18	74.92	74.12
D3	76.18	67.32	**90.13**	**87.47**	74.02	**77.98**	**89.78**	**85.52**	**80.98**
Stars	80.35	**93.42**	67.54	67.54	67.54	67.54	67.54	77.89	71.93
Rings	95.31	**98.24**	**99.67**	**99.67**	**99.67**	**99.8**	**99.67**	**99.67**	**99.66**
Image2	70.78	**91.4**	51.48	54.03	64.71	64.16	58.17	65.32	64.78
Spiral	82.35	**82.56**	77.88	77.62	79.78	82.07	78.48	78.44	79.69
Wine	69.92	**72.3**	**72.19**	63.58	61.27	61.31	**72.19**	67.04	62.45
Std Yeast	53.56	**67.9**	**66.97**	**68.47**	**68.49**	**68.19**	**68.2**	**68.25**	**68.22**
Optdigits	62.57	**87.57**	**83.24**	**83.49**	**83.36**	**83.64**	**83.35**	**83.54**	**83.12**
Log Yeast	38.76	**42.55**	**40.10**	**41.55**	**41.55**	**41.9**	**41.55**	**41.55**	**42.11**
Iris	84.48	**93.82**	76.36	77.38	74.49	74.67	78.07	80.11	74.67
House Votes	73.09	**88.32**	**74.02**	**87.47**	**88.32**	**88.32**	**86.08**	**88.41**	**88.51**
Breast Cancer	80.28	**95.89**	77.60	78.64	78.47	**97.06**	**87.35**	**90.48**	**91.53**

By picking the best consensus partition using the Average Confidence measure we obtained better results than the consensus average in 5 of the 7 artificial data sets and in all of the real data sets. We also notice that the average quality of the partitions selected by this index is close to the quality of the best clustering combination scheme in each data set. Overall, the AC measure selected better partitions that the other measures in 8 out of the 14 data sets. With respect to the Average Neighborhood Confidence measure, we verify that the measure is sensible to the neighborhood size m. It is not clear which value for m should be used for each data set. The ANC measure performed better using $m = 5$ neighbors in Bars, D2, D3, Wine and Iris data sets, while in the Rings, Spiral, Optdigits and Breast Cancer data sets ANC obtained better results using $m = 40$. In both cases, ANC achieved better results than the consensus average in 8 out of the 14 data sets. Regarding the Average Dynamic Neighborhood Confidence, it achieved better results than the consensus average in 9 out of the 14 data sets using $\alpha = 0.5$ and in 8 data sets using $\alpha = 1$. By comparing ADNC using $\alpha = 0.5$ and $\alpha = 1$ with the ANC results, we observe that the $Ci(P^*, P^0) \times 100$ values are usually similar to the best results achieved by ANC. These results point out that the Average Confidence index is a very good choice for performing consensus clustering selection. Although, in same situations, computing the confidence of the assignments in the neighborhood of each object is better. In this case, the dynamic definition of the neighborhood's size should be preferred over the static one.

6 Conclusions and Future Work

A clustering combination method, based in the Evidence Accumulation Clustering and the supervised learning boosting methods was proposed. Our approach is based on estimating the degree of confidence of the assignment of objects to clusters and then influence the process of constructing the clustering ensemble using an object-weighted clustering algorithm. We tested three distinct ways of computing the object weights: focusing on the objects hard to cluster, on the objects easy to cluster, and on a mix of the previous. Three consensus clustering validity measures based on the confidence of the objects' assignments were also proposed to selected the best consensus partition. Experimental results suggest that using the Adaptive Evidence Accumulation Clustering method, focusing the construction of the clustering ensemble on the objects that are harder to cluster, is a good choice to perform data clustering. It was also shown that the proposed consensus clustering measures can successfully be used to perform consensus clustering selection, in particular the Average Consistency index.

In the future, we pretend to study the influence of the size of the neighborhood for computing the Average Neighborhood Consistency and Average Dynamic Neighborhood Consistency measures.

Acknowledgements. This work is supported by FEDER Funds through the "Programa Operacional Factores de Competitividade - COMPETE" program and by National Funds through FCT under the projects FCOMP-01-0124-FEDER-PEst-OE/EEI/UI0760/2011 and PTDC/EIA - CCO/103230/2008, and grant SFRH/BD/43785/2008.

References

1. Al-Razgan, M., Domeniconi, C., Barbar, D.: Random Subspace Ensembles for Clustering Categorical Data. In: Okun, O., Valentini, G. (eds.) Supervised and Unsupervised Ensemble Methods and their Applications. SCI, vol. 126, pp. 31–48. Springer, Heidelberg (2008)
2. Domeniconi, C., Al-Razgan, M.: Weighted cluster ensembles: Methods and analysis. ACM Trans. Knowl. Discov. Data 2, 17:1–17:40 (2009)
3. Duarte, F.J., Fred, A.L.N., Rodrigues, M.F.C., Duarte, J.: Weighted evidence accumulation clustering using subsampling. In: Sixth International Workshop on Pattern Recognition in Information Systems (2006)
4. Dudoit, S., Fridlyand, J.: Bagging to Improve the Accuracy of a Clustering Procedure. Bioinformatics 19(9), 1090–1099 (2003)
5. Fern, X.Z., Brodley, C.E.: Random projection for high dimensional data clustering: A cluster ensemble approach, pp. 186–193 (2003)
6. Fern, X.Z., Brodley, C.E.: Solving cluster ensemble problems by bipartite graph partitioning. In: Proceedings of the Twenty-First International Conference on Machine Learning, ICML 2004, pp. 36–43. ACM, New York (2004)
7. Fred, A., Jain, A.: Combining multiple clustering using evidence accumulation. IEEE Trans. Pattern Analysis and Machine Intelligence 27(6), 835–850 (2005)
8. Fred, A., Jain, A.K.: Evidence Accumulation Clustering Based on the K-Means Algorithm. In: Caelli, T.M., Amin, A., Duin, R.P.W., Kamel, M.S., de Ridder, D. (eds.) SPR 2002 and SSPR 2002. LNCS, vol. 2396, pp. 442–451. Springer, Heidelberg (2002)
9. Fred, A.: Finding Consistent Clusters in Data Partitions. In: Kittler, J., Roli, F. (eds.) MCS 2001. LNCS, vol. 2096, pp. 309–318. Springer, Heidelberg (2001)
10. Freund, Y., Schapire, R.E.: A Decision-theoretic Generalization of Online Learning and An Application to Boosting. In: Vitányi, P.M.B. (ed.) EuroCOLT 1995. LNCS, vol. 904, pp. 23–37. Springer, Heidelberg (1995)

11. Hadjitodorov, S.T., Kuncheva, L.I., Todorova, L.P.: Moderate diversity for better cluster ensembles. Inf. Fusion 7(3), 264–275 (2006)
12. Jouve, P., Nicoloyannis, N.: A new method for combining partitions, applications for distributed clustering. In: International Workshop on Paralell and Distributed Machine Learning and Data Mining (ECML/PKDD 2003), pp. 35–46 (2003)
13. MacQueen, J.B.: Some methods for classification and analysis of multivariate observations. In: Cam, L.M.L., Neyman, J. (eds.) Proc. of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297. University of California Press (1967)
14. Minaei-Bidgoli, B., Topchy, A., Punch, W.F.: Ensembles of partitions via data resampling. In: Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC 2004), vol. 2, pp. 188–192. IEEE Computer Society, Washington, DC (2004)
15. Saffari, A., Bischof, H.: Boosting for Model-Based Data Clustering. In: Rigoll, G. (ed.) DAGM 2008. LNCS, vol. 5096, pp. 51–60. Springer, Heidelberg (2008)
16. Sneath, P.H., Sokal, R.: Numerical Taxonomy: The Principles and Practice of Numerical Classification (1973)
17. Sokal, R.R., Michener, C.D.: A statistical method for evaluating systematic relationships. University of Kansas Scientific Bulletin 28, 1409–1438 (1958)
18. Strehl, A., Ghosh, J.: Cluster ensembles — a knowledge reuse framework for combining multiple partitions. J. Mach. Learn. Res. 3, 583–617 (2003)
19. Topchy, A., Jain, A.K., Punch, W.: Combining multiple weak clusterings, pp. 331–338 (2003)
20. Topchy, A., Minaei-Bidgoli, B., Jain, A.K., Punch, W.F.: Adaptive clustering ensembles. In: ICPR 2004: Proceedings of the Pattern Recognition, 17th International Conference on (ICPR 2004), vol. 1, pp. 272–275. IEEE Computer Society, Washington, DC (2004)
21. Vega-Pons, S., Ruiz-Shulcloper, J.: A survey of clustering ensemble algorithms. International Journal of Pattern Recognition and Artificial Intelligence 25(03), 337–372 (2011)
22. Dimitriadou, E., Weingessel, A., Hornik, K.: Voting-Merging: An Ensemble Method for Clustering. In: Dorffner, G., Bischof, H., Hornik, K. (eds.) ICANN 2001. LNCS, vol. 2130, pp. 217–224. Springer, Heidelberg (2001)
23. Zhai, S.L., Luo, B., Guo, Y.T.: Fuzzy clustering ensemble based on dual boosting. In: Proceedings of the Fourth International Conference on Fuzzy Systems and Knowledge Discovery, FSKD 2007, vol. 02, pp. 240–244. IEEE Computer Society, Washington, DC (2007)