Study of consensus functions in the context of ensemble methods for biclustering

Blaise Hanczar*[1] et Mohamed Nadif[1]

[1]LIPADE, Université Paris Descartes

2 avril 2013

Résumé

The ensemble methods are very popular and can improve significantly the performance of classification and clustering algorithms. Their principle is to generate a set of different models, then aggregated them into only one. Recent works have shown that this approach could also be useful in the biclustering problems.The crucial step of this approach is the consensus functions that compute the aggregation of the biclusters. We identify the main consensus functions commonly used in the clustering ensemble and show how to extend them in the biclustering context. We evaluate and analyze the performances of these consensus functions on several experiments based on both artificial and real data.

Keywords : Biclustering, Ensemble methods, Consensus functions.

1 Introduction

Biclustering, also called direct clustering [Har72], co-clustering [DMM03], simultaneous clustering in [Gov95, TBK05] or block clustering in [GN03] is now a widely used method of data mining in various domains in particular in text mining and bioinformatics. For instance, in document clustering, in [Dhi01] the author has proposed a spectral block clustering method which makes use of the clear duality between rows (documents) and columns (words). In the analysis of microarray data, where data are often presented as matrices of expression levels of genes under different conditions, the co- clustering of genes and conditions has overcamed the problem encountered in conventional clustering methods concerning the choice of similarity. Cheng and Church [CC00] were the first to propose a biclustering algorithm for microarray data analysis. They considered that biclusters follow an additive model and used a greedy iterative search to minimize the mean square residue (MSR). Their algorithm identifies the biclusters one by one and applied to yeast cell cycle data, it has enabled to identify several biologically relevant biclusters. Lazzeroni and Owen [LO00] have proposed the popular plaid model which has been improved by Turner et al. [TBK05]. The authors assumed that biclusters are organized in layers and follow a given statistical model incorporating additive two way ANOVA models. The search approach is iterative : once $(K-1)$ layers (biclusters) have been identified, the K-th bicluster minimizing a merit function depending on all layers is selected. Applied to data from the yeast, the proposed algorithm reveals that genes in biclusters share the same biological functions. In [ES10] the authors have developed its localization procedure which improves the performance of a greedy iterative biclustering algorithm. Several other methods have been proposed in the literature, two complete surveys of biclustering methods can be found in [MO04, BPP08].

Here we propose to use the ensemble methods to improve the performance of the biclustering. It is important to note that we do not propose a new biclustering method in competition with the previously cited algorithms. We seek to adapt the ensemble approach to the biclustering problem in order to improve the performance of any biclustering algorithm. The principle of the ensemble biclustering is to generate a set of different biclustering solutions, then aggregated them into only one solution. The crucial step is based on the consensus functions computing the aggregation of the different solutions. In this paper we have identified four types of consensus function commonly used in the ensemble clustering and giving the best results. We show how extend their use in the biclustering context. We evaluate their performances on a set of both numerical and real data experiments.

The paper is organized as follows. In Section 2, we review the ensemble methods in clustering and biclustering.

*blaise.hanczar@parisdescartes.fr

In section 3, we formalize the collection of biclustering solution and show how to construct it from the Cheng & Church algorithm that we have retained for our study. In section 3, we extend four commonly used consensus functions to biclustering context. Section 5 is devoted to evaluate these new consensus functions on several experimentations. Finally, we summarize the main points arising from approach.

2 Ensemble Methods

The principle of ensemble methods is to construct a set of models, then to aggregate them into a single model. It is well-known that these methods often perform better than a single model [Die00]. Ensemble methods first appeared in supervised learning problems. A combination of classifiers is more accurate than single classifiers [Mac97]. A pioneer method boosting, the most popular algorithm of which is adaboost, was developed mainly by Shapire [Sch03]. The principle is to assign a weight to each training example, then several classifiers are learned iteratively and between each learning step the weight of examples is adjusted depending on the classifier results. The final classifier is a weighted vote of classifiers constructed during the procedure. Another type of popular ensemble methods is bagging, proposed by Breiman [Bre96]. The principle is to create a set a classifiers based on bootstrap samples of the original data. The random forests [Bre01] are the most famous application of bagging. They are a combination of tree predictors, and have given very good results in many domains [DUAdA06].

Several works have shown that ensemble methods can also be used in unsupervised learning. Topchy et al. [TLJF04] have theoretically shown that ensemble methods may improve the performance clustering. The principle of boosting has been exploited by Frossyniotis et al. [FLS04] in order to provide a consistent partitioning of the data. The boost-clustering approach creates, at each iteration, a new training set using weighted random sampling from original data, and a simple clustering algorithm is applied to provide new clusters. Dudoit and Fridlyand [DF03] have used bagging to improve the accuracy of clustering in reducing the variability of the PAM (Partitioning Around Medoids) results [vdLPB03]. Their method has been applied to leukemia and melanoma datasets and has allowed to differentiate the different subtypes of tissues. Strehl et al. [SG02] have proposed an approach to combine multiple partitioning obtained from different sources into a single one. They introduced an heuristics based on a voting consensus. Each example is assigned to one cluster for each partition, an example has therefore many assignments as number of partitions in the collection. In the aggregated partition, the example is assigned to the cluster with whom it was the most often assigned. One problem of this consensus is that requires knowledge of the cluster correspondence between the different partitions. They have also proposed a cluster-based similarity partitioning algorithm. The collection is used to compute a similarity matrix of the examples. The similarity between two examples is based on the frequency of their co-association to the same cluster over the collection. The aggregated partition is computed by a clustering of the examples from this similarity matrix. Fern [FB04] formalizes the aggregation procedure by a bipartite graph partitioning. The collection is represented by a bipartite graph. The examples and clusters of partitions are the two sets of vertices. An edge between an examples and a cluster means that example has been assigned to this cluster. A partition of the graph is performed and each sub-graph represents an aggregated cluster. Topchy [TJP04] proposes to modelize the consensus of the collection by a multinomial mixture model. In the collection, each example is defined by a set of labels that represents their assigned clusters in each partition. This can be viewed as a new space in which are defined the examples, each dimension being a partition of the collection. The aggregated partition is computed from a clustering of examples in this new space. Since the labels are discrete variables, a multinomial mixture model is used. Each component of the model represents an aggregated cluster.

Some recent works have shown that the ensemble approach can also be useful in the biclustering problems [HN12]. A bagging version of the biclustering algorithms has been proposed and tested for microarray data [HN10]. Although this last method improves of performance of the biclustering, in some cases it fails and returns empty biclusters, i.e. without examples or features. The reason comes from the consensus function that handles the set of examples and features on the same dimension as in the clustering context. The consensus function must respect the structure of the biclusters. For this reason, the consensus functions cited above, can be applied on biclustering problems. On this paper we adapt these consensus functions to the biclustering context.

3 Biclustering Solution Collection

The first step of ensemble biclustering is to generate a collection of biclustering solution. Here we give the formalization of the collection and a method to generate it from the Cheng and Church algorithm that we have retained for our study.

3.1 Formalization of the collection

Let's a data matrix $\mathbb{X} = \{\mathbb{E}, \mathbb{F}\}$ where $\mathbb{E} = \{e_1, ..., e_N\}$ is the set of N examples represented by M-dimensional vectors and $\mathbb{F} = \{f_1, ..., f_M\}$ is the set of M features represented by N-dimensional vectors. A bicluster B is a submatrix of X defined by a subset of examples and a subset of features : $B = \{(E_B, F_B) | E_B \subseteq \mathbb{E}, F_B \subseteq \mathbb{F}\}$. A biclustering operator Φ is a function that returns a biclustering solution (i.e. a set of biclusters) from a data matrix : $\Phi(X) = \{B_1, ..., B_K\}$ where K is the number of biclusters. Let's ϕ the function giving for each point of the data matrix the labels of the bicluster to which it belongs. The label is 0 for points belonging to no bicluster.

$$\phi(x_{ij}) = \begin{cases} k \ if \ e_i \in E_{B_k} \ and \ f_j \in F_{B_k} \\ 0 \ if \ e_i \notin E_{B_k} \ or \ f_j \notin F_{B_k} \ \forall k \in [1, K]. \end{cases}$$

A biclustering solution can be represented by a label matrix $\mathbf{I}$ giving for each point : $I_{ij} = \phi(x_{ij})$. In the following it will be convenient to represent this label matrix by an label vector indexing by u defined as $u = i * |\mathbb{F}| + (|\mathbb{F}| - j)$, where $|.|$ denotes the cardinality.

Let's the true biclustering solution of the data set $\mathbb{X}$ represented by $\Phi(\mathbb{X})^*$, $\mathbf{I}^*$ and $\mathbf{J}^*$. An estimated biclustering solution is a biclustering solution returned by an algorithm from the data matrix, it is denoted by $\hat{\Phi}(\mathbb{X})$, $\hat{\mathbf{I}}$ and $\hat{\mathbf{J}}$. The objective of the biclustering task is to find the closest estimated biclustering solution from the true biclustering solution. In ensemble methods, we do not use only one estimated biclustering solutions but we generate a collection of several solutions. We denote this collection of biclustering solutions $\mathbb{C} = \{\hat{\Phi}(X)_{(1)}, ..., \hat{\Phi}(X)_{(R)}\}$. This collection can be represented by an $NM \times R$ data matrix $\mathbb{J} = (\mathbf{J}_1^T, \ldots, \mathbf{J}_{NM}^T)^T$ in merging together all label vectors $\mathbf{J}_u = (J_{u1}, \ldots, J_{uR})^T$ where $J_{ur} = \phi(x_{ij})_{(r)}$ with $r \in [1, R]$. The objective of the consensus function is to form an aggregated biclustering solution, represented by $\overline{\Phi}(X)$, $\bar{\mathbf{I}}$ and $\bar{\mathbf{J}}$, from the collection of estimated solution. Each of these functions is illustrated through an example in Figure 1.

3.2 Construction of the collection

The key point of the generation of the collection is to find a good trade-off between the quality and diversity of the biclustering solutions of the collection. If all generated solutions are the same, the aggregated solution will be identical to the biclusters of the collection. Different sources of the diversity are possible. We can use resampling method like bootstrap or jacknife. In applying the biclustering operator on each resampled data, different solutions are produced. We can also include the source of diversity directly in the biclustering operator. In this case the algorithm is not deterministic and will produce different solutions from the same original data.

In our experiments the biclustering operator is the Cheng and Church algorithm (CC). This algorithm returns a set of biclusters minimizing the mean square residue (MSR).

$$MSR(B_k) = \frac{1}{|B_k|} \sum_{i,j} z_{ik} w_{jk} (X_{ij} - \mu_{ik} - \mu_{jk} + \mu_k)^2,$$

where μ_k is the average of B_k, μ_{ik} and μ_{jk} are respectively the means of E_i and F_j belonging to the bicluster B_k.

The CC algorithm is iterative and the biclusters are identified one by one. To detect each bicluster, the algorithm begins with all features and examples, then it drops the feature or example minimizing the mean square residue (MSR) of the remaining matrix. This procedure is totally deterministic. If the algorithm is ran several times, the same biclustering solution will be find each time. We modify the CC algorithm in including a source of diversity in the computation of the bicluster. At each iteration, we select the top α% of the features and examples minimizing MSR of the remaining matrix. The element to be dropped is randomly chosen from this selection. Thus the parameter α controls the level of diversity of the bicluster collection ; in our simulations $\alpha = 5\%$ appears a good threshold. This modified version of the algorithm is used in all our experiments in order to generate the collection of biclustering solution from a dataset.

4 Consensus Functions for Biclustering

The second step of the ensemble approach is the aggregation of the collection of biclustering solutions. We present here four consensus functions that we extend to the biclustering context. Note that these consensus function work even if the numbers of biclusters in the different solutions of the collection are not the equal.

4.1 Co-association Consensus (COAS)

This consensus is based on the bicluster assignation similarity between the points of the data matrix. The similarity between two points is defined by the proportion of times that they are associated to the same bicluster over the whole collection. All these similarities are represented by a distance matrix D defined by :

$$D_{uv} = 1 - \frac{1}{R} \sum_{r=1}^{R} \delta(J_{ur} = J_{vr}),$$

where $\delta(x)$ returns 0 when x is false and 1 when true. From this dissimilarity data matrix, $K+1$ clusters are identified in using the Partioning Around Mediods (PAM) algorithm [DF03]. Each cluster of points represents one aggregated bicluster excepted the largest one that groups all points belonging to no bicluster.

4.2 Voting Consensus (VOTE)

This consensus function is based on the majority vote of the labels. For each point of the data matrix, the consensus returns the most represented label in the collection of the biclustering solution. The main problem of this approach is that there is no correspondence between the labels of two different estimated biclustering solutions. All biclusters of the collection have to be re-labeled according to their best agreement with some chosen reference solution. Any estimated solution can be used as reference, here we use the first one $\hat{\Phi}(X)_{(1)}$. The agreement problem can be solved in polynomial time by the Hungarian method [PS82] which relabels the estimated solution such as the similarity between the solutions is maximized. The similarity between two biclustering solutions is computed in using the F-measure (details in section 4). The label of the aggregated biclustering solution for a point is therefore defined by :

$$\overline{\mathbf{J}}_u = \underset{k}{\operatorname{argmax}} \left(\sum_{r=1}^{R} \delta(\Gamma(J_{ur}) = k) \right).$$

where Γ is the relabelling operator performed by the Hungarian algorithm.

4.3 Bipartite Graph Partitionning Consensus (BGP)

In this consensus the collection of estimated solutions is represented by a bipartite graph where the vertices are divided into two sets : the point vertices and the label vertices. The point vertices represent the points of the data matrix $\{(e_i, f_j)\}$ while the set of label vertices represents all estimated biclusters of the collection $\{\hat{B}_{k,(r)}\}$, for each estimated solution there is also a vertice that represents the points belonging to no bicluster. An edge links a point vertice to an label vertice if the point belongs to the corresponding estimated bicluster. The degree of each point is therefore R and the degree of each estimated bicluster represents the number of points that it contains. Finding a consensus consists in finding a partition of this bipartite graph. The optimal partition is the one that maximizes the numbers of edges inside each cluster of nodes and minimizes the number of edges between nodes of different clusters. This graph partitioning problem is a NP-hard problem, so we rely on a heuristic to an approximation of the optimal solution. We use a method based on a spin-glass model and simulated annealing [RB06] in order to identified the clusters of nodes. Each cluster of the partition represents an aggregated bicluster formed by all points contained in this cluster.

4.4 Multivariate Mixture Model Consensus (MIX)

In [TJP04], the authors have used the mixture approach to propose a consensus function. In the sequel we propose to extend it to our situation. In model-based clustering it is assumed that the data are generated by a mixture of underlying probability distributions, where each component k of the mixture represents a cluster. Specifically, the $NM \times R$ data matrix $\mathbb{J}$ is assumed to be an $\mathbf{J}_1, \ldots, \mathbf{J}_u, \ldots, \mathbf{J}_{NM}$ i.i.d sample where $\mathbf{J}_u$ from a probability distribution with density

$$\varphi(\mathbf{J}_u|\Theta) = \sum_{k=0}^{K} \pi_k P_k(\mathbf{J}_u|\theta_k),$$

where $P_k(\mathbf{J}_u|\theta_k)$ is the density of label $\mathbf{J}_u$ from the kth component and the θ_k's are the corresponding class parameters. These densities belong to the same parametric family. The parameter π_k is the probability that an object belongs to the kth component, and K, which is assumed to be known, is the number of components in the mixture. The number of components corresponds to the number of biclusters minus one since one of the components represents the points belonging to no bicluster. The parameter of this model is the vector $\Theta = (\pi_0, \ldots, \pi_K, \theta_0, \ldots, \theta_K)$. The mixture density of the observed data $\mathbb{J}$ can be expressed as

$$\varphi(\mathbb{J}|\Theta) = \prod_{u=1}^{NM} \sum_{k=0}^{K} \pi_k P_k(\mathbf{J}_u|\theta_k).$$

The labels $\mathbf{J}_u$ are nominal categorical variables, we consider the latent class model and assume that all R categorical variables are independent, conditionnally on their memebership of a component ; $P_k(\mathbf{J}_u|\theta_k) = \prod_{r=1}^{R} P_{k,(r)}(J_{ur}|\theta_{k,(r)})$, Note that $P_{k,(r)}(\mathbf{J}_u|\theta_{k,(r)})$ represents the probability to have the vector label $\mathbf{J}_u$ in the kth component for the estimated solution $\hat{\Phi}(X)_{(r)}$. If $\alpha_k^{r(j)}$ is the probability that the rth label takes the value j when an $\mathbf{J}_u$ belongs to the component k, then the probability of the mixture can be written $P_k(\mathbf{J}_u|\theta_k) = \prod_{r=1}^{R} \prod_{j=1}^{K} [\alpha_k^{r(j)}]^{\delta(J_{ur}=j)}$. The parameter of the mixture Θ is fitted in maximizing the likelihood function :

$$\Theta^* = \underset{\Theta}{\operatorname{argmax}} \left(\log \left(\prod_{u=1}^{NM} P(\mathbf{J}_u|\theta) \right) \right).$$

The optimal solution of this maximization problem can not generally be computed, we therefore rely on an estimation given by the EM algorithm [DLR77]. In E-step, we compute the posterior probabilities of each label $s_{uk} \propto P_k(\mathbf{J}_u|\theta_k)$ and in the M-step we estimate the parameters of the mixture as follows

$$\pi_k = \frac{\sum_u s_{uk}}{NM} \text{ and } \alpha_k^{r(j)} = \frac{\sum_u s_{uk}\delta(J_{ur} = j)}{\sum_u s_{uk}}.$$

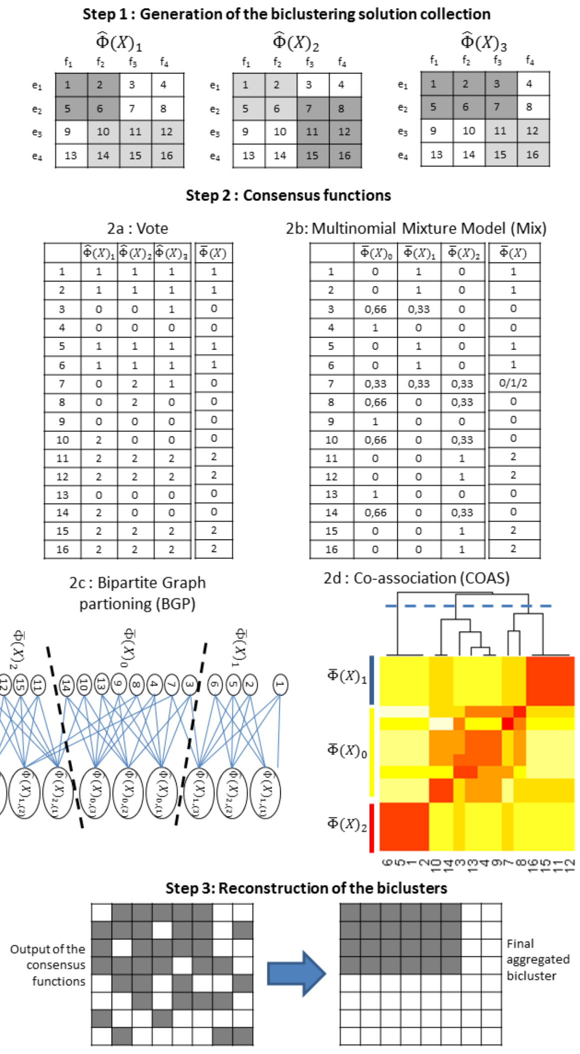
To limit the problems of local minimum during the EM algorithm, we perform the optimization process ten times with different initializations and keep the solution maximizing the log-likelihood. At the convergnce, we consider that the largest π_k corresponds to labels representing the points belonging to no biclusters. The estimators of posterior probabilities give rise to a fuzzy or hard clustering using the maximum a posteriori principle (MAP). Then the consensus function consists in taking for each $\mathbf{J}_u$ the cluster k maximizing its conditional probability $k = \text{argmax}_{\ell=1,\ldots,K}\, s_{u\ell}$, and we obtain the ensemble solution $\overline{\Phi}(\mathbb{X})$.

4.5 Reconstruction of the Biclusters

The four consensus functions presented above, return a partition in $K + 1$ clusters of the points of the data matrix. K of these clusters represent the K aggregated biclusters, the last one groups all points that belong to no biclusters in the aggregated solution. The k aggregated biclusters are not actual biclusters yet. They are just sets of points that do not necessarily form submatrices of the data matrix. A reconstruction step has to be applied on each aggregated bicluster in order to transform them in to a submatrice. This procedure consist to find the submatrix containing the maximum of points that are in the aggregated bicluster and the minimum of points that are not in the aggregated bicluster. The k-th aggregated bicluster is reconstructed in minimizing the following function :

$$\begin{aligned} L(\overline{B_k}) &= \sum_{i=1}^{N}\sum_{j=1}^{M} \delta(e_i \in \overline{E}_{B_k} \wedge f_i \in \overline{F}_{B_k})\delta(\overline{I}_{ij} \neq k) \\ &+ \delta(e_i \notin \overline{E}_{B_k} \vee f_i \notin \overline{F}_{B_k})\delta(\overline{I}_{ij} = k). \end{aligned}$$

This optimization problem is solved by an heuristic procedure. We start with all examples and features involving in the aggregated bicluster. Then iteratively, we drop the example or feature that maximizes the decrease of $L(\overline{B_k})$. This step is iterated until $L(\overline{B_k})$ does not decrease. Once the reconstruction procedure is finished, we obtain the final aggregated biclusters. We give a last comment about the number of biclusters K whose the choice is still an open problem in unsupervised learning. Here the number of biclusters K is given by the users. In almost all algorithms the number of biclusters is a parameter or there a parameter that controls indirectly this number. In consequence the number of biclusters in each estimated solution is assumed equal.



Step 1 : Generation of the biclustering solution collection
$\widehat{\Phi}(X)_1$
$\widehat{\Phi}(X)_2$
$\widehat{\Phi}(X)_3$
Step 2 : Consensus functions
2a : Vote
2b: Multinomial Mixture Model (Mix)
2c : Bipartite Graph partioning (BGP)
2d : Co-association (COAS)
Step 3: Reconstruction of the biclusters
Output of the consensus functions
Final aggregated bicluster

2a : Vote

	$\widehat{\Phi}(X)_1$	$\widehat{\Phi}(X)_2$	$\widehat{\Phi}(X)_3$	$\overline{\Phi}(X)$
1	1	1	1	1
2	1	1	1	1
3	0	0	1	0
4	0	0	0	0
5	1	1	1	1
6	1	1	1	1
7	0	2	1	0
8	0	2	0	0
9	0	0	0	0
10	2	0	0	0
11	2	2	2	2
12	2	2	2	2
13	0	0	0	0
14	2	0	0	0
15	2	2	2	2
16	2	2	2	2

2b: Multinomial Mixture Model (Mix)

	$\overline{\Phi}(X)_0$	$\overline{\Phi}(X)_1$	$\overline{\Phi}(X)_2$	$\overline{\Phi}(X)$
1	0	1	0	1
2	0	1	0	1
3	0,66	0,33	0	0
4	1	0	0	0
5	0	1	0	1
6	0	1	0	1
7	0,33	0,33	0,33	0/1/2
8	0,66	0	0,33	0
9	1	0	0	0
10	0,66	0	0,33	0
11	0	0	1	2
12	0	0	1	2
13	1	0	0	0
14	0,66	0	0,33	0
15	0	0	1	2
16	0	0	1	2

FIGURE 1 – Procedure of ensemble biclustering with the four consensus functions.

5 Results and Discussion

5.1 Performance of consensus functions

In our simulations, we consider six different data structures with $M = N = 100$ in which a true biclustering solution is included. The number of biclusters varies from

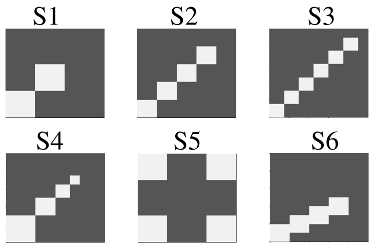


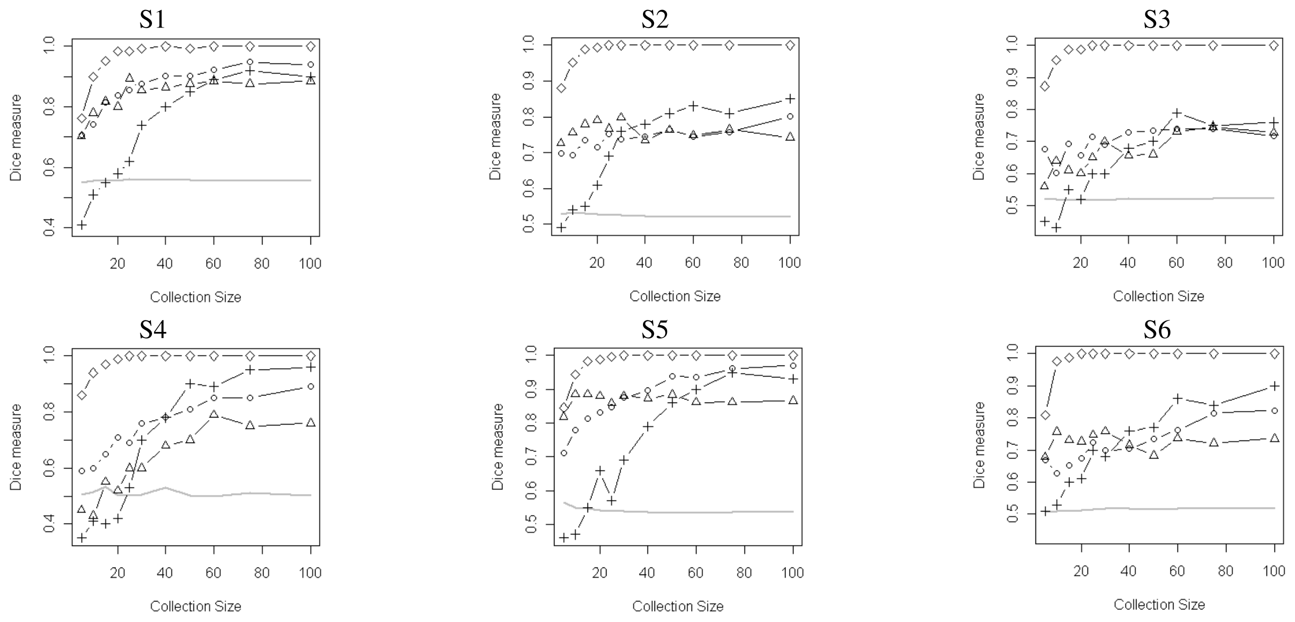
FIGURE 2 – Six data structures considered in the experiments.

2 to 6 and their sizes from 10 examples by 10 features to 30 examples by 30 features. We have defined six different structures of biclusters depicted in Figure 2. For each data, from each true bicluster an estimated bicluster is generated, then a collection of estimated biclustering solutions is obtained. The quality of the collection is controlled by the parameters α_{pre} and α_{rec} that are the average precision and recall between estimated biclusters and their corresponding true biclusters. To generate an estimated bicluster we started with the true bicluster, then we have randomly removed features/examples and have added features/examples that were not in the true bicluster in order to obtain the target precision α_{pre} and recall α_{rec}. Once the collection is generated, the four consensus functions are applied to obtain the aggregated biclustering solutions. Finally to evaluate the performance of each aggregated solution we computed the F-measure (noted Δ) between the obtained solution $\overline{\Phi}(\mathbb{X})$ and the true biclustering solution $\Phi(\mathbb{X})^*$; $\Delta(\Phi(\mathbb{X})^*, \overline{\Phi}(\mathbb{X})) = \frac{1}{K}\sum_{k=1}^{K} M_{Dice}(B_k^*, \overline{B}_k)$ where $M_{Dice}(B_k^*, \overline{B}_k) = \frac{|B_k^* \cap \overline{B}_k|}{|B_k^*| + |\overline{B}_k|}$ is the Dice measure.

Figure 3 shows the performance of the different consensus in function on the size of the biclustering solution collection R with $\alpha_{pre} = \alpha_{rec} = 0.5$. Each of the six panels gives the results on the six data structures. The dot, triangle, cross and diamond curves represent respectively the F-measure in function of R for VOTE, COAS, BGP and MIX consensus. The full gray curve represents the mean of the performance of the biclustering collection. In the six panels, the performance of collection is constant around 0.5. That is not a surprise, since the performance of the collection does not depend on its size and by construction the theoretical performance of each estimated solution is 0.5. On the six dataset structures, from $R \geq 40$, all consensus functions give much better performances than the estimated solutions of the collection. The performances of MIX in all situations are strongly increasing with the size of the collection. Mix does not require a high value of R to record good result, for $R \geq 20$ it converges to their maximum and reaches 1 in all panels. The curves of BGP have the same shape, they begin with a strong increase then they converge to their maximums, but the increase phase is much longer than in MIX. It also worth to note that BGP begins with very low performances for small values of R, it is often lower than the performances of the collection. BGP reaches its best performances with $R \geq 60$, in four panels it obtains the second best results and the third on the two last panels. The performance of VOTE increases slowly and more or less linearly with the collection size. Even with very low values of R, the performance of the consensus is significantly better than the collection. VOTE gives the second best performances for S1 and S5 and the third best for the four other data structures. The performance of COAS is about constant whatever R ; it obtains the worst results in all panels.

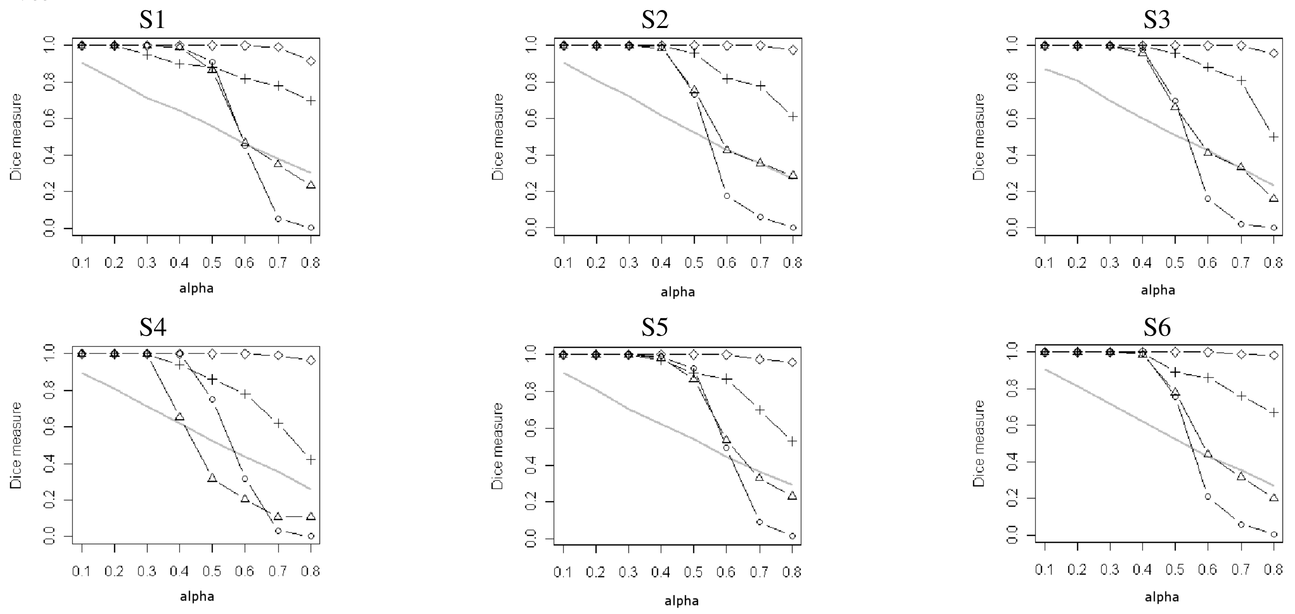
Figure 4 shows the performances of the different consensus in function of the performances of the estimated solution collection controlled by the parameter $\alpha = \alpha_{pre} = \alpha_{rec}$. The performances of all consensus are naturally decreasing with α. By definition the performances of the collection follow the line $y = 1 - x$. For $\alpha \leq 0.4$ and in all cases the consensus functions give the almost perfect biclustering solution with $\Delta \approx 1$, expected for COAS in S4. MIX is still clearly the best consensus, it produces almost the perfect biclustering and its performances are never less than 0.9. BGP is the second best consensus, it is always significantly better than the collection whatever the value of α. VOTE and COAS have similar behavior. They begin with the perfect biclustering solution then, when $\alpha \geq 0.5$, their performances decrease and are at best, for VOTE, around the collection performance.

The F-measure can be decomposed into a combination of precision and recall. In looking the results in detail we see that for VOTE and COAS the precision is much greater than the recall. That means these consensus produce smaller biclusters than the true ones, the features and examples associated to biclusters are generally good but these biclusters are incomplete i.e. examples and features are missing. In the opposite BGP produces biclusters with high recall and low precision. The aggregated biclusters are generally complete but they also contain some extra wrong features and examples. MIX gives equilibrated biclusters with equal precision and recall. The experiment on S4 allows to observe the influence of the size of the biclusters on the results. We see that COAS obtains very bad performance on the small biclusters, since the recall on two smallest biclusters is 0. MIX, VOTE, COAS are independent of the size of the biclusters, their performances are similar with the four biclusters.



○ : VOTE, △ : COAS, + : BGP, ◇ : MIX.

FIGURE 3 – Performance of the consensus in function of R (size of the biclustering solution collection) with $\alpha_{pre} = \alpha_{rec} = 0.5$.



○ : VOTE, △ : COAS, + : BGP, ◇ : MIX.

FIGURE 4 – Performance of the consensus in function on the mean precision α_{pre} and recall α_{rec} of the biclustering solution collection with $\alpha = \alpha_{pre} = \alpha_{rec}$.

5.2 Computing time

Although the performances of consensus functions are good, they have also some critical drawbacks. The use of these methods requires large amount of resources. Table 1 gives the computing time of each consensus function with $R = 50$. VOTE is the fastest method followed by MIX which is about ten times slower than VOTE, this inconvenient can be overcomed by using the eLEM algorithm pro-

TABLE 1 – Computing time (in s) of the consensus functions.

	S1	S2	S3	S4	S5	S6
VOTE	2.6	2.9	2.8	2.6	2.5	2.5
MIX	13.1	28.5	60.7	54	12.3	10.8
COAS	199.6	205.6	234.1	240.8	247.4	248.1
BGP	2502	3147.2	3345.1	3043.5	2806.6	2834.5

posed in [JN07] or the classification EM algorithm [CG93]. COAS is the third, about ten times slower than MIX and BGP needs the most computing time, about ten times more than COAS. In observing S1, S2, S3 we note that the number of biclusters has an impact of the computing time, specially for MIX. VOTE and MIX require loading a $NM \times R$ matrix than contain all labels of the collection. BGP has to generate a graph containing $NM + R$ vertices while COAS requires computing resources for large distance matrices of size $NM \times NM$. When the dimension are large, these methods may become intractable.

5.3 Results on real data

To evaluate our approach in performance term on real datasets, we have used four datasets :

– Nutt : Gene expression data on the classification of gliomas in brain.
– Pomeroy : Gene expression data on different types of tumor in the central nervous system.
– Sonar : Sonar signal from metal objects or rocks.
– Wdbc : Biological data on breast cancer.

The description of these datasets in size term is given in Table 2.

TABLE 2 – Description of the four datasets

Data sets	N	M
Nutt	50	500
Pomeroy	42	500
Sonar	208	60
Wdbc	569	30

Unlike to numerical experimental and since we do not known the true biclustering solutions, the measures of performance can be based on external indices, like Dice score. Obviously, the quality of a biclustering solution can be measured by the AMSR i.e. the average of MSR computed from each bicluster belonging to the biclustering solution ; the lower AMSR, the better the solution. A problem of this approach is that the MSR is biased by the size of the biclusters. Indeed, AMSR advantages the solutions given the smallest biclusters. To remove this size bias we set the size of the biclusters in the parameters of the algorithms. All methods will therefore return biclusters of the same size. The better solutions will be those minimizing AMSR. To compare the different consensus functions, we compute their gain that is the percentage of AMSR decreasing from the single biclustering solution i.e. the solution obtained by the classic CC algorithm without the ensemble approach. This is computed by :

$$Gain = 100\frac{AMSR(\overline{\Phi}_{single}) - AMSR(\overline{\Phi}_{ensemble})}{AMSR(\overline{\Phi}_{single})},$$

where $\overline{\Phi}_{single}$ and $\overline{\Phi}_{ensemble}$ are the biclustering solution returned respectively by the single and ensemble approaches.

TABLE 3 – Gain of each consensus function on the four real datasets in function of the size of the biclusters.

Nutt dataset

	50	100	200	300	400	600	800
VOTE	94	64	18	20	34	27	27
MIX	13	3	43	39	36	18	3
COAS	28	37	14	14	32	5	6
BGP	73	68	74	1	30	22	16

Pomeroy dataset

	50	100	200	300	400	600	800
VOTE	79	85	79	69	32	63	60
MIX	84	83	69	52	37	75	74
COAS	69	78	21	36	30	43	39
BGP	68	80	21	22	30	46	51

Sonar dataset

	50	100	200	300	400	600	800
VOTE	20	30	41	75	93	86	88
MIX	29	47	55	88	92	77	82
COAS	28	17	33	45	72	36	76
BGP	34	51	50	46	20	21	32

Wdbc dataset

	50	100	200	300	400	600	800
VOTE	15	20	28	20	4	11	3
MIX	26	19	42	32	23	21	12
COAS	-4	-18	-15	-7	-17	-8	-25
BGP	6	13	37	31	2	10	-4

Table 3 gives the gain of each consensus function for all datasets in function on the size of the biclusters. We observe that :

– In all situations, all consencus functions give an interesting gain, expected for COAS for Wdbc dataset.

We know that in the merge process, once a cluster is formed it does not undo what was previously done ; no modifications or permutations of objects are therefore possible. This disadvantage can be a handicap for COAS in some situations such as in Wdbc dataset.

– VOTE and MIX outperform BGP in most cases. In addition their behavior seem not depend on the size of biclusters. In Nutt and Sonar datasets, their performance has increased or decreased respectively.
– VOTE appears more efficient than MIX for Nutt dataset which is the larger. However, the size of biclusters seems unaffacted MIX in other experiments.
– The difference of performance between VOTE/MIX and BGP/COAS is large. We observe that the size of the bicluster may impact the performance of the methods but there is no clear rule, it is only depending on the data. Other investigations will be necessary.

In summary VOTE and MIX produce the best performances, the third is BGP and the last is COAS. Knowing that VOTE and MIX require less computing time than BGP, both appear therefore more fruitful.

6 Conclusion

Unlike to the classical clustering context, the biclustering considers the both dimensions of the matrix in order to produce homogeneous submatrices. In this work, we have presented the approach of ensemble biclustering which consists to generate a collection of biclustering solutions then to aggregate them. Firstly, we have showed how to use the CC algorithm to generate the collection. Secondly, for the aggregation of the collection of biclustering solutions, we have extended the use of four consensus functions commonly used in the clustering context. Thirdly we have evaluated the performance of each of them.

On simulated and real datasets, the ensemble approach appears fruitful. The results show that it improves significantly the performance of biclustering whatever the consensus function among VOTE, MIX and BGP. Specifically, VOTE and MIX give clearly the best results in all experiments and require less computing than BGP. Then we recommend to use one of these two methods for the ensemble biclustering problems.

Références

[BPP08] S. Busygin, O. Prokopyev, and P.M. Pardalos. Biclustering in data mining. *Computers and Operations Research*, 35(9) :2964–2987, 2008.

[Bre96] Leo Breiman. Bagging predictors. *Machine Learning*, 24 :123–140, 1996.

[Bre01] Leo Breiman. Random forests. *Machine Learning*, 45 :5–32, 2001.

[CC00] Y. Cheng and G. M. Church. Biclustering of expression data. *Proc Int Conf Intell Syst Mol Biol*, 8 :93–103, 2000.

[CG93] C. Celeux and Govaert. Comparison of the mixture and the classification maximum likelihood in cluster analysis. *J. Statistical Computation Simulation*, 47 :127–146, 1993.

[DF03] S. Dudoit and J. Fridlyand. Bagging to improve the accuracy of a clustering procedure. *Bioinformatics*, 19(9) :1090–1099, 2003.

[Dhi01] Inderjit S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '01, pages 269–274, 2001.

[Die00] Thomas G. Dietterich. Ensemble methods in machine learning. *Lecture Notes in Computer Science*, 1857 :1–15, 2000.

[DLR77] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society, series B*, 39(1) :1–38, 1977.

[DMM03] Inderjit S. Dhillon, Subramanyam Mallela, and Dharmendra S. Modha. Information-theoretic co-clustering. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '03, pages 89–98. ACM, 2003.

[DUAdA06] R. Diaz-Uriarte and S. Alvarez de Andres. Gene selection and classification of microarray data using random forest. *BMC Bioinformatics*, 7(3), 2006.

[ES10] Cesim Erten and Melih Sözdinler. Improving performances of suboptimal greedy iterative biclustering heuristics via localization. *Bioinformatics*, 26 :2594–2600, 2010.

[FB04] Xiaoli Zhang Fern and Carla E. Brodley. Solving cluster ensemble problems by bipartite graph partitioning. In *Proceedings of the twenty-first international conference on Machine learning*, ICML '04, pages 36–, 2004.

[FLS04] D. Frossyniotis, A. Likas, and A. Stafylopatis. A clustering method based on boosting. *Pattern Recognition Letters*, 25 :641–654, 2004.

[GN03] G. Govaert and M. Nadif. Clustering with block mixture models. *Pattern Recognition*, 36 :463–473, 2003.

[Gov95] G. Govaert. Simultaneous clustering of rows and columns. *Control and Cybernetics*, 24(4) :437–458, 1995.

[Har72] J. A. Hartigan. Direct clustering of a data matrix. *Journal of the American Statistical Association*, 67(337) :123–129, 1972.

[HN10] B. Hanczar and M. Nadif. Bagging for biclustering : Application to microarray data. In *European Conference on Machine Learning*, volume 1, pages 490–505, 2010.

[HN12] B. Hanczar and M. Nadif. Ensemble methods for biclustering tasks. *Pattern Recognition*, 45(11) :3938–3949, 2012.

[JN07] F.-X. Jollois and M. Nadif. Speed-up for the expectation-maximization algorithm for clustering categorical data. *Journal of Global Optimization*, 37(4) :513–525, 2007.

[LO00] Laura Lazzeroni and Art Owen. Plaid models for gene expression data. Technical report, Stanford University, 2000.

[Mac97] Richard Maclin. An empirical evaluation of bagging and boosting. In *In Proceedings of the Fourteenth National Conference on Artificial Intelligence*, pages 546–551. AAAI Press, 1997.

[MO04] S. C. Madeira and A. L. Oliveira. Biclustering algorithms for biological data analysis : a survey. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 1(1) :24–45, 2004.

[PS82] Christos H. Papadimitriou and Kenneth Steiglitz. *Combinatorial optimization : algorithms and complexity*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1982.

[RB06] Jörg Reichardt and Stefan Bornholdt. Statistical mechanics of community detection. *Phys. Rev. E*, 74 :016110, 2006.

[Sch03] R. Schapire. The boosting approach to machine learning : An overview. *in Nonlinear Estimation and Classification, Springer*, 2003.

[SG02] A. Strehl and J. Ghosh. Cluster ensembles - a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 3 :583–617, 2002.

[TBK05] Heather Turner, Trevor Bailey, and Wojtek Krzanowski. Improved biclustering of microarray data demonstrated through systematic performance tests. *Computational Statistics & Data Analysis*, 48(2) :235–254, 2005.

[TJP04] Alexander Topchy, Anil K. Jain, and William Punch. A mixture model of clustering ensembles. In *Proc. SIAM Intl. Conf. on Data Mining*, 2004.

[TLJF04] A. P. Topchy, M. H. C. Law, A. K. Jain, and A. L. Fred. Analysis of consensus partition in cluster ensemble. In *Fourth IEEE International Conference on Data Mining.*, pages 225–232, 2004.

[vdLPB03] Mark van der Laan, Katherine Pollard, and Jennifer Bryan. A new partitioning around medoids algorithm. *Journal of Statistical Computation and Simulation*, 73(8) :575–584, 2003.