Bi-clustering continuous data with self-organizing map

Khalid Benabdeslem, Kais Allab

Neural Comput & Applic (2013) 22:1551–1562
DOI 10.1007/s00521-012-1047-6

ORIGINAL ARTICLE

Bi-clustering continuous data with self-organizing map

Khalid Benabdeslem · Kais Allab

Received: 30 March 2011 / Accepted: 22 June 2012 / Published online: 13 July 2012

Abstract In this paper, we present a new SOM-based bi-clustering approach for continuous data. This approach is called Bi-SOM (for Bi-clustering based on Self-Organizing Map). The main goal of bi-clustering aims to simultaneously group the rows and columns of a given data matrix. In addition, we propose in this work to deal with some issues related to this task: (1) the topological visualization of bi-clusters with respect to their neighborhood relation, (2) the optimization of these bi-clusters in macro-blocks and (3) the dimensionality reduction by eliminating noise blocks, iteratively. Finally, experiments are given over several data sets for validating our approach in comparison with other bi-clustering methods.

Keywords Bi-clustering · SOM · Dimensionality reduction

1 Introduction

Clustering is an important task in machine learning and data mining processes. It aims at organizing data into homogeneous groups (or clusters), such as all instances in the same group are similar to each other (cohesion property), while dissimilar from instances in other groups (separation property). This organization is only made on rows according to all columns of the data matrix. However, it is clear that in the obtained partition, each class of instances is characterized according to a subset of features that participate the most on its construction. Thus, it makes good sense to simultaneously search for the specific relation that may exist between the instances and the features. This defines the block clustering (bi-clustering or cross-classification) whose purpose is to reorganize the data matrix into homogeneous blocks according to a given similarity measure.

Since the first algorithm, called Block Clustering and proposed in [17], several other bi-clustering methods have been proposed in diverse application domains, including image processing [32], text mining [10] and bioinformatics [23]. In the direct clustering approach (*Block Clustering*), the data matrix is divided into several sub-matrices corresponding to blocks. The division of a block depends on the variance of its values. Indeed, more the variance is low, more the block is constant. The quality of the final partition is then estimated by the sum of all the variances of these blocks [17].

In 1975, Hartigan [18] proposed two other algorithms of bi-clustering: the first one (*One-Way Splitting*) is mainly based on partitioning the instances with features having an intra-class variance greater than a given threshold in order to split the associated class. As any technique based on direct clustering, a minimal threshold yields a significant number of classes with low density and vice-versa. The second algorithm (*Two-Way Splitting*) proceeds by successive divisions of rows and columns. It calculates, at each iteration, a large number of variances. This is not adaptable to very large databases. In addition, the choice of the threshold is not obvious and requires either prior knowledge or additional tests.

In 1983, Govaert [16] proposed three algorithms of bi-clustering, *Croeuc* for continuous data, *Crobin* for binary

K. Benabdeslem (✉)
University of Lyon 1, LIRIS, CNRS-UMR 5205,
69622 Lyon, France
e-mail: kbenabde@univ-lyon1.fr; kbenabde@univ-lyon.fr

K. Allab
University of Lyon 1, GAMA, 69622 Lyon, France
e-mail: kais.allab@etu.univ-lyon1.fr

data and *Croki2* for contingency tables. The three algorithms proceed by optimizing the partitions of rows and columns by an iterative procedure based on the euclidean distance as an objective function to minimize. These algorithms can be applied to large data sets, but they require the number of both rows and columns.

The main drawback of the above algorithms is that the division (in rows and/or columns) is irreversible. In other words, these methods never reconsider the choice of a division once done. To get around this problem, Cheng and Church [7] proposed in 2000, *δ-clusters* approach, a greedy function for creating bi-clusters by adding rows (or columns) that maximize a local gain. This approach uses the residual mean square (RMS) as similarity measure. In 2003, Yang et al. improved *δ-clusters* by proposing another method called *FLOC* (*Flexible Overlapped Clusters*). They introduced an additional function dealing with the missing data and the overlapping [35]. In 2002, Tanay et al. [33] proposed a graph-based method called *Samba*, which enumerates exhaustively all the cliques modeling the possible bi-clusters in a bipartite graph from the data matrix. Other statistical models were used in an approach called *Plaid Model* proposed by Lazzaroni and Owen [21] and in a method of spectral bi-clustering proposed by Klugar et al. [19].

We may also cite methods of generative bi-clustering proposed by Govaert et al. [15] based on mixture models and the bi-clustering approaches proposed by Pensa et al. [27] which improve the relevance of the partitions according to background knowledge in form of constraints.

Several other methods of bi-clustering using various techniques were proposed: hierarchical bi-clustering [11], bi-clustering based on evolutionary algorithms [25], bayesian bi-clustering [24], bi-clustering using simulated annealing [5] and finally bi-clustering using random walk [1].

Furthermore, there are some methods of bi-clustering based on self-organizing maps (SOM) as *DCC* (*Double Conjugated Clustering*) [6] and KDISJ (*Kohonen for Disjonctive Table*) [8]. The approach DCC has problems relative to the use of two maps (one for the instances and the other for the features), which are built independently with the same dimension. KDISJ is dedicated to categorical data and will be more explained in the next section.

The aforementioned bi-clustering approaches have several shortcomings. The iterative clustering on rows and columns tries to optimize an objective function applied to one of both partitions (rows or columns) by fixing the other one. In some cases, this process of alternative optimization can be inefficient on large data sets. Indeed, the inconvenience of divisive and greedy approaches is that the decisions (division, addition/deletion of rows/columns) are taken once for all, and a bad choice could imply the loss of potential good bi-clusters [7]. Besides, the high complexity of the statistical and enumerative approaches is prohibitive. Finally, several methods require the intervention of the user to fix the values of some parameters (number of bi-clusters, thresholds, etc.).

To overcome some of these shortcomings, we propose an approach of topological bi-clustering (*Bi-SOM*) based on Self-Organizing Maps. SOM is a very powerful tool for analyzing and visualizing numerical data. It is used by Bi-SOM to cluster instances and features simultaneously in a single map and visualize the obtained blocks. In addition, Bi-SOM uses an iterative process of dimensionality reduction by eliminating blocks of irrelevant features considered as "noise".

The rest of the paper is organized as follows. The next section describes a topological approach of bi-clustering called *KDISJ* [8, 9]. Then, we present our proposed approach (Bi-SOM) in the third section. Finally, we provide some experimental results to validate this approach and report comparison results with other bi-clustering methods on real and synthetic data.

2 Related work

SOM is a very popular tool used for visualizing high-dimensional data spaces. It can be considered as doing vector quantization and/or clustering while preserving the spatial ordering of the input data rejected by implementing an order of the reference vectors (also called prototype vectors, cluster centroids or reference vectors) in a one- or two-dimensional output space. The SOM consists of neurons organized on a regular low-dimensional grid, called the map.

More formally, the map is described by a graph (Γ, E). Γ is a set of k interconnected neurons having a discreet topology defined by E. For each pair of neurons (c, r) on the map, the distance $\delta(c, r)$ is defined as the shortest path between c and r on the graph. This distance imposes a neighborhood relation between neurons. Each neuron c is represented by a p-dimensional reference vector $w^c = \{w_1^c, \ldots, w_p^c\}$, where p is equal to the dimension of the input vectors. The number of neurons may vary from a few dozen to several thousand depending on the application.

The SOM training algorithm resembles *k-means* [22]. The important distinction is that in addition to the best matching reference vector, its neighbors on the map are updated. The final result is that neighboring neurons on the grid correspond to neighboring regions in the input space. The training made by SOM introduces the preservation of the topology and imposes that two nearby neurons c, r by report to the discreet topology, are associated with two

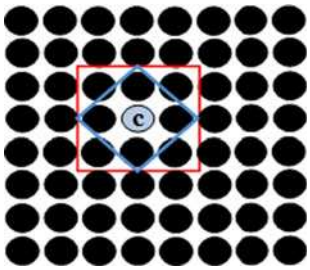

Fig. 1 Two-dimensional topological map with 1-neighborhood of a neuron *c*. Rectangular (*red*) with 8 neighbors and diamond (*blue*) with 4 neighbors (colour figure online)

nearby vectors w^c, w^r, by report to the distance chosen between the data (Fig. 1).

The SOM algorithm is proposed on two versions: stochastic (on-line) or batch (off-line) versions. In this paper, we choose the first version for its ability to well organize the map providing a good topology preservation from data. Theoretical comparisons between both versions can be found in [12].

The first SOM-based bi-clustering approach has been proposed by Cottrell et al. [9]. The method, called KDISJ (*Kohonen for Disjonctive Table*), is an extension of the algorithm KORRESP that has been introduced to analyze contingency tables [8]. The KDISJ algorithm is dedicated to categorical data and aims to simultaneously analyze the instances and the modalities of the qualitative features that describe them.

Let A be a data matrix of dimension $(n \times p)$, we build the complete disjunctive table D of dimension $(n \times m)$, such as m is the number of the modalities of the p features. D is then corrected (D^c) by the following formula:

$$d_{ij}^c = \frac{d_{ij}}{\sqrt{d_{i.}d_{.j}}} \tag{1}$$

$$\text{where: } d_{i.} = \sum_{j=1}^{m} d_{ij} \quad \text{and} \quad d_{.j} = \sum_{i=1}^{n} d_{ij} \tag{2}$$

Note that $d_{i.}$ is equal the number of features p, and $d_{.j}$ represents the number of instances associated with the modality j. After this transformation, we use the Euclidian distance on D^c which is equivalent to use the weighted χ^2 distance on D.

We then consider SOM algorithm and associate with each unit u a reference vector w^u that is comprised of $(m + n)$ components, with the first m components evolving in the space of instances (represented by the rows of D^c) and the last n components evolving in the space of modalities (represented by the columns of D^c). The SOM algorithm lends itself to a double learning process. At each step, we alternatively draw from D^c a row (instance i) or a D^c column (modality j).

When we draw an instance i, we associate the modality $j(i)$ defined by:

$$j(i) = arg\,max_j(d_{ij}^c) = arg\,max_j\left(\frac{d_{ij}}{\sqrt{pd_{.j}}}\right) \tag{3}$$

that maximizes the coefficient d_1^c, that is the rarest modality out of all of the corresponding ones in the total population. This modality is the most characteristic for this instance. In case of ex-aequo, we do as usual in this situation, by randomly drawing a modality among the candidates. We then create an extended instance vector $X = (i, j(i))$, of dimension $(m + n)$ (Fig. 2). Subsequently, we look for the closest of all the reference vectors, in terms of the Euclidean distance restricted to the first m components. Let us denote by u^* the winning unit:

$$u^* = arg\,min_u \|X_{(1\ldots m)} - w^u_{(1\ldots m)}\|^2. \tag{4}$$

Next, we update the reference vector of the unit u^* and its neighbors closer to the extended vector $X = (i, j(i))$:

$$w^u_{(1\ldots m)}(t+1) = w^u_{(1\ldots m)}(t) + \varepsilon\,\sigma(u,u^*)(X_{(1\ldots m)} - w^u_{(1\ldots m)}(t)) \tag{5}$$

where ε is the adaptation parameter (positive, decreasing with time) and σ is the neighborhood function, defined by $\sigma(u,u^*) = 1$ if u and u^* are neighbors in the Kohonen network, and = 0 if not. The radius of the neighborhood varies in the time too.

When we draw a modality j with dimension n (a column of D^c), we do not associate an instance with it. For explaining the principle, let us denote by Y (see Fig. 2) the

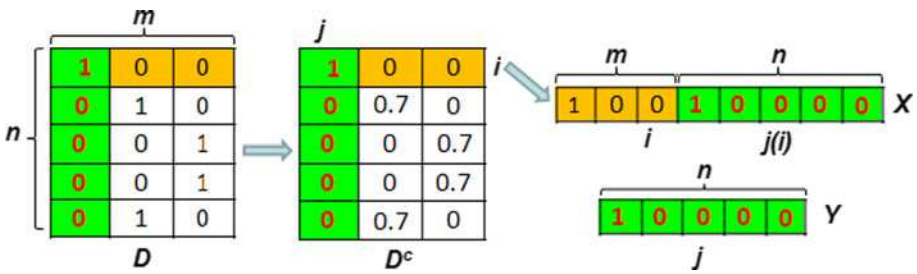

Fig. 2 The data matrix D, the corrected matrix D^c and the vectors X (*line* + *column*), Y (*column*)

$n-$column vector corresponding to modality j. We then seek the closest reference vector, in terms of the Euclidean distance restricted to the last n components.

Let v^* be the winning unit:

$$v^* = arg\,min_u \|Y_{(m+1\ldots m+n)} - w^u_{(m+1\ldots m+n)}\|^2. \quad (6)$$

We then update the last n components of the reference vector associated with v^* and its neighbors closer to the corresponding components of the modality vector Y, without modifying the first m components. This step can be formulated by:

$$w^u_{(m+1\ldots m+n)}(t+1) = w^u_{(m+1\ldots m+n)}(t) + \varepsilon\,\sigma(u, v^*)(Y_{(m+1\ldots m+n)} - w^u_{(m+1\ldots m+n)}(t)) \quad (7)$$

We so apply the SOM algorithm to the instances and modalities while maintaining them associated. Generally, $(n + m)$ iterations are enough for obtaining the convergence [9]. In the end, neighboring instances and modalities are classified in the same class or in neighboring classes. As we mentioned above, this approach handles nominal or discrete features, and therefore, continuous features need to be properly discretized. This could imply a dramatic loss of information in some cases. We suggest to solve this problem by the denition of a new link between rows and columns while respecting the nature of the continuous data.

3 Proposed approach

In this section, we describe our approach that we call Bi-SOM. The proposal carries out a bi-clustering and dimensionality reduction from a continuous data matrix. The principle of Bi-SOM, on a set of n instances described by p features, aims to create a partition of homogeneous data blocks; containing both instances and features that characterize them. The idea is based on this notion of characterization as the link between features and instances for clustering them simultaneously in one map.

Let A be a data matrix of dimension $(n \times p)$. We consider a Kohonen network in which we associate with each unit u a reference vector w^u that consists of $(p + n)$ components. The first p components represent the feature space, and the last n components represent the sample space. The Kohonen algorithm lends itself to a double learning process. At each step, we alternatively draw a vector X (a row or a column) from A.

When X is a row (instance), we seek its nearest neighbor (Y in A) in terms of Euclidean distance. We then create two extended vectors V_1, V_2 of dimension $(p + n)$ by concatenating X and Y with the same feature K which characterizes them at best. The feature K corresponds to the minimum value among the distances between the components of X and those of Y (Fig. 3).

$$K = arg\,min_j(X_j - Y_j)^2 \quad (8)$$

Note that if there are several minimum values, we build several vectors for the same pair of instances X and Y with different columns for K.

For each vector V_1 (V_2 respectively), we seek for the closest of all the reference vectors, in terms of the Euclidean distance using all components $(p + n)$ (contrary to KDISJ which uses the Euclidian distance restricted on the first p components).

Let us denote by u_1^* (u_2^*) the winning unit of V_1 (V_2)

$$u_1^* = arg\,min_u \|V_1 - w^u\|^2 \quad (9)$$

Next, we update the reference vectors of the unit u_1^* (u_2^*) and its neighbors in the map:

$$w^u_{(1\ldots p+n)}(t+1) = w^u_{(1\ldots p+n)}(t) + \varepsilon\,\sigma(u, u_1^*)(V_{1(1\ldots p+n)} - w^u_{(1\ldots p+n)}(t)) \quad (10)$$

Note that if Y is the closest instance to X, and K is the column which characterizes them at best, then $V_1 = [X\ K]$, $V_2 = [Y\ K]$ ($[X\ K]$ denotes de concatenation of X and K). However, X may not be necessarily the closest

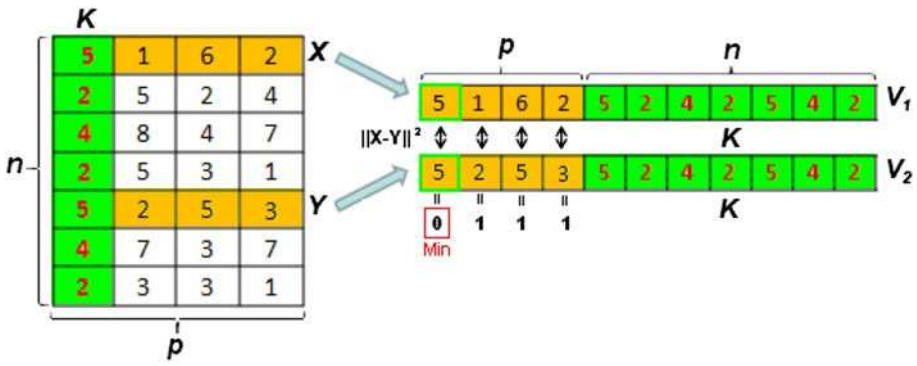

Fig. 3 The data matrix, the constructed vectors of dimension $(n + p)$: $V_1 = [XK]$ and $V_2 = [YK]$

Table 1 Characteristics of used data sets

Data set	Kind	n	p	# class	References
Breast	Real	699	9	2	[13]
Heart	Real	303	13	2	[13]
Ovarian	Real	54	1,536	2	[31]
Leukemia	Real	72	1,762	2	[14]
Prelic1	Synthetic	100	50	10	[28]
Prelic2	Synthetic	100	100	10	[28]
Prelic3	Synthetic	100	50	10	[28]
Prelic4	Synthetic	100	100	10	[28]

instance to Y. In this case, let X' be the closest instance to Y and K' the feature that characterizes them at best, thus we construct $V_3 = [Y\ K']$ and $V_4 = [X'\ K']$. If V_2 and V_3 are classified in different neurons, then Y belongs to several classes. Consequently, this property leads the overlapping of these classes on Y.

When X is a column (feature), with dimension n, we seek the reference vector (neuron) that is the closest, in terms of the Euclidean distance restricted to the last n components.

Let v^* be the winning unit of X:

$$v^* = arg\,min_u \|X_{(p+1\ldots p+n)} - w^u_{(p+1\ldots p+n)}\|^2 \tag{11}$$

We then update the last n components of the winning reference vector associated with v^* and its neighbors closer to the corresponding components of X, without modifying the first p components:

$$w^u_{(p+1\ldots p+n)}(t+1) = w^u_{(p+1\ldots p+n)}(t) + \varepsilon\sigma(u, v^*)(X_{(p+1\ldots p+n)} - w^u_{(p+1\ldots p+n)}(t)) \tag{12}$$

In the obtained partition, a block (neuron) could be empty, containing features and instances, or containing only features. In the last case, Bi-SOM considers these features, which do not characterize no instance, as being irrelevant. Thus, they are deleted from the data matrix, and the bi-clustering is then restarted on the new data matrix. This operation is repeated until the elimination of all the blocks containing only features.

Finally, Bi-SOM is clustered by an Ascendant Hierarchical Clustering (AHC) applied on the reference vectors of the map. This Post-clustering aims at grouping together the close blocks into macro-blocks, in order to optimize their initial number.

3.1 Algorithm

Bi-SOM takes in input a data matrix A of dimension $(n \times p)$ and returns a partition Γ in a single topological map. The reference vectors, representing the neurons of the map, are of dimension $(p + n)$ since each vector is a result of the concatenation of a row and a column of A.

In the worst case, each row of A will be concatenated with all the columns by building every time two vectors (the row and its nearest neighbor). Thus, the number of elements to be treated by SOM is of $2n$ vectors (for rows) plus p vectors (for columns). The complexity of SOM is $O(kn)$, where k is the number of neurons in the map. Consequently, the complexity of Bi-SOM is $O(k(2n + p))$.

Note that B in the algorithm represents the "Noisy" feature set. It contains the features that do not characterize no instances in the map. In other terms, the features are considered as irrelevant if they belong to neurons having no instances. Therefore, they are deleted from the data matrix, and the bi-clustering is repeated on the new data matrix until obtaining blocks with both instances and features.

4 Experiments

4.1 Data sets

Bi-SOM is implemented in Matlab with the Toolbox available in http://www.cis.hut.fi/projects/somtoolbox/. In the first phase of experiments, we applied the approach to data describing several diseases: *Breast cancer wisconsin*, *Heart disease*, *Ovarian cancer* and *Leukemia disease* (Table 1). The *Leukemia* data set was used for the first time by Golub et al. [14] and by other authors afterward [19, 23] . The data consist of 72 individuals with two types of Leukemia named *ALL (Acute Lymphocytic Leukemia)* and *AML (Acute Myelogenous Leukemia)*. The data contain initially 7,129 features (genes). In [6], it was suggested to delete the "affymetrix" control-genes and the genes having a value <20 (biologically, the low levels of expression are difficult to interpret). Finally, 1,762 features are kept.

Algorithm 1 : Bi-SOM

Input: A (Data matrix $= I \times F$), I: the set of instances and F: the set of features.
Output: Γ : obtained partition (blocks).
Begin
- Initialization of the map (with k neurons of dimension $(p + n)$).
repeat
  repeat
    - Draw randomly a vector X of A (row or column).
    if X is a row **then**
      - Look for its closest instance Y in A.
      for *all* K such that $K = argmin_j \|X_j - Y_j\|^2$ **do**
        - Create $V_1 = [XK]$ and $V_2 = [YK]$.
        - Determine the winning neurons of V_1 and V_2 using (eq. 9).
      end for
      - Update the reference vectors of the map using (eq. 10).
    else
      - X is a column : - Determine the winning neuron of X using (eq. 11).
      - Update the reference vectors of the map using (eq. 12).
    end if
  until Treatment of all the rows and the columns of A
  - $B = \{j \in F | \exists r \in \Gamma \wedge j \in r \Rightarrow \neg \exists i \in I \wedge i \in r\}$.
  - $F = F - B$.
  - $A = I \times F$.
until $B = \emptyset$.
- Γ = The set of the k obtained neurons.
- Rearrangement of A (rows and columns) according to Γ.
End

The other data sets are synthetic and provided by Prelic et al. [28] for analyzing the effects of noise and overlapping on the bi-clustering performances.

4.2 Evaluation criteria

It is always difficult to compare two methods of bi-clustering, because of their unsupervised nature. Nevertheless, there are some indices (internal and external) allowing to estimate the quality of the obtained partition.

4.2.1 Internal indices

Internal indices compare intrinsic information about data with the bi-clustering results. In this case, no a priori information further than the raw data are available [30].

An internal index is calculated from two matrices P and M. The matrix P contains the information on the proximity between elements (rows or columns) such that: $P_{ij} = P_{ji} = distance\ (x_i, x_j)$. The matrix M is built as described in external indices, but inversed so higher values correspond to objects (rows or columns) not grouped together. For example $M_{ij} = 1/(1 + k)$ such as k is the number of times that objects i and j are classified together. M_{ij} is included between 0 and 1; it is equal to 1 if i and j are never classified in the same class and close to 0 if they are often classified in the same class. These two matrices are compared by using the normalized Hubert index [28]:

$$H(P,M) = \frac{\frac{1}{h}\sum_{i=1}^{l-1}\sum_{j=i+1}^{l}(P_{ij} - \mu_P)(M_{ij} - \mu_M)}{\sigma_P \sigma_M} \tag{13}$$

where l is the number of objects of the matrix P, and $h = l(l - 1)/2$. μ_P, μ_M are the means of matrices P and M and σ_P, σ_M their variances, respectively. We can thus calculate the index H_I for the instances (rows) and the index H_F for features (columns), and both indices in:

$$H_{IF} = \frac{nH_I + pH_F}{n + p} \tag{14}$$

where n is the number of instances and p is the number of features. The best bi-clustering is the one that minimizes the value of H_{IF}.

4.2.2 External indices

The external indices are used to compare the obtained partition with the correct partition of the data, when it is available.

Let S and R be two partitions, and F a cross table $(S \times R)$. F_{ij} indicates the similarity between the bi-cluster i from S and the bi-cluster j from R. We use the matrix F to calculate the *F-score* index [34].

- Let g_i be the number of instances in the bi-cluster i and c_i the number of features in the bi-cluster i $(n_i = g_i c_i)$;
- $g_{i \cap j}$ is the number of instances belonging to both bi-clusters i and j;

- $c_{i\cap\ j}$ is the number of features belonging to both bi-clusters i and j.

The *F-score* between i and j is calculated as follows:

$$F\text{-}score(i,j) = \frac{2(g_{i\cap j})(c_{i\cap j})}{n_i + n_j} \quad (15)$$

From Eq. (15), we can calculate the overall relevance between the two partitions R and S by:

$$Relevance(R,S) = \frac{1}{|R|}\sum_{i=1}^{|R|} max_{j=1}^{|S|}(F\text{-}score(i,j)) \quad (16)$$

We can also use another measure to evaluate Bi-SOM by calculating the degree of similarity between the obtained partition and the correct partition (Match Score) [2]. Let S and R be two partitions. G_1 the set of rows in S, G_2 the set of rows in R, C_1 the set of columns in S and C_2 the set of columns in R. The Match Score between S and R is:

$$MatchScore(S,R) = \frac{1}{|S|}\sum_{(G_1,C_1)\in S} max_{(G_2,C_2)\in R}\frac{|G_1\cap G_2| + |C_1\cap C_2|}{|G_1\cup G_2| + |C_1\cup C_2|} \quad (17)$$

Finally, three other external indices are used: Jaccard index, Rand index (RI) [29] and Adjusted Rand index (ARI) [2]. We note by:

a: the number of pairs of objects classified together in S and in R.
b: the number of pairs of objects classified together in S but not in R.
c: the number of pairs of objects classified together in R but not in S.
d: the number of pairs of objects not classified neither in S nor in R.

$$\text{Jaccard}(S,R) = \frac{a}{a+b+c} \quad (18)$$

$$\text{RI}(S,R) = \frac{a+d}{a+b+c+d} \quad (19)$$

$$\text{ARI}(S,R) = \frac{2(ad-bc)}{(a+b)(b+d)+(a+c)(c+d)} \quad (20)$$

4.3 Results on synthetic data

In the first phase of experiments, we applied Bi-SOM to synthetic data sets provided by Prelic et al. [28]. These data sets illustrate various scenarios allowing to study the effects of noise and overlapping on the proposed approach. Two types of bi-clusters are considered (constant and additive) (Table 2).

Table 2 Examples of two types of bi-clusters

(a) Constant bi-cluster			
1.0	**2.0**	**3.0**	**4.0**
1.0	2.0	3.0	4.0
1.0	2.0	3.0	4.0
1.0	2.0	3.0	4.0
(b) Additive bi-cluster			
1.0	**2.0**	**5.0**	**0.0**
2.0	3.0	6.0	1.0
4.0	5.0	8.0	3.0
5.0	6.0	9.0	4.0

The reference rows are in *bold*

4.3.1 *Effect of noise*

In this phase, we applied Bi-SOM on two data sets, *Prelic1* (for constant bi-clusters) and *Prelic2* (for additive bi-clusters). The aim is to analyze the sensitivity of the approach to noise, by using the Match Score as quality measure.

The first set of data *Prelic1* is of dimension (100 × 50). It illustrates constant bi-clusters and contains a first correct version of the data and other versions corresponding to rates of 5, 10, 15, 20 and 25 % of noise. The second set of data *Prelic2* is of dimension (100 × 100). It illustrates additive bi-clusters and contains a first correct version of the data, and other versions corresponding to rates of 2, 4, 8 and 10 % of noise.

Note that this notion of noise is different from the features's relevance study, previously described in Bi-SOM. The Noise here is imitated by adding random values drawn from a normal distribution to each cell of the original matrix.

Figure 4 shows the comparison of the results obtained by Bi-SOM with those of the methods (*BiMax* [28], *ISA* [4], *Samba* [33], *CC* [7], *OPSM* [3] and *x-Motif* [26]). In the absence of noise, Bi-SOM, *ISA* and *Samba* realize an excellent score more than 99 %, as well for the constant bi-clusters as the additive bi-clusters. *BiMax* realizes weak and unstable scores in the case of additive bi-clusters. The other methods have weak results for both scenarios. The most important in our case is that in the presence of the noise, Bi-SOM realizes an excellent Match Score often equal to 100 % (Fig. 4), which means that the method is robust to noise, and particularly competitive to the other methods such as *ISA* and *Samba*.

Figure 5 illustrates the results obtained by Bi-SOM applied on the data *Prelic1* relative to the constant bi-clusters, with a rate of noise equal to 5 %. In Fig. 5b, we show the obtained blocks (bi-clusters) after the segmentation of the map. Each block corresponds to all the neurons

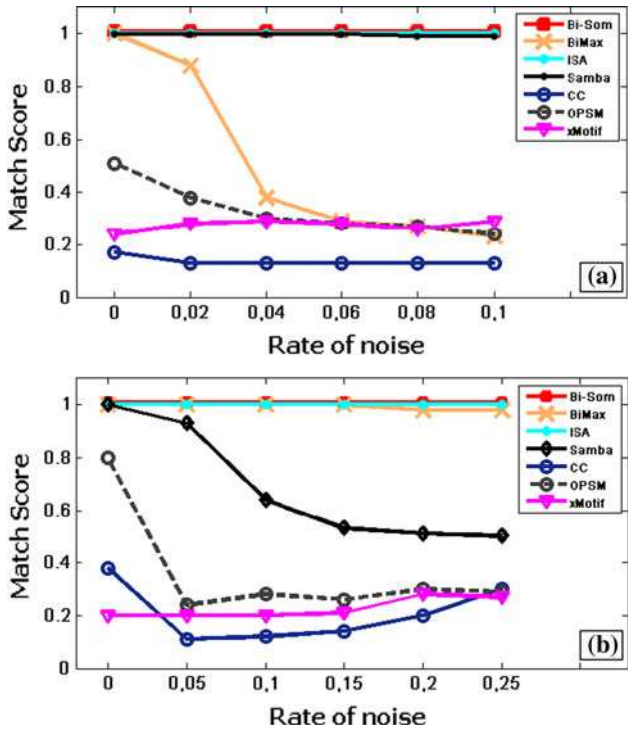

Fig. 4 Comparison between Bi-SOM and some bi-clustering methods, applied to noisy data. **a** Constant bi-clusters. **b** Additive bi-clusters

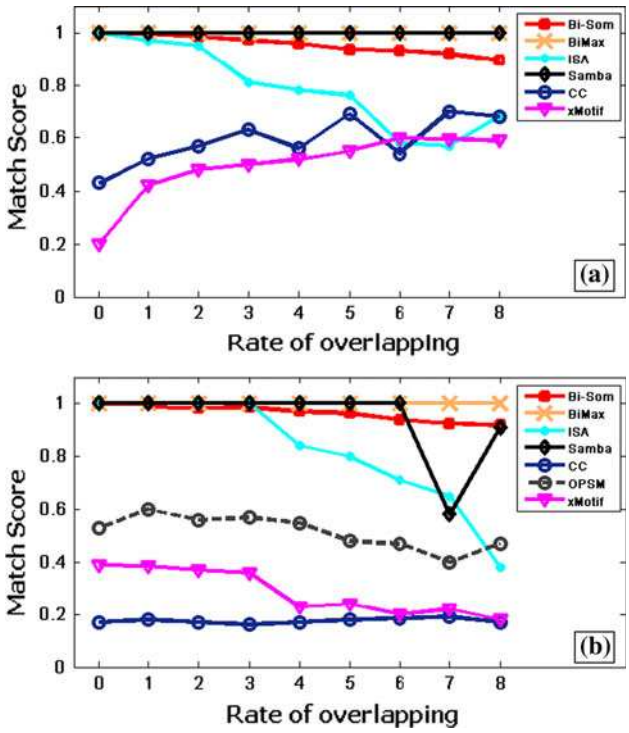

Fig. 6 Comparison between Bi-SOM and some bi-clustering methods, applied to overlapped data. **a** Constant bi-clusters. **b** Additive bi-clusters

having the same color in Fig. 5a. We can see that Bi-SOM obtains the optimal number of ten (10) blocks, corresponding to the correct partition (10 not overlapped blocks of dimension (10 × 5)). The blocks are visualized in the form of a perfect diagonal in (Fig. 5b) according to the topology illustrated in the map of Fig. 5a.

4.3.2 *Effect of overlapping*

In this phase, we applied Bi-SOM on two other data sets, *Prelic3* (for constant bi-clusters) and *Prelic4* (for additive bi-clusters). These data sets are not noisy but contain a first correct version of the data, and other versions containing bi-clusters overlapped in rows and columns. The rate of overlapping varies from 0 to 8 elements (rows and columns). The goal is to analyze the sensitivity of the approach to overlapped bi-clusters.

BiMax is the only method that is stable, whatever the rate of overlapping and whatever the type of bi-clusters. Bi-SOM, *Samba* and *ISA* realize good scores, but they are sensitive to the overlapping (*ISA* is the most sensitive one). Although *OPSM* is not affected by the rate of overlapping in the case of the additive bi-clusters, it remains inefficient for the constant bi-clusters, which explains its absence in Fig. 6a.

Figure 7 shows the results obtained by Bi-SOM applied on the data *Prelic4* relative to additive bi-clusters with a rate of overlapping equal to 10 % (in rows and columns). Figure 7b shows the obtained blocks (bi-clusters) after the segmentation of the map. Each block corresponds to all

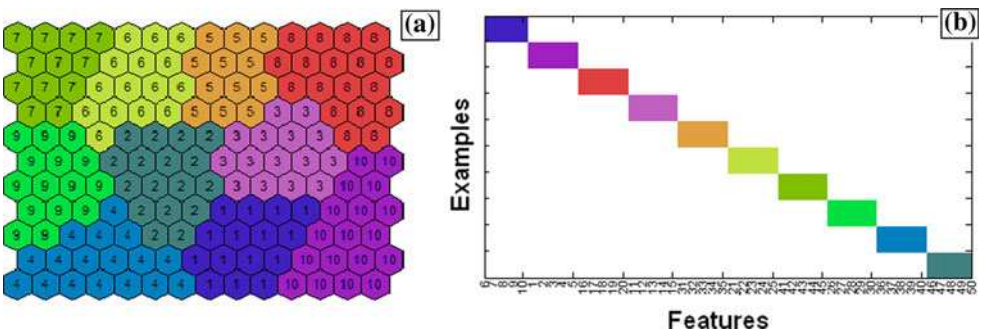

Fig. 5 Results of Bi-SOM applied to the correct version of *Prelic1* data set. **a** The segmented map. **b** The obtained blocks

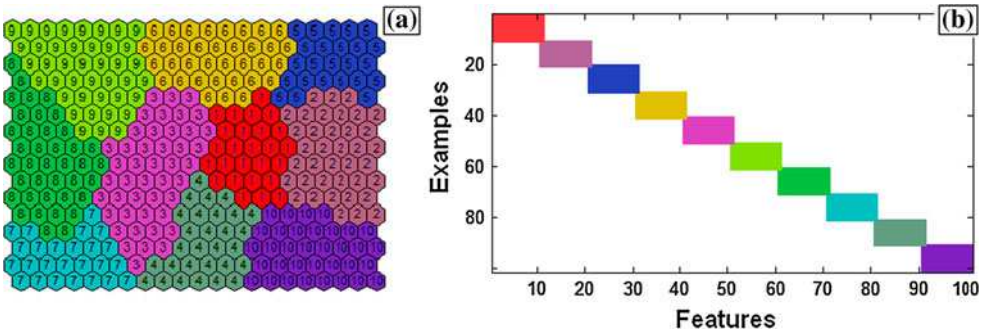

Fig. 7 Results of Bi-SOM applied to the correct version of *Prelic3* data set, with a rate of overlapping equal to 10 %. **a** The segmented map. **b** The obtained blocks

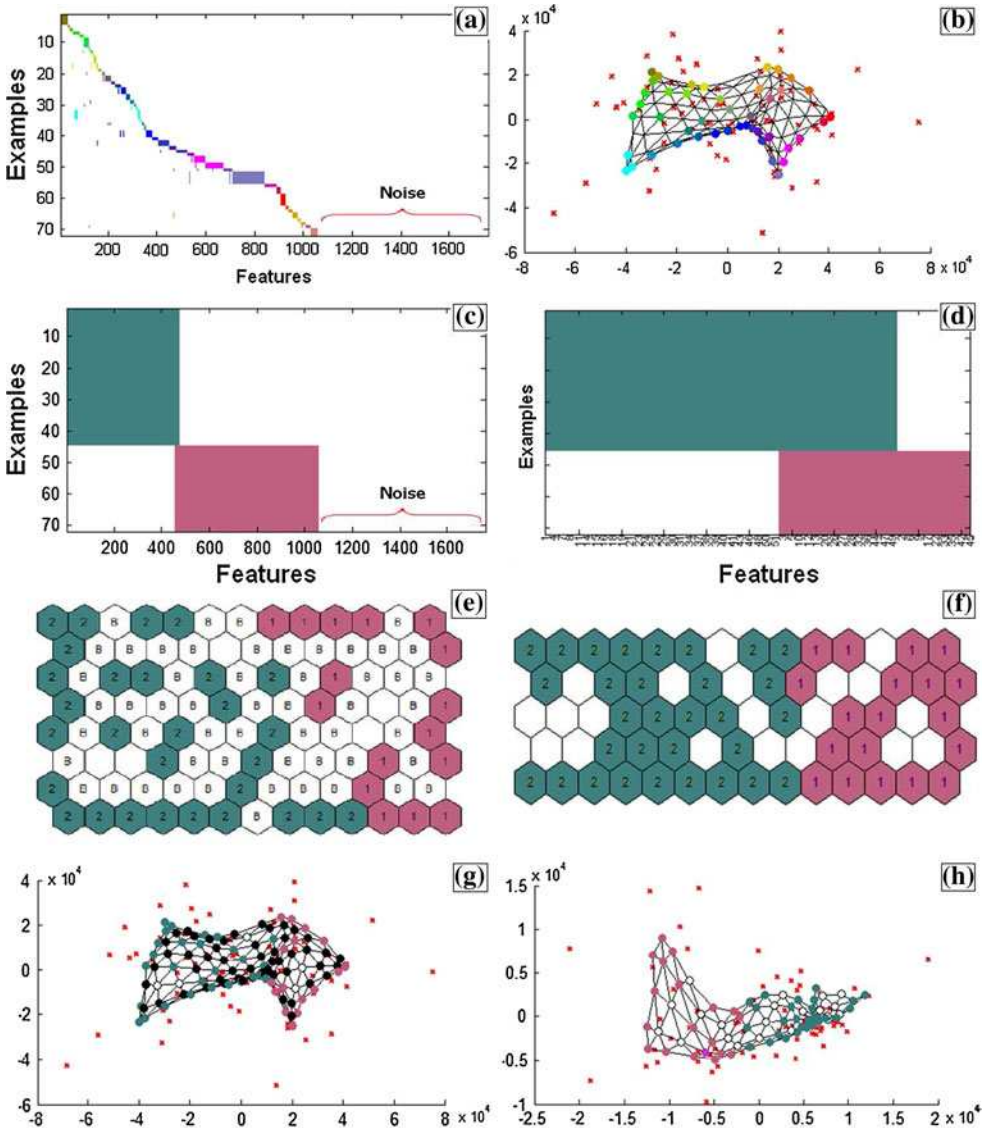

Fig. 8 Results of the bi-clustering on Leukemia data set: **a** the obtained blocks. **b** The map projected on the data space. **c** The obtained macro-blocks. **d** Macro-blocks after the reduction of dimension. **e**, **g** the map optimized before the reduction of dimension. **f**, **h** the map optimized after the reduction of dimension

neurons having the same color in Fig. 7a. We then obtained ten (10) blocks of dimension (10 × 10), each block is overlapped by one row and one column with the block which precedes it in the diagonal. That corresponds perfectly to the correct partition of this data set (Match Score = 1).

4.4 Results on real data

In this phase of experiments, Bi-SOM is applied on the *Leukemia* data set in order to visualize the results of the bi-clustering and the dimensionality reduction. First, we applied the Kohonen heuristic for calculating the dimensions of the map [20]. We obtained (13 × 8).

After the learning of the map, we build bi-clusters relative to neurons, and we create a binary matrix D, such that $D_{ij} = 1$ if the row i and the column j are classified together, $D_{ij} = 0$ otherwise. This matrix is rearranged to visualize the obtained blocks, where each block in Fig. 8a represents the neuron having the same color in Fig. 8b. After that, we apply the AHC algorithm (with "Ward" based agglomerative criterion) to group the close neurons together in macro-bloks. The Fig. 8e, g show the neurons of the map distributed in two classes (1:AML and 2:ALL) corresponding to the two macro-blocks in Fig. 8c. The Fig. 8f, h show the distribution of the map's neurons in two classes corresponding to the two macro-blocks in Fig. 8d after dimensionality reduction.

The neurons having the label (B) represent the classes containing irrelevant features, which corresponds to the black neurons in Fig. 8g. The white neurons (without labels) in Fig. 8g, h represent the empty classes that do not contain neither instances nor features. Their existence represents the smoothing property of the SOM algorithm.

After the first learning of the map, the neurons are projected according to the distribution of the data (Fig. 8b). After the rearrangement of the binary matrix D, each bi-cluster corresponds to an active neuron which contains instances and features (Fig. 8a). We can remark that our method allows the overlapping in rows and/or columns.

The most important remark is that the white space in the right of the bi-clusters in Fig. 8a, c corresponds to the features that are not classified in classes containing instances. These features are considered as irrelevant for the bi-clustering and so eliminated from the initial data matrix. Therefore, the dimension of the map is reduced (Fig. 8g before and Fig. 8h after the dimensionality reduction).

For comparing our approach with the other bi-clustering approaches, we used the tool **BiCAT** *(Bi-clustering Analysis Toolbox)* available in http://www.tik.ee.ethz.ch/∼sop/bicat/. It is a tool of biological data analysis, with two known clustering methods (k-means and AHC (with "Ward" based agglomerative criterion)) and five bi-clustering approaches: (*BiMax, ISA, Samba, CC, OPSM*).

We applied all the methods on the real data sets (*Breast*, *Heart*, *Ovarian* and *Leukemia*). The performances of the obtained partitions are presented in Tables 3, 4, 5, 6. Note that **BicAT** did not detect bi-clusters by applying the method *ISA* on *Breast* data set (Table 3).

Table 3 Performances of Bi-SOM on Breast data set

Index	K-means	AHC	ISA	CC	OPSM	BiMax	Bi-SOM
Jaccard	0.242	0.548	–	0.618	0.511	0.650	0.884
RI	0.583	0.553	–	0.737	0.576	0.730	0.934
ARI	0.219	0.016	–	0.492	0.107	0.448	0.866
Match Score	0.194	0.137	–	0.173	0.458	0.049	0.928
Relevance	0.105	0.260	–	0.156	0.136	0.007	0.963

Table 4 Performances of Bi-SOM on Heart data set

Index	K-means	AHC	ISA	CC	OPSM	BiMax	Bi-SOM
Jaccard	0.118	0.490	0.500	0.403	0.502	0.376	0.547
RI	0.513	0.502	0.500	0.530	0.502	0.522	0.581
ARI	0.030	0.007	0.005	0.060	0.000	0.044	0.128
Match Score	0.125	0.114	0.428	0.114	0.372	0.227	0.603
Relevance	0.061	0.264	0.267	0.114	0.124	0.292	0.683

Table 5 Performances of Bi-SOM on Ovarian data set

Index	K-means	AHC	ISA	CC	OPSM	BiMax	Bi-SOM
Jaccard	0.208	0.453	0.396	0.497	0.497	0.585	0.701
RI	0.546	0.491	0.586	0.497	0.497	0.620	0.747
ARI	0.088	0.013	0.171	0.000	0.000	0.474	0.534
Match Score	0.150	0.130	0.394	0.556	0.324	0.686	0.898
Relevance	0.092	0.259	0.228	0.133	0.443	0.814	0.927

Table 6 Performances of Bi-SOM on Leukemia data set

Index	K-means	AHC	ISA	CC	OPSM	BiMax	Bi-SOM
Jaccard	0.204	0.529	0.515	0.393	0.520	0.564	0.540
RI	0.549	0.559	0.489	0.516	0.520	0.588	0.604
ARI	0.148	0.059	0.017	0.021	0.000	0.284	0.325
Match Score	0.169	0.162	0.435	0.099	0.241	0.662	0.578
Relevance	0.118	0.255	0.163	0.106	0.145	0.702	0.681

The overall results of Bi-SOM are very encouraging on all the used indices. We can see that Bi-SOM provides partitions with a best quality in comparison with the other methods, except in some cases with the method BiMax. For example, for *Leukemia* data set, this method produced the best results of Jaccard index, Match score and Relevance.

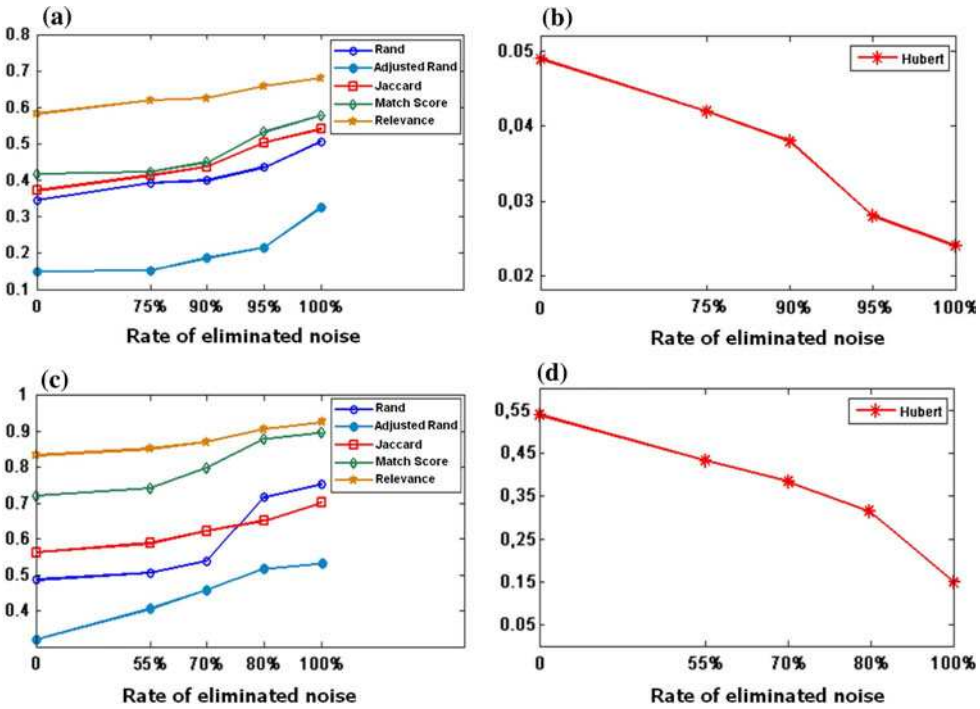

Fig. 9 Evaluation of the obtained partition of Bi-SOM for *Leukemia* and *Ovarian* data sets. **a** Leukemia: external indices. **b** Leukemia: internal index. **c** Ovarian: external indices. **d** Ovarian: internal index

4.5 Results of dimensionality reduction

For Bi-SOM, a feature is irrelevant if it does not characterize no instance. In other terms, if it is classified in neurons that do not contain instances. Bi-SOM proceeds iteratively by deleting the irrelevant features and computes again the bi-clustering with the new obtained data set. This operation is repeated until deleting all irrelevant features that represent the noise. In each iteration, we calculate the performances of the obtained partition (Fig. 9).

Bi-SOM did not detect irrelevant features in both *Breast* and *Heart* data sets, which contain 9 and 13 features, respectively. However, the rate of irrelevant feature detection is of (97 %) for *Leukemia* and of (65 %) for *Ovarian*. We show in Fig. 9a, c that the performances of Bi-SOM are weak when it is applied to the initial data sets with irrelevant features.

The external indices (to be maximized) start with low values and increase gradually. Hence, the Hubert index (to be minimized) (Fig. 9b, d) indicates the quality of the partition obtained by Bi-SOM, by taking into account the instances and the features of bi-clusters. The obtained values are always close to 0 and show that this index is highly influenced by the deletion of the irrelevant features. Bi-SOM realizes the best performance with the totally reduced data set, (0.024) for *Leukemia* for (0.150) with *Ovarian*.

5 Conclusion

In this paper, we proposed a new bi-clustering approach, called Bi-SOM dedicated to continuous data. This approach is based on the definition of a new link associating with each instance, the feature that characterizes it the most. With Bi-SOM, we used the neighborhood information offered by SOM to introduce a topology between the obtained blocks. Then, we exploited this neighborhood property to optimize the obtained partition by grouping the blocks in macro-blocks.

Furthermore, Bi-SOM allowed us to reduce the dimensionality of data, by detecting and deleting the irrelevant features to improve the process of bi-clustering and consequently, the quality of the obtained partition. The comparison of the obtained results with the performances of the other bi-clustering methods showed that our approach is robust to noise, weakly sensitive to overlapping and remains competitive with very encouraging performance rates.

There are several avenues that could be planned for future research. First, since our proposal is sensitive to the overlapping, we could investigate in a new procedure to control this problem. Second, it would be interesting to extend the method to deal with semi-supervised problems. Finally, additional effort is needed to experiment the method on very high-dimensional data.

References

1. Angiulli F, Cesario E, Pizzuti C (2008) Random walk biclustering for microarray data. Inf Sci 178:1479–1497
2. Bandyopadhyay S, Mukhopadhyay A, Maulik U (2007) An improved algorithm for clustering gene expression data. Bioinformatics 21:2859–2865

3. BenDor A, Chor B, Karp R, Yakhini Z (2003) Discovering local structure in gene expression data: the order preserving sub matrix problem. J Comput Biol 10(3–4):373–384
4. Bergmann S, Ihmels J, Barkai N (2004) Defining transcription modules using large-scale gene expression. Bioinformatics 20(13):1993–2003
5. Bryan K, Cunningham P, Bolshakova N (2005) Biclustering of expression data using simulated annealing. CBMS 2005:383–388
6. Busygin S, Jacobsen G, Kramer E (2002) Double conjugated clustering applied to leukemia microarray data. In: Proceedings of the 2nd SIAM international conference on data mining, workshop on clustering high dimensional data
7. Cheng Y, Church G (2000) Biclustering of expression data. In: Proceedings of the 8th international conference on intelligent systems for molecular biology (ISMB'00), vol 8, pp 93–103
8. Cottrell M, Ibbou S, Letrémy P (2004) Som-based algorithms for qualitative variables. Neural Netw 17(8–9):1149–1167
9. Cottrell M, Letrémy MP (2005) How to use the kohonen algorithm to simultaneously analyze individuals and modalities in a survey. Neurocomputing 63:193–207
10. Dhillon IS (2001) 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, pp 269–274
11. Eisen M, Spellman P, Brown P, Botstein D (1998) Cluster analysis and display of genome-wide expression patterns. Proc Natl Acad Sci USA 95(25):14863–14868
12. Fort J, Cottrel M, Letrémy P (2001) Stochastic on-row algorithm versus batch algorithm for quantization and self-organizing maps. Neural networks for signal processing XI, 2001. In: Proceedings of the 2001 IEEE signal processing society workshop, pp 43–52
13. Frank A, Asuncion A (2010) UCI machine learning repository. University of California, Irvine, School of Information and Computer Sciences. http://archive.ics.uci.edu/ml
14. Golub TR, Slonim DK, Tamayo P, Huard C, Gaasenbeek M, Mesirov JP, Coller H, Loh M, Downing JR, Caligiuri MA, Bloomfield CD, Lander ES (1999) Molecular classification of cancer: class discovery and class prediction by gene expression. Science 286:531–537
15. Govaert G, Nadif M (2008) Block clustering with mixture models: comparison of different approaches. Comput Stat Data Anal 52:3233–3245
16. Govaert G (1983) Classification Croisée. Thèse d'état, Université de Paris6
17. Hartigan J (1972) Direct clustering of data matrix. J Am Stat Assoc 67(337):123–129
18. Hartigan J (1975) Direct splitting. Clustering algorithms, Chap. 14. Wiley, New York, pp 251–277
19. Klugar Y, Basri R, Chang J, Gerstein M (2003) Spectral biclustering of microarray data: coclustering genes and conditions. Genome Res 13:703–716
20. Kohonen T (2001) Self-organizing maps. Springer, Berlin
21. Lazzeroni L, Owen A (2000) Plaid models for gene expression data. Stat Sin 12:61–86
22. MacQueen J (1967) Some methods for classification and analysis of multivariate observations. Proc Fifth Berkeley Symp Math Stat Probab 1:281–297
23. Madeira S, Oliveira A (2004) Biclustering algorithms for biological data analysis: a survey. IEEE/ACM Trans Comput Biol Bioinf 1(1):24–45
24. Meeds E, Roweis S (2007) Nonparametric bayesian bi-clustering. Technical report
25. Mitra S, Banka H (2006) Multi-objective evolutionary biclustering of gene expression data. Pattern Recogn 39(12):2464–2477
26. Murali T, Kasif S (2003) Extracting conserved gene expression motifs from gene expression data. Pac Symp Biocomput 8:77–88
27. Pensa R, Boulicaut J-F, Cordero F, Atzori M (2010) Co-clustering numerical data under user-defined constraints. Stat Anal Data Min 3(1):38–55
28. Prelic A, Bleuler S, Zimmermann P, Wille A, Buhlmann P, Gruissem W, Hennig L, Thiele L, Zitzler E (2006) A systematic comparison and evaluation of biclustering methods for gene expression data. Bioinformatics 22(9):1122–1131
29. Rand WM (1971) Objective criteria for the evaluation of clustering methods. J Am Stat Assoc 66:846–850
30. Santamaria R, Quintales L, Theron R (2007) Methods to bicluster validation and comparison in microarray data. In: Proceedings of IDEAL 2007, LNCS4881, pp 780–789
31. Schummer M, Ng W, Bumgarner R, Nelson P, Schummer B, Bednarski D, Hassell L, Baldwin R, Karlan B, Hood L (1999) Comparative hybridization of an array of 21500 ovarian cdnas for the discovery of genes overexpressed in ovarian carcinomas. Gene 238(2):375–385
32. Shi J, Malik J (2000) Normalized cuts and image segmentation. Technical report, University of California at Berkeley, Berkeley, CA, USA
33. Tanay A, Sharan R, Shamir R (2002) Discovering statistically significant biclusters in gene expression data. Bioinformatics 18:36–44
34. Xiaowen L, Wang L (2007) Computing the maximum similarity bi-clusters of gene expression data. Bioinformatics 23(1):50–56
35. Yang J, Wang W, Wang H, Yu P (2003) Enhanced biclustering on expression data. BIBE '03, pp. 321–327