Adaptive batch SOM for multiple dissimilarity data tables

Anderson B. dos S. Dantas*, Francisco de A. T. de Carvalho*
*Center of Informatics – Federal University of Pernambuco (CIn/UFPE)
Av. Prof. Luiz Freire, s/n - Cidade Universitaria, CEP 50740-540, Recife – PE – Brazil
Email: {absd,fatc}@cin.ufpe.br

***Abstract*—This paper introduces a clustering algorithm based on batch Self-Organizing Maps to partition objects taking into account their relational descriptions given by multiple dissimilarity matrices. The presented approach provides a partition of the objects and a prototype for each cluster, moreover the method is able to learn relevance weights for each dissimilarity matrix by optimizing an adequacy criterion that measures the fit between clusters and the respective prototypes. These relevance weights change at each iteration and are different from one cluster to another. Experiments using real-world data bases are considered to show the usefulness of the method.**

***Keywords*-Clustering; Self-Organizing Maps; Relational data; Relevance weight; Multiple dissimilarity matrices**

I. Introduction

Clustering algorithms have been used in machine learning fields including data mining, document retrieval, image segmentation and pattern recognition [1]. Clustering methods aims at organize a data set into clusters in such a way that items belonging to the same cluster have a high degree of similarity, on the other hand, items into different clusters have a high degree of dissimilarity.

Kohonen's SOM (Self-Organizing Maps) [2] are a special type of unsupervised neural network which has clustering and visualization properties [3]. SOM can be considered as an algorithm that maps a high dimensional data space in a one, two or three-dimensional space. This projection enables partitioning the inputs into similar groups and has the property to preserve the topology of the data.

There are two common representations of the objects upon which clustering can be based: feature data and relational data. When each object is described by a vector of quantitative or qualitative values the set of vectors describing the objects is called feature data. When each pair of objects is represented by a relationship, then we have relational data. The most common model of relational data is the case when we have a matrix of dissimilarity data $R = [r_{il}]$, where r_{il} is the pairwise dissimilarity (often a distance) between objects i and l.

References [4] and [5] propose an adaptation of a batch SOM algorithm to dissimilarity data. Reference [6] presents a clustering algorithm that is able to partition objects taking simultaneously into account their relational description given by multiple dissimilarity matrices.

This work proposes an approach based on batch SOM algorithm with adaptive weights to classify objects using multiple dissimilarity matrices. These matrices can be obtained using different sets of variables and a fixed dissimilarity function (the final partition gives a consensus between different views describing the objects), using a fixed set of variables and different dissimilarity functions (the final partition gives the consensus between different dissimilarity functions) or using different sets of variables and dissimilarity functions. The influence of the different dissimilarity matrices is not equally important in the definition of the clusters in the final consensus partition [7]. Thus, in order to obtain a meaningful partition from all dissimilarity matrices, it is necessary to learn cluster-dependent relevance weights for each dissimilarity matrix.

This paper is organized as follows. Section II first describes the Self-Organizing Map algorithm as proposed by [2]. Section III describes a batch Self-Organizing Map algorithm for dissimilarity data proposed by [4]. Section IV presents the algorithm proposed by this paper, the batch Self-Organizing Map for multiple dissimilarity data tables. To show the usefulness of this clustering algorithm, experiments with real-value data sets are considered in section V. Finally, Section VI gives the conclusions.

II. Self-Organizing Maps (SOM)

The class of methods called Self-Organizing Maps [2] belongs to the category of machine learning called competitive or unsupervised [8]. The Self-Organizing Maps algorithm involve iterative procedures to associate a finite number of objects (represented by input vectors) with a finite number of representational points, in such a way that similarity relationships between inputs are respected by these points.

The architecture of SOM algorithm is composed by a set of neurons organized on a regular low-dimensional grid, generally with one, two or three dimensions called map. More formally, the map is described by a non-oriented graph (C, Γ). C is a set of m interconnected neurons having a discrete topology defined by Γ. The graph structure allow the definition of a distance function between two neurons in the map. For each pair of neurons (*c,r*) in the map, $\delta(c, r)$ is the distance between neurons c and r in the graph C. This distance function enables a neighborhood relationship

between neurons. Each neuron c is represented by a p-dimensional referent vector $w_c = \{w_c^1, \ldots, w_c^p\}$, where p is equal to the dimension of the input vectors.

The main feature of Self-Organizing Maps is the possibility of comparing clusters. Each element of the data set is affected to a cluster. Each cluster is projected to a neuron in the map. Similar elements tend to be projected in the same neuron, whereas dissimilarity grows up with the distance between two projections. The neuron with the best correspondence to a given input is updated, as well as its neighbors on the map. The region around the "winner" neuron is stretched towards the training sample presented. This procedure makes the neurons in the map to become ordered, i.e. neighboring neurons have similar referent vectors.

Let $E = \{1, \ldots, n\}$ the set of objects, where each object $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip})(i = 1, \ldots, n)$ belongs to $\mathbb{R}^p$. And, let $h_{\delta(j,l)}$ be a neighborhood function, where $\delta(j, l)$ is a fixed distance function between neurons j and l. $h_{\delta(j,l)} = 1$ when $j = l$ and decreases to zero as $\delta(j, l)$ increases [9].

A. SOM Algorithm

1) Fix m, $\eta(0)$ (start learning rate), N_{iter} (maximum number of iterations) and $h_{\delta(j,l)}(0)$ (start neighborhood function);
2) Initialization: choose randomly m distinct vectors as the initial prototypes $\mathbf{w}_c(0)$, $c = 1, 2, \ldots, m$ and set the counter $t = 1$;
3) Sampling: Grab a random input vector $\mathbf{x}_i(t)$;
4) Selection: find the best Kohonen neuron (winner) $\mathbf{w}_c(t)$ in relation to minimum Euclidean distance rule:
$$f(\mathbf{x}_i(t))^T = \min_{1 \leq j \leq m} \sum_{k=1}^{p} (x_{ik}(t) - w_{jk}(t))^2;$$
5) Weight modification: for all neurons j within a specified neighborhood radius of the winning neuron $\mathbf{w}_c$, adjust the weights according to the following formula:
$$\mathbf{w}_j(t+1) = \mathbf{w}_j(t) + \eta(t) h_{\delta(\mathbf{w}_c, w_j)}(t)(\mathbf{x}_i(t) - \mathbf{w}_j(t));$$
6) Updating: update both learning rate $\eta(t)$ and neighborhood function $h_{l,j}(t)$;
7) Stopping criterion: if $t = N_{iter}$, stop; otherwise go to 3.

III. Batch Self-Organizing Map for dissimilarity data

The batch SOM algorithm for dissimilarity data (B-SOM for dissimilarity data) introduced by [4] stems from the original SOM as described above. The main difference is on the data to be classified. These data are represented by a dissimilarity relationship. Let $E = \{e_1, \ldots, e_n\}$ be a set of n objects and a dissimilarity measure $d(e_i, e_l)$ between the objects e_i and e_l. Each neuron c is represented by a reference vector (or prototype) $g_c = e_j$, $e_j \in E$. In classical SOM each referent vector evolves in the entire input space $\mathbb{R}^p$, in this approach each neuron has a finite number of representations.

The batch training algorithm is an iterative two-step algorithm (affectation and representation steps, discussed later) in which the whole set of input (noted E) is presented to the map before any adjustments be achieved. During learning the following cost function is minimized:

$$J = \sum_{i=1}^{n} \sum_{r=1}^{m} K^T(\delta(f(e_i), r)) d(e_i, g_r) \quad (1)$$

where f is the allocation function and $f(e_i)$ stands for the neuron of the map that is associated to the object e_i and $\delta(f(e_i), r)$ is the distance on the map between a neuron r and the neuron that is allocated to the object e_i. The concept of neighborhood is taken into account through kernel functions K which are positive and such that $\lim_{|x| \to \infty} K(x) = 0$ [10]. Moreover, K^T that is parameterized by T (where T stands for temperature), is the neighborhood kernel function that defines the influence region around each neuron r. The smaller the value of T, the fewer the neurons that belong to the neighborhood of a given neuron r.

This cost function is an extension of the *k-means* cost function, where the Euclidean distance is replaced by a generalized distance denoted d^T:

$$d^T(e_i, g_{f(e_i)}) = \sum_{r=1}^{m} K^T(\delta(f^T(e_i), r)) d(e_i, g_r) \quad (2)$$

This generalized distance is a weighted sum of the euclidean distances between e_i and all the prototypes of the neighborhood of the neuron $f(e_i)$.

When T is kept fixed, the minimization of J is performed iteratively in two steps: affectation and representation.

During affectation step, the function f associates each element e_i to the neuron whose referent vector is "closest" to e_i and decreases the clustering criterion J. Each individual e_i is assigned to its nearest neuron:

$$c = f^T(e_i) = arg \min_{1 \leq r \leq m} d^T(e_i, g_r) \quad (3)$$

During the representation step, new prototypes representing each cluster are selected. The prototype g_r^* of the cluster C_r, which minimizes the cost function J is determined by the equation:

$$g_r^* = arg \min_{e \in E} \sum_{i=1}^{n} K^T(\delta(f^T(e_i), r)) d^T(e_i, e_r) \quad (4)$$

A. The B-SOM Algorithm

1) Initialization.
Fix the number m of neurons (clusters);
Fix δ; Fix the kernel function K^T;
Fix the number of iterations N_{iter};
Fix T_{min}, T_{max}; Set $T \leftarrow T_{max}$; Set $t \leftarrow 0$;
Randomly select m distinct prototypes $g_c^{(0)} \in E(c = 1, \ldots, m)$;
Set the map $L(m, G^0)$, where $G^0 = (g_1^{(0)}, \ldots, g_m^{(0)})$;
Assign each object e_i to the closest neuron (cluster) according to equation 3;

2) Step 1: Representation.
Set $T = T_{max} * (\frac{T_{min}}{T_{max}})^{\frac{t}{N_{iter}-1}}$;
The allocation function is kept fixed;
Select the prototypes $g_c^{(t)}(c = 1, \ldots, m)$ according to equation 4;

3) Step 2: Affectation.
The prototypes $g_c^{(t)}(c = 1, \ldots, m)$ are fixed. Assign each individual $e_i(i = 1, \ldots, n)$ to its nearest neuron according to equation 3;

4) Stopping criterion. If $T = T_{min}$ then Stop; otherwise set $t = t + 1$ and go to 2 (Step 1).

IV. Adaptive batch Self-Organizing Map for multiple dissimilarity data tables

In this section we present an adaptive clustering algorithm based on both SOM for dissimilarity data [4] and the clustering algorithm for multiple dissimilarity data tables [6]. The main objective of the model proposed is to map objects taking into account its relational descriptions given by multiple dissimilarity data tables.

This approach is an adaptive batch SOM algorithm (AB-SOM) that uses different adaptive weights on the dissimilarities matrices. Those weights change at each algorithm iteration and are different from one cluster to another and aims to measure the relevance of each matrix in the process of clustering. Previous works [11] [12] give adaptive versions of stochastic SOM algorithm. The computation of these vectors of weights in this algorithm is inspired from the approach used to compute a weight for each variable in each cluster in the dynamic clustering algorithm based on adaptive distances [13].

Let $E = \{e_1, \ldots, e_n\}$ be a set of n objects and let p dissimilarity nxn matrices $(\mathbf{D}_1, \ldots, \mathbf{D}_j, \ldots, \mathbf{D}_p)$ where $\mathbf{D}_j[i, l] = d_j(e_i, e_l)$, and $d_j(e_i, e_l)$ gives the dissimilarity between the objects e_i and e_l on the dissimilarity matrix $\mathbf{D}_j$. Assume that the prototype g_r of the cluster C_r is an object of the set E, i.e., $g_r \in E \; \forall r = 1, \ldots, c$.

The adaptive batch SOM training algorithm for multiple dissimilarity data tables introduced in this paper is an iterative three-step algorithm (affectation, representation and weighting steps) which aims to minimize the following cost function:

$$J = \sum_{i=1}^{n} \sum_{r=1}^{c} K^T(\delta(f^T(e_i), r)) D_{\lambda_r}(e_i, g_r) \tag{5}$$

where the kernel function K is described by: $K^T(\delta) = \exp(-\frac{\delta^2}{T^2})$ and $d_j(e_i, g_r)$ is the dissimilarity between an example $e_i \in C_r$ and the cluster prototype $g_r \in E$ parameterized by relevance weight vector $\lambda_r = (\lambda_{r1}, \ldots, \lambda_{rj}, \ldots, \lambda_{rp})$ where λ_{rj} is the weight between the dissimilarity matrix $\mathbf{D}_j$ and the cluster C_r.

The relevance weight matrix λ is composed by c relevance weight vectors $\lambda_r = (\lambda_{r1}, \ldots, \lambda_{rj}, \ldots, \lambda_{rp})(k = 1, \ldots, c)$. The weight matrix changes at each iteration of the algorithm and the weight vectors are different for one cluster to another. When T is kept fixed, the algorithm alternates three steps:

Step 1: Definition of the Best Prototypes

In this step, the partition $P = (C_1, \ldots, C_c)$ of E into c clusters and the relevance weight matrix λ are fixed. The prototype $g_r \in E$ of cluster C_r, which minimizes the clustering criterion J, is computed according to:

$$g_r = arg \min_{e \in E} \sum_{i=1}^{n} K^T(\delta(f^T(e_i), r)) \sum_{j=1}^{p} \lambda_{rj} d_j(e_i, e) \tag{6}$$

Step 2: Definition of the Best Relevance Weight Matrix

In this step, the partition $P = (C_1, \ldots, C_c)$ of E and the vector of prototypes $g = (g_1, \ldots, g_c)$ are fixed. The element j of the relevance weight vector $\lambda_r = (\lambda_r 1, \ldots, \lambda_r p)$, which minimizes the clustering criterion J under $\lambda_r j > 0$ and $\prod_{j=1}^{p} \lambda_r j = 1$, is calculated according to the equation:

$$\lambda_{rj} = \frac{\{\prod_{h=1}^{p} (\sum_{i=1}^{n} K^T \delta^2(f^T(e_i), r) d_h(e_i, g_r))\}^{\frac{1}{p}}}{\sum_{i=1}^{n} K^T \delta^2(f^T(e_i), r) d_j(e_i, g_r)} \tag{7}$$

Step 3: Definition of the Best Partition

In this step, the vector of prototypes $g = (g_1, \ldots, g_c)$ and the relevance weight matrix λ are fixed. The cluster C_r, which minimize the criterion J, is updated according the following expression:

$$\begin{aligned} r &= f^T(e_i) \\ &= arg \min_{1 \leq h \leq c} \sum_{r=1}^{c} K^T(\delta(h, r)) \sum_{j=1}^{p} \lambda_{rj} d_j(e_i, g_r) \end{aligned} \tag{8}$$

A. The AB-SOM algorithm with Relevance Weight Matrix

1) Initialization.
   Fix the number c of clusters; Fix δ;
   Fix the number of iterations N_{iter};
   Fix T_{min}, T_{max};
   Set $T \leftarrow T_{max}$; Set $t \leftarrow 0$;
   Randomly select c distinct objects $g_l \in E$;
   Set the Relevance Weight Matrix λ where $\lambda_l = (\lambda_l 1, \ldots, \lambda_l p) = (1, \ldots, 1)$;
   Set the map $M(c, \mathbf{G}^0)$, where $\mathbf{G}^0 = (g_1^{(0)}, \ldots, g_c^{(0)})$;
   Assign each object e_i the the closest prototype in order to obtain the partition $P = (C_1, \ldots, C_c)$ where C_l is constructed according to the equation 8;

2) Step 1: definition of the best prototypes.
   The partition $P = (C_1, \ldots, C_c)$ and the relevance weight matrix λ are fixed;
   Set $T = T_{max} * (\frac{T_{min}}{T_{max}})^{\frac{t}{N_{iter}-1}}$;
   Compute the prototype $g_l \in E$ of cluster C_l according to equation 6.

3) Step 2: definition of the best relevance weight matrix.
   The vector of prototypes g and the partition $P = (C_1, \ldots, C_c)$ are fixed;
   For each l compute the component of the weight vector λ_l according to equation 7.

4) Step 3: definition of the best partition.
   The vector of prototypes g and the relevance weight matrix λ are fixed;
   Assign each individual $e_i (i = 1, \ldots, n)$ to its nearest neuron according to equation 8;

5) Stopping criterion If $T = T_{min}$ then STOP; otherwise set $t = t + 1$ and go to 2 (Step 1).

V. Experiments

To illustrate the usefulness of the proposed algorithm, we used databases from the UCI Machine Learning Repository [14]. Each attribute in the dataset is represented by a dissimilarity matrix. For the batch Self-Organizing Map for dissimilarity data algorithm, there is only one matrix available. For the adaptive approach there is multiple matrices. Some measures will be considered in order to compare the results furnished by the clustering methods, they are: the corrected Rand index (CR) [15], the F-measure [16] and the overall error rate of classification (OERC) [17]. In addition, the confusion matrix will be presented for each experiment. The confusion matrix describes the cardinalities of all pairwise intersections between an a-priori partition and the final partition of the clustering algorithm. Only non-empty clusters are shown, according to the final partition furnished by the algorithms, the matrices may have different sizes.

The variable parameters for the experiments are: topology of the map, the values of T_{min} and T_{max} and the total number of iterations N_{iter}. The topology of the map defines the maximum number of clusters that will be furnished by the algorithm. Table I shows the values of the parameters used on most of the experiments following described, only in the last experiment, with the Wine Quality data set, the parameters were different. Each experiment was repeated 50 times with the same values and was selected the one with the minimum adequacy criterion.

Table I
PARAMETERS FOR THE EXPERIMENTS

Parameter	Value
Topology	2x5
T_{min}	0,3
T_{max}	3,0
N_{iter}	500

A. Wine data set

This data set has results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars, providing three classes of wine. The classes 1, 2 and 3 have, respectively 59, 71 e 48 examples. Each example is described by 13 real value attributes representing the quantities of 13 constituents found in each of the three types of wines.

Table II
WINE: CONFUSION MATRIX FURNISHED BY THE B-SOM ALGORITHM

Cluster/Class	1	2	3	Majority Class
0,0	0	20	10	2
0,1	1	11	16	3
0,2	7	3	5	1
0,3	10	0	0	1
0,4	6	0	0	1
1,0	0	25	2	2
1,1	0	6	9	3
1,2	5	5	6	3
1,3	16	1	0	1
1,4	14	0	0	1

Tables II and III give the confusion matrices furnished, respectively by the batch SOM algorithm and the adaptive approach for multiple dissimilarity tables. The last column (Majority Class) represents the number of the class with most objects in a given cluster. Observing this column it is possible to recognize regions in which objects belonging to the same a-priori class are in neighbors clusters. For example, in table III, the clusters (0,0), (0,1), (1,0) and (1,1), on the left side of the map, have more objects from class 3. The clusters (1,3) and (1,4) have most objects from class 1, the others have more from class 2.

Table III
WINE: CONFUSION MATRIX FURNISHED BY THE AB-SOM FOR MULTIPLE DISSIMILARITY DATA TABLES

Cluster/Class	1	2	3	Majority Class
0,0	0	5	16	3
0,1	0	1	4	3
0,2	0	21	0	2
0,3	2	22	0	2
0,4	6	10	0	2
1,0	0	0	15	3
1,1	0	0	13	3
1,2	0	9	0	2
1,3	24	2	0	1
1,4	27	1	0	1

The results for CR index are 0.42 and 0.31 for, respectively, the adaptive approach for multiple dissimilarity tables and the batch SOM. For the F-measure the results are 0.52 and 0.45, respectively. And for the OERC, the results are 0.09 and 0.27, respectively. The adaptive SOM for multiple dissimilarity table performs better for the wine data set.

Table IV gives the final relevance weight matrix furnished by the adaptive SOM algorithm for multiple dissimilarity data tables. Values above 1 (bold in table) denote a strong influence of a given dissimilarity matrix on the cluster. For example, for the cluster (0,0) the most relevant matrix is the matrix 7 (3.05).

B. Iris data set

This data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant. The 3 classes are: Iris Setosa, Iris Versicolour and Iris Virginica. Each example is described by 4 attributes.

Table V
IRIS: CONFUSION MATRIX FROM B-SOM ALGORITHM

Cluster/Class	1	2	3	Majority Class
0,0	0	1	15	3
0,1	0	0	11	3
0,2	0	0	9	3
0,3	5	0	0	1
0,4	32	0	0	1
1,0	0	26	1	2
1,1	0	7	7	2/3
1,2	0	0	7	3
1,3	0	16	0	2
1,4	13	0	0	1

Tables V and VI give the confusion matrices furnished, respectively by the batch SOM algorithm and the adaptive approach for multiple dissimilarity tables. In table V, the bottom of the map has most objects from class 3 (clusters (1,0), (1,1), (1,2) and (1,3)) and in table VI, class 1 is mostly present on the left side of the map (clusters (0,0) and (1,0)).

Table VI
IRIS: CONFUSION MATRIX FROM AB-SOM FOR MULTIPLE DISSIMILARITY DATA TABLES

Cluster/Class	1	2	3	Majority Class
0,0	13	0	0	1
0,1	0	0	9	3
0,2	0	17	0	2
0,3	0	17	1	2
0,4	0	4	14	3
1,0	37	0	0	1
1,1	0	2	0	2
1,2	0	9	0	2
1,3	0	1	6	3
1,4	0	0	20	3

Table VII
IRIS: FINAL RELEVANCE WEIGHT MATRIX

Cluster/Matrix	1	2	3	4
0,0	0.63	0.18	**3.40**	**2.62**
0,1	0.80	0.81	**1.45**	**1.04**
0,2	0.34	0.26	**2.95**	**3.79**
0,3	0.14	0.29	**10.14**	**2.26**
0,4	0.58	0.32	**4.99**	**1.05**
1,0	0.25	0.07	**5.06**	**11.38**
1,1	**1.59**	0.52	0.95	**1.26**
1,2	0.35	0.50	**1.50**	**3.73**
1,3	0.25	**1.53**	**1.47**	**1.76**
1,4	0.51	0.26	**2.59**	**2.83**

The results for CR index are 0.54 and 0.50 for, respectively, the adaptive approach for multiple dissimilarity tables and the batch SOM. For the F-measure the results are 0.64 and 0.62, respectively. And for the OERC, the results are 0.04 and 0.06, respectively. The adaptive SOM for multiple dissimilarity table performs better for this data set.

Table VII gives the final relevance weight matrix. In this table, the weights for the matrices 3 and 4 have greater values then the other matrices, denoting a greater influence on the objects affectation to the clusters.

C. Thyroid data set

Thyroid data set contains 3 classes of data representing the status of the thyroid gland. It can be at normal state, over function or under function. The three classes (1, 2 and 3) have, respectively, 150, 35 and 30 examples, each one is described by five real-value attributes.

Tables VIII and IX give the confusion matrices furnished, respectively by the batch SOM algorithm and the adaptive approach for multiple dissimilarity tables. On both tables we can see a concentration of the class 1 on the bottom left side of the map.

The results for CR index are 0.37 and 0.41, respectively, for the adaptive approach for multiple dissimilarity tables and the batch SOM. For the F-measure the results are 0.52

Table IV
WINE: FINAL RELEVANCE WEIGHT MATRIX

Cluster/Matrix	1	2	3	4	5	6	7	8	9	10	11	12	13
0,0	**1.03**	0.27	**1.31**	**1.07**	0.48	0.90	**3.05**	0.33	**1.87**	**1.25**	0.57	**2.07**	**2.02**
0,1	0.99	**1.05**	0.55	**1.09**	0.27	0.69	0.79	0.52	0.41	0.92	**1.81**	**3.60**	**8.64**
0,2	**1.28**	0.52	0.41	0.43	**1.47**	0.60	**2.42**	0.90	**1.57**	**5.17**	0.18	0.51	**5.66**
0,3	0.69	0.33	0.93	**2.64**	**1.32**	0.48	**2.29**	**1.22**	**0.45**	**2.39**	0.36	**4.02**	0.62
0,4	**0.77**	**4.19**	0.27	0.99	0.16	**1.92**	0.84	**1.96**	0.24	**3.04**	**3.72**	**2.15**	0.37
1,0	**1.02**	0.37	0.63	0.89	0.47	**1.68**	**2.42**	0.25	**2.79**	0.45	**1.43**	5.12	0.99
1,1	0.46	0.21	0.79	0.90	**1.85**	0.49	**5.07**	0.86	0.24	0.36	**2.21**	**5.23**	**3.48**
1,2	0.42	0.61	0.51	**1.35**	0.16	**2.93**	**3.12**	0.24	**2.27**	**2.96**	**1.33**	**1.72**	**1.01**
1,3	0.77	0.55	0.69	0.16	0.29	**1.38**	**3.13**	**2.14**	**1.12**	**1.95**	**1.01**	**3.90**	0.86
1,4	0.48	**6.01**	0.37	0.33	0.61	**1.64**	**1.65**	0.94	0.40	**1.50**	**1.74**	**2.78**	0.60

Table VIII
THYROID: CONFUSION MATRIX FROM B-SOM ALGORITHM

Cluster/Class	1	2	3	Majority Class
0,0	0	16	0	2
0,1	10	0	3	1
0,2	8	3	0	1
0,3	0	0	7	3
0,4	0	0	5	3
1,0	25	10	0	1
1,1	61	6	1	1
1,2	45	0	1	1
1,3	1	0	4	3
1,4	0	0	9	2

Table IX
THYROID: CONFUSION MATRIX FROM AB-SOM FOR MULTIPLE DISSIMILARITY DATA TABLES

Cluster/Class	1	2	3	Majority Class
0,0	0	34	0	2
0,1	33	1	1	1
0,2	22	0	0	1
0,3	0	0	6	2
0,4	0	0	7	3
1,0	29	0	0	1
1,1	38	0	2	1
1,2	24	0	1	1
1,3	4	0	0	1
1,4	0	0	13	3

and 0.55, respectively. And for the OERC, the results are 0.02 and 0.11, respectively. The batch SOM performs better for this data set.

Table X gives the final relevance weight matrix. We can see that matrix 4 has a great relevance on most of the clusters.

D. E.coli data set

E.coli data set has information about proteins classified in eight classes. There are 336 examples described by 6 attributes, one of the attributes is the class. The classes

Table X
THYROID: FINAL RELEVANCE WEIGHT MATRIX

Cluster/Matrix	1	2	3	4	5
0,0	0.05	0.15	0.05	**27.94**	**85.91**
0,1	0.37	0.17	0.20	**19.29**	**4.14**
0,2	0.21	0.29	**4.45**	**2.61**	**1.35**
0,3	0.14	**2.96**	**3.01**	**1.27**	0.61
0,4	**5.38**	**3.21**	**1.64**	0.86	0.04
1,0	0.28	0.24	0.35	**9.17**	**4.59**
1,1	0.12	0.24	0.38	**15.75**	**5.72**
1,2	0.14	0.64	**1.53**	**27.39**	0.26
1,3	**1.15**	0.91	**4.18**	0.24	0.94
1,4	0.91	**9.47**	**4.83**	0.06	0.38

and their distribution are following: cp (cytoplasm), 143 examples; im (inner membrane without signal sequence), 77 examples; pp (perisplasm), 52 examples; imU (inner membrane, uncleavable signal sequence), 35 examples; om (outer membrane), 20 examples; omL (outer membrane lipoprotein), 5 examples; imL (inner membrane lipoprotein), 2 examples; imS (inner membrane, cleavable signal sequence), 2 examples.

Table XI
E.COLI: CONFUSION MATRIX FROM B-SOM ALGORITHM

Cluster/Class	1	2	3	4	5	6	7	8	Major Class
0,0	47	0	0	0	5	1	0	0	1
0,1	18	1	0	0	0	1	0	0	1
0,2	19	0	0	0	0	0	0	0	1
0,3	0	2	0	0	32	0	0	0	5
0,4	0	25	1	1	24	1	0	0	2
1,0	25	0	0	0	1	0	0	0	1
1,1	27	0	0	0	0	0	0	0	1
1,2	6	0	0	1	0	13	4	8	6
1,3	1	0	1	0	1	35	1	11	6
1,4	0	7	0	0	14	1	0	1	5

Tables XI and XII give the confusion matrices furnished, respectively by the batch SOM algorithm and the adaptive approach for multiple dissimilarity tables. In table XII the

Table XII
E.COLI: CONFUSION MATRIX FROM AB-SOM FOR MULTIPLE DISSIMILARITY DATA TABLES

Cluster/Class	1	2	3	4	5	6	7	8	Major Class
0,0	0	10	0	1	23	0	0	0	5
0,1	0	6	1	0	17	1	0	0	5
0,2	3	1	0	1	0	5	5	1	6/7
0,3	36	0	0	0	8	2	0	0	1
0,4	57	0	0	0	0	2	0	0	1
1,0	0	17	0	0	28	0	0	0	5
1,2	0	0	0	0	0	13	0	11	6
1,3	3	1	1	0	1	29	0	8	6
1,4	44	0	0	0	0	0	0	0	1

cluster (1,1) has no objects, it remain empty because the cluster (1,0) has the same prototype.

Table XIII
ECOLI: FINAL RELEVANCE WEIGHT MATRIX

Cluster/Matrix	1	2	3	4	5
0,0	0.36	0.50	0.54	**2.55**	**3.97**
0,1	0.49	0.75	0.55	**2.28**	**2.16**
0,2	0.97	0.89	0.78	**2.09**	0.71
0,3	0.87	**1.46**	**1.13**	0.78	0.89
0,4	**1.26**	0.90	0.63	**1.21**	**1.15**
1,0	0.50	0.65	0.64	**2.16**	**2.20**
1,1	**1.13**	0.75	0.84	**1.45**	0.96
1,2	**2.44**	0.83	0.26	**1.94**	0.98
1,3	**2.55**	0.53	0.38	**1.54**	**1.25**
1,4	0.48	0.72	0.46	**2.29**	**2.70**

The results for CR index are 0.43 and 0.37, respectively, for the adaptive approach for multiple dissimilarity tables and the batch SOM. For the F-measure the results are 0.52 and 0.52, respectively. And for the OERC, the results are 0.24 and 0.24, respectively. Both algorithms perform similarly except for the CR index, in which the adaptive approach performs better.

Table XIII gives the final relevance weight matrix. The matrix 4 has a great relevance on most clusters.

E. Wine Quality data set (wine red)

Wine quality data set consists of two datasets related to red and white variants of the Portuguese "Vinho Verde" wine [18]. Only the red variation was considered for these experiments. There are 1599 examples, each of them is described by 11 attributes and the classification attribute. This classification is based on sensory data (median of at least 3 evaluations made by wine experts). Each expert graded the wine quality between 0 (very bad) and 10 (very excellent). In this example, the classifications of the wine red are: 3, 5, 4, 7, 6 and 8.

In this experiment, we used a 1x12 topology, and the values of $T_{min} = 0.6$, $T_{max} = 6.0$ and $N_{iter} = 500$.

Table XIV
WINE QUALITY: CONFUSION MATRIX FROM B-SOM ALGORITHM

Cluster/Class	3	5	4	7	6	8	Majority Class
0,0	6	207	28	114	256	11	6
0,1	1	71	5	14	82	2	6
0,2	1	68	5	31	104	0	6
0,3	2	149	9	23	135	3	5
0,6	0	9	1	2	3	0	5
0,10	0	86	3	12	42	2	5
0,11	0	90	2	2	10	0	5

Table XV
WINE QUALITY: CONFUSION MATRIX FROM AB-SOM FOR MULTIPLE DISSIMILARITY DATA TABLES

Cluster/Class	3	5	4	7	6	8	Majority Class
0,0	5	227	29	180	430	17	6
0,3	4	426	23	9	177	1	5
0,8	0	4	0	0	2	0	5
0,9	0	4	1	0	2	0	5
0,10	0	1	0	0	2	0	6
0,11	1	19	0	9	20	0	6

Tables XIV and XV give the confusion matrices furnished, respectively by the batch SOM algorithm and the adaptive approach for multiple dissimilarity tables. In table XV, for example, class 5 is in the central region of the map.

The results for CR index are 0.46 and 0.28, respectively, for the adaptive approach for multiple dissimilarity tables and the batch SOM. For the F-measure the results are 0.54 and 0.34, respectively. And for the OERC, the results are 0.44 and 0.51, respectively. The adaptive approach for multiple dissimilarity data tables performs better than the original batch SOM for this data set.

Table XVI gives the final relevance weight matrix. The matrix 8 has a great relevance on most clusters.

VI. CONCLUSION

This paper introduced a clustering algorithm based on Self-Organizing Maps that is able to partition objects taking into account their relational descriptions given by multiple dissimilarity matrices. These matrices could have been generated using different sets of variables and dissimilarity functions.

This algorithm is an adaptive batch SOM approach for multiple dissimilarity tables. The algorithm starts from an initial partition and alternates three steps until a stop criterion is reached. In the first step, the algorithm gives the solution for the best prototype of each cluster, in the second step it is determined the solution for the best relevance weight matrix. In the last step, the algorithm gives the solution of the best partition. These iterative steps intend to optimize an adequacy criterion that measures the fit between the partition and their prototypes. The relevance weights

Table XVI
WINE QUALITY: FINAL RELEVANCE WEIGHT MATRIX

Cluster/Matrix	1	2	3	4	5	6	7	8	9	10	11
0,0	0.18	0.17	0.18	0.22	0.48	0.16	0.22	**5290164.45**	0.19	0.28	0.16
0,1	0.19	0.12	0.18	0.23	0.53	0.21	0.26	**4090951.57**	0.19	0.21	0.22
0,2	0.19	0.19	0.16	0.30	0.07	0.15	0.12	**6918621.92**	0.16	0.08	**4.16**
0,3	0.13	0.09	0.07	0.12	0.03	0.06	0.04	**4842774.34**	0.07	0.04	**7624.27**
0,4	0.06	0.04	0.03	0.06	0.01	0.03	0.02	**2269275.41**	0.03	0.02	**14870482.69**
0,5	0.07	0.06	0.05	0.08	0.02	0.05	0.04	**6311.79**	0.05	0.02	**156727397.37**
0,6	**1.08**	**1.10**	**1.08**	**1.11**	**4.03**	0.71	0.38	**1.65**	**1.21**	0.40	0.80
0,7	0.32	**1.19**	0.76	0.48	**2.02**	0.47	0.28	0.49	**1.24**	0.42	**106.08**
0,8	**2.20**	0.90	**1.27**	0.65	**11.91**	**1.67**	0.47	0.04	0.63	0.41	**5.59**
0,9	**1.49**	0.95	0.93	0.92	**11.89**	**2.86**	**1.58**	0.02	0.62	0.79	**1.63**
0,10	**1.32**	**2.05**	**2.19**	0.20	**2.87**	0.88	**1.92**	0.06	**1.73**	**1.46**	**1.15**
0,11	**1.30**	**2.10**	**2.94**	0.13	**2.11**	0.68	**2.51**	0.07	**2.25**	**1.76**	**1.00**

change at each iteration and are different from one cluster to another.

The usefulness of this algorithm was illustrated by an experimental evaluation using real-value data sets. These data sets were extracted from the UCI Machine Learning Repository. The algorithm proposed performed better than the original batch algorithm on most of the datasets.

ACKNOWLEDGMENT

The authors thank the Brazilian agencies CNPq and FACEPE for their financial support.

REFERENCES

[1] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM Comput. Surv.*, vol. 31, pp. 264–323, 1999.

[2] T. Kohonen, "The self-organizing maps," *Proceedings of the IEEE*, vol. 78, pp. 1464–1480, 1990.

[3] K. Gopalakrishnan, S. Khaitan, and A. Manik, "Enhanced clustering analysis and visualization using kohonen's self-organizing feature map networks," *International Journal of Computational Intelligence*, vol. 4, pp. 64–71, 2008.

[4] A. Golli, B. Conan-Guez, and F. Rossi, "A self-organizing map for dissimilarity data," in *Classification, Clustering, and Data Mining Applications*. Springer Berlin Heidelberg, 2004, pp. 61–68.

[5] B. Conan-Guez, F. Rossi, and A. E. Golli, "Fast algorithm and implementation of dissimilarity self-organizing maps," *Neural Networks*, vol. 19, pp. 855–863, 2006.

[6] F. de Carvalho, Y. Lechevallier, and F. de Melo, "Partitioning hard clustering algorithms based on multiple dissimilarity matrices," *Pattern Recognition*, 2011, doi: 10.1016/j.patcog.2011.05.016.

[7] H. Frigui, C. Hwang, and F. Chung-Hoon Rhee, "Clustering and aggregation of relational data with applications to image database categorization," *Pattern recognition*, vol. 40, no. 11, pp. 3053–3068, 2007.

[8] F. Murtagh and M. Hernandez-Pajares, "The kohonen self-organizing map method: An assessment," *Journal of Classification*, vol. 12, pp. 165–190, 1995.

[9] C. Ambroise and G. Govaert, "Constrained clustering and kohonen self-organizing maps," *Journal of classification*, vol. 13, no. 2, pp. 299–313, 1996.

[10] F. Badran, M. Yacoub, and S. Thiria, "Self-organizing maps and unsupervised classification," in *Neural networks: methodology and applications*. Springer-Verlag, 2005, pp. 379–442.

[11] J. A. Kangas, T. K. Kohonen, and J. T. Laaksonen, "Variants of self-organizing maps," *IEEE Transactions on Neural Networks*, 1990.

[12] N. Grozavu, Y. Bennani, and M. Lebbah, "From variable weighting to cluster characterization in topographic unsupervised learning," in *Proceedings of International Joint Conference on Neural Networks (IJCNN'09)*.

[13] E. Diday and G. Govaert, "Classification automatique avec distances adaptatives," *R.A.I.R.O. Informatique Comput. Sci.*, vol. 11, no. 4, pp. 329–349, 1977.

[14] A. Frank and A. Asuncion, "Uci machine learning repository," 2010. [Online]. Available: http://archive.ics.uci.edu/ml

[15] L. Hubert and P. Arabie, "Comparing partitions," *Journal of Classification*, vol. 2, pp. 193–218, 1985.

[16] C. J. van Rijsbergen, *Information Retrieval*. London: Butterworths, 1979.

[17] L. Breiman, J. Friedman, C. J. Stone, and R. Olshen, *Classification and Regression Trees*. Chapman and Hall/CRC, 1984.

[18] P. Cortez, A. Cerdeira, F. Almeida, T. Matos, and J. Reis, "Modeling wine preferences by data mining from physicochemical properties," *Decision Support Systems*, vol. 47, no. 4, pp. 547–553, 2009.