Constraint Selection for Semi-supervised Topological Clustering

Kais Allab and Khalid Benabdeslem

University of Lyon1, GAMA Laboratory,
43, Bd du 11 Novembre 1918, Villeurbanne 69622 France
kais.allab@etu.univ-lyon1.fr,
kbenabde@univ-lyon1.fr

Abstract. In this paper, we propose to adapt the batch version of self-organizing map (SOM) to background information in clustering task. It deals with constrained clustering with SOM in a deterministic paradigm. In this context we adapt the appropriate topological clustering to pairwise instance level constraints with the study of their informativeness and coherence properties for measuring their utility for the semi-supervised learning process. These measures will provide guidance in selecting the most useful constraint sets for the proposed algorithm. Experiments will be given over several databases for validating our approach in comparison with another constrained clustering ones.

Keywords: Constrain selection, semi-supervised clustering, SOM.

1 Introduction

Traditional clustering algorithms only access to variables which describe each data but they do not deal with any other kind of given information. Nevertheless, taking a priori knowledge into account in such algorithms, if there exists, is an important problem and a real challenge in nowadays clustering research. It concerns a recent area in machine learning and data mining research which is constrained clustering [1]. This semi-supervised approach has led to improved performance for several data sets [2,16] as well as for real-world applications, such as person identification from surveillance camera clips [3] , noun phrase coreference resolution and GPS-based map refinement [4], and landscape detection from hyperspectral data [5].

Furthermore, The last ten years have seen extensive work on incorporating instance-level constraints into clustering methods. The first work in this area proposed a modified version of COBWEB that strictly enforced pairwise constrains (COP-COBWEB) [6]. It was followed by an enhanced version of widely used k-means algorithm that could also accommodate constraints, called COP-Kmeans [4]. Moreover, in [7], an exploration of the use of instance and cluster-level constraints was performed with agglomerative hierarchical clustering. In [8] the authors proposed a new graph based constrained clustering algorithm called

COP-Bcoloring where they have shown improvements in quality and computational complexity of clustering by using constraints in a graph b-coloring clustering algorithm. Theoretically, it was proven that clustering with constraints raised an intractable feasibility problem [9,10] for simply finding any clustering that satisfies all constraints via a reduction from graph coloring [8,11,12].

Constraints can be generated from background knowledge about the data set [13] or from a subset of the data with known labels [14]. Practically, it has been illustrated that constraints can improve the results of a variety of clustering algorithms. However, there can be a large variation in this improvement, even for a fixed number of constraints for a given data set [4,10]. In fact, It has been observed that constraints can have ill effects even when they are generated directly from the data labels that are used to evaluate accuracy, so this behavior is not caused by noise or errors in the constraints. Instead, it is a result of the interaction between a given set of constraints and the algorithm being used [13].

In this paper we propose to adopt constraints in a self-organizing based clustering, which represents a prominent tool for high dimensional data analysis, since it provides a substantial data reduction that can be used for visualizing and exploring properties of data. Our first contribution is based on a deterministic model, where constraints are integrated and identified in the neurons of the corresponding topographic neural networks. In this sense, we extend the work of Kohonen [15] by adapting the batch version of Self-Organizing Map (SOM) to hard constraints. Our proposal is different from that proposed in [16], which also address the problem of semi-supervised learning in SOM. However, in that paper, the authors propose a label propagation based strategy over the training database as it is done for the classification problem [17,18,19]. In our case, we trait the problem in a different paradigm, where "Semi-supervised learning = Unsupervised learning + Constraints". In this context, we study two important measures, informativeness and coherence [13], that captures relevant properties of constraint sets. These measures provide insight into the effect a given constraint set has for a specific constrained clustering algorithm. They will be used for selecting the relevant constraints in the clustering process.

2 Integrating Constraints in SOM

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 ordering of the codebook vectors (also called prototype vectors, cluster centroids or reference vectors) in 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 (M, E). M is a set of K interconnected neurons having a discrete 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

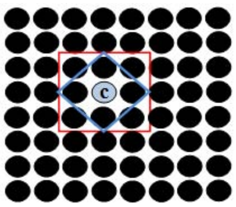

Fig. 1. Two-dimensional topological map with 1-neighborhood of a neuron c. Rectangular (red) with 8 neighbors and diamond (blue) with 4 neighbors.

between neurons (Figure 1). Each neuron c is represented by a D-dimensional reference vector $w_c = w_c^1, ..., w_c^D$, where D is equal to the dimension of the input vectors $x_i \in X$ (Data set).

The SOM training algorithm resembles K-means. The important distinction is that in addition to the best matching reference vector, its neighbors on the map are updated. The end result is that neighboring neurons on the grid correspond to neighboring regions in the input space.

2.1 Constraints

The operating assumption behind all constrained clustering methods is that the constraints provide information about the true (desired) partition, and that more information will increase the agreement between the output partition and the true partition. Constraints provide guidance about the desired partition and make it possible for clustering algorithms to increase their performance [10].

In this paper, we propose to adapt a topology based clustering to hard constraints: Must-Link constraint (***ML***): involving x_i and x_j , specifies that they must be placed into the same cluster. Cannot-Link constraint (***CL***): involving x_i and x_j , specifies that they must be placed into deferent clusters.

2.2 The Topological Constrained Algorithm

The SOM' algorithm is proposed on two versions: Stochastic (on-line) and batch (off-line) versions. In this paper, we consider the second one which is deterministic, fast and more adapted to our proposition to optimize the objective function subject to different considered constraints.

The batch version of SOM is an iterative algorithm in which the whole data set is presented to the map before any adjustments are made. In each training step, the data set is partitioned according to the Voronoi regions of the map reference vectors. More formally, we define an affectation function f from $\mathbb{R}^D$ (the input space) to C (the output space), that associates each element x_i of $\mathbb{R}^D$ to the neuron whose reference vector is "closest" to x_i. This function induces a partition $P = P_c; c = 1...K$ of the set of observations where each part P_c is defined by: $P_c = \{x_i \in X; f(x_i) = c\}$.

The quality of the partition $(P_c)_{c\in C}$ and its associated prototype vectors $(w_c)_{c\in C}$ is given by the following energy function [20]:

$$E^T((P_c)_{c\in C},(w_c)_{c\in C}) = \sum_{x_i\in X}\sum_{c\in C} h^T(\delta(f(x_i),c))\ \|w_c - x_i\|^2 \tag{1}$$

where f represents the assignment function: $f(x_i) = c$ if $x_i \in P_c$ and $h^T(.)$ is the neighborhood kernel around the winner unit c. In practice, we often use $h^T(x) = e^{-\frac{x^2}{T^2}}$ where T represents the neighborhood radius in the map. It can be fixed or decreased from an initial value T_{max} to a final value T_{min}.

In general, $f(x_i) = r$ where $r \in C$, so (1) can be rewritten as follow:

$$E^T((P_c)_{c\in C},(w_c)_{c\in C}) = \sum_{r\in C}\sum_{x_i\in P_r}\sum_{c\in C} h^T(\delta(r,c))\ \|w_c - x_i\|^2 \tag{2}$$

where P_r represents the set of elements which belong to the neuron r. Since $h^T(0) = 1$ (when $r = c$), E^T can be decomposed on two terms:

$$E_1^T = \sum_{r\in C}\sum_{x_i\in P_r} \|w_r - x_i\|^2 \tag{3}$$

$$E_2^T = \sum_{r\in C}\sum_{c\neq r}\sum_{x_i\in P_c} h^T(\delta(r,c))\,\|w_r - x_i\|^2 \tag{4}$$

E_1^Tcorresponds to the distortion used in the partitioning based clustering algorithms like K-Means. E_2^T is specific to SOM algorithm. Note, that if the neighboring relationship is not considered ($\delta(.) = 0$), optimizing the objective function subject to constraints could be resolved by the known COP-Kmeans [4].

Our first contribution consists here to adapt SOM to ML and CL constraints by minimizing the equation (1) subject to these constraints. For this optimization problem we can consider several versions proposed by [21,20,15]. All these versions proceed on tow steps: assignment step for calculating f and adaptation step for calculating w_c.

In this work, we use the version proposed by Heskes and Kappen [21]. The assignment step consists to minimize E^T with w_c fixed and the adaptation step minimizes the same objective function but with the prototypes as fixed. Although the two-optimizations are performed accurately, we can not guarantee that the energy is generally minimized by this algorithm. However, if we fixe the neighborhood structure (T is fixed), the algorithm converges towards a stable state after a finite number of steps [20].

Since the energy is a sum of independent equations, we can replace the two optimization problems by a set of equivalent simple problems. The formulation of (1) shows that the energy is constructed as the sum over all observations of a measure of adequacy of $\mathbb{R}^D \times C$ into $\mathbb{R}^+$ defined by:

$$\gamma^T(x,r) = \sum_{c\in C} h^T(\delta(r,c))\,\|w_c - x\|^2 \tag{5}$$

$$\text{which gives : } E^T((P_c)_{c\in C},(w_c)_{c\in C}) = \sum_{x_i\in X} \gamma^T(x_i, f(x_i)) \tag{6}$$

For optimizing E^T subject to ML and CL constraints, considering the prototypes (w_c)as fixed, it suffices to minimize each sum independently with the incorporation of a new procedure $V(.)$ for controlling the violation of the constraints in the assignment process:

$$f(x_i) = \begin{cases} c^* = arg\,\underset{r\in C}{Min}\gamma^T(x_i, r) & such\, V(x_i, c^*, \Omega_{ML}, \Omega_{CL})\, is\, False \\ \emptyset & otherwise \end{cases} \quad (7)$$

With : $V(x_i, c^*, \Omega_{ML}, \Omega_{CL}) =$

$$\begin{cases} True & if\,\forall x_j \in X|((x_i, x_j) \in \Omega_{ML}) \wedge (f(x_j) \neq c^*) \\ & OR\,\forall x_j \in X|((x_i, x_j) \in \Omega_{CL}) \wedge (f(x_j) = c^*) \vee (f(x_j) \in Neigh(c^*)) \\ False & Otherwise \end{cases} \quad (8)$$

where $Neigh(c^*)$ represents the set of neighborhoods of c^* in the map, Ω_{ML} is the must-link constraint set and Ω_{CL} is the cannot-link constraint set.

Similarly, when the classes $(P_c)$are fixed, the optimization of E^T can be given by minimizing the energy associated to each neuron:

$$E_c^T(w) = \sum_{x_i \in X} h^T(\delta(f(x_i), c))\, \|w - x_i\|^2 \quad (9)$$

subject to ML and CL constraints.

We can easily resolve this problem giving an unique solution as weighting mean of observations with a new defined control function $g_c(.)$:

$$w_c = \frac{\sum_{x_i \in X} h^T(\delta(f(x_i), c))\, x_i\, g_c(x_i)}{\sum_{x_i \in X} h^T(\delta(f(x_i)), c)\, g_c(x_i)} \quad (10)$$

where

$$g_c(x_i) = \begin{cases} 0 & if\,\exists x_j \in X|(f(x_j) = c) \wedge (x_i, x_j) \in \Omega_{CL} \\ 1 & otherwise \end{cases} \quad (11)$$

Equations (7) and (10) represent the modified batch version of SOM algorithm (that we call S3OM, for Semi-Supervised Self Organizing Map) with our changes in both, assignment and adaptation steps. The algorithm takes in a data set (X), a must-link constraints set (Ω_{ML}), and a cannot-link constraints set(Ω_{CL}). It returns a partition of the observations in X that satisfies all specified constraints.

The major modification is that, when updating cluster assignments, we ensure that none of the specified constraints are violated. We attempt to assign each point x_i to its closest neuron c in the map. This will succeed unless a constraint would be violated. If there is another point x_j that must be assigned to the same neuron as x_i, but that is already in some other neuron, or there is another point x_k that cannot be grouped with x_i but is already in c or in the neighboring of c, then x_i cannot be placed either in c or its neighboring.

We continue down the sorted list of neurons until we find one that can legally host x_i. Constraints are never broken; if a legal neuron cannot be found for x_i, the empty partition ($\emptyset$) is returned. Note that with the use of equation(11), each reference vector w_c is updated only by the observations $x_i \in X$ which have not CL constraints with any element x_j belonging to c.

3 Constraint Selection

Regarding to other results obtained in [3,4,14,22,23], we observed that integrating constraints generally improve the clustering performance. But sometimes, they could have ill effects even when they are generated from the data labels that are used to evaluate accuracy. So it is more important to know why do some constraint sets increase clustering accuracy while others have no effect or even decrease accuracy. For that, the authors in [13] have defined two important measures, informativeness and coherence, that capture relevant properties of constraint sets.

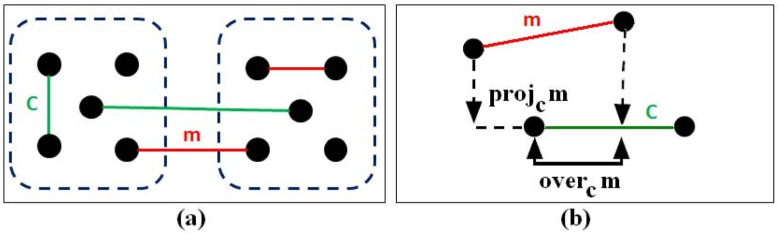

Fig. 2. Two properties of constraints (red lines for ML and green lines for CL): (a) Informatisness: m and c are informative. (b) Coherence: projected overlap between m and c ($over_c m$) is not null. So, The coherence of the subset {m,c} is null.

3.1 Informativeness

This measure represents the amount of conflict between the constraints and the underlying objective function and search bias of an algorithm. It is based on measuring the number of constraints that the clustering algorithm cannot predict using its default bias. Given a possibly incomplete set of constraints Ω and an algorithm A, we generate the partition P_A by running A on the data set without any constraints (Figure 2(a)). We then calculate the fraction of constraints in Ω that are unsatisfied by P_A:

$$I_A(\Omega) = \frac{1}{|\Omega|} \sum_{\alpha \in \Omega} unsat(\alpha, P_A) \tag{12}$$

where $unsat(\alpha, P_A)$ is 1 if P_A does not satisfy α and 0 otherwise. This approach effectively uses the constraints as a hold-out set to test how accurately the algorithm predicts them.

3.2 Coherence

This measure represents the amount of agreement between the constraints themselves, given a metric d that specifies the distance between points. It does not require knowledge of the optimal partition P^* and can be computed directly. The coherence of a constraint set is independent of the algorithm used to perform constrained clustering. One view of an $ML(x, y)$ (or $CL(x, y)$) constraint is that it imposes an attractive (or repulsive) force within the feature space along the direction of a line formed by (x, y), within the vicinity of x and y. Two constraints, one an ML constraint (m) and the other a CL constraint (c), are incoherent if they exert contradictory forces in the same vicinity. Two constraints are perfectly coherent if they are orthogonal to each other. To determine the coherence of two constraints, m and c, we compute the projected overlap of each constraint on the other as follows.

Let $\overrightarrow{m}$ and $\overrightarrow{c}$ be vectors connecting the points constrained by m and c respectively. The coherence of a given constraint set Ω is defined as a fraction of constraints pairs that have zero projected overlap (Figure 2(b)):

$$Coh_d(\Omega) = \frac{\sum_{m \in \Omega_{ML}, c \in \Omega_{CL}} \delta(over_c m = 0 \wedge over_m c = 0)}{|\Omega_{ML}| \, |\Omega_{CL}|} \tag{13}$$

where $over_c m$ represents the distance between the two projected points linked by m over c. δ is the number of the overlapped projections. Please, see [13] for more details.

From the equation (13), we can easily define a specific measure for each constraint as follows:

$$Coh_d(m) = \frac{\sum_{c \in \Omega_{CL}} \delta(over_c m = 0)}{|\Omega_{CL}|} \tag{14}$$

$$Coh_d(c) = \frac{\sum_{m \in \Omega_{ML}} \delta(over_m c = 0)}{|\Omega_{ML}|} \tag{15}$$

3.3 Hard Selection

In this section, we show how to select the relevant constraints according to their informativeness and coherence. To be selected, a constrained α_i must be (1) informative, i.e it must not be satisfied by the classical SOM (without constraints) and (2) fully coherent, i.e. it must not overlaps with any other constraint α_j ($\alpha_j \in \Omega_{CL}$ if $\alpha_i \in \Omega_{ML}$ and vis-versa). This hard fashion to select constraints can be described in Algorithm1.

3.4 Soft Selection

Hard selection is very selective and can considerably reduce the number of relevant constraints Ω_s and thus penalize the constraints having important values of coherence. In fact, in a given constraint set Ω, we could have a great difference

Algorithm 1. Hard selection

```
Input:Constraint set $\Omega = \{\alpha_i\}$
Onput:Selected constraint set $\Omega_s$
Initialize $\Omega_s = \emptyset$
for $i = 1$ to $|\Omega|$ do
  if $unsat(\alpha_i, P_{SOM}) = 1$ then
    if $Coh_d(\alpha_i) = 1$ then
      $\Omega_s = \Omega_s \cup \{\alpha_i\}$
    end if
  end if
end for
```

between the number of ML constraints and the number of CL ones. So the coherence could be favorable for the type of constraints (ML or CL) having the bigger set. For example, let Ω be a set of 4 ML constraints and 10 CL constraints. With hard selection, if a ML constraint overlaps with just one CL constraint (yet, its coherence is 0.9!) it would not be selected. We propose therefore a soft version of selection described in Algorithm 2. The aim is to know if the clustering quality is more efficient with a great number of *softly* selected constraints than with a few number of *hardly* selected constraints.

Algorithm 2. Soft selection

```
Input:Constraint set $\Omega = \{\alpha_i\}$
Onput:Selected constraint set $\Omega'_s$
Initialize $\Omega_s = \emptyset$, $\Omega'_s = \emptyset$
for $i = 1$ to $|\Omega|$ do
  if $unsat(\alpha_i, P_{SOM}) = 1$ then
    $\Omega_s = \Omega_s \cup \{\alpha_i\}$
  end if
end for
for $i = 1$ to $|\Omega_s|$ do
  if $Coh_d(\alpha_i) \geq Coh_d(\Omega_s)$ then
    $\Omega'_s = \Omega'_s \cup \{\alpha_i\}$
  end if
end for
```

4 Experimental Results

Extensive experiments were carried out over the data sets in Table 1. These Data sets are voluntarily chosen for evaluating the clustering performance of S3OM and comparing it with other state of the art techniques: COP-KMeans (CKM) [4], PC-KMeans (PKM), M-KMeans (MKM), MPC-KMeans (MPKM) [14], Cop-Bcoloring (CBC) [8], (lpSM) [16] and Belkin-Niyogi's Approach [16].

Table 1. Used Data sets

Data sets	N	D	$\#classes$	Map' dimensions	Reference
Glass	214	9	6	11 × 7	[2]
Ionosphere	351	34	2	13 × 7	[2]
Iris	150	4	3	16 × 4	[2]
Leukemia	72	1762	2	9 × 5	[24]
Chainlink	1000	3	2	18 × 9	[25]
Atom	800	3	2	14 × 10	[25]
Hepta	212	3	7	9 × 8	[25]
Lsun	400	2	3	13 × 8	[25]
Target	770	2	6	13 × 11	[25]
Tetra	400	3	4	11 × 9	[25]
TwoDiamonds	800	2	2	20 × 7	[25]
Wingnut	1070	2	2	16 × 10	[25]
EngyTime	4096	2	2	21 × 15	[25]

4.1 Evaluation of the Proposed Approach

For generating constraints from each data set, we considered randomly 20% of data equally distributed on the labels. From these labelled data, we generated the set of all *must-link* constraints (Ω_{ML}) and all *cannot-link* ones (Ω_{CL}). Next, we applied a SOM based clustering algorithm on all data set without any constraints. The obtained partition (P_{SOM}) is then used by Algorithm 1 (Algorithm 2) in informativeness study of Ω ($\Omega_{ML} \cup \Omega_{CL}$) to select the most relevent constraints. The selected constraint set (Ω'_s or Ω_s) is exploited in S3OM' algorithm.

For the construction of the S3OM' maps, we used a Principal Component Analysis (PCA) based heuristic proposed by [15] for automatically providing the initial number of neurons and the dimensions of the maps (Table 1). The reference vectors are initialized linearly along the greatest eigenvectors of the associated data set. Then, each S3OM map is clustered by an Ascendant Hierarchical Clustering (AHC) for optimizing the number of clusters (by grouping neurons) [26]. In general, an internal index, like Davies Bouldin or Generalized Dunn [27], are used for cutting the dendrogram. Here, we propose to cut it according to the violation of constraints, i.e., we select the small number of classes which does not violate the constraints especially the CL ones. For the evaluation of the accuracy of S3OM algorithm, we propose to use the Rand index [28].

Table 2 compares the results (averaged over 1000 trials) for each algorithm in terms of its unconstrained and constrained performance, when provided with 25 randomly selected constraints. We added our obtained results to those shown in [4,13,8]. The best result for each algorithm/data set combination is in bold.

In one hand, S3OM shows that integrating constraints in SOM model provides a clear improvement to clustering accuracy. In other hand, the results obtained by S3OM are similar and sometimes better than another known constrained clustering methods. The most important remark is that with our constrained

Table 2. Average Rand Index of S3OM vs five constrained clustering algorithms (for 1000 trials). Firstly without constraints (Unc) and secondly, with 25 randomly selected constraints (Con).

Data	CKM	MKM	PKM	MPKM	CBC	**S3OM**
	Unc/Con	Unc/Con	Unc/Con	Unc/Con	Unc/Con	Unc/ Con
Glass	69.0 **69.4**	39.5 **56.6**	43.4 **68.8**	39.5 **67.8**	66.0 **67.4**	64.3 **68.4**(±2.1)
Iono.	58.6 **58.7**	58.8 **58.9**	58.8 **58.9**	58.9 **58.9**	56.0 **58.8**	50.5 **61.3**(±8.0)
Iris	84.7 **87.8**	88.0 **93.6**	84.3 **88.3**	88.0 **91.8**	82.5 **87.2**	91.2 **92.2**(±3.8)

algorithm the clustering performance often increases significantly with a few number of constraints comparing to another ones. For example, in Table 2, for Glass, S3OM (68.4%) is not better than CKM (69.4%) but S3OM yields an improvement of 4.1% over the baseline and CKM achieves just 0.4% increase in accuracy. Thanks to topological property and neighborhood relation which allowed us to separate two points related by a CL constraint by putting them in two distant clusters (neurons), unlike in the other constrained algorithms in which the assignment is done in the closest cluster which don't violate the constraints.

4.2 Results of Constraint Selection

In Table 2, we have seen that in general, constraints improve clustering performance. However, these constraints were generated randomly, so there exist some ones which are not informative and some ones which are not coherent.

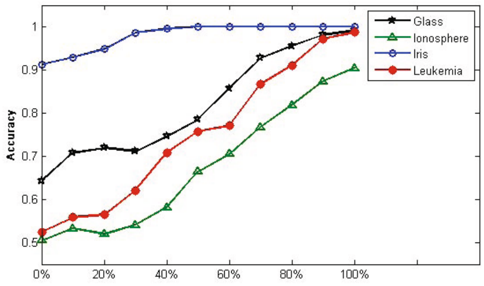

Fig. 3. Results of S3OM according to coherence rate, with fully informative constraints

To understand how this constraint set properties affect our constrained algorithm, we performed the following experiment. We measured the accuracy according to random informative constraint sets but with different rate of coherence (calculated by (13)). We can see that all databases exhibited important increases of accuracy when coherence increases in a set of randomly selected constraints (Figure 3). In fact, for all data sets, the accuracy increases steadily and quickly when integrating "good" background information.

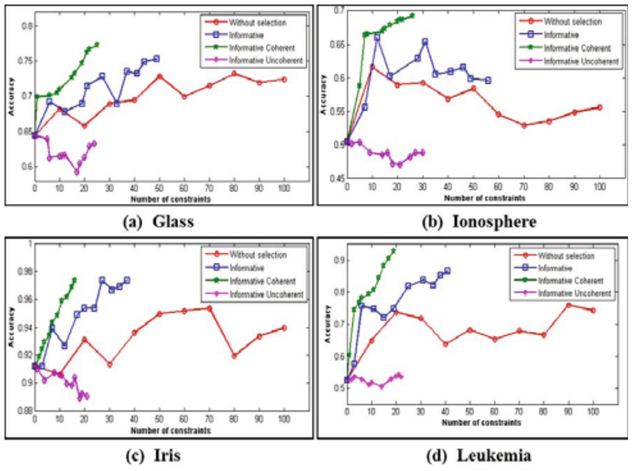

Fig. 4. Performance of S3OM according to both informativeness and coherence

Using randomly selected constraints could decreases clustering performance (Red curves in Figure 4). Thus, for each data set, we selected the informative ones (using Eq. 12). The obtained blue curves are better than the red ones, but they contain some decreasing. For that, we divided the informative constraint sets into two subsets: coherent and incoherent ones. In Figure 4, the constraints are either fully coherent or fully incoherent in each considered subset. The blue curves (mixed constraints) are clearly situated between the green ones (with coherent constraints) and the pink ones (with incoherent constraints) for all databases. For example, for Glass, S3OM achieves an accuracy of 64.3% without any constraint. Overall accuracy increases with the incorporation of constraints, reaching 72.4% after 100 random constraints. However, the performance is better when using just the 49 informative constraints (75.3%) and it is even better when using just the 25 informative and coherent constraints (77.2%) and weak when using the 24 other non coherent ones (63.3%). Remark that for Iris, the accuracy achieves 97.4% with 37 informative constraints (with some incoherent constraints) but when dealing with the coherence ones, it increases quickly and achieves the same accuracy using less number (16) of constraints. That means that uninformative and incoherent constraints could be sometimes dramatic for clustering performance.

4.3 Results of Selection on FCPS Data Sets

In this section, we applied S3OM on FCPS (Fundamental Clustering Problem Suite) data sets [25] using 100 softly selected constraints. Then we compared the obtained results (averaged over 100 trials) to those shown in [16].

Table 3 shows that S3OM can find the correct partition of Atom, Chainlink, Hepta, Lsun and Tetra data sets and realizes similar results than lpSM' approach

Table 3. Average Rand Index of S3OM vs two semi-supervised learning algorithms (for 100 trials) with some FCPS data sets. Firstly without constraints (Unc) and secondly, with 100 softly selected constraints (Con).

Data sets	Belkin and Niyogi	lpSM	**S3OM**	
			Unc	Con
Atom	93	100	63.87	**100**
Chainlink	78	100	57.72	**100**
Hepta	100	100	100	**100**
Lsun	83	100	90.87	**100**
Target	53	87	81.89	**90.6**(±3.4)
Tetra	100	100	96.56	**100**
TwoDiamonds	75	95	90.04	**97.4**(±2.5)
Wingnut	70	90	95.02	**99.2**(±0.8)
EngyTime	57	96	61.21	**96.8**(±3.1)

(Accuracy = 100 %) [16]. We can also remark that S3OM realizes better performance than those of the other methods and confirms the contribution of the selected constraints in the improvement of the quality of the clustering model. For example, for EngyTime, S3OM without constraints realizes an accuracy of 61.21% without any constraint. By using the 100 softly selected constraints, S3OM can reach an optimal averaged accuracy of 96.8%.

4.4 Results of Selection on Leukemia Data Set

The microarray "Leukemia" data is constituted of 72 samples, corresponding to two types of Leukemia: 47 ALL (Acute Lymphocytic Leukemia) and 25 AML (Acute Myelogenous Leukemia). The data set contains expressions for 7129 genes. In [24] it was suggested to remove the *affymetrix* control genes and all genes with an expression below to 20 (too low to be interpreted with confidence). Finally 1762 genes are kept. We tested thus the hard selection and the soft one over the obtained (72×1762) data set. Having 72 labeled samples, we generated the maximum number of constraints (2556: 1381 ML and 1175 CL). Using informativeness, we reduced this number to 1215 (540 ML and 675 CL). We found only 3 coherent constraints using hard selection, achieving an accuracy of 54.3%, yielding an improvement of 2% over the baseline! Thus, we wanted to know if the accuracy could be significantly improved if we use soft selection.

First, we measured accuracy for 1000 trials with randomly selected constraints with different sets of ML and CL. Their sizes vary between 5 and 37. We show in Table 4 the results when the constraints are selected with hard selection and when they are selected with soft selection. From this table, we can see that in general, Hard selection is more efficient than the soft one, which is natural when using the constraints with perfect coherence. However, the 3^{rd} and 5^{th} lines of the table show the opposite, especially when the gap between the CL and the ML is important.

Table 4. Performance of S3OM over "Leukemia" data set with hard selected constraints vs soft selected ones

	Hard			Soft	
#ML	#CL	Acc	#ML	#CL	Acc
1	4	62.0	2	7	55.9
5	5	70.9	6	9	66.5
11	**4**	**75.7**	**13**	**7**	**77.0**
11	9	86.7	14	17	84.7
8	**17**	**89.6**	**11**	**21**	**91.1**
14	16	97.2	16	21	88.6

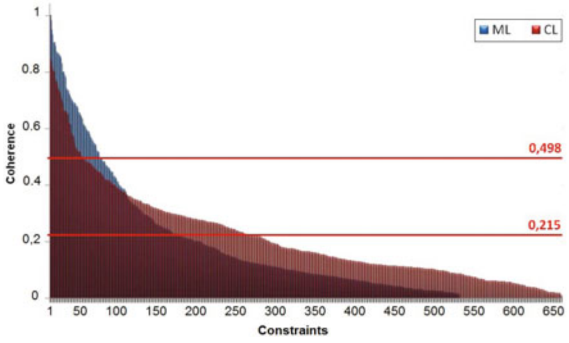

Fig. 5. Ranking constraints according to coherence using soft selection

Second, we sorted the obtained set of 1215 informative constraints according to their coherence. The global coherence of this set was equal to 0.215, which allowed us to obtain 453 (176 ML and 277 CL) constraints softly selected (37% from the informative constraint set). It corresponds to the first cut in Figure 5.

Finally, we studied the behavior of the S3OM quality according to different percents of this constraint set (between 5% and 37%), in order to find the optimal number of coherent constraints which realizes the best accuracy, in Figure 6. The different numbers in this figure represents the number of coherent constraints corresponding to the different percents.

Subsequently, we obtained a perfect accuracy with an optimal number of coherent constraints (111 constraints) which corresponds to 9.1% of the informative data set. Note that the coherence of the 111^{th} coherent constraint is 0.498, that corresponds to the second cut in Figure 5. Thus, in this case, a constraint is selected if its coherence is greater than or equal to 0.498.

4.5 Visualization

In this section, we present some visualization inspections of "Chainlink" data set which represents an important problem for data structure. It describes two chains tied in 3D space. The aim is to see if our proposals are able to disentangle the two clusters and represent the real structure of data.

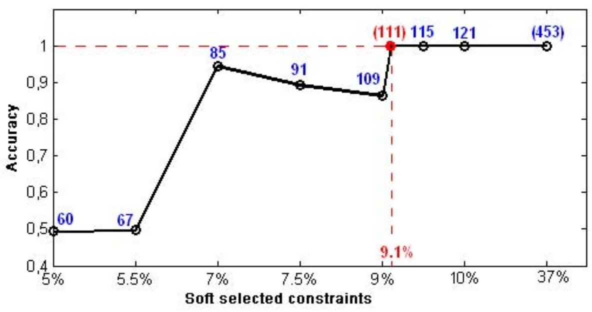

Fig. 6. Performance of S3OM according to soft selected constraints

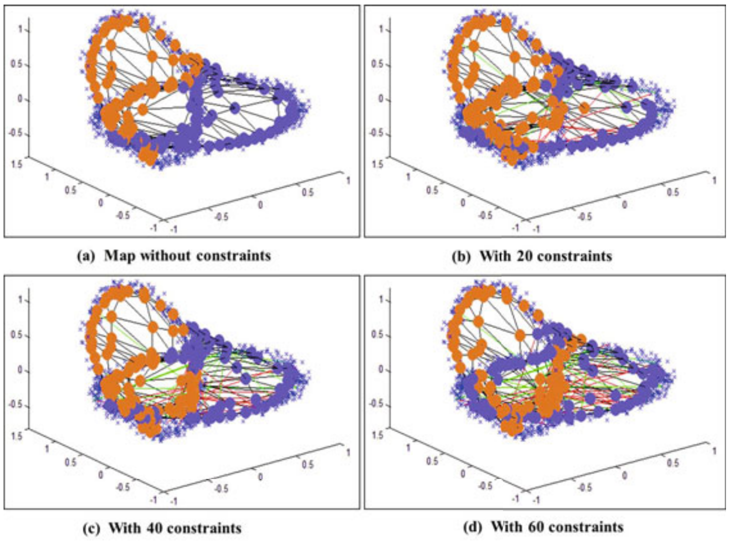

Fig. 7. Behaviors of Chainlink maps done by S3OM. Orange neurons represent the first chain (Majorelle Blue neurons for the second one). Green lines for ML constraints (Red lines for CL).

Figure 7 shows the representation of the topological projection of the map's neurons in "Chainlink" data space. Firstly, without constraints and secondly with 20, 40 and 60 constraints. Note that this projection is done in the $3D$ plan according to the 3 dimensions of data. We can see that we always obtain a partition with 2 classes which corresponds to the correct number of the real partition. However, without any constraints the algorithm (standard SOM) is not able to completely find the structure of data and more we add constraints more the algorithm disentangle the two chains. The best structure is obtained with 60 informative and coherent constraints and the map is well organized.

5 Conclusion

In this work, the contributions are two-fold. First, a new algorithm S3OM was developed for semi-supervised clustering by considering instance level constraints in the objective function optimization of batch version of SOM. The deterministic aspect of this algorithm allowed us to perform hard constraints satisfaction in a topological clustering. Second, we studied a constraint set properties, informativeness and coherence, that provided a quantitative basis for explaining why a given constraint set increases or decreases performance. These measures was used for selecting the most useful constraints for clustering. The constraint selection was done in both, hard and soft fashions.

References

1. Basu, S., Davidson, I., Wagstaff, K.: Constrained clustering: Advances in algorithms, theory and applications. Chapman and Hall/CRC Data Mining and Knowledge Discovery Series (2008)
2. Frank, A., Asuncion, A.: Uci machine learning repository. Technical report, University of California (2010)
3. Bar-Hillel, A., Hertz, T., Shental, N., Weinshall, D.: Learning a mahalanobis metric from equivalence constraints. Journal of Machine Learning Research 6, 937–965 (2005)
4. Wagstaff, K., Cardie, C., Rogers, S., Schroedl, S.: Clustering with instance level constraints. In: Proc. of the 18th International Conference on Machine Learning, pp. 577–584 (2001)
5. Lu, Z., Leen, T.K.: Semi-supervised learning with penalized probabilistic clustering. In: Advances in Neural information Processing Systems 17 (2005)
6. Wagstaff, K., Cardie, C., Rogers, S., Schroedl, S.: Clustering with instance level constraints. In: Proc. of the 17th International Conference on Machine Learning, pp. 1103–1110 (2000)
7. Davidson, I., Ravi, S.S.: Agglomerative hierarchical clustering with constraints: theorical and empirical results. In: Proc. of ECML/PKDD, pp. 59–70 (2005)
8. Elghazel, H., Benabdeslem, K., Dussauchoy, A.: Constrained graph b-coloring based clustering approach. In: Song, I.-Y., Eder, J., Nguyen, T.M. (eds.) DaWaK 2007. LNCS, vol. 4654, pp. 262–271. Springer, Heidelberg (2007)
9. Davidson, I., Ravi, S.S.: The complexity of non-hierarchical clustering with instance and cluster level constraints. Data Mining and Knowledge Discovery 61, 14–25 (2007)
10. Davidson, I., Ravi, S.S.: Clustering with constraints: feasibility issues and the k-means algorithm. In: Proc. of the SIAM International Conference on Data Mining, pp. 138–149 (2005)
11. Kulis, B., Basu, S., Dhillon, I., Mooney, R.: Semi-supervised graph clustering, a kernel approach. In: Proc. of the 22th International Conference on Machine Learning, pp. 577–584 (2005)
12. Davidson, I., Ester, M., Ravi, S.S.: Efficient incremental clustering with constraints. In: Proc. of 13th ACM Knowledge Discovery and Data Mining (2007)
13. Davidson, I., Wagstaff, K., Basu, S.: Measuring constraint-set utility for partitional clustering algorithms. In: Proc. of ECML/PKDD (2006)

14. Bilenko, M., Basu, S., Mooney, R.J.: Integrating constraints and metric learning in semi-supervised clustering. In: Proc. of the 21th International Conference on Machine Learning, pp. 11–18 (2004)
15. Kohonen, T.: Self organizing Map. Springer, Berlin (2001)
16. Herrmann, L., Ultsch, A.: Label propagation for semi-supervised learning in self-organizing maps. In: Proc. of the 6th WSOM (2007)
17. Belkin, M., Niyogi, P.: Using manifold structure for partially labelled classification. In: Proc. of Advances in Neural Information Processing Systems (2003)
18. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: Proc. of COLT: Proc. of the Workshop on Computational Learning Theory, pp. 92–100 (1998)
19. Chapelle, O., Scholkopf, B., Zien, A.: Semi-supervised learning. The MIT Press, Cambridge (2006)
20. Cheng, Y.: Convergence and ordering of kohonen's batch map. Neural Computation 9(8), 1667–1676 (1997)
21. Heskes, T., Kappen, B.: Error potentials for self-organization. In: Proc. of IEEE International Conference on Neural Networks, pp. 1219–1223 (1993)
22. Xing, E.P., Ng, A.Y., Jordan, M.I., Russel, S.: Distance metric learning, with application to clustering with side-information. Advances in Neural Information Processing Systems 15, 505–512 (2003)
23. Klein, D., Kamvar, S.D., Manning, C.D.: From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering. In: Proc. of the 19th International Conference on Machine Learning, pp. 307–313 (2002)
24. Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J.P., Coller, H., Loh, M., Downing, L.R., Caligiuri, M.A., Bloomfield, C.D., Lander, E.S.: Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. Science 15 286(5439), 531–537 (1999)
25. Ultsch, A.: Fundamental clustering problems suite (fcps). Technical report, University of Marburg (2005)
26. Vesanto, J., Alhoniemi, E.: Clustering of the self organizing map. IEEE Transactions on Neural Networks 11(3), 586–600 (2000)
27. Kalyani, M., Sushmita, M.: Clustering and its validation in a symbolic framework. Pattern Recognition Letters 24(14), 2367–2376 (2003)
28. Rand, W.M.: Objective criteria for the evaluation of clustering method. Journal of the American Statistical Association 66, 846–850 (1971)