From Horizontal to Vertical Collaborative Clustering using Generative Topographic Maps

Jérémie Sublime[1*], Nistor Grozavu[2], Guénaël Cabanes[2], Younès Bennani[2] and Antoine Cornuéjols[1]

[1]AgroParisTech, INRA UMR MIA 518, 16 rue Claude Bernard, 75231 Paris, France
[2]LIPN UMR CNRS 7030, 99 av. J-B Clément, 93430 Villetaneuse, France

[*]Corresponding author: jeremie.sublime@agroparistech.fr

Abstract

Collaborative clustering is a recent field of Machine Learning that shows similarities with both ensemble learning and transfer learning. Using a two-step approach where different clustering algorithms first process data individually and then exchange their information and results with a goal of mutual improvement, collaborative clustering has shown promising performances when trying to have several algorithms working on the same data. However the field is still lagging behind when it comes to transfer learning where several algorithms are working on different data with similar clusters and the same features.

In this article, we propose an original method where we combine the topological structure of the Generative Topographic Mapping (GTM) algorithm and take advantage of it to transfer information between collaborating algorithms working on different data sets featuring similar distributions.

The proposed approach has been validated on several data sets, and the experimental results have shown very promising performances.

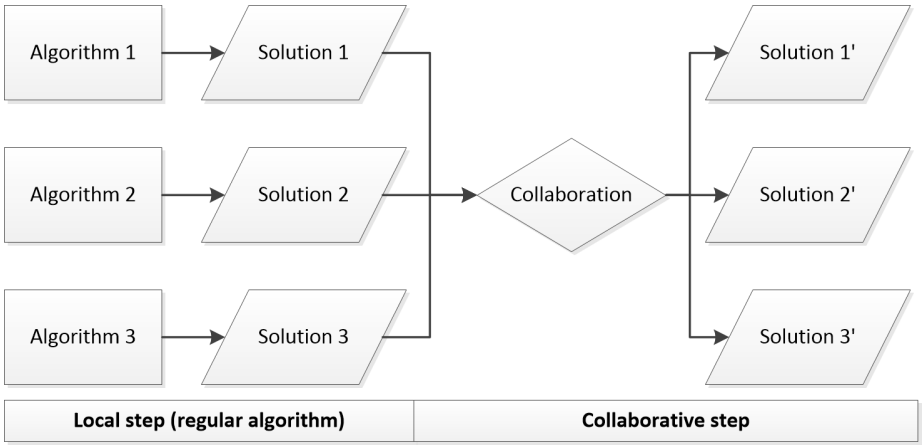

Figure 1: Collaborative clustering

1 Introduction

Data clustering is a difficult task which aims at finding the intrinsic structures of a data set. The goal is to discover groups of similar elements among the data [10]. However, the number and the size of data sets currently expend at an unprecedented rate, increasing difficult for individual clustering algorithms to achieve good performances in a reasonable amount of time. There are two main reasons to explain these difficulties: 1) Finding a satisfying clustering often requires to try several algorithms with different parameter configurations. 2) Regardless of the results' quality, processing huge data sets is time consuming, and there are very few tools to transfer and re-use already mined information from one problem to another with the goal of making the process faster.

Given this context, collaborative clustering is a recent and promising new field with few works available in the literature (e.g. [15, 4, 6, 22]) that offers several solutions for these specific issues. While most of distributed clustering techniques [17, 16] try to obtain a consensus result based on all algorithms' models and solutions, the fundamental concept of collaboration is that the clustering algorithms operate locally (namely, on individual data sets) but collaborate by exchanging information [12]. In short, the goal of collaborative clustering is to have all algorithms improving their results. Most collaborative methods follow a 2-step framework [12] (See Figure 1):

- **Local Step**: First, the algorithms operate locally on the data they have access to. At the end of this step, each algorithm proposes a solution vector, i.e. a vector of cluster labels, one for each data point.
- **Collaborative Step**: Then, the algorithms exchange their information. The information received form the collaborators is used to confirm or improve each local model. Depending on the collaborative method, this step may use different ways of exchanging the information: votes, confusion matrices, prototypes, etc. At the end of the collaborative step, ideally, all solution vectors have been improved based on the shared knowledge.

Depending on the data sets on which the algorithms collaborate, there are three main types of collaboration: Horizontal, Vertical and Hybrid collaboration. The defi-

nitions of Horizontal and Vertical Collaboration have been formalized in earlier works [14, 8]. The Hybrid Collaboration is a combination of both Horizontal and Vertical Collaboration. Both subcategories can be seen as a constrained forms of transfer learning:

- **Horizontal Collaboration**: Several algorithms analyze different representations of the same observations. It can be applied to multi-view clustering, multi-expert clustering, clustering of high dimensional data, or multi-scale clustering, see Figure 2.
- **Vertical Collaboration**: Several algorithms analyze different data sets sharing the same descriptors and having similar data distributions. The Collaborators are therefore looking for similar clusters, see Figure 3. This is equivalent to knowledge transfer in identical feature spaces and can also be applied to process large data sets by splitting them and processing each subset with different algorithms exchanging information.

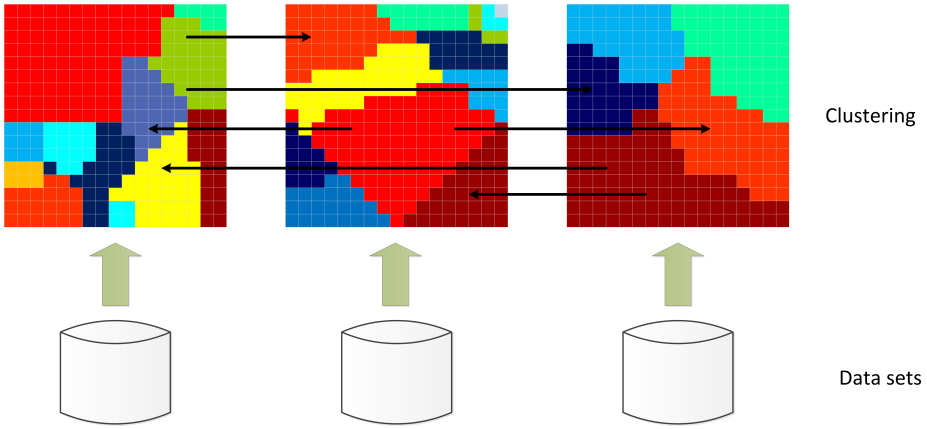

Figure 2: Horizontal clustering principle

In this article we propose to to adapt an horizontal collaboration framework [18] for vertical collaboration purposes. The new method is based on the neural network structure of the Generative Topographic Maps (GTM) [1]. By combining both, we are able to turn our originally horizontal collaboration method into a new and robust vertical collaboration framework. This article is an extension from an original work presented at the 7th International Conference on Soft Computing and Pattern Recognition [19]. This extended version includes some extra theoretical background as well as additional experiments.

The remainder of this article is organized as follows: Section 2 presents recent works on collaborative clustering and explains how they compare to our proposed method. In section 3 we introduce the horizontal collaborative framework. In section 4, we detail the GTM algorithm and we explain how it is combined with the horizontal collaborative framework to achieve a vertical collaboration. Section 5 present the results a set of experiments to assess the performances of our proposed method. Finally, this article ends with a conclusion and a few insights on potential extensions of this work.

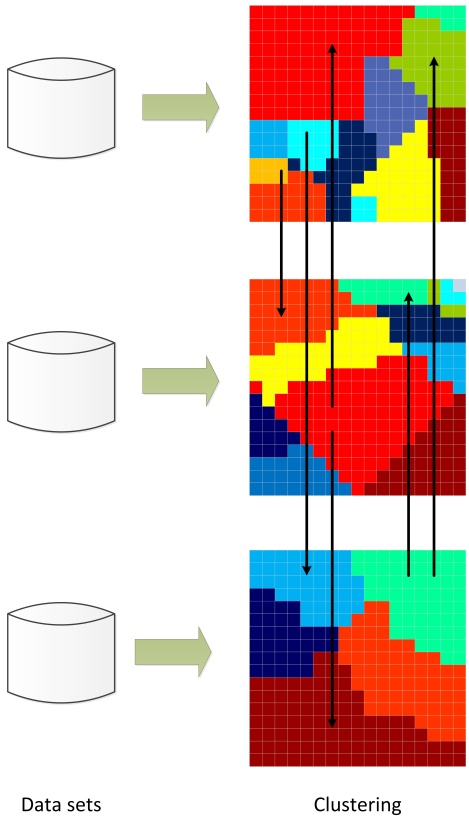

Figure 3: Vertical clustering principle

2 Collaborative Clustering

In this section we shortly describe the closest and most recent related works in the literature:

- The Collaborative Clustering using Heterogeneous Algorithms framework [18]. This framework enables horizontal collaboration as well as reinforcement learning, and is based on the EM algorithm [3]. We use this framework as a tool to build the proposed method of this article.
- The SAMARAH multi-agent system [5]. This framework enable collaboration and consensus for horizontal clustering only, and is based on a complex conflict resolution algorithm that uses probabilistic confusion matrices.
- Fuzzy Collaborative Clustering introduced by Pedrycz using the fuzzy c-means algorithm. The objective function governing the search for the structure in this

case is the following:

$$Q[ii] = \sum_{k=1}^{N[ii]} \sum_{i=1}^{c} u_{ik}^2[ii] d_{ik}^2[ii] + \sum_{jj=1}^{P} \beta[ii, jj] \sum_{i=1}^{c} \sum_{k=1}^{N[ii]} u_{ik}^2 ||v_i[ii] - v_j[jj]||^2$$

where $\beta[ii, jj]$ is a collaboration coefficient supporting an impact of the jj^{th} data set and affecting the structure to be determined in the ii^{th} data set. The number of patterns in the ii^{th} data set is denoted by $N[ii]$. $U[ii]$ and $v[ii]$ denote the partition matrix and the i^{th} prototype produced by the clustering realized for the ii-set of data.

- Two prototype-based collaborative clustering Frameworks have been proposed by Ghassany et al. [6], Grozavu N. and Bennani Y. [9]. These methods have been inspired from the works of Pedrycz et al. [12, 13] on the c-means collaborative clustering. Both these prototype-based approaches can be used for either horizontal or vertical collaboration. It is a derivative method modifying the original SOM algorithm [11]. Since this approach is the closest from ours, the results from both methods are compared in the experiments.

  For the SOM_{col} method [9], the classical SOM objective function is modified in order to take into account the distant neighborhood function $\mathcal{K}_{ij}$ as follows:

$$R_V^{[ii]}(\chi, w) = \sum_{jj \neq ii}^{P} \alpha_{[ii]}^{[jj]} \sum_{i=1}^{N} \sum_{j=1}^{|w|} \mathcal{K}_{\sigma(j,\chi(x_i))}^{[ii]} \|x_i^{[ii]} - w_j^{[ii]}\|^2$$

$$+ \sum_{jj=1, jj \neq ii}^{P} \beta_{[ii]}^{[jj]} \sum_{i=1}^{N^{[ii]}} \sum_{j=1}^{|w|} K_{ij} D_{ij} \tag{1}$$

$$\mathcal{K}_{ij} = \left(\mathcal{K}_{\sigma(j,\chi(x_i))}^{[ii]} - \mathcal{K}_{\sigma(j,\chi(x_i))}^{[jj]} \right)^2$$

$$D_{ij} = \|w_j^{[ii]} - w_j^{[jj]}\|^2$$

  where P represents the number of datasets, N - the number of observations of the ii-th dataset, $|w|$ is the number of prototype vectors from the ii SOM map and which is equal for all the maps.

  For collaborative GTMs [6], we penalize the complete log-likelihood of the M-step, basing on [7], considering the term of penalization as a collaboration term.

One disadvantage of the last two prototype-based collaborative approaches is that they require to fix a collaborative (confidence) parameter which define the importance of the distant clustering. In the case of unsupervised learning there is no available information on the clusters quality and this parameter is often tricky to choose, which is a problem since the final results are very dependent from the confidence parameter. One of the advantages of the method proposed in this article is that it does not require a confidence parameter. Indeed, it benefits from self-regulation by diminishing the influence of solutions with high diversity compared with the other collaborators.

3 Horizontal collaborative clustering with heterogeneous algorithms

In an earlier work, we have proposed a collaborative framework that allows different algorithms working on the same data elements [18] to collaborate and mutually improve their results. This algorithm is described in the subsection thereafter.

3.1 Algorithm

Let us consider a group of clustering algorithms $C = \{c_1, ..., c_J\}$, which we independently apply to our data set (observations) $X = \{x_1, ..., x_N\}, x_i \in \mathbb{R}^d$ resulting in the solution vectors $S = \{S^1, S^2, ...S^J\}$, where S^i is the solution vector provided by a given clustering algorithm c_i searching for K_i clusters. A solution vector contains for each data element the label of the cluster it belongs to. $s^i_n \in [1..K_i]$ is the id of the cluster that algorithm c_i associates to the n^{th} element of X (i.e. x_n). We also note $\boldsymbol{\theta} = \{\boldsymbol{\theta}^1, \boldsymbol{\theta}^2, ..., \boldsymbol{\theta}^J\}$ the parameters of the different algorithms (for example the mean-values and co-variances of the clusters).

The main idea is to consider that each algorithm involved in the collaborative process can be transformed into an optimization problem where we try to optimize an equation similar to Equation (2), with $p(S^i|X, \boldsymbol{\theta}^i)$ that will be a density function representing the observed algorithm depending on its parameters $\boldsymbol{\theta}^i$, and $P(S^i)$ the *a posteriori* probability of the solution vector S^i.

$$\tilde{S^i} = \arg\max_{S^i}(p(S^i|X, \boldsymbol{\theta}^i)) = \arg\max_{S^i}(p(X|S^i, \boldsymbol{\theta}^i) \times P(S^i)) \tag{2}$$

This Equation corresponds to the maximization of a likelihood function. Let us consider the density function $f(x|\boldsymbol{\theta}^i)$ with $\boldsymbol{\theta^i} \in \Omega$ the parameters to be estimated. Then, Equation (2) can be rewritten into Equation (3) where $f(X|\boldsymbol{\theta}^i)_{|S^i}$ depends on the current solution vector S^i as defined in Equation (4).

$$\tilde{S^i} = \arg\max_{S^i}(p(S^i|X, \boldsymbol{\theta}^i)) = \arg\max_{S^i}(f(X|\boldsymbol{\theta}^i)_{|S^i} \times P(S^i)) \tag{3}$$

$$f(X|\boldsymbol{\theta}^i)_{|S^i} = \prod_{n=1}^{N} f(x_n|\boldsymbol{\theta}^i_{s_n}) \tag{4}$$

Since all our collaborating algorithms may not be looking at the same representations of the data, we have a sample abstract space $\mathcal{X}$ and several sampling spaces of observations $\mathcal{Y}_i, i \in [1 \ldots, J]$. Instead of observing the "complete data" $X \in \mathcal{X}$, each algorithm may observe and process sets of "incomplete data" $y = y_i(x) \in \mathcal{Y}_i, i \in [1 \ldots J]$.

For a fixed i, let g_i be the density function of such $y = y_i(x)$, given by:

$$g_i(Y|\boldsymbol{\theta}^i) = \int_{\mathcal{X}(y)} f(x|\boldsymbol{\theta^i})dx \tag{5}$$

with $\mathcal{X}(y) = \{x; y_i(x) = y\}, i \in [1 \ldots J]$.

Using these notations, the problem that we are trying to solve in our collaborative framework is shown in Equation (6) which is the collaborative version of Equation (3). The last term $P(S^i|S)$ is extremely important since it highlights our objective of

taking into account the solutions vectors $S = \{S^1, S^2, ...S^J\}$ proposed by the other algorithms in order to weight the probability of a local solution S^i.

$$\tilde{S^i} = \arg\max_{S^i}(p(S^i|\boldsymbol{\theta}^i, Y, S)) = \arg\max_{S^i}(g_i(Y|\boldsymbol{\theta}^i)_{|S^i} \times P(S^i|S)) \tag{6}$$

This equation can be developed as follows:

$$g_i(Y|\boldsymbol{\theta}^i)_{|S^i} \times P(S^i|S) = \prod_{n=1}^{N} g_i(y_n|\boldsymbol{\theta}^{\boldsymbol{i}}_{\boldsymbol{s_n}}) \times P(s^i_n|S) \tag{7}$$

Solving the latest equation requires to compute the probability $P(s^i_n|S), \forall n \in N$. To do so, we need to map the clusters proposed by the different collaborating algorithms.

To this end, let $\Psi^{i\to j}$ be the *Probabilistic Correspondence Matrix* (PCM) mapping the clusters from an algorithm c_i to the clusters of an algorithm c_j. Likewise $\Psi^{i\to j}_{a,b}$ is the probability of having a data element being put in the cluster b of clustering algorithm c_j if it is in the cluster a of algorithm c_i. These PCM matrices can easily be computed from the solution vectors of the different collaborators using Equation (8), where $|S^i_a \cap S^j_b|$ is the number of data elements belonging to the cluster a of algorithm c_i and to the cluster b of algorithm c_j, and $|S^i_a|$ is the total number of data elements belonging the the cluster a of algorithm c_i. This equation can easily be modified for fuzzy algorithms.

$$\Psi^{i\to j}_{a,b} = \frac{|S^i_a \cap S^j_b|}{|S^i_a|}, \qquad 0 \leq \Psi^{i\to j}_{a,b} \leq 1 \tag{8}$$

While the solution vectors coupled with the PCM matrices may be enough to build a consensus Framework as they did in [5], it is not enough to have a collaborative framework that benefits all collaborators. Doing so would require the local models of each clustering algorithm to be able to use these solution vectors and matrices to improve themselves.

Under the hypothesis that all clustering algorithms are independent from each other, for a given algorithm c_i the right term of Equation (7) can then be developed as shown in Equation (9) where Z is a normalization constant, and $\Psi^{j\to i}_{s_n}$ (shorter version for $\Psi^{j\to i}_{s^j_n, s^i_n}$) the element of the matrix $\Psi^{j\to i}$ that links the clusters associated to the data element x_n by algorithms c_j and c_i.

$$g_i(y_n|\boldsymbol{\theta}^{\boldsymbol{i}}_{\boldsymbol{s_n}}) \times P(s^i_n|S) = \frac{1}{Z} g_i(y_n|\boldsymbol{\theta}^{\boldsymbol{i}}_{\boldsymbol{s_n}}) \times \prod_{j\neq i} \Psi^{j\to i}_{s_n} \tag{9}$$

In practice, we propose to transform any clustering algorithm into its collaborative version by translating it into the same model than shown in Equations (6) and (9).

Equation (9) can then be optimized locally for each algorithm using a modified version of the Expectation Maximization algorithm [3]. This modified algorithm as well as the complete process of our proposed framework is explained in Algorithm (1). As one can see, since one version of this algorithm runs for each algorithm simultaneously, our framework can easily be parallelized.

For two algorithms c_i and c_j, let $H_{i,j}$ be the *normalized confusion entropy* [20, 21] linked to the matrix $\Psi^{i\to j}$ having K_i lines and K_j columns. $H_{i,j}$ is then computed on the lines of $\Psi^{i\to j}$ according to Equation (10).

Algorithm 1: Probabilistic Collaborative Clustering Guided by Diversity: General Framework

Local step:
forall the *clustering algorithms* **do**
  Apply the regular clustering algorithm on the data Y.
  $\rightarrow$ Learn the local parameters $\boldsymbol{\theta}$
end
Compute all $\Psi^{i \rightarrow j}$ matrices
Collaborative step:
while *the system global entropy* $\mathcal{H}$ *is not stable* **do**
  forall the *clustering algorithms* c_i **do**
    forall the $y_n \in Y$ **do**
      Find s_n^i that maximizes Equation (9).
    end
  end
  Update the solution vectors S
  Update the local parameters $\boldsymbol{\theta}$
  Update all $\Psi^{i \rightarrow j}$ matrices
end

$$H_{i,j} = \frac{-1}{K_i \times \ln(K_j)} \sum_{l=1}^{K_i} \sum_{m=1}^{K_j} \Psi_{l,m}^{i \rightarrow j} \ln(\Psi_{l,m}^{i \rightarrow j}) \tag{10}$$

H is a square matrix of size $J \times J$ where J is the number of collaborators. It has null diagonal values. Since the entropies are oriented, the matrix is not properly symmetrical, albeit it exhibits symmetrical similarities. The stopping criterion for this algorithm is based on the global entropy $\mathcal{H}$ which is computed using Equation (11). When $\mathcal{H}$ stops changing between two iterations, the collaborative process stops.

$$\mathcal{H} = \sum_{i \neq j} H_{i,j} = \sum_{i=1}^{J} \sum_{j \neq i} \frac{-1}{K_i \times \ln(K_j)} \sum_{l=1}^{K_i} \sum_{m=1}^{K_j} \Psi_{l,m}^{i \rightarrow j} \ln(\Psi_{l,m}^{i \rightarrow j}) \tag{11}$$

3.2 Adaptation to vertical collaboration

Since the previously introduced algorithm showed good performances for horizontal collaboration applications, our goal was to try to modify this algorithm for transfer learning purposes.

Doing so would require to get rid of the constraint that with this Framework all algorithms must work on the same data, even if they have access to different feature spaces. Instead, what we wanted was to have several algorithms working on different data sets in the same feature spaces and looking for similar clusters.

Unfortunately, modifying the original Framework and its mathematical model to adapt them to this new context proved to be too difficult. Instead of working on a new Framework for vertical collaboration from scratch, we though of a clever way to tweak the original framework by using the properties of unsupervised neural networks based on vector quantization, such as the Self-organizing maps (SOM) [11], or the GTM [1].

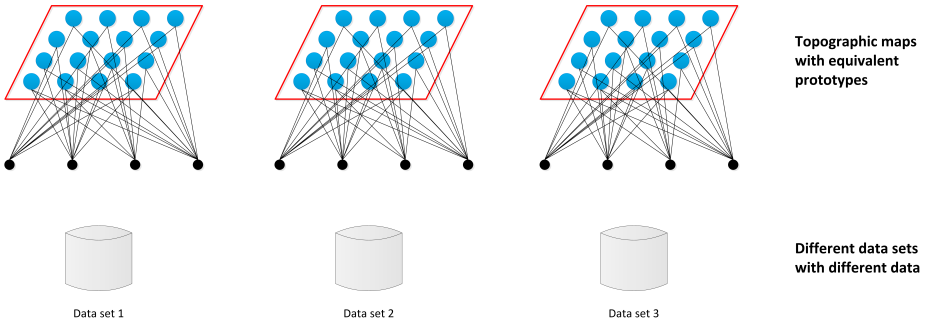

Figure 4: From Different data sets to similar prototypes

The principle of these algorithms is that when initialized properly, and when used on data sets that have similar data distributions and are in the same feature spaces, they output very similar topographic maps where the prototypes are roughly identical from one data set to another (See Figure 4). The outputted maps and their equivalent prototypes can then be seen as a split data set to which it is possible to apply our previous collaborative Framework without any modification. Therefore, using the structure of these unsupervised neural networks, it is possible to solve a vertical collaboration problem using an horizontal collaboration framework.

4 The GTM model as a collaborative clustering local step

4.1 Original GTM algorithm

The GTM algorithm was proposed by Bishop et al. [1] as a probabilistic improvement to the Self-organizing maps (SOM) [11]. GTM is defined as a mapping from a low dimensional latent space onto the observed data space. The mapping is carried through by a set of basis functions generating a constrained mixture density distribution. It is defined as a generalized linear regression model:

$$y = y(z, \mathbf{W}) = \mathbf{W}\Phi(z) \tag{12}$$

where y is a prototype vector in the D-dimensional data space, Φ is a matrix consisting of M basis functions $(\phi_1(z), \ldots, \phi_M(z))$, introducing the non-linearity, $\mathbf{W}$ is a $D \times M$ matrix of adaptive weights w_{dm} that defines the mapping, and z is a point in latent space. The standard definition of GTM considers spherically symmetric Gaussians as basis functions, defined as:

$$\phi_m(x) = \exp\left\{-\frac{\|x - \mu_m\|^2}{2\sigma^2}\right\} \tag{13}$$

where μ_m represents the centers of the basis functions and σ - their common width. Let $X = (x_1, \ldots, x_N)$ be a data set containing N data points. A probability distribution of a data point $x_n \in \mathbb{R}^D$ is then defined as an isotropic Gaussian noise distribution with a single common inverse variance β:

$$
\begin{aligned}
p(x_n|z, \mathbf{W}, \beta) &= \mathcal{N}(y(z, \mathbf{W}), \beta) \\
&= \left(\frac{\beta}{2\pi}\right)^{D/2} \exp\left\{-\frac{\beta}{2}\|x_n - y(z, \mathbf{W})\|^2\right\}
\end{aligned} \tag{14}
$$

The distribution in x-space, for a given value of $\mathbf{W}$, is then obtained by integration over the z-distribution

$$
p(x|\mathbf{W}, \beta) = \int p(x|z, \mathbf{W}, \beta)p(z)dz \tag{15}
$$

and this integral can be approximated defining $p(z)$ as a set of K equally weighted delta functions on a regular grid,

$$
p(z) = \frac{1}{K}\sum_{i=1}^{K} \delta(z - z_k) \tag{16}
$$

So, equation (15) becomes

$$
p(x|\mathbf{W}, \beta) = \frac{1}{K}\sum_{i=1}^{K} p(x|z_i, \mathbf{W}, \beta) \tag{17}
$$

For the data set X, we can determine the parameter matrix $\mathbf{W}$, and the inverse variance β, using maximum likelihood. In practice it is convenient to maximize the log likelihood, given by:

$$
\begin{aligned}
\mathcal{L}(\mathbf{W}, \beta) &= \ln \prod_{n=1}^{N} p(x_n|\mathbf{W}, \beta) \\
&= \sum_{n=1}^{N} \ln \left\{\frac{1}{K}\sum_{i=1}^{K} p(x_n|z_i, \mathbf{W}, \beta)\right\}
\end{aligned} \tag{18}
$$

4.2 The EM Algorithm

The maximization of (18) can be regarded as a missing-data problem in which the identity i of the component which generated each data point x_n is unknown. The EM algorithm for this model is formulated as follows:

The posterior probabilities, or responsibilities, of each Gaussian component i for every data point x_n using Bayes' theorem are calculated in the E-step of the algorithm in this form

$$
\begin{aligned}
r_{in} &= p(z_i|x_n, \mathbf{W}_{old}, \beta_{old}) \\
&= \frac{p(x_n|z_i, \mathbf{W}_{old}, \beta_{old})}{\sum_{i'=1}^{K} p(x_n|z_i', \mathbf{W}_{old}, \beta_{old})} \\
&= \frac{\exp\{-\frac{\beta}{2}\|x_n - \mathbf{W}\phi(z_i)\|^2\}}{\sum_{i'=1}^{K} \exp\{-\frac{\beta}{2}\|x_n - \mathbf{W}\phi(z_i')\|^2\}}
\end{aligned} \tag{19}
$$

As for the M-step, we consider the expectation of the complete-data log likelihood in the form

$$\mathbf{E}[\mathcal{L}_{comp}(\mathbf{W}, \beta)] = \sum_{n=1}^{N} \sum_{i=1}^{K} r_{in} \ln\{p(x_n|z_i, \mathbf{W}, \beta)\} \tag{20}$$

The parameters $\mathbf{W}$ and β are now estimated maximizing (20), so the weight matrix W is updated according to:

$$\Phi^T G \Phi \mathbf{W}_{new}^T = \Phi^T R X \tag{21}$$

where, Φ is the $K \times M$ matrix of basis functions with elements $\Phi_{ij} = \phi_j(z_i)$, R is the $K \times N$ responsibility matrix with elements r_{in}, X is the $N \times D$ matrix containing the data set, and G is a $K \times K$ diagonal matrix with elements

$$g_{ii} = \sum_{n=1}^{N} r_{in} \tag{22}$$

The parameter β is updated according to

$$\frac{1}{\beta_{new}} = \frac{1}{ND} \sum_{n=1}^{N} \sum_{i=1}^{K} r_{in} \|x_n - \mathbf{W}^{new} \phi(z_i)\|^2 \tag{23}$$

4.3 Clustering of the obtained map

The result of the GTM algorithm is a topographic map in the form of linked prototypes. These topographic maps can be seen as a compression of the original data set, with the prototype being representative of different clusters from the original data set.

However, the number of prototype is usually much higher than the number of clusters that one can expect to find in a data set. Therefore, the initial GTM algorithm is usually followed by a clustering of the acquired prototypes in order to map them to the final clusters. This process is analogue to building a second layer of neurons over the topographic map. The prototypes of the final clusters are usually computed using the EM algorithm for the Gaussian Mixture Model [3] on the prototypes from the topographic map.

4.4 Collaborative clustering applied to the clustering step of the GTM algorithm

Our idea here is to apply the previously proposed collaborative framework to the second step of the GTM algorithm: The clustering of the final prototypes using the EM algorithm. To do so, we use the prototypes vectors $\mathbf{W}$ as input data sets for our collaborative model.

If we note s_q^i the cluster linked to the q^{th} map prototype w_q for the i^{th} map, then when adapting equation (9), the local equation to optimize in the collaborative EM for the i^{th} topographic map is the following:

$$P(w_q|s_q^i, \theta_{s_q}^i) \times P(s_q^i|S) = \frac{1}{Z} P(w_q|s_q^i, \theta_{s_q}^i) \times \prod_{j \neq i} \Psi_{s_q}^{j \to i} \tag{24}$$

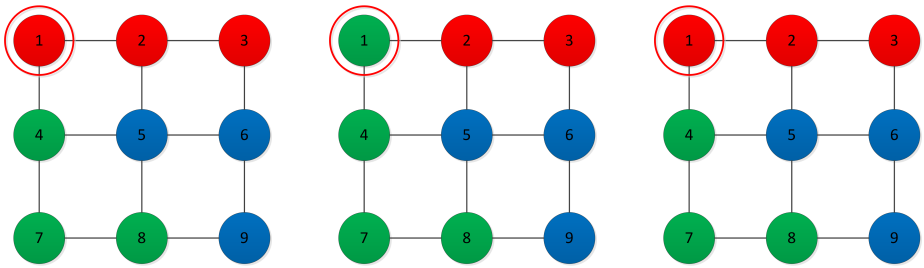

Figure 5: Example of 3 collaborating topographic maps. Since they had the same initialization and are used on data that are assumed to have similar distributions, the neurons are equivalent from one map to another. This simple example shows a conflict on the cluster associated to the first neuron. Using our collaborative method, the first neuron will most likely be switched to the red cluster in the second map. With bigger maps, more algorithms and more clusters, conflicts will be more difficult to resolve than in this simple example.

Under the hypothesis that all topographic maps have the same number of prototypes, underwent the same initialization, if we suppose that the different data sets have similar distributions, and knowing that we use the batch version of the GTM algorithm, the prototypes outputted by different GTM algorithms can be seen as a data set the attributes of which have been split between the different GTM algorithm instances. Therefore, since each prototype has a unique equivalent in each other topographic map, we can apply the collaborative framework for Heterogeneous algorithms.

Let's now suppose that we are running several GTM algorithms on different data sets that have the same features and for which we can assume the same cluster distributions can be found. If we use the same initialization for the prototypes of the topographic maps as described before, then we will have the prototype equivalents on the different maps. In this context, using the map prototypes $\mathbf{W}$ and their temporary cluster labels S from the local EM algorithm, we can apply a collaborative step to the EM algorithm. By doing so, the whole framework would be equivalent to a transfer learning process between the different data sets using vertical collaboration.

Based on the collaborative version of the EM algorithm, the transfer learning algorithm with Generative Topographic Maps using Collaborative Clustering is described in Algorithm 2. Figure 5 is an illustration of the kind of result we can expect from this Framework applied to topographic maps.

5 Experiments

5.1 Data sets

To evaluate the proposed Collaborative Clustering approach, we applied our algorithm on several data sets of different sizes and complexity. We chose the following: Waveform, Wisconsin Diagnostic Breast Cancer (wdbc), Madelon and Spambase.

- *waveform data set*: This data set consists of 5000 instances divided into 3 classes. The original base included 40 variables, 19 are all noise attributes with mean 0 and variance 1. Each class is generated from a combination of 2 of 3 "base" waves.

Algorithm 2: Vertical Collaborative Clustering using GTM : V2C-GTM

Data transformation
forall the *Data sets* X^i **do**
  Apply the regular GTM algorithm on the data X^i.
  Run a first instance of the EM algorithm on the prototypes $\mathbf{W^i}$
end
Retrieve the prototypes $\mathbf{W^i}$ and their clustering labels S^i
Local step:
forall the *clustering algorithms* **do**
  Apply the regular EM algorithm on the prototypes matrix $\mathbf{W}$.
  $\rightarrow$ Learn the local parameters Θ
end
Compute all $\Psi^{i \rightarrow j}$ matrices
Collaborative step:
while *the system global entropy is not stable* **do**
  forall the *clustering algorithms* c_i **do**
    forall the $w_q \in \mathbf{W^i}$ **do**
      Find s_q^i that maximize Equation (24).
    end
  end
  Update the solution vectors S
  Update the local parameters Θ
  Update all $\Psi^{i \rightarrow j}$ matrices
end

- *Wisconsin Diagnostic Breast Cancer (WDBC)*: This data has 569 instances with 32 variables (ID, diagnosis, 30 real-valued input variables). Each data observation is labeled as benign (357) or malignant (212). Variables are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image.

- *Spam Base*: The SpamBase data set is composed from 4601 observations described by 57 variables. Every variable described an e-mail and its category: spam or not-spam. Most of the attributes indicate whether a particular word or character was frequently occurring in the e-mail. The run-length attributes (55-57) measure the length of sequences of consecutive capital letters.

- *Madelon*: Madelon is an artificial dataset, which was part of the NIPS 2003 feature selection challenge. This is a two-class classification problem with continuous input variables. MADELON is an artificial dataset containing data points grouped in 32 clusters placed on the vertices of a five dimensional hypercube and randomly labelled +1 or -1. The five dimensions constitute 5 informative features. 15 linear combinations of those features were added to form a set of 20 (redundant) informative features. Based on those 20 features one must separate the examples into the 2 classes (corresponding to the +/-1 labels).

5.2 Indexes

As criteria to validate our approach we consider the purity (accuracy) index of the map which is equal to the average purity of all the cells of the map. A good GTM map should have a high purity index.

The cells purity is the percentage of data belonging to the majority class. Assuming that the data labels set $L = l_1, l_2, ..., l_{|L|}$ and the prototypes set $C = c_1, c_2, ..., c_{|C|}$ are known, the formula that expresses the purity of a map is the following:

$$purity = \sum_{k=1}^{|C|} \frac{c_k}{N} \times \frac{max_{i=1}^{|L|}|c_{ik}|}{|c_k|} \tag{25}$$

where $|c_k|$ is the total number of data associated with the cell c_k, and $|c_{ik}|$ is the number of data of class l_i which are associated to the cell c_k and N - the total number of data.

We define a_{11} as the number of object pairs belonging to the same cluster in $P1$ and $P2$, a_{10} denotes the number of pairs that belong to the same cluster in $P1$ but not in $P2$, and a_{01} denotes the pairs in the same cluster in $P2$ but not in $P1$. Finally, a_{00} denotes the number of object pairs in different clusters in $P1$ and $P2$. N is the total number of objects, n_i the number of objects in cluster i in $P1$, n_j the number of objects in cluster j in $P2$ and n_{ij} the number of object in cluster i in $P1$ and j in $P2$.

$$AR = \frac{a_{00} + a_{11} - n_c}{a_{00} + a_{01} + a_{10} + a_{11} - n_c} \tag{26}$$

For the Adjusted Rand Index (ARI), n_c is the agreement we would expect to arise by chance alone using the regular Rand index.

Finally, we also used a clustering index: The Davies-Bouldin index (DB index) [2] which assess that the resulting clusters are compact and well separated. The Davies-Bouldin index is not normalized and a lower value indicates a better quality. It is computed as follows:

Let S_i be a measure of scatter within a cluster i of size T_i and of centroid μ_i. Let $x_j \in X$ be a data associated to this cluster and μ_i, then:

$$S_i = \frac{1}{T_i} \sum_{j=1}^{T_i} ||x_j - \mu_i||_2$$

Let $M_{i,j}$ be a measure of separation between two clusters i and j so that:

$$M_{i,j} = ||\mu_i - \mu_j||_2$$

From these values and given K clusters, we define the Davies-Bouldin Index as shown in Equation (27).

$$DB = \frac{1}{K} \sum_{i=1}^{K} max_{j \neq i} \frac{S_i + S_j}{M_{i,j}} \tag{27}$$

5.3 Experimental Results

The experimental protocol was the following: All data sets were randomly shuffled and split into 5 subsets with roughly equivalent data distributions in order to have the topographic maps collaborating between the different subsets.

First, we ran the local step, to obtain a GTM map for every subset. The size of all the used maps were fixed to 12×12 for the SpamBase and Waveform data sets and 4×4 for the wdbc and Madelon data sets. Then we started the collaborative step using our proposed collaborative framework with the goal of improving each local GTM by exchanging based on the maps found for the other subsets. We evaluated the maps purity, the Adjusted Rand index of the final cluster, and the Davies-Bouldin Index of the clusters, based on the new GTMs after collaboration.

The results are shown in Table 1. Improved results and results that have not been deteriorated during the collaborative process are shown in bold.

As one can see, the results are different depending on the considered indexes. Overall our proposed method gives good results at improving the Adjusted Rand Index with excellent performances on all data sets except for the wdbc data set. The results for the purity index are also very satisfying with a post-collaboration improvement for more than 50% (12/20) of the data sets sub-samples. The results on the Davies-Bouldin index are more contrasted with only 11 cases out of 20 when the internal index remains stable or improves. These results are similar with those of other works on collaborative learning and highlight that while the goal of a general improvement of all collaborators is usually difficult to achieve, the average results' improvements remains positive.

Furthermore, our main goal was to take into account distant information from other algorithms working on similar data distribution and to build a new map. This procedure being unsupervised, it can deteriorate the different quality indexes when collaborating with data sets the distributions of which do not exactly match between each other, or simply when the quality of their proposed maps is too low.

5.4 Comparison with other algorithms

In this section we compare our algorithm to the vertical version of the collaborative clustering using prototype-based techniques (GTM_{Col}) introduced in [6]. While the two methods may seem similar, there are some major differences: 1) In our proposed method the collaboration occurs after building the maps, while in the GTM_{Col} the collaboration occurs while building the maps. 2) In our method the collaborations is simultaneously enabled between all algorithms, while GTM_{Col} only enables pairwise collaborations. Given these two differences the results that we show thereafter have to be taken with caution: While the two methods have the same goals and applications, they are very different in the way they work.

In Table 2, we show the comparative results of the average gain of purity measured before and after collaboration.

As one can see, while both methods give mild performances at improving the purity of a GTM map for our algorithm and a SOM map for the GTM_{Col} method, our algorithm is always positive on average for all data sets and our global results are also slightly better.

It is easy to see that the proposed V2C-GTM method outperforms other methods by increasing every time the accuracy index after the collaboration step. Even, if for Madelon dataset, the purity index after the collaboration is higher for the GTM_{Col} and SOM_{Col} methods, we have to note here that for these indexes the accuracy gain

Table 1: Experimental results of the horizontal collaborative approach on different data sets

Dataset	Map	Purity	ARI	DB index
SpamBase	GTM_1	51.1%	0.2	2.15
	GTM_2	53.3%	0.17	1.87
	GTM_3	58.4%	0.12	1.72
	GTM_4	64.89%	0.38	1.47
	GTM_5	75.97%	0.61	0.91
	GTM_{col1}	**59.8%**	**0.3**	**1.68**
	GTM_{col2}	**59.2%**	**0.27**	**1.65**
	GTM_{col3}	57.8%	**0.12**	1.77
	GTM_{col4}	**65.58%**	**0.45**	**1.23**
	GTM_{col5}	68.43%	0.52	1.09
WDBC	GTM_1	62.66%	0.32	1.37
	GTM_2	67.65%	0.37	1.29
	GTM_3	73.78%	0.48	0.94
	GTM_4	61%	0.35	1.48
	GTM_5	56.13%	0.241	1.63
	GTM_{col1}	58.66%	0.258	1.56
	GTM_{col2}	67.45%	0.36	1.34
	GTM_{col3}	71.62%	0.462	1.12
	GTM_{col4}	**63.12%**	**0.374**	**1.38**
	GTM_{col5}	**62.45%**	**0.369**	**1.44**
Madelon	GTM_1	51%	0.22	13.35
	GTM_2	56.5%	0.27	15.25
	GTM_3	52.5%	0.245	12.16
	GTM_4	50.75%	0.209	11.56
	GTM_5	50.25%	0.2	11.69
	GTM_{col1}	**51%**	**0.223**	**13.35**
	GTM_{col2}	55.5%	**0.27**	15.71
	GTM_{col3}	**52.5%**	**0.245**	**12.16**
	GTM_{col4}	**56.25%**	**0.257**	14.82
	GTM_{col5}	**51.5%**	**0.234**	14.05
Waveform	GTM_1	67.25%	0.46	1.54
	GTM_2	72.12%	0.58	1.27
	GTM_3	74.28%	0.61	1.22
	GTM_4	69.47%	0.507	1.49
	GTM_5	71.09%	0.564	1.3
	GTM_{col1}	**67.79%**	**0.472**	**1.46**
	GTM_{col2}	71.76%	**0.62**	**1.27**
	GTM_{col3}	72.59%	0.59	1.25
	GTM_{col4}	**71.52%**	**0.617**	**1.24**
	GTM_{col5}	**71.1%**	**0.603**	**1.23**

depends on the collaboration parameter β which is fixed in the algorithm (the higher this parameter is, the higher the distant collaboration will be used in the local learning process).

Another important aspect of the GTM and SOM based collaboration methods is that these approaches can attempt collaboration only between two collaborators in both direction which explain the $\pm$ in the results (without having an a priori knowledge about the quality of the collaborators the accuracy gain can be positive or negative). We note here that the proposed V2C-GTM approach can use several distant information from several collaborators without fixing any collaboration parameters and usually the accuracy gain is positive.

Table 2: Comparison of the average gain of purity before and after collaboration

Dataset	Purity		
	V2C-GTM	GTM_{Col}	SOM_{Col}
SpamBase	+1.43%	-2.31%	-2.4%
WDBC	+0.416%	-2.45%	$\pm$0.32%
Madelon	+1.15%	+2.85%	+2.1%
Waveform	+0.11%	+0.07%	$\pm$2.6%

These results are quite interesting because unlike the GTM_{Col} method that was specifically thought and developed with the idea of using it with semi-organized maps or generative topographic maps, the collaborative framework that we use was thought to be as generic as possible and not particularly adapted to the GTM algorithm.

The conclusion we could draw from these results is that perhaps the probabilistic approach used by our framework is more effective than the derivative approach used in the other method.

6 Conclusion

In this article, we have proposed an original collaborative learning method based on collaborative clustering principles and applied to the Generative Topographic Mapping (GTM) algorithm. Our framework consists in applying the GTM algorithm on different data sets where similar clusters can be found (same feature spaces and similar data distributions). Our proposed method makes it possible to exchange information between different instances of the GTM algorithm with the goal of a faster convergence and better tuning of the topographic maps parameters.

Our experimental results have shown our framework to be very effective at improving the final clustering of the maps involved in the collaborative process at least based on external indexes such as the maps purity and the Adjusted Rand Index, thus fulfilling its intended purpose. Furthermore, the results on both internal and internal indexes are better or similar with those already observed with other collaborative methods. Sadly some of the caveats observed with other methods in the literature seem to apply to our proposed method as well, in the way that while the global results' improvement after

collaboration remain positive, it is still unlikely to achieve performances above those of the best collaborator.

One attractive perspective for our work would be to find a way to remove both constraints that either the observed data or the feature spaces have to be identical in order to use either horizontal or vertical collaboration. Getting rid of both constraints would enable transfer learning between data sets that are very different but have similar clusters structures. Doing so would require to find a solution to train the topographic maps with either the SOM or the GTM algorithm in a way that despite the different feature spaces the parallel maps would learn from all data sets and still have similar features once built. We look forward to finding a solution for this problem.

Acknowledgements

This work has been supported by the ANR Project COCLICO, ANR-12-MONU-0001.

References

[1] BISHOP, C. M., SVENSEN, M., AND WILLIAMS, C. K. I. Gtm: The generative topographic mapping. *Neural Computation 10* (1998), 215–234.

[2] DAVIES, D. L., AND BOULDIN, D. W. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence 1*, 2 (Feb. 1979), 224–227.

[3] DEMPSTER, A. P., LAIRD, N. M., AND RUBIN, D. B. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B 39*, 1 (1977), 1–38.

[4] DEPAIRE, B., FALCON, R., VANHOOF, K., AND WETS, G. PSO Driven Collaborative Clustering: a Clustering Algorithm for Ubiquitous Environments. *Intelligent Data Analysis 15* (January 2011), 49–68.

[5] FORESTIER, G., GANCARSKI, P., AND WEMMERT, C. Collaborative clustering with background knowledge. *Data & Knowledge Engineering 69*, 2 (2010), 211–228.

[6] GHASSANY, M., GROZAVU, N., AND BENNANI, Y. Collaborative clustering using prototype-based techniques. *International Journal of Computational Intelligence and Applications 11*, 3 (2012).

[7] GREEN, P. J. On Use of the EM Algorithm for Penalized Likelihood Estimation. *Journal of the Royal Statistical Society. Series B (Methodological) 52*, 3 (1990), 443–452.

[8] GROZAVU, N., AND BENNANI, Y. Topological collaborative clustering. *Australian Journal of Intelligent Information Processing Systems 12*, 3 (2010).

[9] GROZAVU N., B. Y. Topological collaborative clustering. *in LNCS Springer of ICONIP'10 : 17th International Conference on Neural Information Processing* (2010).

[10] JAIN, A. K., MURTY, M. N., AND FLYNN, P. J. Data clustering: a review. *ACM Computing Surveys 31*, 3 (1999), 264–323.

[11] KOHONEN, T. *Self-organizing Maps*. Springer Berlin, 2001.

[12] PEDRYCZ, W. Collaborative fuzzy clustering. *Pattern Recognition Letters 23*, 14 (2002), 1675–1686.

[13] PEDRYCZ, W. Fuzzy clustering with a knowledge-based guidance. *Pattern Recogn. Lett. 25*, 4 (2004), 469–480.

[14] PEDRYCZ, W. *Knowledge-Based Clustering*. John Wiley & Sons, Inc., 2005.

[15] PEDRYCZ, W., AND HIROTA, K. A consensus-driven fuzzy clustering. *Pattern Recognition Letters 29*, 9 (2008), 1333–1343.

[16] SILVA, A. D., LECHEVALLIER, Y., DE A. T. DE CARVALHO, F., AND TROUSSE, B. Mining web usage data for discovering navigation clusters. In *ISCC* (2006), P. Bellavista, C.-M. Chen, A. Corradi, and M. Daneshmand, Eds., IEEE Computer Society, pp. 910–915.

[17] STREHL, A., AND GHOSH, J. Cluster ensembles — a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research 3* (2002), 583–617.

[18] SUBLIME, J., GROZAVU, N., BENNANI, Y., AND CORNUÉJOLS, A. Collaborative clustering with heterogeneous algorithms. In *2015 International Joint Conference on Neural Networks, IJCNN 2015, Killarney, Ireland, July 12-18, 2015* (2015).

[19] SUBLIME, J., GROZAVU, N., BENNANI, Y., AND CORNUÉJOLS, A. Vertical collaborative clustering using generative topographic maps. In *IEEE 7th International Conference on Soft Computing and Pattern Recognition, SocPaR 2015, Fukuoka, Japan, November 13-15, 2015* (2015).

[20] WANG, X.-N., WEI, J.-M., JIN, H., YU, G., AND ZHANG, H.-W. Probabilistic confusion entropy for evaluating classifiers. *Entropy 15*, 11 (2013), 4969–4992.

[21] WEI, J.-M., YUAN, X.-J., HU, Q.-H., AND WANG, S.-Q. A novel measure for evaluating classifiers. *Expert System Applications 37*, 5 (2010), 3799–3809.

[22] ZARINBAL, M., ZARANDI, M. F., AND TURKSEN, I. Relative entropy collaborative fuzzy clustering method. *Pattern Recognition 48*, 3 (2015), 933 – 940.