Enriched topological learning for cluster detection and visualization

Guénaël Cabanes, Younès Bennani, Dominique Fresneau

Enriched topological learning for cluster detection and visualization

Guénaël Cabanes[a], Younès Bennani[a], Dominique Fresneau[b]

[a]*LIPN-CNRS, UMR 7030, 99 Avenue J-B. Clément, 93430 Villetaneuse, France*
[b]*LEEC, EA 4443, 99 Avenue J-B. Clément, 93430 Villetaneuse, France*

Abstract

The exponential growth of data generates terabytes of very large databases. The growing number of data dimensions and data objects presents tremendous challenges for effective data analysis and data exploration methods and tools. Thus, it becomes crucial to have methods able to construct a condensed description of the properties and structure of data, as well as visualization tools capable of representing the data structure from these condensed descriptions. The purpose of our work described in this paper is to develop a method of describing data from enriched and segmented prototypes using a topological clustering algorithm. We then introduce a visualization tool that can enhance the structure within and between groups in data. We show, using some artificial and real databases, the relevance of the proposed approach.

Keywords: Self-Organizing Map, Prototypes Enrichment, Two-Level Clustering, Coclustering, Visualization.

1. Introduction

The exponential growth of data generates terabytes of very large databases [1]. The growing number of data dimensions and data objects presents tremendous challenges for effective data analysis and data exploration methods and tools. Thus, it becomes crucial to have methods able to construct a condensed description of the properties and structure of data [2, 3, 4], as well as visualization tools capable of representing the data structure from these condensed descriptions.

The purpose of the work described in this paper is to develop a method of describing data from enriched and segmented prototypes using a topological clustering algorithm. An important contribution of the proposed approach is the ability to provide data visualizations via maps and graphs, to provide a comprehensive exploration of the data structure. We propose here a method of describing data from enriched prototypes, based on learning a Self-Organizing Map (SOM) [5]. Prototypes of the SOM are segmented using an adapted clustering algorithm. This method is flexible enough to be adapted to a high variety of different problems. A new coclustering algorithm is proposed to illustrate this flexibility, and we show an example of real application for this algorithm. We then introduce a visualization tool of enriched and segmented SOM that can enhance the structure within and between groups of data.

The remainder of this paper is organized as follow. Section 2 presents the learning of the data structure to obtain a condensed description. Section 3 show a new SOM-based coclustering algorithm and the results of an experimental application. Visualization tool is described in Section 4 and some examples are shown. A conclusion is given in section 5.

2. Learning data structure

We propose here a method to learn data structure, based on the automated enrichment and segmentation of a group of prototypes representing the data to be analyzed [6]. We suppose that these prototypes have been previously computed from data thanks to an adapted algorithm, such as Neural Gas (NG) [7] or Self Organizing Map (SOM) [8, 5]. In this paper we focus on the use of the SOM algorithm as a basis of data quantization and representation. A SOM consists of a set of artificial neurons that represent the data structure. These neurons are connected with their neighbors according to topological connections (also called neighborhood connections). The dataset to analyze is used to organize the SOM under topological constraints of the input space. Thus, a correspondence between the input space and the mapping space is built. Two observations close in the input space should activate the same neuron or two neighboring neurons of the SOM. Each neuron is associated with a prototype and, to respect the topological

constraints, neighboring neurons of the best match unit of a data (BMU, the most representative neuron) also update their prototype for a better representation of this data. This update is important because the neurons are close neighbors of the best neuron.

2.1. Principle

The first step is the learning of the enriched SOM. During the learning, each SOM prototype is extended with novel information extracted from the data. This information will be used in the following step to find clusters in the data and to infer the density function. More specifically, the information added to each prototype are:

- *Density modes*. It is a measure of the data density surrounding the prototype (local density). The local density is an information about the amount of data present in an area of the input space. We use a Gaussian kernel estimator [9] for this task.
- *Local variability*. It is a measure of the data variability that is represented by the prototype. It can be defined as the average distance between the prototypes and the represented data.
- *The neighborhood*. This is a prototype's neighborhood measure. The neighborhood value of two prototypes is the number of data that are well represented by each one.

The second step is the clustering of the data using density and connectivity information so as to detect low-density boundary between clusters. We propose a clustering method that directly uses the information learned during the first stage.

2.2. Prototypes Enrichment

The enrichment algorithm proceeds in three phases:

Input:

- The distance matrix $Dist(w, x)$ between the M prototypes w and the N data x.

Output:

- The density D_i and the local variability s_i associated to each prototype w_i.
- The neighborhood values $v_{i,j}$ associated with each pair of prototype w_i and w_j.

Algorithm:

- **Density estimate**:

$$D_i = 1/N \sum_{k=1}^{N} \frac{e^{-\frac{Dist(w_i, x^{(k)})^2}{2\sigma^2}}}{\sigma\sqrt{2\pi}}$$

  with σ a bandwidth parameter chosen by user.

- **Estimate neighborhood values**:
  - For each data x, find the two closest prototypes (BMUs) $u^*(x)$ and $u^{**}(x)$:

$$u^*(x) = argmin_i(Dist(w_i, x))$$

    and

$$u^{**}(x) = argmin_{i \neq u^*(x)}(Dist(w_i, x))$$

  - Compute $v_{i,j}$ = the number of data having i and j as two first BMUs.

- **Local variability estimate**: For each prototype w, variability s is the mean distance between w and the L data x_w represented by w:

$$s_i = 1/L \sum_{j=1}^{L} Dist(w_i, x_w^{(j)})$$

The proposed method for estimating the mode density is very similar to that proposed by [10]. It has been shown that when the number of data approaches infinity, the estimator D converges asymptotically to the true density function [11]. The choice of the parameter σ is important for good results. If σ is too large, all data will influence the density of all the prototypes, and close prototypes will be associated to similar densities, resulting in decreased accuracy of the estimate. If σ is too small, a large proportion of data (the most distant prototypes) will not influence the density of the prototypes, which induces a loss of information. A heuristic that seems relevant and gives good results is to define σ as the average distance between a prototype and its nearest neighbor.

At the end of this step, each prototype is associated with a density and variability value, and each pair of prototypes is associated with a neighborhood value. Much of the information on the data structure is stored in these values. There is no more need to keep data in memory.

2.3. Clustering of prototypes

Various prototypes-based approaches have been proposed to solve the clustering problem [12, 13, 14, 15]. However, the obtained clustering is never optimal, since part of the information contained in the data is not represented by the prototypes. We propose a new method of prototypes' clustering, that uses density and neighborhood information to optimize the clustering. The main idea is that the core part of a cluster can be defined as a region with high density. Then in most cases the cluster borders are defined either by low density region or "empty" region between clusters (i.e. large inter cluster distances) [16].

At the end of the enrichment process, each set of prototypes linked together by a neighborhood value $v > 0$ define well separate clusters (i.e. distance-defined). This is useful to detect borders defined by large inter cluster distances (Fig.2(b)). The estimation of the local density (D) is used to detect cluster borders defined by low density. Each cluster is defined by a local maximum of density (density mode, Fig. 2(c)). Thus, a "Watersheds" method [17] is applied on prototypes' density for each well separated cluster to find low density area inside these clusters, in order to characterize density defined sub-clusters (Fig.2(d)). For each pair of adjacent subgroups we use a density-dependent index [18] to check if a low density area is a reliable indicator of the data structure, or whether it should be regarded as a random fluctuation in the density (Fig.2(e)). This process is very fast because generally the number of prototypes is small. The combined use of these two types of group definition can achieve very good results despite the low number of prototypes in the map and is able to detect automatically the number of cluster (cf. [19]).

The algorithm proceed in three steps:

Input:

- Density values D_i.
- Neighborhood values $v_{i,j}$.

Output:

- The clusters of prototypes.

1. **Extract all groups of connected units**:
   Let $P = \{C_i\}_{i=1}^{L}$ the L groups of linked prototypes (see Fig.2(b)):

   $$\forall m \in C_i, \exists n \in C_i \text{ such as } v_{m,n} > threshold$$

   In this paper $threshold = 0$.

2. **For each** $C_k \in P$ **do:** :

   - Find the set $M(C_k)$ of density maxima (see Fig.2(c)).

     $$M(C_k) = \{w_i \in C_k \mid D_i \geq D_j, \forall w_j \text{ neighbor to } w_i\}$$

     Prototypes w_i and w_j are neighbor if $v_{i,j} > threshold$.
   - Determine the merging threshold matrix (see Fig. 1):

     $$S = [S(i,j)]_{i,j=1...|M(C_k)|}$$

     with

     $$S(i,j) = \left(\frac{1}{D_i} + \frac{1}{D_j}\right)^{-1}$$

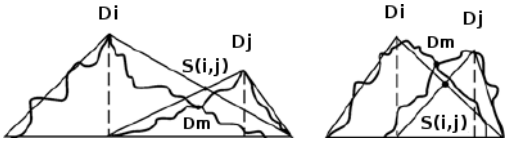


Figure 1: Threshold computation

   - For all prototype $w_i \in C_k$, label w_i with one element $label(i)$ of $M(C_k)$, according to an ascending density gradient along the neighborhood. Each label represents a micro-cluster (see Fig.2(d)).
   - For each pair of neighbors prototypes (w_i, w_j) in C_k, if:

     $$label(i) \neq label(j)$$

     and if both

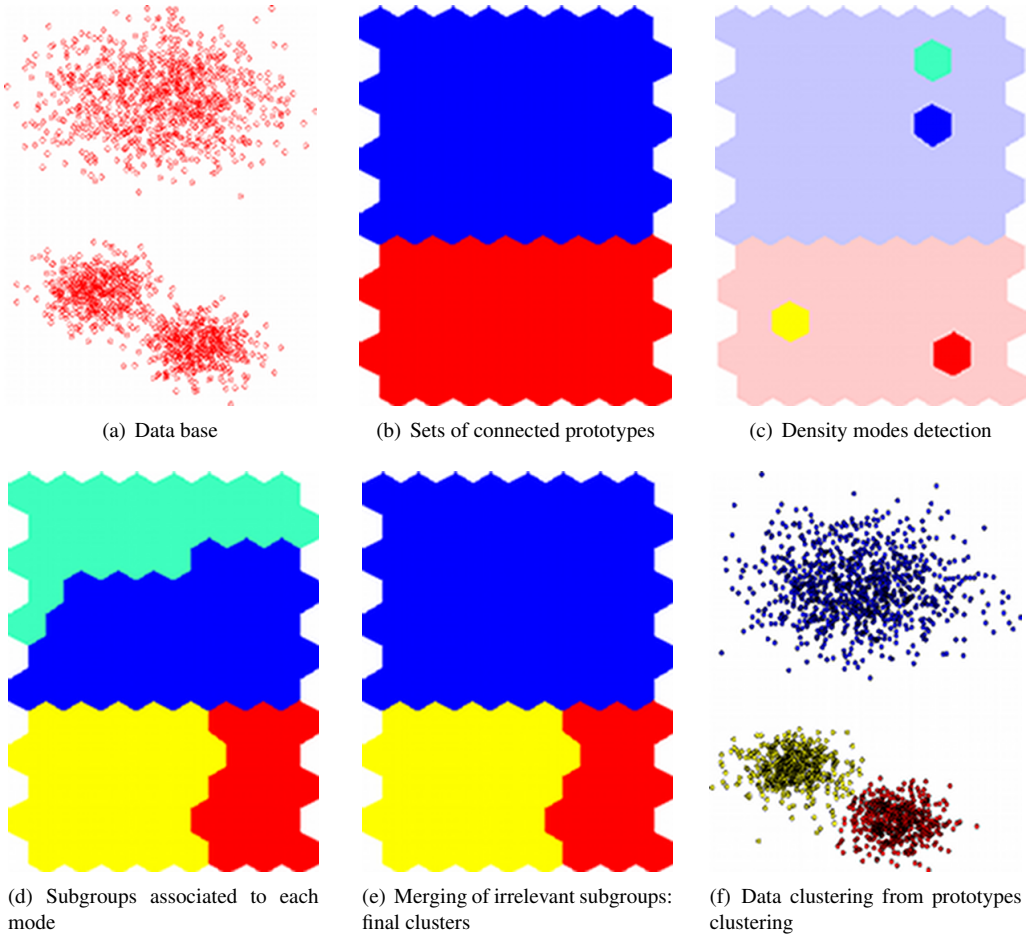
     $$D_i > S(label(i), label(j))$$

     and

     $$D_j > S(label(i), label(j))$$

     then merge the two micro-clusters (Fig.2(e)).

The effectiveness of the proposed clustering method have been demonstrated in [19] by testing the performances on 10 databases presenting various clustering difficulties. It was compared to S2L-SOM [20] (using only neighborhood information) and to some traditional two levels methods, in term of clustering quality (Jaccard and Rand indexes [21]) and stability (sub-sampling based method [22]). The selected traditional algorithms



(a) Data base (b) Sets of connected prototypes (c) Density modes detection

(d) Subgroups associated to each mode (e) Merging of irrelevant subgroups: final clusters (f) Data clustering from prototypes clustering

Figure 2: Example of a sequence of the different stages of the clustering algorithm.

for comparison are K-means and Ascendant Hierarchical Clustering (AHC) applied (i) to the data and (ii) to the prototypes of the trained SOM. The Davies-Bouldin [23] index was used to determine the best cutting of the dendrogram (AHC) or the optimal number K of centroids for K-means. Our algorithm determines the number of clusters automatically and do not need to use this index. In AHC, the proximity of two clusters was defined as the minimum of the distance between any two objects in the two different clusters. The results for the external indexes show that for all the databases the proposed clustering algorithm is able to find without any error the expected data segmentation and the right number of clusters. This is not the case of the other algorithms, when the groups have an arbitrary form, when there is no structure (i.e. only one cluster) in the data or when clusters are in contact. Considering the stability, the new algorithm shows excellent results, whatever the dimension of data or the clusters' shape. It is worth noticing that in some case the clustering obtained by the traditional methods can be extremely unstable.

We present here additional tests that have been done to compare the new method with other usual clustering algorithms that generally perform better than K-Means and AHC. These algorithms are DBSCAN [24], CURE [25] and Spectral Clustering [26]. In [27], the authors show that these algorithms fail in resolving some clustering problems, especially when clusters' shape is not hyper-spherical or when clusters are in contact. Fig. 3 to 5 show that our method success in resolving this kind of problems (datasets are the same as in [27]).

To summarize, the proposed method presents some interesting qualities in comparison to other clustering algorithms:

- The number of cluster is automatically detected by the algorithm.
- No linearly separable clusters and non hyper-

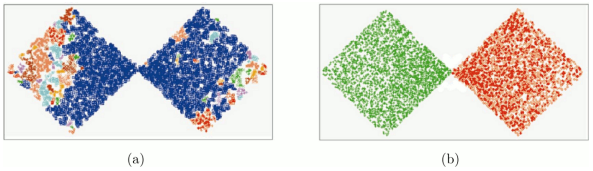


Figure 3: Clustering obtained with (a) DBSCAN and (b) the proposed method.

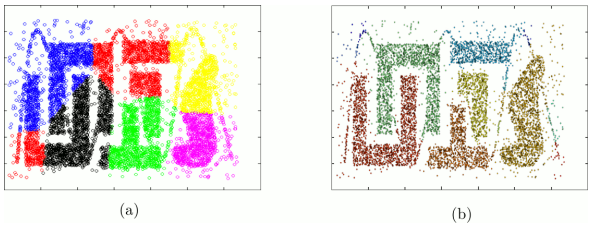


Figure 4: Clustering obtained with (a) Spectral Clustering and (b) the proposed method.

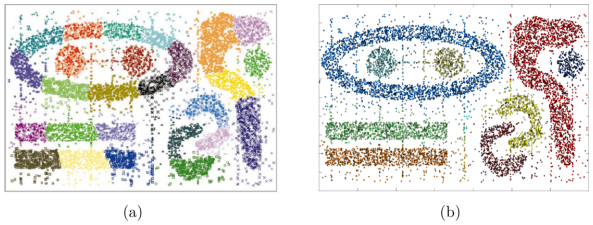


Figure 5: Clustering obtained with (a) CURE and (b) the proposed method.

spherical clusters can be detected.

- The algorithm can deal with noise (i.e. touching clusters) by using density estimation.

2.4. Modeling data distributions

The objective of this step is to estimate the density function which associates a density value to each point of the input space. An estimation of some values of this function have been calculated (i.e. D_i) at the position of the prototypes representing a cluster. An approximation of the function must now be inferred from these values.

The hypothesis here is that this function may be properly approximated in the form of a mixture of Gaussian kernels. Each kernel K is a Gaussian function centered on a prototype. The density function can therefore be written as:

$$f(x) = \sum_{i=1}^{M} \alpha_i K_i(x)$$

with

$$K_i(x) = \frac{1}{N\sqrt{2\pi}h_i} e^{-\frac{d(w_i,x)^2}{2{h_i}^2}}$$

The most popular method to fit mixture models (i.e. to find h_i and α_i) is the expectation-maximization (EM) algorithm [28]. However, this algorithm needs to work in the data input space. As here we work on enriched SOM instead of dataset, we cannot use EM algorithm.

Thus, we propose a heuristic to choose h_i:

$$h_i = \frac{\sum_j \frac{v_{i,j}}{N_i+N_j}(s_i N_i + d_{i,j} N_j)}{\sum_j v_{i,j}}$$

$d_{i,j}$ is the distance between prototypes w_i and w_j. The idea is that h_i is the standard deviation of data represented by K_i. These data are also represented by w_i and their neighbors. Then h_i depends on the variability s_i computed for w_i and the distance $d_{i,j}$ between w_i and his neighbors, weighted by the number of data represented by each prototype and the connectivity value between w_i and his neighborhood.

Now, since the density D for each prototype w is known ($f(w_i) = D_i$), a gradient descent method can be used to determine the weights α_i. The α_i are initialized with the values of D_i, then these values are reduced gradually to better fit $D = \sum_{i=1}^{M} \alpha_i K_i(w)$. To do this, the following criterion is minimized:

$$R(\alpha) = \frac{1}{M} \sum_{i=1}^{M} \left[\sum_{j=1}^{M} \left(\alpha_j G_j(w_i) \right) - D_i \right]^2$$

Algorithm:

1. **Initialization**:
$$\forall i, \alpha_i = D_i$$
2. **Error calculation**:
$$\forall i, Err(i) = \sum_{j=1}^{M} \alpha_j G_j(w_i) - D_i$$
3. **Coefficients update**:
$$\forall i, \alpha_i(t) = max\,[0\,;\, \alpha_i(t\text{-}1) - \epsilon * Err(i)]$$
with ϵ the gradient step. Here we use $\epsilon = 0.1$.
4. **As** $mean(|Err|) > threshold$: go to 2, else return α_i. The threshold is chosen by user, here we choose 1% of the mean density.

Thus, we have a density function that is a model of the dataset represented by the enriched SOM. Some examples of estimated density are shown on Fig. 6 and 7.

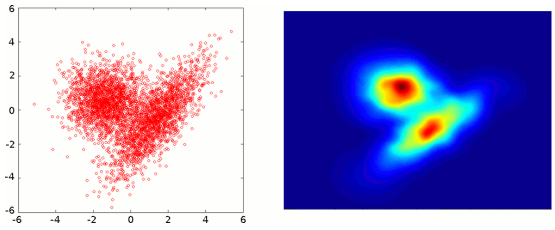


Figure 6: “Engytime” dataset and the estimated density function.

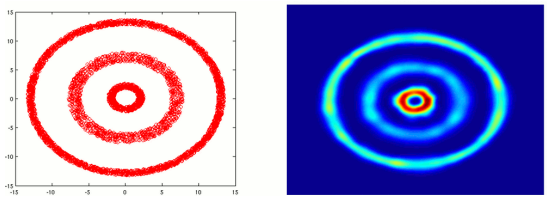


Figure 7: “Rings” dataset and the estimated density function.

3. An application to coclustering

The algorithms presented in section 2 can be easily adapted for the analysis of a variety of problems (see [29, 30]). We propose in this section an adaptation to coclustering problem and a real application of this adaptation.

It can be sometime very interesting to be able to regroup and visualize the attributes used to describe the data, in addition to the clustering of these data. This allows, for example, to combine in a simple way each cluster of data with the characteristic features of this cluster, but also to visualize correlations between attributes. Coclustering, biclustering, or two mode clustering is a data mining technique which allows simultaneous clustering of rows and columns of data sets (data matrix) [31]. Given a set of m rows in n columns (i.e., an $m \times n$ matrix), the coclustering algorithm generates coclusters - a subset of rows which exhibit similar behavior across a subset of columns, or vice versa. The most popular application for such methods is gene expression analysis, i.e. to identify local patterns in gene expression data (see [32]).

The use of SOM to perform coclustering have been proposed in [33, 34]. However, in these works, each cluster is represented by an unique prototype of the SOM, which leads to an inappropriate number of clusters. The proposed method will combine a modified SOM with a two-level coclustering of the SOM prototypes ables to detect automatically the correct number of clusters.

3.1. SOM adaptation for disjunctive data

The basis algorithm of our approach is the KDisj method proposed in [33]. This algorithm is an adaptation of SOM that allow to project on the map both data and features used to describe them. This algorithm is designed for the quantization of qualitative data in the form of a disjunctive table T: each feature has several mutually exclusive modalities (e.g. the attribute “color” may have the modalities “yellow”, “green”, etc ...). Features can therefore be encoded as a vector size equal to the number of modalities with a value of zero in all dimensions except one. We can code in the same way several attributes by a vector of size equal to all the modalities of the various features with as many non-zero values as the number of attributes. The main idea of KDisj is that one can describe a data based on the modalities associated with (row vector), but it is also possible to describe a modality based on the set of data (column vector). All data and modality can then be represented in a space of dimension $A + E$ (number of modalities for all features + number of data). A SOM can be learned in this space by presenting alternately a data and a modality during the learning. The distance between a data (size A) and a prototype of the map (size $A + E$) will be calculated on the A first dimensions, while the distance between a modality (size E) and a prototype will be calculated on the last E dimensions. To ensure a link between the A first dimensions and the E last, prototypes will be adjusted on all dimensions during the adaptation phase, by associating to each data its not-null modality the most characteristic (i.e. the rarest in the data set). Thus, the first A dimensions of each prototype are adapted based on the presented data and the last E dimensions are adapted depending on the associated modality. Note that it is not possible to do this even when a modality is presented, since there is no rare data in the description of the set of modalities (each data is characteristic of exactly as many modalities as the number of attributes).

3.2. A new Two-Levels coclustering algorithm

The proposed algorithm uses a stochastic learning process: prototype update and enrichment (limited here to connections values) are performed incrementally by presenting data in a random order. Whenever a data is presented, the value of the connection between the two most representative prototypes is increased whereas other connections values are decreased. In the same time prototypes are updated. The version presented here is modified to be adapted to data expressed in frequency or proportion, i.e. we associate a percentage to each

modality of a feature, the sum of terms for an attribute is thus equal to 1 (or 100%). This data type is widely used in many fields (time management, budget, modality varying in time or space,...). The only difference with a disjunctive table, in this case, is that you can associate a characteristic data to each modality. This allow to update prototypes in all dimensions ($A + E$) whatever is presented (data or modality).

The stochastic coclustering algorithm is the following:

1. **Initialization**:
   - Correct disjunctive table T into T_c:
   $$t_{ij}^c = \frac{t_{ij}}{\sqrt{t_{i.}t_{.j}}}$$
   $$\text{with } t_{i.} = \sum_{j=1}^{N} t_{ij} \text{ and } t_{.j} = \sum_{i=1}^{N} t_{ij}$$
   In that way using euclidean distance on T_c is similar to use weighted χ^2 distance on T [33].
   - Initialize randomly the prototypes $w_j = (w_{Aj}, w_{Ej})$.
   - Initialize to 0 connections values v_{ij} between each pair of neurons i et j.

2. **Present a data** $x^{(k)}$: i.e. a row of T_c, randomly chosen.
   - Associate to $x^{(k)}$ modality $y(x^{(k)})$ defined by
   $$y(x^{(k)}) = \underset{y}{Argmax}\, t_{xy}^c$$
   and create vector $Z_x^{(k)} = (x^{(k)}, y(x^{(k)}))$.
   - *Competition step*:
     - Choose the two most representatives neurons $u^*(x^{(k)})$ and $u^{**}(x^{(k)})$ over the A first dimensions:
     $$u^*(x^{(k)}) = \underset{1\le i\le M}{Argmin} \parallel x^{(k)} - w_{Ai} \parallel^2$$
     $$u^{**}(x^{(k)}) = \underset{1\le i\le M, i\neq u^*}{Argmin} \parallel x^{(k)} - w_{Ai} \parallel^2$$
     - Update connection value between $u^*(x^{(k)})$ and its neighbors according to the learning step $\varepsilon(t)$, a decreasing function of time in $[0, 1]$, inversely proportional to time:
     $$v_{u^*u^{**}}(t) = v_{u^*u^{**}}(t\text{-}1)\text{-}\varepsilon(t)\,(v_{u^*u^{**}}(t\text{-}1)\text{-}1)$$
     $$v_{u^*i}(t) = v_{u^*i}(t\text{-}1)\text{-}\varepsilon(t)\,(v_{u^*i}(t\text{-}1))$$
     $$\forall i \neq u^{**}, i \text{ neighbor of } u^*$$
   - *Adaptation step*:
     - Update prototypes w_j for each neuron j on all dimensions, according to the neighbor function H:
     $$w_j(t) = w_j(t\text{-}1)\text{-}\varepsilon(t)H_{ju^*(x^{(k)})}(w_j(t\text{-}1)\text{-}Z_x^{(k)})$$

3. **Present a modality** $y^{(k)}$: i.e. a column of T_c, randomly chosen.
   - Associate to $y^{(k)}$ modality $x(y^{(k)})$ defined by
   $$x(y^{(k)}) = \underset{x}{Argmax}\, t_{xy}^c$$
   and create vector $Z_y^{(k)} = (x(y^{(k)}), y^{(k)})$.
   - *Competition step*:
     - Find the two best representatives neurons $u^*(y^{(k)})$ and $u^{**}(y^{(k)})$ over the E last dimensions and update connection values between $u^*(y^{(k)})$ and its neighbors as in step 2.
   - *Adaptation step*:
     - Update prototypes w_j for each neuron j, according to the neighbor function H:
     $$w_j(t) = w_j(t\text{-}1)\text{-}\varepsilon(t)H_{ju^*(y^{(k)})}(w_j(t\text{-}1)\text{-}Z_y^{(k)})$$

4. **Repeat steps 2 and 3** until convergence.

At the end of the clustering process, a cluster is a set of prototypes which are linked together by neighborhood connections with positive values. Thus,the right number of cluster is determined automatically.

In comparison to most existing coclustering methods, our algorithm is able to perform at the same time a fast clustering of both data and features, and a two dimensional quantization of the data, which allows an easy visualization of this structure. Moreover, our algorithm is able to detect automatically the right number of coclusters, whatever the shape of theses clusters. Most prototypes based coclustering (such as [35] for example) are unable to detect automatically the number of coclusters

to find, as this number must be given as a parameter. They also cannot detect non-hyperspherical clusters and they do not propose any two-dimensional visualization. SOM-based algorithms such as [33, 34] allow visualization, but are unable to correctly detect the coclusters, as the number of coclusters found is always the same as the number of prototypes in the SOM. Our algorithm overcomes this problem by learning a coclustering of the prototypes during the learning of the SOM.

3.3. Application

The application part of this work is to analyze and visualize biological experimental data. These data comes from a study on the ants' spatial and social organization [36]. A queen (R), a male (Mc), a young (J) and 43 workers (2-44) were observed in an artificial nest composed of 9 rooms (Loc2 to Loc10), a tunnel leading outside (Loc1) and a foraging area (Loc0, see Figure 8). For each individual, we know the proportion of time spent in each room and in 20 different activities extracted from a set of pictures of all individuals in the nest and the foraging area.

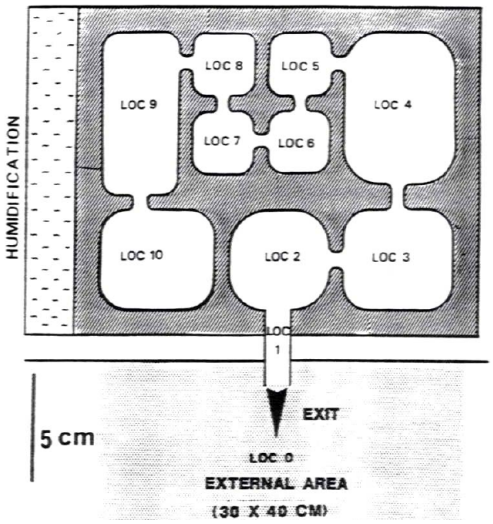


Figure 8: The artificial nest used for the experimental study.

The main goal of this study is to determine the existence of clusters of similar ants and to link each group of ants with some characteristic behaviors, in order to understand the social role of the group, as well as the relevant location, in order to understand how each group manage the allocated space to perform its task. The new algorithm is then a relevant algorithm to perform these tasks, as it is able to produce cluster regrouping at the same time individuals and features modalities.

The results obtained with the new algorithm from these data are shown in Figure 9. The entire learning process took a few seconds. Codes C0 to C10 represent the ten rooms. Ants behaviors are represented by 20 activities, each coded with a two or three letters, the last one giving the general category (T: entry and exit of the nest, N: Management of food, C: cocoons care, L: larvae care, O: eggs care).

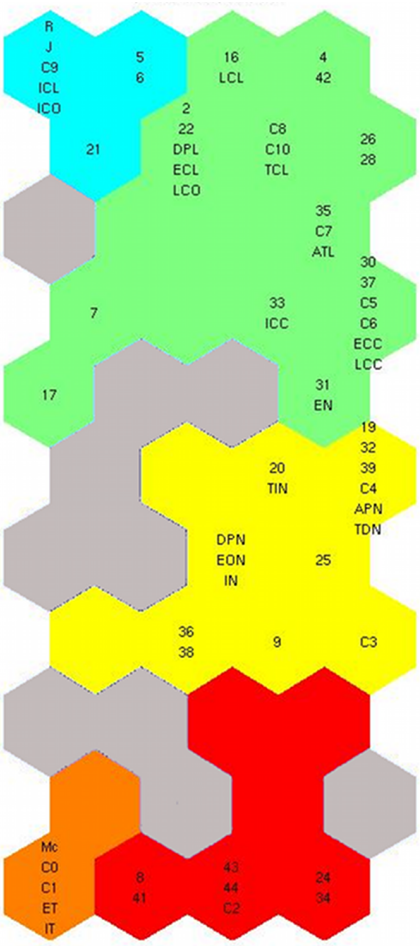


Figure 9: Clusters of ants (numbers), behaviors (letters) and location (C + number) obtained automatically. Each hexagon is a visualization of a neuron of the map. Neurons sharing a color represent individuals and features belonging to the same cluster. Grey neurons are not representative and do not belong to any clusters.

These results show that the queen, the young and a few other individuals are related to Room 9 and are characterized by "immobility on eggs and larvae" behavior ("blue" cluster). This is relevant as the queen need to be in a big room far from the entrance (for protection, [37]). Also, as the queen spend her life to lay

eggs, there is always in eggs and sometime larvae in her room, as well as young ants that don't have any social activity yet [37]. The "green" cluster regroup rooms 5, 6, 7, 8 and 10 with activity of larvae and cocoons care. This group is representative of the social role "nurses" which is essential in the colony's life. Ants in this group take care of the brood in order to guaranty-ing its survival. As during the development of the larva and the cocoon the need in humidity and temperature may vary, it have been observed hat the nurses displace frequently the members of the brood to find optimal location [37], it is therefore not surprising to find many different rooms in this cluster. In the same way, cluster "yellow" is a group of ant managing food in room 3 and 4, not far from the foraging area (where the food is given). The "red" cluster represent ants spending most of their time in room 2, without any related social activities. These kind of ant are known to be "generalist" in a colony, they are able to perform any task, especially foraging task, depending on the need of the colony [36]. The last cluster ("Orange") regroup rooms 0 and 1 (the tunnel and the foraging area) with input and output behavior. Theses relations are obvious. The male is also in the cluster, which indicate that he is mature to flight out the nest to find a female and fund a new colony.

One should also note that the linear disposition of the rooms inside the nest is also kept on the map.

4. Visualization

4.1. Description of the visualization process

The clustering is accompanied by a set of information that can be used to complete the analysis of data. This information is the matrix of distances between prototypes and the density matrix, but also the values of connections that can be used to determine relative importance of each prototype for the representation of data. It is possible to represent all this information into a single figure for a detailed analysis of the structure of each group and their relationships (see also [6]):

- The prototypes are projected in a two-dimensional space (possibly three) using a projection of Sammon, which retains the best initial distances between prototypes [38].
- The size of the disks representing the prototype is proportional to the density associated with each prototype.
- The color of each prototype depends on the cluster to which it is associated.
- Neighborhood connections (local topology) are represented by a segment connecting the neighboring prototypes.
- Local values of density and variability allow us to estimate the density variations in the representation space. These variations are represented in the form of contour lines. The projection of contour lines in the plane is operated by a projection of the Gaussian mixture in the space of representation.

This visualization provides information on both inter-group structure (number of clusters, similarities between clusters) but also intra-group structure (local topology, local density and density variation within the cluster, and data variability).

4.2. Visualization examples

We applied this method to eight artificial and real databases, using a Self-Organizing Map algorithm to learn prototypes.

Figures 10 and 11 show some visualization examples that can be obtained from low-dimensional datasets.

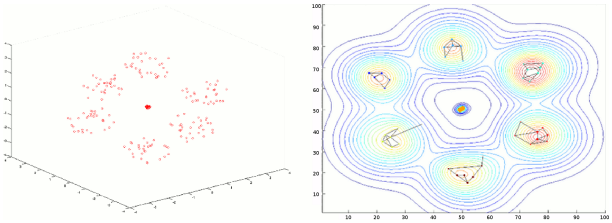


Figure 10: "Hepta" dataset (left) and their visualization (right).

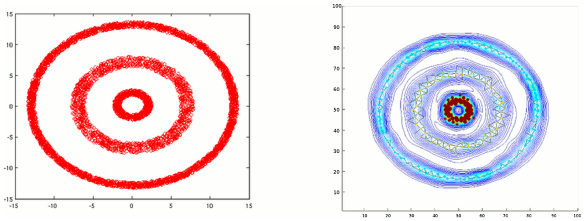


Figure 11: "Rings" dataset (left) and their visualization (right).

One notices that the data structure is well preserved by the quantization and clustering algorithm and is well represented by the visualization process. The data density is easily represented by the size of the prototypes and the level lines. Furthermore, these lines allow two-dimensional view of the general form of the different

clusters and their relative size. Visualization of connections, added to the different colors associated with the prototypes, allow for a visual description of the segmentation of data into different clusters. In addition, visualization is sufficiently detailed to allow representation of complex data distribution, as illustrated in Figures 11.

Figures 12 to 14 show some examples of visualizations that can be obtained from real data. " Iris" data describes three different species of flowers using four features. The "Ants" data describe the activity of each individual of a colony of ants (11 features). Finally, "Children" data is a description of time spent in various gaming activities in a group of children (8 features).

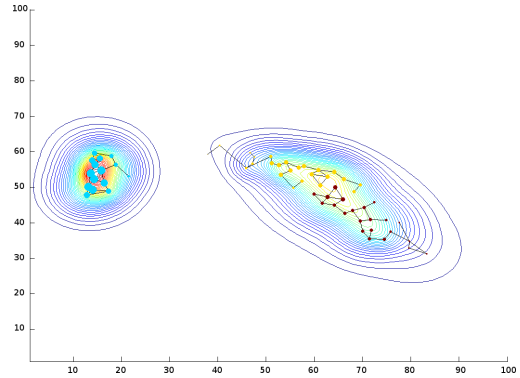


Figure 12: Visualization of "Iris" data.

The visualization of these databases, which have small size but dimension greater than three, illustrates the ability of the visualization method to project the relevant information in a two-dimensional space. For example, the " Iris" data (Fig. 12) are structured into two distinct groups, one of these groups is further subdivided into two very close subgroups. The three clusters are automatically discovered by the clustering algorithm and correspond to three distinct species of flowers.

Regarding "Ants" data (Fig. 13), each cluster detected by the algorithm corresponds to a behavior and a different social role within the colony (hunters, nurses, cleaners, guards, etc.. ..). Here, there is no clear separation in terms of density between the groups, which means that certain behaviors are possible intermediates. The existence of these intermediaries are known in biology, especially thanks to the presence of generalist ants which can perform any task, based on the needs of the colony [37].

Finally, the data "Children" [39] (Fig. 14) represent the activities of kindergarten children playing at recess. The data are divided into two sets of density fairly well separated, each subdivided into two subsets. The central

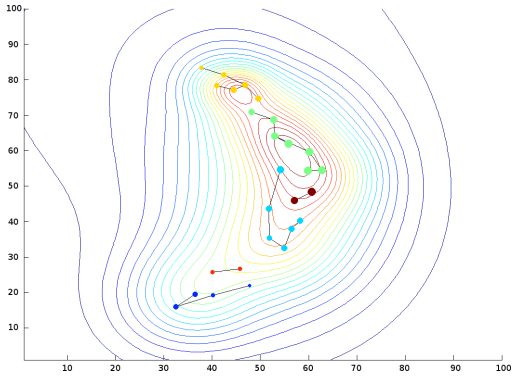


Figure 13: Visualization of "Ants" data.

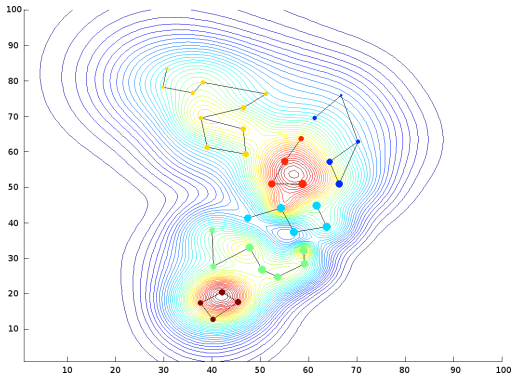


Figure 14: Visualization of "Children" data.

subgroup itself is subdivided into three clusters by the algorithm. It is interesting to note that, overall, group order from top to bottom corresponds to an increase in the age of the child and increase the complexity of game activities. The yellow group is composed almost exclusively of children in the first year of kindergarten, while the vast majority of children in the last year are in the brown group. The subdivision of the two intermediate years into four clusters reflects individual differences in the dynamics of child development. The decrease in density between the blue group and the green group separates the child spending most of their time in social games (with their peers) of children playing mostly alone. This indicates that a child who began playing with others will not return, or rarely, to solitary play. All this information is in agreement with the domain knowledge [40, 39].

5. Conclusion

In this paper, we proposed a new data structure modeling method, based on the learning of prototypes.

A new coclustering algorithm is also proposed, as an example of adaptation of the main algorithm to solve different kind of problems. We applied this algorithm to analyze characteristics of spatial and social organization in an ant colony. Obtained results are easy to read and understand, and are perfectly compatible with biologists knowledge.

We finally proposed a method of visualization able to enhance the data structure within and between groups. We have shown, using some artificial and real examples, the relevance of the proposed method.

References

[1] P. Lyman, H. R. Varian, How Much Information, http://www.sims.berkeley.edu/how-much-info-2003 (2003).

[2] J. Gehrke, F. Korn, D. Srivastava, On computing correlated aggregates over continual data streams, in: Special Interest Group on Management of Data Conference, 2001, pp. 13–24.

[3] G. S. Manku, R. Motwani, Approximate frequency counts over data streams, in: Very Large Data Base, 2002, pp. 346–357.

[4] C. C. Aggarwal, J. Han, J. Wang, P. S. Yu, A framework for clustering evolving data streams, in: Very Large Data Base, 2003, pp. 81–92.

[5] T. Kohonen, Self-Organizing Maps, Springer-Verlag, Berlin, 2001.

[6] G. Cabanes, Y. Bennani, Coupling Clustering and Visualization for Knowledge Discovery from Data, in: Proceeding of the International Joint Conference on Neural Networks (IJCNN'11), San Jose, USA, 2011, pp. 2127–2134.

[7] T. M. Martinetz, K. J. Schulten, A "neural-gas" network learns topologies, in: T. Kohonen, K. Mäkisara, O. Simula, J. Kangas (Eds.), Artificial Neural Networks, Elsevier Science Publishers, Amsterdam, 1991, pp. 397–402.

[8] T. Kohonen, Self-Organization and Associative Memory, Springer-Verlag, Berlin, 1984.

[9] B. Silverman, Using kernel density estimates to investigate multi-modality, Journal of the Royal Statistical Society, Series B 43 (1981) 97–99.

[10] S. R. Pamudurthy, S. Chandrakala, C. C. Sakhar, Local density estimation based clustering, Prodeeding of International Joint Conference on Neural Networks (2007) 1338–1343.

[11] B. W. Silverman, Density Estimation for Statistics and Data Analysis, Chapman & Hall/CRC, 1986.

[12] E. L. J. Bohez, two level cluster analysis based on fractal dimension and iterated function systems (ifs) for speech signal recognition, IEEE Asia-Pacific Conference on Circuits and Systems (1998) 291–294.

[13] M. F. Hussin, M. S. Kamel, M. H. Nagi, An efficient two-level SOMART document clustering through dimensionality reduction, in: ICONIP, 2004, pp. 158–165.

[14] E. E. Korkmaz, A two-level clustering method using linear linkage encoding, in: International Conference on Parallel Problem Solving From Nature, Lecture Notes in Computer Science, Vol. 4193, Springer-Verlag, 2006, pp. 681–690.

[15] V. G. Kaburlasos, S. E. Papadakis, Granular self-organizing map (grSOM) for structure identification, Neural Networks 19 (5) (2006) 623–643.

[16] A. Ultsch, Clustering with SOM: U*C, in: Proceedings of the Workshop on Self-Organizing Maps, 2005, pp. 75–82.

[17] L. Vincent, P. Soille, Watersheds in digital spaces: An efficient algorithm based on immersion simulation, IEEE Trans. Pattern Anal. Mach. Intell. 13 (1991) 583–598.

[18] S.-H. Yue, P. Li, J.-D. Guo, S.-G. Zhou, Using greedy algorithm: DBSCAN revisited II, Journal of Zhejiang University SCIENCE 5 (11) (2004) 1405–1412.

[19] G. Cabanes, Y. Bennani, A local density-based simultaneous two-level algorithm for topographic clustering, in: Proceeding of the International Joint Conference on Neural Networks (IJCNN), 2008, pp. 1176–1182.

[20] G. Cabanes, Y. Bennani, A simultaneous two-level clustering algorithm for automatic model selection, in: Proceedings of the International Conference on Machine Learning and Applications (ICMLA'07), 2007, pp. 316–321.

[21] M. Halkidi, Y. Batistakis, M. Vazirgiannis, Cluster Validity Methods, Special Interest Group on Management of Data Record 31 (2,3) (2002) 40–45, 19–27.

[22] A. Ben-Hur, A. Elisseeff, I. Guyon, A stability based method for discovering structure in clustered data, in: Pacific Symposium on Biocomputing, Vol. 7, 2002, pp. 6–17.

[23] D. L. Davies, D. W. Bouldin, A cluster separation measure, IEEE Transactions on Pattern Recognition and Machine Intelligence 1 (2) (1979) 224–227.

[24] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, AAAI Press, 1996, pp. 226–231.

[25] S. Guha, B. Harb, Approximation algorithms for wavelet transform coding of data streams, IEEE Transactions on Information Theory 54 (2) (2008) 811–830.

[26] J. Shi, J. Malik, Normalized Cuts and Image Segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (8) (2000) 888–905.

[27] G. Karypis, E.-H. Han, V. Kumar, Chameleon: Hierarchical clustering using dynamic modeling, IEEE Computer 32 (8) (1999) 68–75.

[28] A. P. Dempster, N. M. Laird, D. B. Rubin, Maximum likelihood from incomplete data via the EM algorithm, Journal of the Royal Statistical Society, Series B 39 (1977) 1–38.

[29] G. Cabanes, Y. Bennani, C. Chartagnat, D. Fresneau, Topographic connectionist unsupervised learning for RFID behavior data mining, in: Q. Z. Sheng, Z. Maamar, S. Zeadally, M. Cameron (Eds.), The Second International Workshop on RFID Technology (IWRT), INSTICC PRESS, 2008, pp. 63–72.

[30] G. Cabanes, Y. Bennani, Unsupervised topographic learning for spatiotemporal data-mining, Advances in Artificial Intelligence 2010, Article ID 832542, 12 pages.

[31] B. Mirkin, Mathematical Classification and Clustering, volume 11 of Nonconvex Optimization and Its Application., 1996.

[32] S. C. Madeira, A. L. Oliveira, Biclustering algorithms for biological data analysis: a survey, Trans. on Computational Biology and Bioinformatics 1 (1) (2004) 24–45. doi:.2.

[33] M. Cottrell, P. Letrémy, E. Roy, Analysing a contingency table with kohonen maps: A factorial correspondence analysis, in: IWANN, 1993, pp. 305–311.

[34] T. Hoang, M. Olteanu, SOM biclustering – coupled self-organizing maps for the biclustering of microarray data, in: IDAMAP 03, Workshop Notes, 2003, pp. 40–46.

[35] A. Banerjee, I. Dhillon, J. Ghosh, S. Merugu, D. S. Modha, A generalized maximum entropy approach to Bregman coclustering and matrix approximation, Journal of Machine Learn-

ing Research 8 (2007) 1919–1986.
[36] D. Fresneau, Biologie et comportement social d'une fourmis ponérine néotropicale (*Pachycondyla apicalis*), Ph.D. thesis, Université Paris-Nord (Paris 13), Paris (1994).
[37] B. Hölldobler, E. Wilson, The ants, Harvard University Press, 1990.
[38] J. Sammon Jr., A nonlinear mapping for data structure analysis, IEEE Transactions on Computer 18 (5) (1969) 401–409.
[39] S. Barbu, G. Cabanes, G. Le Maner-Idrissi, Boys and girls on the playground: Sex differences in social development are not stable across early childhood, PLoS ONE 6 (1) (2011) e16407.
[40] D. P. Fromberg, D. Bergen, Play from birth to Twelve: Contexts, perspectives, and Meanings, Routledge, New York, 2006.