# Self-Organizing Maps for clustering and visualization of bipartite graphs

Madalina Olteanu[1] & Nathalie Villa-Vialaneix[2]

[1] *SAMM, Université Paris 1, 90 rue de Tolbiac, 75634 Paris cedex 13 -*
[2] *INRA, UR 0875 MIA-T, BP 52627, 31326 Castanet Tolosan cedex - France France*

**Résumé.** Les graphes (souvent appelés réseaux) ont connu un intérêt croissant ces dernières années car on les retrouve de manière naturelle dans un nombre important d'applications en sciences sociales, biologie, informatique... Cet article propose une méthode de fouille de données pour visualiser et classer les sommets d'une classe particulière de graphes, les graphes bipartis. La méthode proposée est basée sur un algorithme de carte auto-organisatrice et s'appuie sur une extension de cette approche à des données décrites par une matrice de dissimilarité.

**Mots-clés.** graphe, graphe biparti, classification non supervisée, visualisation, carte auto-organisatrice (SOM)

**Abstract.** Graphs (also frequently called networks) have attracted a burst of attention in the last years, with applications to social science, biology, computer science... The present paper proposes a data mining method for visualizing and clustering the nodes of a peculiar class of graphs: bipartite graphs. The method is based on a self-organizing map algorithm and relies on an extension of this approach to data described by a dissimilarity matrix.

**Keywords.** graph, bipartite graph, clustering, visualization, self-organizing map (SOM)

## 1 Introduction

Graphs (also frequently called networks) have attracted a burst of attention in the last years, with applications to social science, biology, computer science... A significant class of graphs arises naturally in a number of real-life applications: bipartite graphs. In such graphs, vertices are labeled by two distinct labels (say, "1" and "2"), such that no pair of vertices with the same label share an edge. Examples of such graphs are people attending events, scientists publishing articles, actors playing in movies, people working in firms.

Combining clustering and visualization has been proven a useful tool to help the user decipher the main characteristics of a graph [12]. Several approaches have been developed so far to perform this task: (i) using a node clustering prior to visualizing the graph induced by the clustering, which is a simplified representation of the graph in

which clusters are symbolized by a single node; (ii) visualizing the whole graph by adding constraints on the node locations [6] that can be derived from a prior node clustering or (iii) combining in one step clustering and visualization by using topology preserving methods such as self-organizing maps (SOM) [3, 14]. In the present article, we propose to extend SOM to bipartite graphs. This approach is based on a combination of the so-called *relational SOM* that can handle data described by a dissimilarity matrix (see Section 2.1) with ideas taken from the "korresp" algorithm which is designed to process contingency tables (see Section 2.2). The following section describes more precisely the SOM algorithm and its extension to non-vectorial data. It finally describes the "bipartite SOM" approach.

# 2 SOM for bipartite graphs

## 2.1 SOM for numeric data and extensions to data described by a dissimilarity matrix

The Self-Organizing Map (SOM) algorithm aims at mapping $n$ input data $x_1, \ldots, x_n \in \mathbb{R}^d$ into a low dimensional grid composed of $U$ neurons. A prototype $p_u$, taking values in the same space as the input data, is associated to each unit $u \in \{1, \ldots, U\}$ of the grid. The grid is equipped with a distance $d$ between neurons (usually chosen as the length of the shortest path on the grid). The algorithm aims at clustering together similar observations while preserving the original topology of the data set on the map (i.e., close observations are clustered into close neurons on the map). The original algorithm for numerical vectors iterates over the two steps below:

- an *assignment step* in which one observation (on-line version) is picked at random and is affected to the closest prototype: $f(x_i) = \arg\min_{u=1,\ldots,U} \|x_i - p_u\|$,
- a *representation step* in which all prototypes are updated according to the new assignment by mimicking a stochastic gradient descent scheme: $p_u^{\text{new}} = p_u^{\text{old}} + \mu H\left(d\left(f(x_i), u\right)\right)\left(x_i - p_u^{\text{old}}\right)$, where $H$ is the neighborhood function verifying the assumptions $H : \mathbb{R}^+ \to \mathbb{R}^+$, $H(0) = 1$ and $\lim_{x\to+\infty} H(x) = 0$, and $\mu$ is a training parameter. Generally, $H$ and $\mu$ are supposed to be decreasing with the number of iterations during the training procedure.

In the case where $(x_i)_i$ do not take value in $\mathbb{R}^d$ but are described by a dissimilarity matrix $D = (\delta_{ii'})_{i,i'=1,\ldots,n}$ that contains a measure of pairwise dissimilarities between entities $(x_i)_i$, several approaches have been proposed to extend the SOM algorithm: some choose to use prototypes that take values in $(x_i)_i$ [10, 4], leading to a discrete optimization scheme. However, this method, called "median SOM" increases the computational time, while prototypes remain restricted to the original data set and may generate possible sampling or sparsity issues. *Relational SOM* [9, 13] overcomes this difficulty by defining

the prototypes as (implicit) "convex combinations" of the original data: $p_u \sim \sum_{i=1}^{n} \gamma_{ui} x_i$[1]. More precisely, the two steps of the algorithm described above are modified by:

- *assignment step*:
$$f(x_i) = \arg \min_{u=1,\ldots,U} (D\gamma_u)_i - \frac{1}{2}\gamma_u^T D \gamma_u \tag{1}$$
which is exactly equivalent to the assignment step in the standard SOM if the dissimilarity matrix $D$ is the squared Euclidean distance;

- *representation step*:
$$\gamma_u^{\text{new}} \leftarrow \gamma_u^{\text{old}} + \mu H(d(f(x_i), u)) \left(\mathbf{1}_i - \gamma_u^{\text{old}}\right) \tag{2}$$
where $\mathbf{1}_i$ is a vector with a single non null coefficient at the $ith$ position, equal to one. Once again, this modification is exactly equivalent to the representation step in the standard SOM if the dissimilarity matrix $D$ is the squared Euclidean distance.

The following section explains how to use the relational SOM algorithm for clustering the nodes of bipartite graphs.

## 2.2 Bipartite SOM for bipartite graphs

In this section, the following notations will be used: $\mathcal{G} = (V, E, L, W)$ denotes a bipartite graph for which

- $V = \{x_1, \ldots, x_n\}$ is the set of vertices of the graph;
- $L = (c_i)_{i=1,\ldots,n}$ is a set of labels $c_i \in \{1, 2\}$ for the nodes $x_i$. The number of vertices labeled "1" is denoted by $n_1$ and the number of vertices labeled "2" is denoted by $n_2$ (and thus $n_1 + n_2 = n$). For the sake of simplicity, the vertices will then be supposed to be ordered according to their labels: $c_1 = c_2 = \ldots = c_{n_1} = 1$ and $c_{n_1+1} = \ldots = c_{n_1+n_2} = 2$;
- $E$ is the set of edges, which is a subset of $\{(x_i, x_j) : \ c_i = 1 \text{ and } c_j = 2\}$;
- $W$ (optionally) is a $n_1 \times n_2$-dimensional weight matrix that possibly encodes the strength of the link between two nodes ($W_{ij} \geq 0$ and $W_{ij} = 0 \Leftrightarrow (x_i, x_{n_1+j}) \notin E$).

When faced with such data, a common approach is to define the univariate projections of the graph, i.e., the graphs $\mathcal{G}^l$ $(l = 1, 2)$ with vertices $(x_i^{(l)})_{i=1,\ldots,n_l} := \{x_i : \ c_i = l\}$[2],

[1] Note that $\sum_{i=1}^{n} \gamma_{ui} x_i$ has no real meaning since, as in the case of graphs, the sum might not be defined between entities $x_i$ which a priori take values in any (possibly non-vectorial) space. However the notation is used symbolically in reference to a pseudo-euclidean space embedding, as defined in [7].

[2] In the following, we will, depending on the context, use the notation $x_i$ or $x_i^{(1)}$ $(i = 1, \ldots, n_1)$ for nodes labeled "1" and $x_j$ or $x_{j-n_1}^{(2)}$ $(j = n_1 + 1, \ldots, n_1 + n_2)$ for nodes labeled "2".

linked by edges that are weighted by the number of common neighbors shared by the two edges in the bipartite graph. However, [8] shows that a clustering based on one of the two projections gives unreliable results. Since then, several methods have been proposed to handle the peculiar case of bipartite graph clustering [1, 17, 16], most of them being based on a modification of the modularity quality criterion [11] for fitting the structure of bipartite graphs.

Our approach is rather different since we intend to simultaneously perform clustering and visualization using SOM, as explained in the introduction. The problem at hand is very similar to the one solved with the so-called "korresp" algorithm [5] in which the values of two factor variables summarized by a contingency table are simultaneously clustered. This method is based on a modification of the standard SOM that is similar to factor correspondence analysis. Using a similar idea, we propose to deal with bipartite graphs by defining a *augmented profile* for the prototypes. Unlike the "korresp" case, the update of the augmented profiles is not based on the row/column profiles but on the bipartite structure of the graph itself. As in the "korresp" algorithm, the vertices labeled, respectively, "1" and "2" are iteratively processed. More precisely,

- first, a dissimilarity measure is computed for all pairs of nodes in the two projection graphs $\mathcal{G}^l$, $l = 1, 2$, denoted by $(\delta^{(l)}_{ii'})_{i,i'=1,\dots,n_l}$. This dissimilarity measure can be, for instance, the shortest path length between two nodes or the distance induced by one of the numerous kernels for graphs:

$$\delta^{(l)}_{ii'} = \sqrt{K(x^{(l)}_i, x^{(l)}_i) + K(x^{(l)}_{i'}, x^{(l)}_{i'}) - 2K(x^{(l)}_i, x^{(l)}_{i'})}$$

  with $K$ being e.g., a kernel based on a regularization of the graph Laplacian (see [15]). The dissimilarity measure defines the *reduced profile* of $(x^{(l)}_i)$, which is the $n_l$-dimensional vector $\left(\delta^{(l)}_{i,1}, \dots, \delta^{(l)}_{i,n_l}\right)$;

- then, similarly as in the relational SOM algorithm, prototypes are initialized as (symbolic) convex combinations of the original inputs, with two parts (each specific to one of the two labels): $p_u = \left(p^{(1)}_u, p^{(2)}_u\right)$ with $p^{(l)}_u = \sum_{i=1}^{n_l} \gamma^{(l)}_{ui} x^{(l)}_i$, $\gamma^{(l)}_{ui} \geq 0$ and $\sum_{i=1}^{n_l} \gamma^{(l)}_{ui} = 1$ for $l = 1, 2$. As in the "korresp" algorithm, the method iterates over the following steps:
  - a node $x^{(1)}_i$ is picked at random within the nodes labeled "1" and is affected to its closest prototype using the distance calculation of relational SOM based on the corresponding reduced prototypes $(p^{(1)}_u)_u$, as in Equation (1);
  - the full prototypes are then updated using their augmented profiles and a gradient-descent like step as in Equation (2). More precisely, the part of the prototype that corresponds to label "1" is updated as usual:

$$\forall\, j = 1, \dots, n_1,\ \gamma^{\text{new}}_{uj} = \gamma^{\text{old}}_{uj} + \mu H\left(d\left(f(x_i), u\right)\right)\left(\mathbf{1}_{ji} - \gamma^{\text{old}}_{uj}\right)$$

(with $\mathbf{1}_{ji} = 1$ if $j = i$ and 0 otherwise) and the part of the prototype that corresponds to label "2" is updated using the neighborhood of the node $x_i^{(1)}$:

$$\forall\, j = n_1+1, \ldots, n_1+n_2,\ \gamma_{uj}^{\text{new}} = \frac{\gamma_{uj}^{\text{old}} + \mu H\left(d\left(f(x_i), u\right)\right)\left(\sum_{k\in\mathcal{N}(x_i)} \mathbf{1}_{jk} - \gamma_{uj}^{\text{old}}\right)}{1 + \mu H\left(d\left(f(x_i), u\right)\right)(d_i - 1)}$$

where $\mathcal{N}(x_i)$ is the set of neighboring nodes and $d_i$ is the degree of node $x_i^{(1)}$. The constant $\frac{1}{1+\mu H(d(f(x_i),u))(d_i-1)}$ is used so that the sum over $j = n_1+1, \ldots, n_1+n_2$ of the $\gamma_{uj}$ is still equal to 1.

– the last two steps are symmetric to the first two and concern the nodes labeled "2".

# 3 Conclusion and perspectives

The present paper introduces a novel approach to handle clustering and visualization of bipartite graphs. The method is currently being tested on a large dataset of board members of CAC 40 firms. Relational SOM and the "korresp" algorithm are implemented in the R package **SOMbrero**[3], the extension described in this paper is scheduled to be included in a future release of the package.

# References


[1] M.J. Barber. Modularity and community detection in bipartite networks. *Physical Review E*, 76:066102, 2007.

[2] J. Boelaert, L. Bendhaïba, M. Olteanu, and N. Villa-Vialaneix. SOMbrero: an r package for numeric and non-numeric self-organizing maps. In *Proceedings of WSOM 2014*, 2014. Forthcoming.

[3] R. Boulet, B. Jouve, F. Rossi, and N. Villa. Batch kernel SOM and related laplacian methods for social network analysis. *Neurocomputing*, 71(7-9):1257–1273, 2008.

[4] B. Conan-Guez, F. Rossi, and A. El Golli. Fast algorithm and implementation of dissimilarity self-organizing maps. *Neural Networks*, 19(6-7):855–863, 2006.

[5] M. Cottrell, P. Letrémy, and E. Roy. Analyzing a contingency table with Kohonen maps: a factorial correspondence analysis. In J. Cabestany, J. Mary, and A. (Eds.) Prieto, editors, *Proceedings of International Workshop on Artificial Neural Networks (IWANN 93)*, Lecture Notes in Computer Science, pages 305–311. Springer Verlag, 1993.

[6] P. Eades and M.L. Huang. Navigating clustered graphs using force-directed methods. *Journal of Graph Algorithms and Applications*, 4(3):157–181, 2000.

[7] L. Goldfarb. A unified approach to pattern recognition. *Pattern Recognition*, 17(5):575–582, 1984.


[3] **SOMbrero** is available on R-Forge at http://sombrero.r-forge.r-project.org/ or can be used on-line at http://shiny.nathalievilla.org/sombrero, see [2] for further details.

[8] R. Guimerà, M. Sales-Pardo, and L.A.N. Amaral. Module identification in bipartite and directed networks. *Physical Review E*, 76:036102, 2007.

[9] B. Hammer and A. Hasenfuss. Topographic mapping of large dissimilarity data sets. *Neural Computation*, 22(9):2229–2284, September 2010.

[10] T. Kohohen and P.J. Somervuo. Self-organizing maps of symbol strings. *Neurocomputing*, 21:19–30, 1998.

[11] M.E.J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review, E*, 69:026113, 2004.

[12] A. Noack. Energy models for graph clustering. *Journal of Graph Algorithms and Applications*, 11(2):453–480, 2007.

[13] M. Olteanu and N. Villa-Vialaneix. On-line relational and multiple relational SOM. *Neurocomputing*, 2013. Forthcoming.

[14] F. Rossi and N. Villa-Vialaneix. Optimizing an organized modularity measure for topographic graph clustering: a deterministic annealing approach. *Neurocomputing*, 73(7-9):1142–1163, 2010.

[15] A.J. Smola and R. Kondor. Kernels and regularization on graphs. In M. Warmuth and B. Schölkopf, editors, *Proceedings of the Conference on Learning Theory (COLT) and Kernel Workshop*, Lecture Notes in Computer Science, pages 144–158, 2003.

[16] K. Suzuki and K. Wakita. Extracting multi-facet community structure from bipartite networks. In *Proceedings of International Conference on Computational Science and Engineering (CSE '09)*, volume 4, pages 312–319, Vancouver, BC, Canada, 2009.

[17] P. Zhang, J. Wang, X. Li, M. Li, Z. Di, and Y. Fan. Clustering coefficient and community structure of bipartite networks. *Physica A*, 387:6869–6875, 2008.