Multiple kernel self-organizing maps

Madalina Olteanu, Nathalie N. Villa-Vialaneix, Christine Cierco-Ayrolles

Multiple Kernel Self-Organizing Maps

Madalina Olteanu[1], Nathalie Villa-Vialaneix[2] and Christine Cierco-Ayrolles[2]

1- SAMM, Université Paris 1, Paris, France
2- Unité BIA, INRA de Toulouse, Auzeville, France

Abstract. In a number of real-life applications, the user is interested in analyzing several sources of information together: a graph combined with the additional information known on its nodes, numerical variables measured on individuals and factors describing these individuals... The combination of all sources of information can help him to understand the dataset in its whole better. The present article focuses on such an issue, by using self-organizing maps. The use a kernel version of the algorithm allows us to combine various types of information and automatically tune the data combination. This approach is illustrated on a simulated example.

1 Introduction

In a number of real-life applications, the user is interested in analyzing several sources of information together: a graph combined with the additional information known on its nodes, numerical variables measured on individuals and factors describing these individuals... The combination of all sources of information can help him to understand the dataset in its whole better. The present article focuses on such an issue, by using self-organizing maps (SOM).

SOM has already been extended to the framework of non numerical data, using various approaches. Historically, median SOM, which consists in replacing the standard computation of the prototypes by an approximation in the original dataset, was introduced first. This principle was used to extend SOM to dissimilarity data in [1]. Several variants of the original algorithm have been proposed since: [2, 3, 4, 5] extended the framework of SOM for dissimilarity data by proposing a more flexible representation for the prototypes, either in batch or in online versions. A closely related approach is to rely on kernels to map the original data into an (implicit) Euclidean space where the standard SOM can be used [6, 7, 8]. Several kernels have been designed to handle complex data such as strings, nodes in a graph or several graphs [9].

In the present paper, we propose to extend kernel SOM by using a combination of kernels, each of them dedicated to a particular aspect of the entire dataset. The methodology is described in Section 2 and illustrated in Section 3.

2 Method

The purpose of the present paper is to design a self-organizing map algorithm in which data coming from various sources can be used as inputs in an optimal combination. Let us suppose that we are given D sets of input data,

$(x_i^d)_{i=1,\ldots,n,\, d=1,\ldots,D}$, all measured on the same individuals ($i = 1, \ldots, n$), and taking values in arbitrary spaces $(\mathcal{G}_d)_d$. For example, $(x_i^d)_i$ may represent p-dimensional numerical vectors or factor (non-numerical) variables describing the n individuals. More specifically, $(x_i^d)_i$ may describe the n nodes in a graph and the kind of relation between them or some text variables naming or describing the data.

Multiple kernel SOM

Suppose also that for each of these D datasets, a *kernel* is known, i.e., a function $K_d : \mathcal{G}_d \times \mathcal{G}_d \to \mathbb{R}$ such that K_d is symmetric ($\forall\, z, z' \in \mathcal{G}_d$, $K_d(z, z') = K_d(z', z)$) and positive ($\forall\, N \in \mathbb{N}$, $\forall\, (z_j)_{j=1,\ldots,N} \subset \mathcal{G}_d$ and $\forall\, (\alpha_j)_{j=1,\ldots,N} \subset \mathbb{R}$, $\sum_{j,j'} \alpha_j \alpha_{j'} K_d(z_j, z_{j'}) \geq 0$). For example, the linear kernel, the Gaussian kernel or the polynomial kernel, among others, may be used for numerical variables. For nodes of a graph, the heat kernel [10], the commute time kernel [11] or any other regularized version of the Laplacian can be used as kernels [12], while for textual variables, string kernels, based on the number of occurences of common substrings [13] may be considered.[1]

From these D kernels, a new kernel can be defined by computing a convex combination: $K(x_i, x_{i'}) = \sum_{d=1}^{D} \alpha_d K_d(x_i^d, x_{i'}^d)$, where $\alpha_d \in [0, 1]$, $\sum_d \alpha_d = 1$ and $x_i = (x_i^1, \ldots, x_i^d) \in \mathcal{G} = \mathcal{G}_1 \times \ldots \times \mathcal{G}_d$. Obviously, such an application is also symmetric and positive and thus [14] proves that there is a Hilbert space, $(\mathcal{H}, \langle ., . \rangle)$ (called the *feature space*) and an application $\phi : \mathcal{G} \to \mathcal{H}$ (called the *feature map*) such that $K(x_i, x_{i'}) = \langle \phi(x_i), \phi(x_{i'}) \rangle$.

In the SOM algorithm, the individuals $i = 1, \ldots, n$ have to be clustered into a low dimensional grid made of M neurons, $\{1, \ldots, M\}$. These neurons are related to each other by a neighborhood relationship, h. Each neuron is also represented by prototypes, $(p_j)_j$ ($j = 1, \ldots, M$) taking values in the previously defined feature space $\mathcal{H}$. Following the general framework of kernel SOM described in [6, 15, 8], the prototypes are written as a convex combination of the input data in the feature space $\mathcal{H}$: $p_j = \sum_i \gamma_{ji} \phi(x_i)$. The (squared-)distance between an individual x_i and a prototype p_j is then computed by

$$\|p_j - \phi(x_i)\|^2 = K(x_i, x_i) - 2 \sum_{l=1}^{n} \gamma_{jl} K(x_i, x_l) + \sum_{l,l'=1}^{n} \gamma_{jl} \gamma_{jl'} K(x_l, x_{l'}). \qquad (1)$$

Online multiple kernel SOM

If the $(\alpha_d)_d$ are given, the standard kernel SOM aims at optimizing (over $(\gamma_{ji})_{ji}$) the following energy function :

$$\mathcal{E}((\gamma_{ji})_{ji}, (\alpha_d)_d) = \sum_{i=1}^{n} \sum_{j=1}^{M} h\left(f(x_i), j\right) \left\| \phi^{\alpha}(x_i) - p_j^{\alpha}(\gamma_j) \right\|_{\alpha}^2 ,$$

where $f(x_i) \in \{1, \ldots, M\}$ is the index of the cluster to which x_i is assigned. The notations ϕ^{α}, p_j^{α} and $\|.\|_{\alpha}$ are used to emphasize that these quantities depend

[1] Note that this methodology can also been applied to combine several kernels on the same data (e.g., different kinds of string kernels, encapsulating different features of the texts).

on α. The novelty of this paper is to include the optimization of the convex combination of kernels into the on-line algorithm which trains the map. More precisely, a stochastic gradient descent step is added to the original on-line SOM algorithm to optimize the energy $\mathcal{E}((\gamma_{ji})_{ji}, (\alpha_d)_d)$ over both $(\gamma_{ji})_{ji}$ and $(\alpha_d)_d$. To perform the stochastic gradient descent step on the (α_d), the computation of the derivative of $\mathcal{E}|_{x_t} = \sum_{j=1}^{M} h\left(f(x_t), j\right) \left\|\phi^{\alpha}(x_t) - p_j^{\alpha}(\gamma_j)\right\|_{\alpha}^2$ (the contribution of the randomly chosen observation x_t to the energy) with respect to α is needed. As $\frac{\partial}{\partial \alpha_d}\left[K(x_i, x_{i'})\right] = K_d(x_i^d, x_{i'}^d)$, we have that

$$\mathcal{D}_d := \frac{\partial \mathcal{E}|_{x_t}}{\partial \alpha_d} = \sum_{j=1}^{M} h\left(f(x_t), j\right) \left(K_d(x_t^d, x_t^d) - 2\sum_{l=1}^{n} \gamma_{jl} K_d(x_t^d, x_l^d) + \sum_{l,l'=1}^{n} \gamma_{jl}\gamma_{jl'} K_d(x_l^d, x_{l'}^d) \right)$$

Following an idea similar to that of [16], the SOM is trained by performing, alternatively, the standard steps of the SOM algorithm (i.e., affectation and representation steps) and a gradient descent step for the $(\alpha_d)_d$. The full process is given in Algorithm 1.

Algorithm 1 Multiple online kernel SOM

1: $\forall\, j = 1, \ldots, M$ and $\forall\, i = 1, \ldots, n$, initialize γ_{ji}^0 randomly in $\mathbb{R}$ such that $\gamma_{ji}^0 \geq 0$ and $\sum_{i=1}^{n} \gamma_{ji}^0 = 1$ and $\forall\, d = 1, \ldots, D$, initialize α_d^0 in $[0, 1]$ such that $\sum_d \alpha_d^0 = 1$.

2: **for** $t = 1 \to T$ **do**

3: Randomly choose an input $x_l \in \{x_i\}_i$

4: *Assignment step.* Find the unit of the closest prototype

$$f^t(x_l) \leftarrow \arg\min_{j=1,\ldots,M} \left\| \phi^{\alpha^{t-1}}(x_l) - p_j^{\alpha^{t-1}}(\gamma_j^{t-1}) \right\|_{\alpha^{t-1}}$$

5: *Representation step.* Update all the prototypes: $\forall\, j = 1, \ldots, M$,

$$\gamma_{ji}^t \leftarrow \gamma_{ji}^{t-1} + \mu(t) h(f^t(x_l), j)\left(\delta_{li} - \gamma_{ji}^{t-1}\right)$$

with $\delta_{li} = 1$ if $l = i$ and 0 otherwise.

6: *Gradient descent step.* Update the kernel:

$$\forall\, d = 1, \ldots, D,\ \alpha_d^t \leftarrow \alpha_d^{t-1} + \nu(t)\mathcal{D}_d^t \qquad \text{and} \qquad K^t \leftarrow \sum_d \alpha_d^t K_d.$$

7: **end for**

To ensure that the gradient step respects the constraints on α ($\alpha_d \geq 0$ and $\sum_d \alpha_d = 1$), the following strategy can be applied: first, the gradient $\mathcal{D}_d^t = \left(\frac{\partial \mathcal{E}^t|_{x_t}}{\partial \alpha_d}\right)_d$ is reduced and it is projected so that the non-negativity of α is

ensured (the reduced gradient projection procedure is described in [17, 18, 16]). This leads to the descent direction $\mathcal{D}_d^t$. The descent step is decreased with the standard rate of ν_0/t.

In practice, to ensure similar scales for all kernels, a normalization step is performed before computing the first convex combination: all kernels are scaled to have a unit norm, according to the Frobenius norm. The tuning procedure then automatically finds out the (α_d) that give the best map, i.e., that best organizes the observations according to the global kernel K. As illustrated in [19], this can be particularly useful when the data can be described by several kernels, all briging up a different insight on the similarities between individuals, as it is often the case in social sciences (see [20, 21] for a discussion on this topic).

Finally, the additional cost of tuning the (α_d) is moderate because it is performed online: in practice, we simply multiplied by two the number of steps usually needed to train the map and this choice led to a converging algorithm.

3 Application

In this section, a simple example is used to test the algorithm and illustrate its behavior in the presence of complementary information. 200 observations, divided into 8 groups (numbered from 1 to 8 in the following), were generated using three different kinds of data: 1/ an unweighted graph, simulated similarly as described in [22]. The nodes of the groups 1 to 4 and the nodes of the groups 5 to 8 could not be distinguished in the graph structure: the edges within these two sets of nodes were randomly generated with a probability equal to 0.3. The edges between these two sets of nodes were randomly generated with a probability equal to 0.01; 2/ numerical data coming from two dimensional Gaussian distributions. The variables were simulated by Gaussian vectors with independent components having a variance equal to 0.3 and a mean equal to $(0, 0)$ for the odd groups and to $(1, 1)$ for the even groups; 3/ a factor with 2 levels. Observations of groups 1, 2, 5, and 7 were affected to the first level and observations of the other groups to the second level.

Hence, only the combined knowledge of the three datasets gave access to the eight original groups. The online multiple kernel SOM algorithm was applied to this problem with the commute time kernel for the graph and a Gaussian kernel for the numerical data and for the factor variable (re-coded in disjunctive form). The algorithm was compared with a standard kernel SOM approach using one of the three datasets only. It was also compared to kernel SOM using numerical and factor data or all the three datasets but used as if they were issued from the same dataset with a single Gaussian kernel (when the graph was added to the numerical and factor data, it was under the form of its adjacency matrix). The performance of the different approaches were compared by using the normalized mutual information (NMI) [23] compared to the original classes. This measure has values between 0 and 1 and is equal to 1 when the two partitions are identical. The results are given in Figure 1.

The multiple kernel SOM clearly outperformed all the other approaches, in-

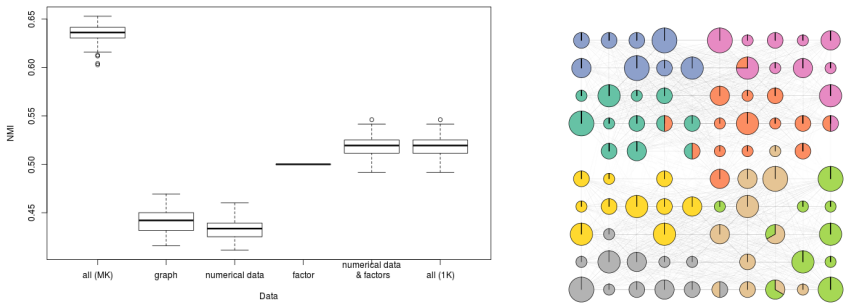

Fig. 1: NMI (left) of online kernel SOM when using all datasets in the multiple kernel framework or only one dataset or all datasets in a single kernel framework. Example of a map resulting from the use of the algorithm (right): each node represents a neuron of the map and has an area proportional to the number of observations classified inside. The colors display the proportion of the eight original classes in the clusters.

cluding the one where all datasets are simply aggregated together. The example of the resulting map (given on the right hand side of the figure) shows a good classification and a good organization according to the three types of information: the eight groups are well distinguished by the algorithm.

4 Conclusion

This article proposes a way to combine multiple sources of heterogenous information in self-organizing maps. Our approach can help to select the most relevant sources of information or the most relevant ways to describe similarities between observations. The computational cost of such an approach is moderate but can however be prohibitive when run with very large datasets or a large number of kernels. A sparse approach is currently under study for improving this aspect of the algorithm.

References

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

[2] 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.

[3] B. Hammer, A. Hasenfuss, F. Rossi, and M. Strickert. Topographic processing of relational data. In Bielefeld University Neuroinformatics Group, editor, *Proceedings of the 6th Workshop on Self-Organizing Maps (WSOM 07)*, Bielefeld, Germany, September 2007.

[4] B. Hammer, A. Gisbrecht, A. Hasenfuss, B. Mokbel, F.M. Schleif, and X. Zhu. Topographic mapping of dissimilarity data. In J. Laaksonen and T. Honkela, editors, *Advances in Self-Organizing Maps (Proceedings of the 8th Workshop on Self-Organizing Maps,*

WSOM 2011), volume 6731 of *Lecture Notes in Computer Science*, pages 1–15, Espoo, Finland, 2011. Springer.

[5] M. Olteanu, N. Villa-Vialaneix, and M. Cottrell. On-line relational som for dissimilarity data. In P.A. Estevez, J. Principe, P. Zegers, and G. Barreto, editors, *Advances in Self-Organizing Maps (Proceedings of WSOM 2012)*, volume 198 of *AISC (Advances in Intelligent Systems and Computing)*, pages 13–22, Santiago, Chile, 2012. Springer Verlag, Berlin, Heidelberg.

[6] D. Mac Donald and C. Fyfe. The kernel self organising map. In *Proceedings of 4th International Conference on knowledge-based intelligence engineering systems and applied technologies*, pages 317–320, 2000.

[7] P. Andras. Kernel-Kohonen networks. *International Journal of Neural Systems*, 12:117–135, 2002.

[8] 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.

[9] T. Gärtner. *Kernel for Structured Data*. World Scientific, 2008.

[10] R.I. Kondor and J. Lafferty. Diffusion kernels on graphs and other discrete structures. In *Proceedings of the 19th International Conference on Machine Learning*, pages 315–322, 2002.

[11] F. Fouss, A. Pirotte, J.M. Renders, and M. Saerens. Random-walk computation of similarities between nodes of a graph, with application to collaborative recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 19(3):355–369, 2007.

[12] 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.

[13] C. Watkins. Dynamic alignment kernels. In A. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 39–50, Cambridge, MA, USA, 2000. MIT P.

[14] N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3):337–404, 1950.

[15] N. Villa and F. Rossi. A comparison between dissimilarity SOM and kernel SOM for clustering the vertices of a graph. In *6th International Workshop on Self-Organizing Maps (WSOM)*, Bielefield, Germany, 2007. Neuroinformatics Group, Bielefield University.

[16] A. Rakotomamonjy, F.R. Bach, S. Canu, and Y. Grandvalet. SimpleMKL. *Journal of Machine Learning Research*, 9:2491–2521, 2008.

[17] D. Luenberger. *Linear and Nonlinear Programming*. Addison-Wesley, 1984.

[18] F. Bonnans. *Optimisation Continue*. Dunod, 2006.

[19] M. Olteanu, S. Massoni, and N. Villa-Vialaneix. Which distance use when extracting typologies in sequence analysis? An application to school to work transitions. Submitted for publication.

[20] A. Abbott and Tsay. Sequence analysis and optimal matching methods in sociology. *Review and Prospect. Sociological Methods and Research*, 29(1):3–33", 2000.

[21] L. Wu. Some comments on "Sequence analysis and optimal matching methods in sociology, review and prospect". *Sociological Methods and Research*, 29(1):41–64, 2000.

[22] A. Condon and R.M. Karp. Algorithms for graph partitioning on the planted partition model. *Random Structures and Algorithms*, 18(2):116–140, 2001.

[23] L. Danon, A. Diaz-Guilera, J. Duch, and A. Arenas. Comparing community structure identification. *Journal of Statistical Mechanics*, page P09008, 2005.