ESANN 2017 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 26-28 April 2017, i6doc.com publ., ISBN 978-287587039-1. Available from http://www.i6doc.com/en/.

Accelerating stochastic kernel SOM

Jérôme Mariette[1], Fabrice Rossi[2], Madalina Olteanu[2] and Nathalie Villa-Vialaneix[1]

1- MIAT, Université de Toulouse, INRA, 31326 Castanet-Tolosan, France

2- SAMM, EA 4543, Université Paris 1, F-75634 Paris, France

Abstract. Analyzing non vectorial data has become a common trend in a number of real-life applications. Various prototype-based methods have been extended to answer this need by means of kernalization that embed data into an (implicit) Euclidean space. One drawback of those approaches is their complexity, which is commonly of order the square or the cube of the number of observations. In this paper, we propose an efficient method to reduce complexity of the stochastic kernel SOM. The results are illustrated on large datasets and compared to the standard kernel SOM. The approach has been implemented in the last version of the R package **SOMbrero**[1].

1 Introduction

In a number of real-life applications, data cannot be described by numerical variables or cannot be compared in a meaningful way using standard Euclidean metrics. This is the case, for instance, with graphs, trees, categorical data, ... A generic way to handle this type of dataset is to use a measure of similarity or dissimilarity between the data, which can be expertly designed.

Self-organizing maps (SOM) have first been extended to the framework of non numerical data through the median SOM approach [1, 2]. The method replaces the standard computation of the prototypes by an approximation in the original dataset. However, the representation of the prototypes is very restricted and generates representation issues and therefore poor performances. A different approach is taken in the relational and kernel versions of SOM [3, 4, 5, 6, 7]. In these versions, the original data are embedded into an (implicit) Euclidean or pseudo-Euclidean space in which the algorithm is performed using only the dissimilarity or the kernel. A side effect of this method is that it requires that the prototypes are expressed as a convex combination of the original data, leading to a quadratic complexity for the batch and the on-line version of the algorithm [8]. The induced computation cost is a serious bottleneck to analyze datasets with more than a few thousands observations.

In the present paper, we propose an efficient method to reduce the complexity of the stochastic kernel SOM (K-SOM) so that it is linear in the number of observations. The approach is based on an efficient on-line update of the prototypes, which only requires to store two matrices. The standard K-SOM algorithm is briefly described in Section 2 and our proposal for reducing its computational complexity is discussed in Sections 3 (kernel version) and 4 (adaptation to arbitrary dissimilarity datasets). Finally, the approach is illustrated in Section 5.

[1]`https://cran.r-project.org/web/packages/SOMbrero`, version 1.2

2 A brief description of the stochastic K-SOM

In the sequel, we consider a set of observations $(x_i)_{i=1,\ldots,n}$ taking values in an arbitrary space $\mathcal{X}$. $\mathcal{X}$ is equipped with a kernel $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ that provides pairwise similarity between the observations, $K_{ij} := K(x_i, x_j)$. K is assumed symmetric and positive which ensures the existence of a unique Hilbert space, $(\mathcal{H}, \langle ., . \rangle_{\mathcal{H}})$, and a unique mapping ϕ from $\mathcal{X}$ into $\mathcal{H}$ such that $K(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{H}}$ [9].

In K-SOM [4, 5], the U prototypes $(p_u)_{u=1,\ldots,U}$, representing the units of the low dimensional grid on which the observations are projected, are expressed as convex combinations of the images of $(x_i)_{i=1,\ldots,n}$ by ϕ: $p_u = \sum_{i=1}^n \alpha_{ui}\phi(x_i)$, with $\alpha_{ui} \geq 0$ and $\sum_{i=1}^n \alpha_{ui} = 1$.

In the on-line version of the algorithm, the following two steps are performed for each iteration t, starting from randomly chosen values α_{ui}^0:

- the assignment step in which an observation x_i is picked at random and affected to its closest prototype, $f^t(x_i) = \arg\min_{u=1,\ldots,U} \|\phi(x_i) - p_u^t\|^2_{\mathcal{H}}$. This step is equivalent to $f^t(x_i) = \arg\min_{u=1,\ldots,U} \sum_{j,j'=1}^n \alpha_{uj}^t \alpha_{uj'}^t K_{jj'} - 2\sum_{j=1}^n \alpha_{uj}^t K_{ij}$, for $K_{ij} = K(x_i, x_j)$;
- the representation step in which the prototypes are updated according to a gradient descent-like approach: $\forall u = 1, \ldots, U$, $p_u^{t+1} \leftarrow p_u^t + \mu(t)h^t(d(f^t(x_i), u))(\phi(x_i) - p_u^t)$, which is equivalent to $\alpha_u^{t+1} \leftarrow \alpha_u^t + \mu(t)h^t(d(f^t(x_i), u))(\mathbf{1}_i - \alpha_u^t)$ for d the distance between units on the grid, h^t a decreasing function such that $h^t(0) = 1$ and $\lim_{x \to +\infty} h^t(x) = 0$, $\mathbf{1}_i$ the n-dimensional vector in which only the entry indexed by i is non zero and equal to 1 and μ is a positive number. Usually $\mu(t)$ and h^t are chosen so as to vanish when t increases.

The complexity of the assignment and representation steps are, respectively, $\mathcal{O}(n^2U)$ and $\mathcal{O}(nU)$, which leads to a total complexity of $\mathcal{O}(n^2U)$ for one iteration. To obtain good convergence properties, the algorithm requires a number of iterations of the order of βn, as shown in [10], yielding a complexity of $\mathcal{O}(\beta n^3 U)$. Hence, the K-SOM is not adapted to large datasets and cannot be used to analyze more than a few thousands observations. [11] have proposed two approximate versions to overcome this issue, using sparse representations of the prototypes or DR pre-processing techniques. In the present article, we propose a modification of the SOM update steps so as to produce a solution which is exactly equivalent to the original algorithm, but with a reduced computational time.

3 Reducing the complexity of stochastic K-SOM

Using a re-formulation of the assignment step, the computational cost of one iteration can be reduced to $\mathcal{O}(nU)$. At iteration t, an observation x_i is picked at random within $(x_{i'})_{i'=1,\ldots,n}$ and the assignment step is written $f^t(x_i) =$

$\arg\min_{u\in 1,\ldots,U} A^t_u - 2B^t_{ui}$ in which $A^t = \left(\sum_{j,j'=1}^n \alpha^t_{uj}\alpha^t_{uj'}K_{jj'}\right)_{u=1,\ldots,U}$ is a vector of size U and $B^t = \left(\sum_{j=1}^n \alpha^t_{uj}K_{i'j}\right)_{u=1,\ldots,U,\, i'=1,\ldots,n}$ is a $(U \times n)$-matrix.

The updates of A^t and B^t are performed during the representation step, which is thus equivalent to $\forall u = 1,\ldots,U,\quad \alpha^{t+1}_u = (1-\lambda_u(t))\alpha^t_u + \lambda_u(t)\mathbf{1}_i$, in which $\lambda_u(t) = \mu(t)h(d(f^t(x_i),u))$. This leads to the following updates:

$$B^{t+1}_{ui'} = \sum_{j=1}^n \alpha^{t+1}_{uj}K_{i'j} = (1-\lambda_u(t))B^t_{ui'} + \lambda_u(t)K_{i'i},$$

and the vector A is modified using the equation given by

$$A^{t+1}_u = \sum_{j,j'=1}^n \alpha^{t+1}_{uj}\alpha^{t+1}_{uj'}K_{jj'} = (1-\lambda_u(t))^2A^t_u + \lambda_u(t)^2K_{ii} + 2\lambda_u(t)(1-\lambda_u(t))B^t_{ui}.$$

Thus, the computational cost of the algorithm can be decomposed into:

- computing $A^0_u = \sum_{j,j'=1}^n \alpha^0_{uj}\alpha^0_{uj'}K_{jj'}$ and $B^0_{ui'} = \sum_{j=1}^n \alpha^0_{uj}K_{i'j}$ for all $u = 1,\ldots,U$ and all $i' = 1,\ldots,n$. This step is performed only once and has a complexity of $\mathcal{O}(n^2)$ and $\mathcal{O}(n)$ for A^0 and B^0, respectively;
- performing the assignment step which complexity does not depend on n since the distances are pre-computed and stored;
- the update of A^t and B^t (representation step), which have respective complexities equal to $\mathcal{O}(U)$ and $\mathcal{O}(nU)$.

Since the assignment and representation steps are usually performed $\mathcal{O}(\beta n)$ times, the total complexity of the algorithm is dominated by $\mathcal{O}(\beta n^2 U)$. This computational cost is obtained using the additional storage of A^t and B^t which requires a memory of $\mathcal{O}(U)$ and $\mathcal{O}(nU)$, respectively.

4 The case of dissimilarity data

In some cases, the dataset is described by a measure of dissimilarity ($\delta(x_i,x_j) = \delta_{ij}$), rather than by a kernel. We will suppose that the dissimilarity is symmetric ($\delta_{ij} = \delta_{ji}$), with positive elements ($\delta_{ij} \geq 0$) and a null diagonal ($\delta_{ii} = 0$) but it might be not Euclidean. In [6, 7] are described relational versions of the SOM algorithm that are adapted to this case and are based on a pseudo-Euclidean framework [12]. The principle is fairly similar to the one described in Section 2 except that the assignment step writes:

$$f^t(x_i) = \arg\min_{u=1,\ldots,U}\left(\sum_{j=1}^n \alpha^t_{uj}\delta_{ij} - \frac{1}{2}\sum_{j,j'=1}^n \alpha^t_{uj}\alpha^t_{uj'}\delta_{jj'}\right)$$

(the representation step is unchanged compared to the kernel version). The complexity of the method is thus identical to the one of the kernel SOM. In the case where the dissimilarity is computed from a kernel K by $\delta_{ij} = K_{ii} + K_{jj} - 2K_{ij}$, the dissimilarity SOM and the relational SOM are identical.

The adaptation of kernel SOM described in Section 3 has a straightforward equivalent in the case of the relational SOM: $A^t = \left(\sum_{j,j'=1}^n \alpha_{uj}^t \alpha_{uj'}^t \delta_{jj'}\right)_{u=1,\ldots,U}$ and $B^t = \left(\sum_{j=1}^n \alpha_{uj}^t \delta_{i'j}\right)_{u=1,\ldots,U,i'=1,\ldots,n}$. The assignment step is thus $f^t(x_i) = \arg\min_{u=1,\ldots,U} \left(B_{ui}^t - \frac{1}{2}A_u^t\right)$ and the updates of A^t and B^t are performed during the representation step as described in Section 3.

5 Application

In this section, different simulations are performed to compare the standard K-SOM to the accelerated version described in Sections 3 and 4. To assess the computational cost of the methods, three large size datasets are used:

- a graph, denoted by "polblogs", in which the 1,222 nodes are blogs on US politics (recorded in 2005 by [13][2]). The edges in this graph represent hyper-links between the blogs. For this dataset, the shortest path lengths between pairs of nodes have been computed and used in the relational version of this algorithm. This dissimilarity is not Euclidean.

- a DNA barcoding dataset, denoted by "cowrie", that contains 2,036 samples issued from the cowries family introduced in [14]. DNA barcoding data are composed of sequences of nucleotides, *i.e.*, sequences of "A", "C", "G", "T" letters in high dimension (hundreds or thousands of sites) allowing to assign biological specimens to a species. Only 1,414 samples were used (those corresponding to species with very few observations were removed). The Kimura-2P [15] dissimilarity is computed between pairs of sequences and used in the relational version of the algorithm. Again, this dissimilarity is not Euclidean.

- a (standard) numerical dataset, denoted by "wine", which is related to red variants of the Portuguese "Vinho Verde" wine [16][3]. The dataset contains 4 898 observations of 12 numeric variables based on physicochemical tests, such as the pH, the sulphates or the residual sugar. The Gaussian kernel has been computed between any pair of wines: $K_{ij} = e^{-\sigma\|x_i - x_j\|^2}$ with σ equals to the median of $\left\{\frac{1}{\|x_i - x_j\|^2}\right\}_{i<j}$.

Hence, the first two datasets are true relational dataset which requires to use an adapted version of the SOM algorithm to be processed, whereas the third one is a numeric dataset, which can be processed in a standard manner with the original SOM algorithm.

[2]The graph is available at `http://www-personal.umich.edu/~mejn/netdata/polblogs.zip`.
[3]The dataset is available at `https://archive.ics.uci.edu/ml/datasets/Wine+Quality`.

Table 1: CPU time in seconds of the different versions of the SOM algorithm (average over 100 maps and standard deviation between parenthesis)

	numeric SOM	rSOM	acc. rSOM
“polblogs”	*NA*	1112.34 (165.86)	**36.99** (4.49)
“cowrie”	*NA*	1715.40 (249.60)	**42.52** (2.32)
“Wine”	**16.40** (2.13)	8527.16 (874.58)	206.32 (38.24)

All simulations have been performed with the R package **SOMbrero**[4] that contains implementation of the standard numeric SOM and of the relational version of the algorithm. **SOMbrero** uses a stochastic learning, as is described in this article. Version 1.1 has been used to obtain the computational time provided by the original version of the algorithm and version 1.2 to obtain the computational time provided by the accelerated version. For a fixed random seed, the results of both versions (version 1.1 with the standard implementation and version 1.2 with the accelerated implementation) were always exactly identical for the dissimilarity versions.

All maps were trained for a 10×10 grid with respectively 6000 (“polblogs”), 7000 (“cowrie”) and 25000 (“wines”) iterations. All grids were equipped with a piecewise linear neighborhood with the Euclidean distance between units on the grid calculated using the unit coordinates in $\mathbb{N}^{*2}$. Following the same methodology, 100 maps were trained using the standard numeric version of the SOM on “wine”, the only numerical dataset, as a basis for comparison: the computational cost of the numeric version of the algorithm is $\mathcal{O}(\beta n \times Up)$ in which $p \ll n$ is the number of variables in the dataset (see [8]). It is thus expected that this direct approach is still faster than the relational version. Results (in terms of clustering on the map) are also different between the relational version and the numeric version but these differences are related to the choice of a good dissimilarity in a given dataset, which is out of the scope of this paper. We thus only report computational times here.

Table 1 provides the computational cost in seconds obtained over the 100 maps (mean and standard deviation) for the numeric SOM, the standard relational or kernel SOM and the accelerated version. *NA* (not applicable) values in the numeric SOM column of both “polblogs” and “cowrie” datasets come from the fact that these datasets are not in the framework of the numeric SOM.

Results show that the accelerated version allows to highly reduce the computational time of the relational or kernel SOM. When comparing the results obtained on the different datasets, the accelerated version is 30 times faster than the original approach on “polblogs” and more than 40 times faster on both “cowrie” and “wine”. Compared to the numerical version, the computational cost is increased but still comparable.

[4]`https://cran.r-project.org/web/packages/SOMbrero`

6 Conclusion

We have proposed an efficient version of the stochastic K-SOM and relation SOM, with a complexity of $\mathcal{O}(\beta n^2 U)$. The experiments performed on several datasets demonstrate that the presented method strongly decreases the overall computational time. With the introduction of this method, a step is taken to allow the SOM algorithm to deal with massive non numerical datasets.

References

[1] T. Kohonen 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] T. Graepel, M. Burger, and K. Obermayer. Self-organizing maps: generalizations and new optimization techniques. *Neurocomputing*, 21:173–190, 1998.

[4] 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.

[5] K.W. Lau, H. Yin, and S. Hubbard. Kernel self-organising maps for classification. *Neurocomputing*, 69:2033–2040, 2006.

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

[7] M. Olteanu and N. Villa-Vialaneix. On-line relational and multiple relational SOM. *Neurocomputing*, 147:15–30, 2015.

[8] F. Rossi. How many dissimilarity/kernel self organizing map variants do we need? In T. Villmann, F.M. Schleif, M. Kaden, and M. Lange, editors, *Advances in Self-Organizing Maps and Learning Vector Quantization (Proceedings of WSOM 2014)*, volume 295 of *Advances in Intelligent Systems and Computing*, pages 3–23, Mittweida, Germany, 2014. Springer Verlag, Berlin, Heidelberg.

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

[10] 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.

[11] J. Mariette, M. Olteanu, and N. Villa-Vialaneix. Efficient interpretable variants of online SOM for large dissimilarity data. *Neurocomputing*, 225:31–48, 2017.

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

[13] L.A. Adamic and N. Glance. The political blogosphere and the 2004 us election: divided they blog. In *Proceedings of the 3rd LINKDD Workshop*, pages 36–43, New York, NY, USA, 2005. ACM Press.

[14] C.P. Meyer and G. Paulay. DNA barcoding: error rates based on comprehensive sampling. *PLoS Biology*, 3(12), 11 2005.

[15] M. Kimura. A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences. *Journal of Molecular Evolution*, 16:111–120, 1980.

[16] P. Cortez, A. Cerdeira, F. Almeida, T. Matos, and J. Reis. Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems*, 47(4):547–553, 2009.