Speeding up the self organizing map for dissimilarity data

Aïcha El Golli

Projet AxIS
INRIA-Rocquencourt,
Domaine De Voluceau, BP 105,
78153 Le Chesnay Cedex, France
(e-mail: aicha.elgolli@inria.fr)

Abstract. This paper presents an optimization of the self organizing map for dissimilarity data. In fact, practical clustering algorithms for dissimilarity data are extremely costly because of the calculation of the dissimilarity table and require multiple data scans to achieve convergence. Therefore, we propose working on sample set data to speed up the training process and also to handle large data set.
Keywords: Self organizing map, dissimilarity, random sampling.

1 Introduction

The self organizing map (SOM) [Kohonen, 1982a], [Kohonen, 1982b] and [Kohonen, 1997] is considered as a clustering method and also a projection method. It can be used at the same time both to reduce the amount data by clustering, and for projecting the data nonlinearly onto a lower dimensional display. Due to its unsupervised learning and topology preserving proprieties it has proven to be especially suitable in analysis of complex systems. The SOM algorithm implements a nonlinear topology preserving mapping from a high-dimensional input metric vector data space, $\mathbb{R}^p$, into a two-dimensional network or grid of neurons. To understand what the SOM really shows, it is important to understand that it actually performs two tasks: vector quantization and vector projection. Vector quantization creates from the original data a smaller, but still representative, data set to be worked with. The set of prototype vectors reflects the properties of the data space. The projection performed by the SOM is nonlinear and restricted to a regular grid (the map grid). The SOM tries to preserve the topology of the data space rather than relative distances.

The Kohonen's SOM is based on the notion of center of gravity and unfortunately, this concept is not applicable to many kinds of complex data. The extension of the self organizing map to dissimilarity data [El Golli *et al.*, 2004] is an alternative solution for new forms of complex data and so allows its process on dissimilarity measures rather than on raw data. With this alternative only the definition of a dissimilarity for each type of data is necessary to apply the method and so treat complex data. This extension is an adaptation of the batch-learning version of the SOM to dissimilarity data. At each stage, the learning is performed by alternating an assignment step and a representation step.

We focus on the problem of clustering large data set. In fact, when we work with this kind of data this extension of the SOM to complex data is extremely costly because of the calculation of the dissimilarity table. In order to solve this problem we propose to work on a sample set either on the whole learning set.

The paper is organized as follows: we first recall our adaptation of the SOM algorithm in its batch version for the dissimilarity data. Then we describe the algorithm working with a sample set.

2 Batch self-organizing map for dissimilarity data

The SOM can be considered as carrying out vector quantization and/or clustering while preserving the spatial ordering of the prototype vectors (also called referent vectors) in one or two dimensional output space. The SOM consists of neurons organized on a regular low-dimensional map. More formally, the map is described by a graph (C, Γ). C is a set of m interconnected neurons having a discrete topology defined by Γ.

For each pair of neurons (c, r) on the map, the distance $\delta(c, r)$, is defined as the shortest path between c and r on the graph. This distance imposes a neighborhood relation between neurons. The batch training algorithm is an iterative algorithm in which the whole data set (noted Ω) is presented to the map before any adjustments are made. We note z_i an element of Ω and $\mathbf{z_i}$ the representation of this element in the space D called representation space of Ω. In our case, the main difference with the classical batch algorithm is that the representation space is not $\mathbb{R}^p$ but an arbitrary set on which dissimilarity (denoted d) is defined.

Each neuron c is represented by a set $A_c = z_1, ..., z_q$ of elements of Ω with a fixed cardinality q, where z_i belongs to Ω. A_c is called an individual referent. We denote A the set of all individual referents, i.e. the list $A = A_1, ..., A_m$. In our approach each neuron has a finite number of representations. We define a new adequacy function d^T from $\Omega \times P(\Omega)$ to $\mathbb{R}^+$ by:

$$d^T(z_i, A_c) = \sum_{r \in C} K^T(\delta(r, c)) \sum_{z_j \in A_r} d^2(\mathbf{z_i}, \mathbf{z_j}) \tag{1}$$

d^T is based on the kernel positive function K. $K^T(\delta(c, r))$ is the neighborhood kernel around the neuron r. This function is such that $\lim_{|\delta| \longrightarrow \infty} K(\delta) = 0$ and allows us to transform the sharp graph distance between two neurons on the map ($\delta(c, r)$) into a smooth distance. K is used to define a family of functions K^T parameterized by T, with $k^T(\delta) = K(\frac{\delta}{T})$. T is used to control the size of the neighborhood [Anouar *et al.*, 1997], [Dreyfus *et al.*, 2002]; when the parameter T is small, there are few neurons in the neighborhood. A simple example of K^T is defined by $K^T(\delta) = e^{-\frac{\delta^2}{T^2}}$.

During the learning, we minimize a cost function E by alternating an assignment step and a representation step. During the assignment step, the assignment function f assigns each individual z_i to the nearest neuron, here in terms of the function d^T:

$$f(z_i) = arg \min_{c \in C} d^T(z_i, A_c) \tag{2}$$

If there is equality, we assign the individual z_i to the neuron with the smallest label.

During the representation step, we have to find the new individual referents A^* that represent the set of observations in the best way in terms of the following cost function E:

$$E(f, A) = \sum_{z_i \in \Omega} d^T(z_i, A_{f(z_i)}) = \sum_{z_i \in \Omega} \sum_{r \in C} K^T(\delta(f(z_i), r)) \sum_{z_j \in A_r} d^2(\mathbf{z_i}, \mathbf{z_j}) \tag{3}$$

This function calculates the adequacy between the induced partition by the assignment function and the map referents A.

The criterion E is additive so this optimization step can be carried out independently for each neuron. Indeed, we minimize the m following functions:

$$E_r = \sum_{z_i \in \Omega} K^T(\delta(f(z_i), r)) \sum_{z_j \in A_r} d^2(\mathbf{z_i}, \mathbf{z_j}) \tag{4}$$

In the classical batch version, this minimization of E function is immediate because the positions of the referent vectors are the averages of the data samples weighted by the kernel function.
Here is the algorithm:

Initialization: iteration $k = 0$, choose an initial codebook A^0. Fix $T = T_{max}$ and the total number of iterations N_{iter}

<u>Iteration</u>: At iteration k, the set of individual referents of the previous iteration A^{k-1} is known. Calculate the new value of T:

$$T = T_{max} * (\frac{T_{min}}{T_{max}})^{\frac{k}{N_{iter-1}}}$$

► **affectation step:** up date the affectation function f_{A^k} associated to the A^{k-1} referent. Affecting each individual z_i to the referent as defined in equation (2).

► **representation step:** determine the new codebook A^{k*} that minimizes the $E(f_{A^k}, A)$ function (with respect to A) A_c^{k*} is defined from equation (4).

Repeat **<u>Iteration</u>** until $T = T_{min}$

3 Incorporating sampling

The extension of the SOM to dissimilarity data (DisSom) is a solution for different kind of complex data since we can define a dissimilarity but the computational complexity constitute a problem when we have a large data sets. In order to handle large data sets, we need an efficient mechanism for reducing the size of the learning set of the DisSom. One approach to achieving this is via random sampling ($S \subset \Omega$), the key idea is to apply DisSom's clustering algorithm to the new learning set S drawn from the data set rather than the entire data set. Typically, the random sample S will fit in main memory and will be much smaller than the original data set. Consequently, significant improvements in execution times for DisSom can be realized. When we choose a random samples S of moderate sizes we preserve information about the geometry of clusters fairly accurately, thus enabling DisSom to correctly cluster the input. We propose to use the algorithm [**?**] for drawing a sample randomly from data using constant space.

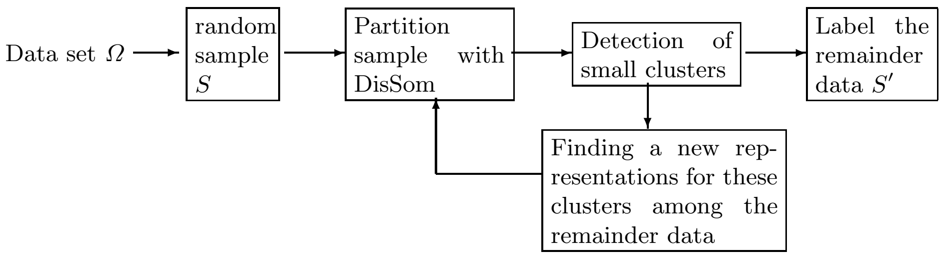
Data set Ω
random sample S
Partition sample with DisSom
Detection of small clusters
Label the remainder data S'
Finding a new representations for these clusters among the remainder data

Fig. 1. Overview of the steps of optimized DisSom

Once clustering of the random sample S is completed, the individual referent of each cluster is used to label the remainder of the data set (S'), where $\Omega = S \cup S'$ and $S \cap S' = \emptyset$. Each data point $z_i \in S'$ will be assigned to the closest individual referent of the map using the assignment function f (equation 2). But since we do not consider the entire data set, information about certain clusters may be missing in the input. As a result, our clustering algorithm may miss out certain clusters or incorrectly identify certain clusters. To this end, before labelling the remainder of the learning set we detect small clusters (c) and we have to find new individuals referent from the remainder data S' that minimizing the inertia criterion of the clusters c:

$$arg \min_{z_i \in S'} \sum_{z_j \in c} d^2(z_i, z_j)$$

To detect outliers we can use the Chernoff bounds. In fact, assuming that each cluster has a certain minimum size, we can use Chernoff bounds

[Motwani and Raghavan, 1995] to calculate the minimum sample size for which the sample contains, with high probability, at least a fraction *fr* of every clusters [Guha *et al.*, 1998].

The steps involved in clustering with DisSom are described in Figure 1. Since the learning set of the DisSom clustering algorithm is a set of randomly sampled points from original data set, the final k clusters involve only a subset of the entire set of points. In DisSom, the algorithm for assigning the appropriate cluster labels to the remaining data points employs a fraction of randomly selected individuals referent for each of the final k clusters. Each data point is assigned to the cluster containing the individual referent closest to it.

4 Conclusion

In this paper, we propose to speed up the self organizing map on dissimilarity data for large data sets. In fact, we propose to employ random sampling that allows to handle large data sets efficiently.

References

[Anouar *et al.*, 1997]F. Anouar, F. Badran, and S. Thiria. Self organized map, a probabilistic approach. 1997.

[Dreyfus *et al.*, 2002]G. Dreyfus, J.M. Martinez, M. Samuelides, M. Gordon, F. Badran, S. Thiria, and L. Hérault. *Réseaux de neurones méthodologie et applications*. Eyrolles, Paris, 2002.

[El Golli *et al.*, 2004]A. El Golli, B. Conan-Guez, and F. Rossi. a self-organizing map for dissimilarity data. In D. Banks, L. House, F. R. McMorris, P. Arabie, and W. Gaul, editors, *Classification, Clustering and Data Mining Application (Proceeding of IFCS)*, pages 61–68, Chicago, Illinois, 2004. Springer.

[Guha *et al.*, 1998]S. Guha, R. Rastogi, and K. Shim. Cure: An efficient clustering algorithm for large databases. In *In ACM SIGMOD Conf.*, pages 73–84, 1998.

[Kohonen, 1982a]T. Kohonen. Analysis of a simple self-organizing process. *Biol. cybern.*, 44:135–140, 1982.

[Kohonen, 1982b]T. Kohonen. Self-organized formation of topologically correct feature map. *Biol. Cybern*, 43:59–69, 1982.

[Kohonen, 1997]T. Kohonen. *Self-Organizing Maps*. Springer Verlag, New York, 1997.

[Motwani and Raghavan, 1995]R. Motwani and P. Raghavan. *Randomized algorithms*. In Cambridge university press, 1995.