Adaptive batch SOM for multiple dissimilarity data tables

Anderson B. S. Dantas

31 de outubro de 2011

Clustering methods organize a set of items into clusters such that items within a given cluster have a high degree of similarity, whereas those of different clusters have a high degree of dissimilarity. These methods have been widely applied in fields such as taxonomy, image processing, information retrieval and data mining.

There are two common representations of the objects upon which clustering can be based : feature data and relational data. When each object is described by a vector of quantitative or qualitative values, the set of vectors describing the objects is called a *feature data*. Alternatively, when each pair of objects is represented by a relationship, then it is called *relational data*. The most common case of relational data is when one has (a matrix of) dissimilarity data, say $R = [r_{kl}]$, where r_{kl} is the pairwise dissimilarity (often a distance) between objects k and l.

This paper extends the dynamic hard clustering algorithm for relational data into hard clustering algorithms that are able to partition objects taking into account simultaneously their relational descriptions given by multiple dissimilarity matrices. The main idea is to obtain a collaborative role of the different dissimilarity matrices to obtain a final partition.

The influence of the different dissimilarity matrices can not be equally important in the definition of the clusters in the final partition. Thus, to obtain a meaningful partition from all dissimilarity matrices, the relational hard clustering algorithms given in this paper are designed to give a partition and a prototype for each cluster as well as to learn a relevance weight for each dissimilarity matrix by optimizing an adequacy criterion that measures the fitting between the clusters and their representatives. These relevance

weights change at each algorithm's iteration and can either be the same for all clusters or different from one cluster to another.

The batch SOM algorithm for relational data based on multiple dissimilarity matrices is an iterative three-step algorithm (representation, weighting and affectation steps) in which the whole data set is presented to the map before that any adjustments are made.

The cost function of this batch SOM algorithm is given by:

$$J = \sum_{e_i \in E} \sum_{l=1}^{c} K^T(\delta(\chi(e_i), l)) D_{\boldsymbol{\lambda}_l}(e_i, G_l) \tag{1}$$

in which $D_{\boldsymbol{\lambda}_l}$ is the global matching between an example $e_i \in P_l$ and the cluster prototype $G_l \in E^{(q)}$, parameterized by the relevance weight vector $\boldsymbol{\lambda}_l = (\lambda_{l1}, \ldots, \lambda_{lp})$ of the dissimilarity matrices $\mathbf{D}_j$ into cluster $P_l \, (l = 1, \ldots, c)$.

The concept of neighbourhood is taken into account through kernel functions K which are positive and such that $\lim_{|x| \to \infty} K(x) = 0$. Moreover, K^T that is parameterized by T, is the neighbourhood kernel function that defines the influence region around each neuron r. The smaller the value of T, the fewer the neurons that belong to the neighbourhood of a given neuron r.

The main idea of the matching functions which are defined by a vector of weights in which the weights are estimated locally for each cluster is to compare clusters and their prototypes using a different matching measure associated with each cluster that changes at each iteration, i.e., the distance is not determined absolutely and is different from one cluster to another.

$$D_{\boldsymbol{\lambda}_l}(e_i, G_l) = \sum_{j=1}^{p} \lambda_{lj} D_j(e_i, G_l) = \sum_{j=1}^{p} \lambda_{lj} d_j(e_i, G_l)$$

$D_j(e_i, G_l) = \sum_{e \in G_l} d_j(e_i, e)$ is the local dissimilarity between an example $e_i \in P_l$ and the cluster prototype $G_l \in E^{(q)}$ on dissimilarity matrix $\mathbf{D}_j \, (j = 1, \ldots, p)$.

When T is kept fixed, the minimization of J is performed iteratively in three steps: representation, weighting and affectation.

During the representation step, the partition $P^{(t-1)} = (P_1^{(t-1)}, \ldots, P_c^{(t-1)})$ and the vectors of weights ${\boldsymbol{\lambda}_l}^{(t-1)} \, (r = 1, \ldots, c)$ are kept fixed. The cost function J is minimized with respect to the prototypes.

During the weighting step, the partition $P^{(t-1)} = (P_1^{(t-1)}, \ldots, P_c^{(t-1)})$ and the prototypes $G_l^{(t)} \in E^{(q)}\,(l = 1, \ldots, c)$ are fixed. The cost function J is minimized with respect to the vectors of weights.

During the affectation step, the reference vectors (prototypes) $G_l^{(t)} \in E^{(q)}\,(l = 1, \ldots, c)$ and the vectors of weights $\boldsymbol{\lambda}_l{}^{(t-1)}\,(r = 1, \ldots, c)$ are kept fixed. The cost function J is minimized with respect to the allocation function.

To compare the clustering results furnished by the clustering methods, an external index – the corrected Rand index (CR)– as well as the $F-measure$ and the overall error rate of classification $(OERC)$ will be considered.

CR index assesses the degree of agreement (similarity) between an a priori partition and a partition furnished by the clustering algorithm. Moreover, the CR index is not sensitive to the number of classes in the partitions or the distribution of the items in the clusters. Finally, CR index takes its values from the interval [-1,1], in which the value 1 indicates perfect agreement between partitions, whereas values near 0 (or negatives) correspond to cluster agreement found by chance.

The traditional $F - measure$ between class $P_i\,(i = 1, \ldots, m)$ and cluster $Q_j\,(j = 1, \ldots, K)$ is the harmonic mean of precision and recall. The $F - measure$ index takes its values from the interval [0,1], in which the value 1 indicates perfect agreement between partitions.

The $OERC$ index aims to measure the ability of a clustering algorithm to find out the *a priori* classes present in a data set.

All these data sets are described by a data matrix of "objects $\times$ real-valued attributes". Several dissimilarity matrices are obtained from these data matrices. One of these dissimilarity matrices has the cells that are the dissimilarities between pairs of objects computed taking into account simultaneously all the real-valued attributes. All the other dissimilarity matrices have the cells which are the dissimilarities between pairs of objects computed taking into account only a single real-valued attribute. In this paper, the dissimilarity between pairs of objects were computed according to the Euclidean (L_2) distance.