PartSOM: A Framework for Distributed Data Clustering Using SOM and K-Means

Flavius L. Gorgônio and José Alfredo F. Costa
Federal University of Rio Grande do Norte
Brazil

1. Introduction

Cluster analysis can be defined as the process of partition data into a certain number of clusters (or groups) of similar objects, where each group consists of similar objects amongst themselves (internal homogeneity) and different from the objects of the other groups (external heterogeneity), i.e., patterns in the same cluster should be similar to each other, while patterns in different clusters should not (Xu & Wunsch II, 2005). Typical clustering applications include pattern recognition, data mining, data compression, market segmentation and dimensionality reduction, among others.

More formally, given a set of N input patterns: $X = \{x_1, \ldots, x_N\}$, where each $x_i = (x_{i1}, \ldots, x_{ip})^{\mathrm{T}} \in \Re^p$ represents a p-dimensional vector and each measure x_{ij} represents a attribute (or variable) from dataset, a clustering process attempts to seek a K partition of X, denoted by $C = \{C_1, \ldots, C_K\}$, $(K \leq N)$ such that:

a) $C_i \neq \phi, i = 1 \ldots K;$

b) $\bigcup_{i=1}^{K} C_i = X;$

c) $C_i \cap C_j = \phi,\ i, j = 1 \ldots K; i \neq j.$

Traditional clustering algorithms operate over a single dataset. In most cases, before applying traditional algorithms on distributed databases, all distributed data must be centered on a main site. However, in some applications, e.g. medical and business, privacy-preserving and security policies disallow data combination, because confidential information can be reconstructed (Silva & Klusch, 2006).

In very large databases, the integration of several datasets into a single location is discouraged (Forman & Zhang, 2000). If an organization has very large distributed databases and needs to gather all the data to apply clustering algorithms, processing can demand excessive data transfers, which can be slow and expensive.

Moreover, any change that occurs in the distributed data, as for example, the inclusion of new information or alterations of existing data, will have to be updated into the central database. This demands a complex process of information updating, which causes a data transfer overload within the system.

For that reason, several algorithms have appear aimed at clustering dispersed data into several locations and summarizing results in a central unit, ensuring data security and confidentiality. This approach is known as distributed data clustering (DDC).

Clustering algorithms can be based on a wide variety of theories and techniques, including graph theory, combinatorial search techniques, fuzzy set theory, artificial neural networks, and kernels techniques (Xu & Wunsch II, 2005). Artificial neural networks are an important computational tool, with strong inspiration neurobiological and widely used in the solution of complex problems, which cannot be handled with traditional algorithmic solutions (Haykin, 1999). Applications for neural networks include pattern recognition, signal analysis and processing, analysis tasks, diagnosis and prognostic, data classification and clustering.

Competitive neural networks provide a family of algorithms used for data representation, visualization and clustering. Among the unsupervised neural network models, the self-organizing map (SOM) plays a major role. SOM features include information compression while trying to preserve the topological and metric relationship of the primary data space (Kohonen, 2001). The SOM network defines, via unsupervised learning, a mapping of a continuous p-dimensional space to a set of model vectors, or neurons, usually arranged as a 2-D array.

This work proposes a novel strategy for cluster analysis in distributed databases using a recently proposed architecture, named *partSOM*, and typical clustering algorithms, such as SOM and K-Means. In this approach, the clustering algorithm is applied separately in each distributed dataset, relative to database vertical partitions, to obtain a representative subset of each local dataset. In the sequence, these representative subsets are sent to a central site, which performs a fusion of the partial results. Next, a clustering algorithm is applied again to obtain a final result.

The main contribution of this paper is to show that, in situations where the volume of data is very large or when data privacy and security requirements impede consolidation at a single location, the results obtained with the application of this strategy justify its use.

The remainder of the chapter is organized as follows: section 2 presents a brief bibliographical review about distributed data clustering algorithms and section 3 describes the main aspects of the SOM and K-Means algorithm. Section 4 presents the proposed strategy, detailing its operation and the advantages obtained with its use. Section 5 describes the methodology used in the experiments and section 6 present the results of the application of the proposed strategy for some datasets, comparing them with the results obtained with the traditional approaches. Finally, section 7 presents conclusions and the direction of future algorithm research.

2. A Review on Distributed Data Clustering

Cluster analysis algorithms groups data based on the similarities between objects or patterns. The complexity of the cluster analysis process increases with data cardinality (N, the number of objects in a database) and dimensionality (p, the number of attributes). Clustering methods range from those that are largely heuristic to more formal procedures based on statistical models. The most frequently used methods are hierarchical (or heuristic) and partitioning (or iterative) methods. Several algorithms have been developed based on different strategies, including hierarchical clustering, vector quantization, graph theory, fuzzy logic, artificial neural networks, combinatory search, and others. Xu & Wunsch II (2005) and Berkhin (2006) present two recent surveys of clustering algorithms.

Hierarchical methods have been dominant in the clustering literature and proceed by stages, producing a sequence of partitions, each corresponding to a different number of clusters (Costa & Andrade Netto, 2001). Partitioning methods produce one partition with *K* groups, usually by minimizing some objective criterion. The most common method is the K-Means, which uses heuristics for reducing the within-group sum of squares. Each method has its advantages and drawbacks. Most partitional methods, for example, require a priori choice of the number of clusters and may be sensitive to initialization. Furthermore, many validation criteria are currently available, but if left to the user's choice, may result in disagreement due to the possible differences of the geometric structure imposed on the data space from clustering algorithm and the validation index.

Searching for clusters in high-dimensional databases is a non trivial task. Some common algorithms, such as traditional agglomerative hierarchical methods, are unsuitable for large datasets. Moreover, the increase in the number of attributes of each entrance not only has a negative influence on the processing time of the algorithm, but also hinders cluster identification. An alternative approach is to divide the database into partitions and to perform data clustering of each one separately.

Some current applications have databases that are large enough that cannot be maintained integrally in the main memory, even on very powerful computers. Kantardzic (2003) describe three approaches to solving this problem:

a. The data are stored in secondary memory and data subsets are clustered separately, obtaining the results in a subsequent stage that contains the whole group;
b. An algorithm of incremental grouping is used. Each element is individually stored in the main memory and associated to one of the existing groups or allocated to a new group;
c. A parallel implementation is used and several algorithms work simultaneously on the stored data.

Two approaches are frequently used to partition dataset: the first, and more common, is to divide the database horizontally, creating homogeneous subsets of the data, so that each algorithm operates on the same attributes, although treating of different registers. Another approach is to divide the database vertically, creating heterogeneous data subsets. In this case, each algorithm operates on the same registers, but handles on different sets of attributes.

There have been recent published studies about distributed data clustering. Forman & Zhang (2000) present a method that parallels several algorithms to obtain greater efficiency in the data mining process of distributed databases. These authors reinforce the need for reducing communication overload among the bases, to reduce processing time and to minimize the need for powerful machines with extended storage capacities.

Other factors that motivate the existence of distributed databases are related to security and privacy (Lam et al., 2004). Privacy-preserving clustering focuses its efforts on algorithms that ensure data privacy and security, as for example, in medical or business databases. Several organizations maintain geographically distributed databases as a form of increasing the security of their information. Thus, if security fails, the intruder can access only part of the information.

İnan et al. (2007) present a method for clustering horizontally partitioned databases, based on constructing the dissimilarity matrix of objects from different sites, which can be used for privacy preserving clustering. Vaidya & Clifton (2003) presents an approach for vertically partitioned databases using a distributed K-Means algorithm. On the other hand, Jagannathan et al. (2006) present a K-means variant algorithm for clustering horizontally partitioned databases. Oliveira & Zaïane (2004) propose a spatial data transformation method for protecting attributes values when sharing data for clustering, called RBT, which is independent of any clustering algorithm.

In databases with a large number of attributes, another approach is sometimes used to perform the analysis considering only an attribute subset, instead of considering all of them. An obvious difficulty with this approach is to identify which attributes are the most important in the cluster identification process. Some studies have used statistical methods such as Principal Components Analysis (PCA) and Factor Analysis to deal with this problem (Friedman & Meulman, 2004; Damian et al., 2007).

Kargupta et al. (2001) describe a PCA-based technique, called Collective Principal Component Analysis (CPCA), to cluster high-dimensional heterogeneous distributed databases. The authors focus on reducing data transfer rates between distributed sites.

He et al. (2005) analyze the influence of data types on the clustering process and present a strategy for dividing the group of attributes into two subsets, one with only the numerical attributes and the other with only the categorical ones. Authors then propose to cluster separately each of these subsets, using appropriate algorithms for each type. In the end, the results of each cluster are combined into a new database, which is once again submitted to a clustering algorithm for categorical data.

*Part*SOM is a simple and efficient architecture to cluster distributed datasets, based on multiples self-organizing maps disposed in a parallel arrangement (Gorgônio, 2009). This architecture applies SOM algorithm separately in each distributed dataset, corresponding to vertical partitions of data, to obtain a representative subset of each local dataset. In the sequence, these representative subsets are sent to a central site, which performs a fusion of the results and applies the SOM algorithm again to obtain the final result (Gorgônio and Costa 2008a; Gorgônio and Costa, 2008b).

There are two main advantages of the *part*SOM architecture over traditional approaches. At first, since only a reduced data volume is transferred between local and central units, the architecture is efficient in situations where the data volume is very large. Second, since only representative pattern are sent to central unit, the architecture is particularly interesting when data privacy and security policies disable data consolidation into a single location.

This work extends this approach, showing that strategy is efficient not only with SOM, but also using others clustering algorithms, such as K-Means or a combination of both in the same process. A series of experiments we carry out with the objective to determine the efficacy of proposed strategy, using a set of public domain databases and experimental results are compared with traditional SOM and K-Means approaches.

3. Clustering Algorithms

3.1 K-Means

K-Means is the most commonly-used clustering algorithm, particularly for its simplicity and ease of implementation. The algorithm objective is to group a set of *N* item in *K* groups, based on some similarity measure, normally the Euclidean distance, given for:

$$\|x_i - x_j\| = \sqrt{\sum_{f=1}^{p} \left(x_{if} - x_{jf}\right)^2} \quad (1)$$

where x_i and x_j represent two elements from dataset and x_{if} represents the *f*-th attribute from *i*-th element.

The algorithm starts choosing a random set of *K* centroids, where *K* value is previously defined by the user. Next, the algorithm calculates a distance matrix between each one of the dataset elements and the centroids.

In the following step, each element is associated with its nearest centroid. The value of each centroid then is recomputed, based on mean of the values of the elements that belongs to this centroid. A new distances matrix is calculated and the process repeats until the convergence. The algorithm finishes when each element is associated to the cluster represented for its centroid.

3.2 Self-Organizing Maps

The self-organizing feature map (SOM) has been widely studied as a software tool for visualizing high-dimensional data. Important features include information compression while preserving the topological and metric relationship of the primary data items (Kohonen, 2001). The SOM is composed of two layers of neurons: an input layer and an output (or competitive) layer. The output layer neurons are connected to adjacent neurons by a neighboring relation, defining the topology of the map.

Training is accomplished by presenting one input pattern x at a time in a random sequence and comparing it, in parallel, with all the reference vectors. The best match unit (BMU), which can be calculated using the Euclidean metric, represents the weight vector with the greatest similarity to that input pattern. Denoting the winning neuron by c, the BMU can be formally defined as the neuron for which

$$\|x - x_c\| = \min_i \{ \|x - x_i\| \} \quad (2)$$

where $\|.\|$ is the measure of distance.

The input is thus mapped to this location. The weight vectors of BMU as well as the neighboring nodes are moved closer to the input data vector. The magnitude of the attraction is governed by the learning rate. The SOM update rule for the weight vector of the unit *i* is

$$x(t+1) = x_i(t) + h_{ci}(t) \cdot \left[x(t) - x_i(t) \right] \quad (3)$$

where *t* denotes time, *x*(*t*) is the input vector randomly drawn from the input data set at time *t* and $h_{ci}(t)$ is the neighborhood kernel around the winning unit *c* at time *t*. This last term is a non-increasing function of time and of the distance of unit *i* from BMU and usually composed of two components: the learning rate function α(*t*) and the neighborhood function *h*(*d*,*t*):

$$h_{ci}(t) = \alpha(t) \cdot h(\|r_c - r_i\|, t) \tag{4}$$

where r_i denotes the location of unit *i* on the map grid.

As learning proceeds and new input vectors are given to the map, the learning rate α(*t*) gradually decreases to zero, according to the specified learning rate function type. Along with learning rate, the neighborhood radius decreases as well.

Despite the SOM is an excellent tool in clustering tasks, almost always, another complementary approaches may be needed to increase the obtained results, particularly when there are many clusters or cluster borders are not well defined. A number of methods for visualizing data relations in a trained SOM have been proposed, such as multiple views of component planes, mesh visualization using projections and 2D and 3D surface plots of distance matrices.

The U-matrix method (Ultsch, 1993) enables visualization of the topological relations of the neurons in an organized SOM. A gradient image (2D) or a surface plot is generated by computing the distances between adjacent neurons. High values in the U-matrix encode dissimilarities between neurons and correspond to cluster borders.

A host of strategies for cluster detection using the U-matrix were proposed in the literature (Costa & Andrade Netto, 2001a; Costa & Andrade Netto, 2003). Three main algorithms were presented: mathematical morphology derived map segmentation (Costa & Andrade Netto, 2001b); a graph partitioning approach (Costa & Andrade Netto, 2003) and contiguity-constrained hierarchical clustering approaches (Gonçalves et al., 2008; Murtagh, 1995). Both algorithms were developed for automatic partitioning and labeling of a trained SOM network.

The first approach uses image processing algorithms such as the watershed transform, which are used to obtain connected regions of neurons representing similar stimuli classes. The second approach uses rules to partition the map by analyzing inconsistent neighboring relations between neurons. Each resulting neuron cluster is a sub-graph that defines complex and non-parametric geometries in the input space, which approximately describes the shape of the clusters. Regarding the last approach, Gonçalves et al. (2008) present improvements of contiguity-constrained hierarchical clustering approaches using validation indexes.

Some works apply clustering algorithms over the U-matrix to segment the map in well defined regions. Vesanto & Alhoniemi (2000) propose strategies for cluster detection using different approaches to group similar neurons in the SOM, including the use of hierarchical agglomerative clustering and partitive clustering, using the K-Means algorithm.

4. The Proposed Strategy

Distributed clustering algorithms usually work in two stages. Initially, the data are analyzed locally in each unit that is part of the distributed database. In a second stage, a central instance gathers partial results and combines them into an overall result.

This section presents a strategy for clustering similar objects located in distributed databases using traditional clustering algorithms, such as self-organizing maps and K-Means. An imposed restriction is that proposed strategy requires a prototype-based algorithm to perform the dataset partition into subsets and select a set of one or more representative elements to each cluster. This representative set is normally named prototype (or codebook) and will be used in a process similar to vectorial quantization in order to select a sample of each remote unit.

The entire process is also divided into two stages:

1. The clustering algorithm is applied locally in each one of the distributed bases, in order to elect a representative subset from input data, which is send to central unit;
2. The clustering algorithm is applied again, this time to the representatives of each one of the distributed bases, that are unified in a central unit;

The proposed strategy, consisting of five steps, is presented as follows:

```
Proposed Strategy

1. Each local site applies any clustering algorithm to its data, obtaining a reference vector (codebook)
2. Each local site codes its data, using the codebook, to obtain a vector of data indexes that will represent the original data
3. Each local site send indexes and reference vectors to the central site
4. The central site remounts the dataset based on each index and reference vectors received
```

Step 1 applies anyone traditional clustering algorithm in each local dataset, relative to vertical partitions from the database. Thus, the algorithm is applied to an attribute subset in each of the remote units, obtaining a reference vector from each data subset. This reference vector, known as the codebook, contains a representative subset of each local unit.

Step 2 codes the input data of each local unit using the codebook obtained in the previous stage. Each input is presented to the obtained codebook and the index corresponding to the closest vector present in the codebook is stored in an index vector. So, a data index is created based on representative objects instead of original objects. Despite the difference from the original dataset, representative objects in the index vector are very similar to the original data.

In step 3, each remote unit sends its index and reference vector to the central unit, which is responsible for unifying all partial results. An additional advantage of the proposed algorithm is that the amount of transferred data is considerately reduced, since index vectors have only one column (containing an integer value) and the codebook is usually mush less than the original data. So, reducing data transfer and communication overload are considered by the proposed algorithm.

Step 4 is responsible for receiving the index vector and the codebook from each local unit and combining partial results to remount a database based on received data. To remount each dataset, index vector indexes are substituted by the equivalent value in the codebook. Datasets are combined juxtaposing partial datasets; however, it is important to ensure that objects are in the same order as that of the original datasets. Note that the new database is slightly different from the original data, but data topology is maintained, similar to procedure used for vectorial quantization.

In step 5, any clustering algorithm is again applied over the complete database obtained in step 4. The expectation is that the results obtained in that stage can be generalized as being equivalent to the clustering process of the entire database. The data obtained after the step 4 and that will serve as input in stage 5 correspond to values close to the original, because vectors correspondents in codebook are representatives of input dataset.

An overview of the complete architecture is showed in Fig. 1, considering the SOM algorithm. Similarly, K-Means or other prototype-based clustering algorithms can be used in proposed strategy.

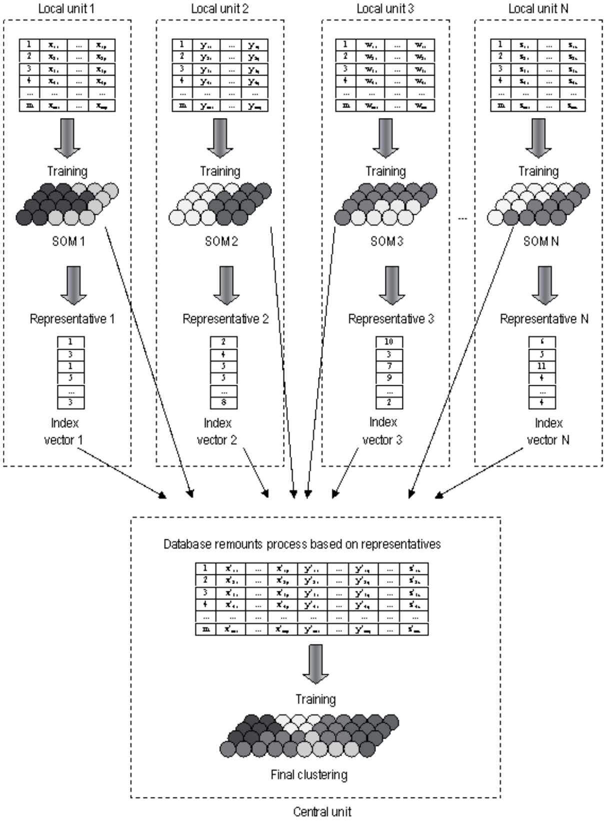

Fig. 1. Complete architecture of proposed strategy using the SOM algorithm

5. Methodology

In order to verify the precision of the proposed strategy, results with this approach are compared with those of traditional clustering using traditional SOM and/or K-Means algorithms. Experiments were carried out using the SOM Toolbox 2.0 package, a SOM implementation for Matlab, available at http://www.cis.hut.fi/projects/somtoolbox/ (Vesanto, 2000).

As a form of validating the proposed strategy, several comparative criteria were used, including the individual counting of errors obtained in the application of the algorithm over well known datasets, the use of quality measures present in SOM toolbox and visual comparison of the trained maps and U-Matrix between the traditional SOM algorithm and the proposed approach.

Several quality measures have been proposed to evaluate and compare clustering algorithms. The SOM Toolbox includes two:

1. Data representation accuracy, measured using average quantization error between data vectors and their BMUs on the map, and
2. Dataset topology representation accuracy, measured using topographic error, which is the percentage of data vectors for which the first and second BMUs are not adjacent units.

The databases used were obtained from the UCI data repository (Asuncion & Newman, 2007). Additional information about number of instances, number of attributes and clusters contained in each dataset are presented in Table 1.

Dataset	Instances	Attributes	Attribute Type	Clusters
Iris	150	4	Numerical	3
Wine	178	13	Numerical	3
Breast Cancer	699	10	Numerical	2
Mushroom	8124	22	Categorical	2

Table 1. Characteristics of datasets used in experiments

6. Experiments

6.1 Iris Dataset

The well-known Iris dataset was used in a variety of studies on pattern recognition. It presents 150 instances containing width and length measures of the sepals and petals of three species of the flower Iris: 'Setosa', 'Versicolor' and 'Virginical'. With 4 attributes and 3 classes, each containing 50 objects, the aim is to cluster similar species based on their measurements. One class is linearly separable 'Setosa' whereas classes 'Versicolor' and 'Virginica' have a degree of mixture.

In the first experiment, Iris dataset was analyzed initially using the traditional SOM algorithm, in a plan hexagonal lattice map with 11 x 6 neurons. Dataset was then vertically partitioned into two subsets, each containing two attributes. The SOM algorithm was applied to two subsets separately and, finally, on the representatives of each one. In this phase, we used two maps, with 9 x 7 and 21 x 3 neurons in the local units and a final map with 11 x 6 neurons, both plan and hexagonal lattice. Maps size was automatic defined by

SOM Toolbox, based on data distribution in input space (Vesanto, 2000). In all experiments maps were randomly initialized and batch SOM algorithm was used.

Two training phases were performed: rough and fine tuning training phases. In rough phase of the traditional SOM approach, neighborhood radius was defined as $\sigma_{initial}$ = 2 and σ_{final} = 1 and *trainlen* was defined to 5 epochs. In fine tuning phase, σ = 1 and *trainlen* was defined to 18 epochs.

In proposed strategy approach, was used over again a randomly initialized map and batch training method. In rough phase, neighborhood radius was defined as $\sigma_{initial}$ = 2 in the first map and as $\sigma_{initial}$ = 3 in the second. Final map used same values of the traditional SOM experiment (11 x 6 neurons). Final neighborhood radius was σ = 1 to all three maps. The *trainlen* parameter was defined as 5 epochs in rough phase in the three maps and as 17 epochs in fine tuning phase for two first maps (encoding phase) and as 18 epochs in the final map.

Two criteria were used to compare the results. The first consists of labeling each of the neurons on the map with information about the class number it belongs to, in agreement with the number of input data instances that it represents. For this to occur, each input data instance had to have a label to identify its class. Note that privileged information was not used during the training phase, only during the algorithm accuracy verification.

In the first approach, traditional SOM was 96% accurate, which corresponds to 6 missing classifications out of 150 instances. In the alternative approach, the accuracy of the proposed strategy increased to 96.7%, which corresponds to 5 missing classifications in the analyzed group.

The second criterion is more subjective, because it is based on a visual comparison. As previously described, the U-matrix enables visualization of clusters (similarity and dissimilarity regions) detected by SOM algorithm. Fig. 2 shows the self-organizing maps obtained with traditional SOM and the proposed strategy using the Iris dataset. Maps were manually colored to identify the class that the neuron belongs to.

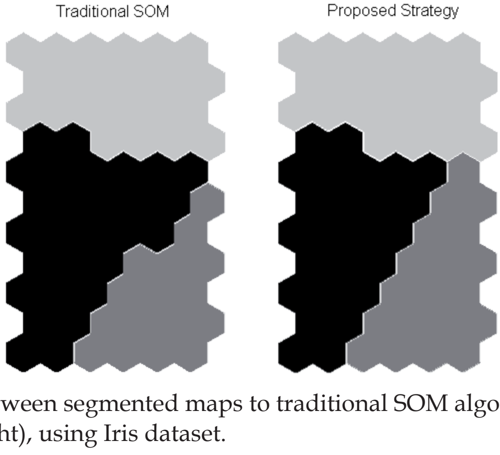

Fig. 2. Comparison between segmented maps to traditional SOM algorithm (left) and proposed strategy (right), using Iris dataset.

In the second experiment, Iris dataset was analyzed using combined SOM and K-Means algorithms. As in previous experiment, the dataset was vertically partitioned into two subsets, each containing two attributes. The SOM algorithm was applied to two subsets separately and K-Means was applied on the representatives of each one. Maps size and lattice were identical to those defined in the previous experiment. In the central unit, centroids number was defined as $k = 3$, identical to classes number.

Table 2 summarize average values obtained in the first experiment and present other clustering quality measures like QE (average quantization error) and TE (topographic error).

Algorithm	QE	TE	Errors	Hits (%)
Traditional SOM approach	0.3927	0.0133	6	96.0%
Proposed Strategy with SOM + SOM	0.2934	0.0067	5	96.7%

Table 2. Comparatives missing cluster values betweens traditional SOM and the proposed strategy (Iris Dataset)

Table 3 summarize average values obtained in the second experiment, using a combined approach with SOM and K-Means algorithms.

Algorithm	Errors	Hits (%)
Traditional K-Means approach	27	82.0%
Proposed Strategy with SOM + K-Means	25	83.3%

Table 3. Comparatives missing cluster values betweens traditional K-Means and the proposed strategy (Iris Dataset)

6.2 Wine Dataset

The Wine dataset has 178 instances and 13 attributes, which correspond to the results of chemical analyses performed with three types of wines that are produced in the same region of Italy, but from different cultivations. Attributes include alcoholic content, acidity, alkalinity, color intensity, among others. The dataset has 59 instances of the first class, 71 instances of the second class and 48 instances of the third. The classes are labeled as '1', '2' and '3', in order to facilitate a subsequent identification and the validation of the algorithm. The dataset has well defined classes, so that the identification of these classes is performed with relatively little difficulty.

The strategy used in experiments was similar to the previous one, with Iris database. The dataset was divided into two subsets, one with the first six attributes and the other with the seven remaining attributes. The clustering algorithm was applied separately over each one of the subsets and, later, on their representatives.

In the first approach, using traditional SOM, we have used a map size with 11 x 6 neurons in a hexagonal lattice. Training parameters used are $\sigma_{initial} = 2$, $\sigma_{final} = 1$ and 4 epochs in rough phase and $\sigma_{initial} = \sigma_{final} = 1$ and 15 epochs in fine tuning phase.

In the second approach, using proposed strategy, two maps were used, with 9 x 7 and 11 x 6 neurons in local phase, and a map with 11 x 6 neurons in final phase, both plans and hexagonal lattice. Training parameters used in all three maps are $\sigma_{initial} = 2$, $\sigma_{final} = 1$ and 4 epochs in rough phase and $\sigma_{initial} = \sigma_{final} = 1$ and 15 epochs in fine tuning phase. In both approaches, maps size was automatic defined by SOM Toolbox, considering principal components of data distribution.

The results obtained with the proposed strategy were very near to those of the traditional approach, where all the variables were analyzed simultaneously. The U-matrix obtained with the Wine dataset, using both approaches, is presented in Fig. 3.

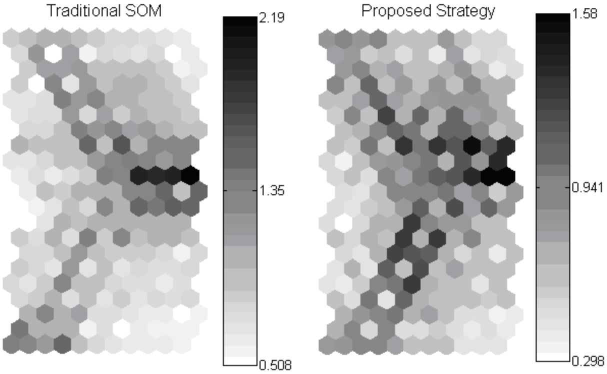

Fig. 3. U-matrix of the SOM algorithm (left) and the proposed strategy (right), using the Wine dataset

Additionally, Fig. 4 shows final self-organizing map with the respective labels that identify each neuron. The numbers represent the class that the neuron belongs to. This is obtained by a voting process on how many elements of the original dataset the neuron is associated to. Fig. 5 presents the same maps labeled to improve the visualization quality. Maps were manually colored based on labels obtained in previous stage.

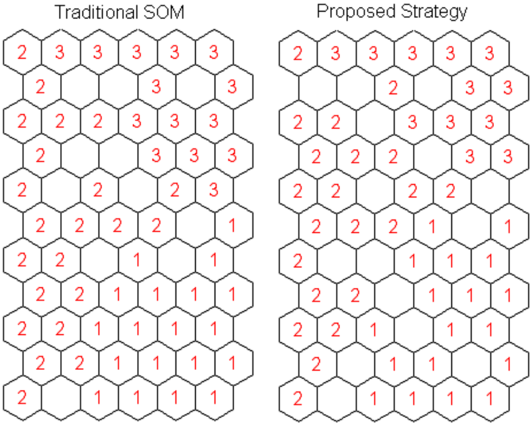

Fig. 4. The self-organizing map, with the respective labels that identify the class of each neuron, for the traditional SOM algorithm (left) and the proposed strategy (right), using the Wine dataset

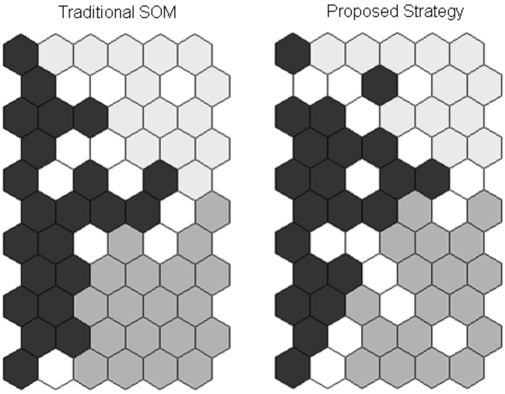

Fig. 5. The manually coloured self-organizing map for the traditional SOM algorithm (left) and the proposed strategy (right), using the Wine dataset.

Table 4 summarizes the average results obtained with traditional SOM and proposed strategy. It presents QE (average quantization error) and TE (topographic error) to Wine dataset. In a similar form of previous section, results obtained with the proposed algorithm were very near to those of traditional methods, using a single dataset. Furthermore, the experiment was replicated using different attribute subsets. The dataset was divided into three, four and five partitions to verify how the partitioning method affects the proposed algorithm performance. In both cases, obtained results were very similar to those presented in Table 4.

Algorithm	QE	TE	Errors	Hits (%)
Traditional SOM	1.8833	0.0169	4	97.8%
Proposed Strategy	2.0967	0.0674	7	96.1%

Table 4. Comparatives missing cluster values betweens traditional SOM and the proposed strategy (Wine Dataset)

In a similar experiment, the Wine dataset was analyzed initially using the traditional K-Means algorithm, and next, using the proposed strategy with the K-Means algorithm. As in previous experiment, the dataset was divided into two subsets, with six and seven attributes, respectively. The K-Means algorithm was applied separately over each one of the subsets and, later, on their representatives.

In traditional approach, the centroids number was defined as k = 3, identical to classes number. In distributed approach, the centroids number was defined as k = 66 in local stage and as k = 3 in central stage. At first stage, the k value was maximized for two reasons: first, a large number of prototypes improve the performance of the algorithm, and second, was used the same number of SOM neurons in previous experiment, with the objective to become more adequate the comparative between both approach. Table 5 summarizes the average results obtained with traditional K-Means and distributed strategy.

Algorithm	Errors	Hits (%)
Traditional K-Means	6	96.6%
Distributed K-Means Strategy	7	96.1%

Table 5. Comparatives missing cluster values betweens traditional K-Means and the proposed strategy (Wine Dataset)

6.3 Wisconsin Breast Cancer Dataset

The Wisconsin Breast Cancer dataset has 699 instances of the cytological analysis of fine needle aspiration of breast tumors. Each instance contains 10 attributes that are computed from a digitized image of a fine needle aspiration of a breast mass. Attributes include radius, texture, perimeter, area, smoothness, compactness, concavity, concave points, symmetry, and fractal dimension. The dataset contain 458 (65.52%) benign instances and 241 (34.48%) malignant instances.

The SOM algorithm was applied to the entire database and to some partitions of the same dataset, according to proposed strategy. The first five attributes were trained in a map and the four remaining attributes were trained separately in another. As before, representatives of both groups were gathered and combined to form a new dataset, which was later clustered using the SOM algorithm again.

In the first experiment, using traditional SOM, was used a 2D hexagonal lattice map with size 22 x 6 neurons. In the second experiment (proposed strategy) two maps with 19 x 7 and 17 x 8 neurons in local phase and a map with 22 x 6 neurons in final phase were used. All SOM were 2D hexagonal lattice maps. As in previous experiments, maps size was automatic defined by SOM Toolbox, based on input data distribution. In both experiments, training parameters used are $\sigma_{initial}$ = 3, σ_{final} = 1 and *trainlen* = 2 epochs in rough phase and $\sigma_{initial}$ = σ_{final} = 1 and *trainlen* = 15 epochs in fine tuning phase.

Table 6 summarizes the obtained results with traditional SOM and proposed strategy, including QE (average quantization error) and TE (topographic error) to Wisconsin Breast Cancer dataset.

Algorithm	QE	TE	Errors	Hits (%)
Traditional SOM	0.9461	0.0272	17	97.57%
Proposed Strategy	1.0009	0.0615	13	98.14%

Table 6. Comparatives Missing Cluster Values Betweens Traditional SOM and the Proposed Strategy (Wisconsin Breast Cancer Dataset)

Fig. 6 presents the final self-organizing map, which was manually colored to improve the visualization quality. Darker tones represent benign instances and light tones represent malignant instances. As in previous experiment, each neuron is labeled based on amount of instances from original dataset associated with this neuron and privileged information is used only to label the map and verify the algorithm accuracy.

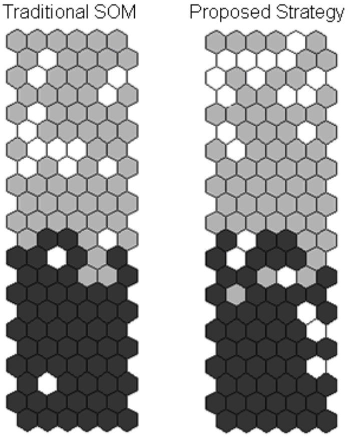

Fig. 6. The manually labelled self-organizing map for the traditional SOM algorithm (left) and the proposed strategy (right), using the Wisconsin Breast Cancer dataset.

6.4 Mushroom Dataset

The Mushroom dataset contain 8124 instances and 22 categorical attributes, with descriptions of hypothetical samples corresponding to several species of gilled mushrooms. Attributes include shape, surface, color, and others. Each species is identified as definitely edible, definitely poisonous, or of unknown edibility and not recommended. This latter class was combined with the poisonous one.

Initially, a pre-processing step was required to convert all categorical data to numeric data, because traditional SOM algorithm handles only numeric data. This processing increases the complexity of the clustering process, since attributes number (dimensionality) is enlarged. After pre-processing, the number of database attributes increased to 117.

The strategy used in experiments was similar to the previous one. In traditional approach, the SOM algorithm was applied to the entire database. However, in the proposed approach, dataset was partitioned into two, three e four subsets. In all experiments, the partitions were carrying out in order that similar attributes were grouped together. For example, 'cap_shape', 'cap_surface' and 'cap_color' attributes were grouped in the same subset.

In the first approach, using traditional SOM, was used a map size with 23 x 19 neurons in a hexagonal lattice. Training parameters used are $\sigma_{initial}$ = 3, σ_{final} = 1 and 1 epoch in rough phase and $\sigma_{initial}$ = σ_{final} = 1 and 3 epochs in fine tuning phase.

In the second approach, using proposed strategy, the dataset was divided into two subsets, one with the first 49 attributes and the other with the 68 remaining attributes. Two maps were used, with 25 x 18 and 23 x 19 neurons in local phase, and a map with 23 x 19 neurons in final phase, both plans and hexagonal lattice. Training parameters used are $\sigma_{initial}$ = 4, σ_{final} = 1 in first map and $\sigma_{initial}$ = 3 and σ_{final} = 1 in second map, and training are performed using 1 epoch in rough phase and 2 epochs in fine tuning phase. In final map, were used $\sigma_{initial}$ = σ_{final} = 1 and 1 epoch in rough phase and 2 epochs in fine tuning phase.

Next, the dataset was divided into three subsets, with 49, 33 e 35 attributes respectively. The clustering algorithm was applied separately over each one of the subsets and, later, on their representatives. Three maps were used, with 25 x 18, 25 x 18 and 23 x 19 neurons in local phase. Training parameters used in local phase were $\sigma_{initial}$ = 2, σ_{final} = 1 and 1 epoch in rough and tuning phase. In final phase, was used a plan and hexagonal lattice map with 25 x 18 neurons. Training parameters used were $\sigma_{initial}$ = σ_{final} = 1, 1 epoch in rough and 3 epochs in fine tuning phase.

Finally, the dataset was divided into four subsets, with 31, 18, 33 e 35 attributes respectively. The clustering algorithm was applied separately over each one of the subsets and, later, on their representatives. Four maps were used, with 25 x 18, 23 x 19, 25 x 18 and 23 x 19 neurons in local phase, and a map with 23 x 19 neurons in final phase, both plans and hexagonal lattice. Training parameters used in first and third maps are $\sigma_{initial}$ = 4, σ_{final} = 1, 1 epoch in rough and fine tuning phase. In all remain maps, parameters used are $\sigma_{initial}$ = 4 e σ_{final} = 1 and 3 epochs in fine tuning phase.

Table 7 summarizes the average results obtained with Mushroom dataset, using traditional SOM and proposed strategy with two, three and four partitions. The average quantization error (QE) and the topographic error (TE) are presented. Also, are presented the number of errors and hits obtained in the application of the algorithm in a classification task. As in previous section, results obtained with the proposed algorithm were very near to those of traditional methods, using a single dataset.

Algorithm	QE	TE	Errors	Hits (%)
Traditional SOM	5.545	0.042	295	96.37%
Proposed Strategy with 2 partitions	5.859	0.049	292	96.41%
Proposed Strategy with 3 partitions	8.048	0.159	331	95.93%
Proposed Strategy with 4 partitions	5.783	0.062	292	96.41%

Table 7. Comparatives missing cluster values betweens traditional SOM and the proposed strategy (Mushroom Dataset)

One of the main advantages of the proposed strategy in relation to the traditional approach is the data transfer reduction between remotes and central units. Suppose the input data consist of $N \times p$ matrix, where N denotes the number of instances existent in the database and p denotes the data dimensionality. The codebook can be represented as a $k \times p$ matrix, where k denotes the number of prototypes in the codebook and the indexes vector can be represented as a $N \times 1$ matrix.

Therefore, since only the indexes vector and the codebook are transferred from remotes units to central unit, and generally, $k \ll N$, the strategy appears to be sufficiently adequate for application in distributed data clustering, because the traffic among units is reasonably decreased.

Algorithm	Unit1	Unit2	Unit3	Unit4	Total
Traditional SOM with 2 partitions	398,076	552,432	-	-	950,508
Proposed Strategy with 2 partitions	30,174	37,840	-	-	68,014
Traditional SOM with 3 partitions	398,076	268,092	284,340	-	950,508
Proposed Strategy with 3 partitions	30,174	22,974	23,419	-	76,567
Traditional SOM with 4 partitions	251,844	146,232	268,092	284,340	950,508
Proposed Strategy with 4 partitions	22,074	15,990	22,974	23,419	84,457

Table 8. Comparatives data volume transferred betweens traditional SOM and the proposed strategy (Mushroom Dataset)

Table 8 presents a comparative between traditional SOM and the proposed approach, in relation to the data volume transferred, considering two, three and four remote units. Only 1 byte for each attribute was considered, since all the attributes were converted to binary in pre-processing step.

7. Commented Results

The main contribution of this work is to present an alternative strategy for distributed data clustering in vertically partitioned databases using common algorithms. This approach can be performed in a distributed way, executing several instances of the same (or another) algorithm in parallel, each at one site, and combining all results in a central site.

In the first experiment, the Iris dataset was analyzed with proposed strategy using two approaches: first, using SOM in both local and central stages, and second, using SOM in the local stage and K-Means in the central stage. We compared traditional approach with distributed strategy and showed that the very similar results can be obtained in both approaches.

In the second experiment, the Wine dataset was analyzed with traditional SOM and traditional K-Means algorithms and with the proposed strategy using multiple SOM and multiple K-Means instances. The results show that the number of errors in the application of the strategy in a classification task is near to obtained with traditional approaches. Additionally, we carried out other experiments, with 3, 4, 5 and 6 partitions, obtaining similar results.

In the third experiment, the Wisconsin Breast Cancer dataset was analyzed with traditional SOM algorithm and with the proposed strategy using multiple SOM instances. The results show that the number of errors in the application of the strategy in a classification task is near to obtained with traditional approach.

In the fourth experiment, the Mushroom dataset was analyzed with traditional SOM and proposed strategy. We carried out experiments with two, three and four partitions, and results show the efficiency of proposed strategy. Additionally, we analyze the traffic among units and it was showed that using the proposed strategy, the data volume transferred can be efficiently reduced.

8. Conclusion

There is a growing need for effective approaches to analyze data distributed among several sites. This is motivated by the increase in geographically distributed databases and security policies. Merging several parties of a database into a single site is not recommended in some applications, because confidential information can be remounted. Distributed data clustering algorithms are an alternative approach to cluster analysis tasks in distributed databases, because do not require very large databases and reduce inter-sites communication.

Two common algorithms, SOM and K-Means have been widely used in data clustering applications. K-Means is the most used, because is simple and easy of implementation. Self-organizing maps have some advantages in visualization process, including topology preserving mapping, which maps similar data vectors together.

In this work, we propose a novel strategy for data clustering in vertically partitioned databases using a distributed approach that execute typical clustering algorithms (such as self-organizing maps and K-Means) in local units of a distributed database and combine partial results in a central unit. This strategy can be applied to cluster geographically distributed databases, as to avoid excessive data overload from remote units as to maintain the data security and privacy.

We carried out several experimental using UCI databases for demonstration purposes and showed that the proposed strategy, running in a distributed environment, obtained similar results to those with same clustering algorithms, running in traditional approach.

Also, were discussed previously published studies and was presented situations in which for security and privacy reasons or due to high cost of transferring data to a central unit, a central database cannot be recommended. Otherwise, it was demonstrated that privacy-preserving data is guaranteed, since only reference vectors are sent to the central site.

Future research directions will use a mean and residual-vector quantization approach to increase security and data representation.

9. References

Asuncion, A. & Newman, D. J. (2007). *UCI Machine Learning Repository*, available online on: http://www.ics.uci.edu/~mlearn/MLRepository.html, Department of Information and Computer Science, University of California, Irvine, CA.

Berkhin, P. (2006). A Survey of Clustering Data Mining Techniques, In: *Grouping Multidimensional Data: Recent Advances in Clustering*, J. Kogan, C. Nicholas and M. Teboulle (Ed.), pp. 25–71, Springer-Verlag, New York.

Costa, J. A. F. & Andrade Netto, M. L. (2001a). A new tree-structured self-organizing map for data analysis, *Proc. of the Intl. Joint Conf. on Neural Networks*, Washington, DC, Vol. 3, pp. 1931-1936.

Costa, J. A. F. & Andrade Netto, M. L. (2001b). Clustering of complex shaped data sets via Kohonen maps and mathematical morphology, In: *Data Mining and Knowledge Discovery*, B. Dasarathy (Ed.), pp. 16-27, Proceedings of the SPIE, Florida, USA.

Costa, J. A. F. & Andrade Netto, M. L. (2003). Segmentação do SOM Baseada em Particionamento de Grafos. *Proceedings of Brazilian Neural Networks Conference*, pp. 301-308, São Paulo, Brazil, June 2003.

Damian, D.; Orešič, M.; Verheij, E.; Meulman, J.; Friedman, J.; Adourian, A.; Morel, N.; Smilde A. & van der Greef, J. (2007). Applications of a new subspace clustering algorithm (COSA) in medical systems biology, *Metabolomics Journal*, Vol. 3, No. 1, Mar. 2007, pp. 69-77.

Forman, G. & Zhang, B. (2000). Distributed data clustering can be efficient and exact. *SIGKDD Explorations Newsetterl*. Vol. 2, Dec. 2000, pp. 34-38.

Friedman, J. H. & Meulman, J. J. (2004). Clustering objects on subsets of attributes (with discussion), *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, Vol. 66, No. 4, pp. 815-849.

Gonçalves, M.; Andrade Netto, M. L. ; Costa, J. A. F. & Zullo, J. (2008). A new method for unsupervised classification of remotely sensed images using Kohonen self-organizing maps and agglomerative hierarchical clustering methods, *International Journal of Remote Sensing*, Vol. 29, pp. 3171-3207.

Gorgônio, F. L. & Costa, J. A. F. (2008a). Combining Parallel Self-Organizing Maps and K-Means to Cluster Distributed Data, *Proceedings of the 11th IEEE International Conference on Computational Science and Engineering*, CSE'2008, São Paulo, Brazil, 2008, pp. 53-58.

Gorgônio, F. L. & Costa, J. A. F. (2008b). Parallel Self-organizing Maps with Application in Clustering Distributed Data, *Proceedings of the International Joint Conference on Neural Networks*, IEEE World Congress on Computational Intelligence, Hong-Kong, 2008, Vol. 1, pp. 420.

Gorgônio, F. L. (2009). *Uma Arquitetura para Análise de Agrupamentos sobre Bases de Dados Distribuídas*, Tese de Doutorado, Federal University of Rio Grande do Norte, UFRN/PPgEEC, Natal, RN, Brazil.

Haykin, S. (1999). *Neural networks: A comprehensive foundation*, 2nd ed., Macmillan College Publishing Company, New York.

He, Z.; Xu, X. & Deng, S. (2005). Clustering Mixed Numeric and Categorical Data: A Cluster Ensemble Approach, Technical Report, available online on: http://aps.arxiv.org/ftp/cs/papers/0509/0509011.pdf, 2005.

İnan, A.; Kaya, S. V.; Saygın, Y.; Savaş, E.; Hintoğlu; A. A. & Levi, A. (2007). Privacy preserving clustering on horizontally partitioned data, Data & Knowledge Engineering, Vol. 63, No. 3, Dec. 2007, pp. 646-666.

Jagannathan, G.; Pillaipakkamnatt, K. & Wright, R. N. (2006). A New Privacy-Preserving Distributed k-Clustering Algorithm, Proc. of the 2006 SIAM International Conference on Data Mining, pp. 492-496.

Kantardzic, M. (2003). *Data Mining: Concepts, Models, Methods, and Algorithms*, IEEE Press.

Kargupta, H.; Huang, W. ; Sivakumar, K. & Johnson, E. (2001). Distributed Clustering Using Collective Principal Component Analysis, *Knowledge and Information Systems Journal*, Vol. 3, No. 4, May 2001, pp. 422-448.

Kohonen, T. (2001). *Self-Organizing Maps*, 3rd ed., Springer-Verlag, New York.

Lam, C. M.; Zhang, X. F. & Cheung, W. K. (2004). Mining Local Data Sources for Learning Global Cluster Models, *Proceedings of International Conference on Web Intelligence*, Iss. 20-24, Sept. 2004, pp. 748-751.

Murtagh, F. (1995). Interpreting the Kohonen self-organizing feature map using contiguity-constrained clustering, *Pattern Recognition Letters*, Vol. 16 , No. 4, April 1995, pp. 399-408.

Oliveira, S. R. M. & Zaïane, O. R. (2004). Privacy Preservation When Sharing Data For Clustering, Proc. of the International Workshop on Secure Data Management in a Connected World, 2004.

Silva, J. C. & Klusch, M. (2006). Inference in distributed data clustering. *Engineering Applications of Artificial Intelligence*, Vol. 19, pp. 363-369.

Ultsch, A. (1993). Self-Organizing Neural Networks for Visualization and Classification, In: *Information and Classification*, O. Opitz et al. (Ed), pp. 301-306, Springer, Berlin.

Vaidya, J. & Clifton, C. (2003). Privacy-preserving k-means clustering over vertically partitioned data. Proc. of the Ninth ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining, pp. 206-215, ACM, New York, NY.

Vesanto, J. (2000). *Using SOM in Data Mining*, Licentiate's Thesis, Department of Computer Science and Engineering,Helsinki University of Technology, Espoo, Finland.

Vesanto, J. & Alhoniemi, E. (2000). Clustering of the Self-Organizing Map, *IEEE Transactions on Neural Networks*, Vol. 11, No. 3, May 2000, pp. 586–600.

Xu, R. & Wunsch II, D. (2005). Survey of Clustering Algorithms. *IEEE Transaction on Neural Networks*, Vol. 16, No. 3, May 2005, pp. 645-678.

Self-Organizing Maps
Edited by George K Matsopoulos

ISBN 978-953-307-074-2
Hard cover, 430 pages
Publisher InTech
Published online 01, April, 2010
Published in print edition April, 2010

The Self-Organizing Map (SOM) is a neural network algorithm, which uses a competitive learning technique to train itself in an unsupervised manner. SOMs are different from other artificial neural networks in the sense that they use a neighborhood function to preserve the topological properties of the input space and they have been used to create an ordered representation of multi-dimensional data which simplifies complexity and reveals meaningful relationships. Prof. T. Kohonen in the early 1980s first established the relevant theory and explored possible applications of SOMs. Since then, a number of theoretical and practical applications of SOMs have been reported including clustering, prediction, data representation, classification, visualization, etc. This book was prompted by the desire to bring together some of the more recent theoretical and practical developments on SOMs and to provide the background for future developments in promising directions. The book comprises of 25 Chapters which can be categorized into three broad areas: methodology, visualization and practical applications.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Flavius L. Gorgonio and Jose Alfredo F. Costa (2010). PartSOM: A Framework for Distributed Data Clustering Using SOM and K-Means, Self-Organizing Maps, George K Matsopoulos (Ed.), ISBN: 978-953-307-074-2, InTech, Available from: http://www.intechopen.com/books/self-organizing-maps/partsom-a-framework-for-distributed-data-clustering-using-som-and-k-means

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821