

# Tree-Structured Self-Organizing Maps Modelling for Imputation and Editing

Pasi Piela

Statistics Finland

FIN-00022 STATISTICS FINLAND

FINLAND

Pasi.Piela@stat.fi

## 1. Tree-Structured Self-Organizing Maps

The tree-structured self-organizing maps based approach for imputation and editing is shortly presented here. Self-organizing maps (SOM) is an one iterative method for classification and can thus also be used in finding the imputation classes. Imputations are made within clusters, located by corresponding neurons, in several ways which can be based on both classical and neural methods.

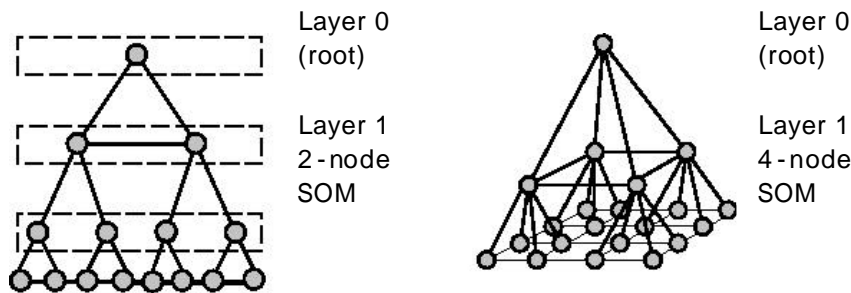
The basic SOM defines a mapping from the input data space  $\mathbf{R}^n$  onto a latent space consisted typically of a two-dimensional array of nodes or neurons [2]. A parametric reference or weight vector  $\mathbf{w}_i$  is chosen for each neuron  $i$  from the discrete set  $i = \{1, 2, \dots, N\}$ . Now, let  $\mathbf{x} \in \mathbf{R}^n$  be a stochastic, random data vector. Usually the smallest of the Euclidean distances  $\|\mathbf{x} - \mathbf{w}_i\|$  is made to define the best matching unit (BMU) for the vector  $\mathbf{x}$ , denoted by the subscript  $b$ , that is,  $b = \operatorname{argmin}\{\|\mathbf{x} - \mathbf{w}_i\|\}$ . Then SOM algorithm updates the weight vectors of the BMU and in its neighbouring neurons  $i$  as follows:

$$\mathbf{w}_i(t+1) = \mathbf{w}_i(t) + h_{bi}(t)[\mathbf{x}(t) - \mathbf{w}_i(t)]$$

where  $h_{bi}(t) = \mathbf{a}(t)H\left(\frac{v_b - v_i}{s}\right)$  defines the neighbourhood kernel  $H(\cdot)$  over the lattice points and the learning rate ( $0 < \mathbf{a}(t) < 1$ ) at iteration step  $t$ . The map can also be created in a non-stochastic way by using the so-called *batch algorithm* in which data points are associated with their BMUs in each iteration, and then all the prototypes are updated at a time.

The Tree-Structured Self-Organizing Map is made of several SOMs arranged to a tree structure (see Figure 1.1). Thus, each neuron has its own associated subgroup of data, four subgroups on layer 1 in two-dimensional case but one group, the data set itself on layer 0. The subgroup forms the cluster of which centroid is the weight vector of the best matching unit  $b$ ,  $\mathbf{w}_b$ . Furthermore, the centroid on layer 0 defines the mean of all data.

**Figure 1.1.** Illustrations of one and two-dimensional TS-SOM structures.



The training is repeated layer by layer using knowledge about the BMU of the frozen layer  $l-1$  in the search of the BMU on next layer  $l$ . That is, the search is restricted to a small set of units that are linked to the BMU and its neighbours on level  $l-1$ . As a significant advantage of the TS-SOM, this reduces the computational complexity in the search of BMU when compared to basic SOM.

The training is usually made with the batch algorithm [1] (referring to Koikkalainen's paper). During each epoch the BMUs are searched for all data vectors using the tree search, and then new centroids  $\mathbf{w}_b^{new}$  are computed using the rule:

$$\mathbf{w}_b(t+1) = \frac{1}{N_b + \sum_{i \in N_c(b)} \mathbf{a} N_i} \left( N_b \mathbf{w}_b^{new} + \sum_{i \in N_c(b)} \mathbf{a} N_i \mathbf{w}_i^{new} \right)$$

where  $N_c(b)$  is a set of indices of neighbours of  $b$ , and  $N_i$  is the number of data records in the Voronoi region (cluster)  $i$ . The smoothing is partially controlled through the parameter  $\mathbf{a} \in [0..1]$ .

## 2. TS-SOM based Imputation and Editing

Nearest neighbour imputation can be made by filling the missing component  $x_i(j)$  of the data vector from the nearest data vector within the same cluster.

$$\mathbf{x}_i(j) = \mathbf{x}_n(j), \quad n = \operatorname{argmin}_{k \in cl_b} (\|\mathbf{x}_i - \mathbf{x}_k\|).$$

An obvious problem in this kind of hot-deck type of imputation is its computational complexity (related to computation time) that is  $O(N^2)$  due to the full search among  $N$  data vectors. By using TS-SOM the complexity can be reduced to  $O(M \log(N) + N^2/M^2)$  [1], where  $M$  is number of imputation clusters. In practice, this was clearly seen when imputing quite large data mass of 200,000 observations and 14 variables: the computation time using Pentium® II processor was more than 12 hours without TS-SOM mapping (layer = 0) and 20 minutes with TS-SOM layer 3 as imputation layer (64 neurons).

Group means imputation can be made by taking the centroid of the cluster,  $\mathbf{m}_b$ , and replacing the missing component  $j$  of the data vector  $\mathbf{x}_i$  by corresponding  $\mathbf{m}_b(j)$ , which is actually the fastest way to impute. The structure of TS-SOM gives possibilities to solve problematic situations: the centroid can be interpolated from a parent neuron of the upper TS-SOM layer or it is possible to use neighbouring clusters and neurons as well in finding appropriate donor or the centroid needed.

As for taking into account local distributions of the clusters, the regression based methods like MLP network with backpropagation algorithm can be used, and thus take values for missing components from these models that are generalizations of the local distributions of each cluster.

But as Koikkalainen [3] shows, TS-SOM can be seen as a hierarchical Gaussian mixture model, where each neuron is a Gaussian Generator, it is possible to create imputation models which are connected to the posterior probabilities of the neurons [1].

After imputations, TS-SOM can be used in finding subgroups of high differences between imputed and corresponding true values. Respectively for editing problems, TS-SOM modelling helps in error localization. It can be trained to observe different abnormalities, and in the ideal situation the erroneous observations are, for example, distinguished in their own clusters.

This methodology is tested and applied to practical editing & imputation problems of the different types of data sets, such as large business survey and household survey data sets, for the EU/FP5 project EUREDIT. Results are interesting and encouraging.

## REFERENCE

- [1] Häkkinen, E. (2001). Design, Implementation and Evaluation of the Neural Data Analysis Environment. PhD thesis, Diss. Jyväskylä Studies in Computing. Univ. of Jyväskylä, Finland.
- [2] Kohonen, T. (1997). Self-Organizing Maps. Springer, Berlin, Heidelberg.
- [3] Koikkalainen, P. (1999). Tree Structured Self-Organizing Maps. In Oja, E. and Kaski, S., eds., Kohonen Maps, pages 121-130. Elsevier, The Netherlands.

## RÉSUMÉ

L'approche moderne basée sur des cartes arborescentes à auto-organisation pour l'imputation et l'édition est présentée ici en bref. Les imputations sont effectuées dans des clusters localisés par des neurones correspondants, de plusieurs manières qui peuvent être basées sur des méthodes tant classiques que neurales.