Proceedings of the VI Brazilian Conference on Neural Networks - VI Congresso Brasileiro de Redes Neurais
pp. 457–462, June 2–5, 2003 – Centro Universitário da FEI, São Paulo - SP - Brazil

A New Structured Self-Organizing Map with Dynamic Growth Applied to Image Compression

José Alfredo F. Costa[1], A. Duarte D. Neto[2], and Márcio L. A. Netto[3]

[1]Departamento de Engenharia Elétrica
[2]Departamento de Engenharia de Computação e Automação
Universidade Federal do Rio Grande do Norte
Natal – RN – Brazil

[3]DCA - Faculdade de Eng. Elétrica e de Computação
Universidade Estadual de Campinas – Campinas/SP - Brazil
E-mail: *marcio@dca.fee.unicamp.br*

Abstract **¾ A new hierarchical structure of the Self-Organizing Map (SOM) with dynamic growth is presented and applied to codebook design in vector quantization (VQ) and image compression. The tree-structured approach for codebook design is motivated for reducing the high computational efforts in the training and image coding phases in traditional VQ algorithms. The DHSOM has the ability to self determine the structure of the network through heuristic rules, and its final structure reflects the variability of the data (image blocks). It is shown that training and coding times obtained with DHSOM algorithm are faster than conventional SOM and LBG algorithms while the qualitative results are equivalent.**

1. Introduction

The importance of image and signals compression algorithms in computer systems and communication is ever increasing due to the many and massive uses of digital data storage and transmission devices.

Traditional lossy compression methods include scalar predictive coding, transform coding and hybrid techniques. The first approach is relatively simple but do not achieve much compression, while the transform methods can result in higher compression at the expense of increased computational complexity [1].

The method of vector quantization has been widely used preferentially when there is the need of high compression rate. The traditional algorithm to design the codebook (or reference vectors) is the LBG algorithm [2] which generates it by an iterative process that optimizes a cost function of distortion (or mean square error – MSE). The drawbacks of the LBG include the need to choose the size (K) of the codebook *a priori*, problems due to inappropriate initialization and local minima during adaptation. The VQ encoding process involves a high computational load, which increases with K, but the decoder is a simple look-up table. The quality of the decoded image is related to the size and quality of the designed codebook.

A diversity of artificial neural networks models have been employed for image compression, including feed-forward multilayer perceptrons [3], Hopfield [4] and competitive / unsupervised algorithms [5-8]. Self-Organizing Maps (SOM) have shown capable to develop codebooks for vector quantization with superior performances of the LBG [8-9], and some of its advantages include robustness to the codebook initialization and topographic map formation.

The search for the winner neuron for a given pattern in the map is the heaviest task in both the training and image coding phases. In the conventional SOM the search is sequential and requires a computational effort proportional to the number of neurons. For a good statistical significance of the codebook, and a better reconstructed image, the number of neurons should be large, however it implies in higher training and encoding times. Tree-structured vector quantization approaches have been developed to overcome these limitations, that is, to have a fair codebook size at low searches iterations [6-7,10-11].

In a previous work [5], a fixed hierarchical structure of the algorithm SOM was considered. The structure of the tree was defined *a priori* and the results of this implementation are presented in this work for effect of comparison with the dynamic algorithm proposed. The new proposal includes the dynamic growth of the tree and was denominated DHSOM – Hierarchical and Dynamic Self-Organizing Map. This algorithm will be applied in the development of codebook and its growth and prunning, of the network and sub-networks, are made through heuristical rules. The main objectives are minimizing the quantization error and the processing time.

This organization of this paper is as follows: section 2 describes Kohonen hierarchical structure and section 3 presents the rules of growth of the dynamic hierarchical network and the algorithm DHSOM. Results are described in section 4 and conclusion and final comments are presented in section 5.

2. Hierarchical structure of the SOM

The SOM algorithm is based on unsupervised learning and it generates a topology-preserving map of the training data where the location of a unit carries semantic information [7]. It has been used in a diversity of problems including data visualization, clustering, and data analysis [13-17]. The algorithm is described very briefly. A detailed exposition can be found in ref. [7].

SOM network consists essentially of two layers of neurons. The components of an input vector are fed into all neurons of the input layer. The SOM defines a mapping from the high dimensional input data space onto a regular, usually, two-dimensional array of nodes. Each neuron i of the SOM is represented by an p-dimensional weight vector $m_i = [m_{i1}, m_{i2}, \ldots, m_{ip}]^T$, where p is equal to the dimension of the input vectors. The neurons of the map are connected to adjacent neurons by a neighborhood relation dictating the structure of the map. In the 2-dimensional case the neurons of the map can be arranged either on a rectangular or hexagonal lattice.

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

$$\| \boldsymbol{x} - \boldsymbol{m}_c \| = \min_i \{ \| \boldsymbol{x} - \boldsymbol{m}_i \| \} \qquad (1)$$

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

$$\boldsymbol{m}_i(t+1) = \boldsymbol{m}_i(t) + h_{ci}(t) \cdot [\boldsymbol{x}(t) - \boldsymbol{m}_i(t)] \qquad (2)$$

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 winner 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 is formed of two components: the learning rate function $\alpha(t)$ and the neighborhood function $h(d, t)$:

$$h_{ci}(t) = a(t) \cdot h(\| \boldsymbol{r}_c - \boldsymbol{r}_i \|, t) \qquad (3)$$

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

As the learning proceeds and new input vectors are given to the map, the learning rate and the neighborhood radius gradually decreases to zero according to the specified functions. The correct choice for parameters is not a straightforward task and there are several *rules-of-thumb*, found through experiments. After the training has been performed, the map should be topologically ordered. This means that vectors that are close in the input space would be mapped onto neighbor neurons or even in the same neuron. Some recent improvements include linear initialization and its parallel implementation, or batch algorithm [8].

The use of tree-structured Kohonen maps objectives the reduction of the computational effort necessary for obtaining, or designing, large codebooks [5,10-11]. In a previous work, the authors described a static and pyramidal structure, where each tree node was a SOM map trained with the traditional algorithm. After the end of training in each tree level the map's neurons generate new child map which receive data assigned to its parent neuron. The training of the HSOM is sequential, i.e., top down, and the structure was determined previously the beginning of the learning. This model allows the creation of sub-maps that possess hierarchical relations with maps in superior levels. The main advantage, regarding the use of the conventional full-search SOM, is the training and coding times reduction, which is a important feature in signal and image segmentation and compression algorithms.

The drawback is that the structure is fixed *a priori*, what it means that does not have mechanism of dynamic growth for sub-maps. The modification presented in this paper aim to flexibly the algorithm by allowing only neurons with good activation to generate sub-maps. The hierarchical and dynamic SOM thus produces a data-driven tree of maps with growth controlled for heuristic rules calculated for each neuron in each level of the hierarchy. It can emphasize high variance parts of the input space allowing more neurons to that region.

3. Hierarchical and dynamic SOM (DHSOM)

The DHSOM appeared to surpass the limitations of the HSOM, keeping many of its characteristics. The DHSOM does not require the priori definition of the structure, therefore growth rules are established by the proper algorithm thus enabling expansion (or contraction, via pruning) in a dynamical way. The main characteristics of the DHSOM are:

- Determination of the heuristic rules that define if a node will or not be expanded. It will be taken in consideration the relative activity of the neuron regarding the other neurons in the same map.
- Determination of the flexibility criterion to expansion the network in each level. It will establish criterion of expansion less rigid in the beginning of the tree. Otherwise highest levels will have a more rigid criterion.
- Determination of the size of the sub-maps as a function of its activity and importance to the quantization error minimization.
- Pruning test for maps: determination of the viability of maintaining sub-maps just generated according to its contribution for reducing the quantization error of the parent node.

3.1 Determination of the Heuristic Rules

The training in each level is the traditional SOM algorithm. After finishing the training in the root map it can initiate the phase of hierarchical expansion. Heuristic rules have been established to control the dynamics of growth of the network. There are two main conditions for considering a particular neuron for growing (expansion to a new sub-map): activity and its quantization error. The observance of these criterions will enable the neuron expansion.

Defining activation as the number of patterns assigned to a neuron, a node can be considered active for purpose of expansion if its activation is superior to the average activation in the map.

The other criterion is related to quantization error (QE) associated in each neuron. It is of interest to expand neurons with high data variance and its child map will act to decrease the error in a next level of the structure. The QE factor establish that neurons with error superior to the average quantization error are candidate to be expanded.

Finally, a factor that considers a minimum of data assigned for a sub-map, which prevents undesirable large growth of the tree. This rule acts as a stopping criterion of the algorithm, because this rule can disable the network expansion when the representation of the neuron will be minor that a percentage that is about 3 to 5% of the data set.

3.2. Criterion of Flexibility for Expanding the Network

The heuristic rules of activity and representation can have their average values modified by factors to allow the flexibility of the network. This is made using a monotonic function that privileges more expansion of the network in initial levels. Thus, mean of activity and mean of quantization error are modulated to lower values in levels near to the root map, and to great than mean values as the level increases.

An example could be written as *factor*(*level*) = a**level* ^ ^{b}, where $0 < a,b < 1$. Figure 1 shows the case when $a = 0.5$ and $b = 0.6$. The level is in the x axis. It can be seen that in the beginning, for $x = 1$ or 2, the attenuation is lower than 1 and the criterion value is decreased. Otherwise, for values upper than 3 result in high criterion to be accomplished, making the node forced to exceed a high criterion value to expand to a child map. For example, in the case of activity criterion, for level = 3 the neuron has to exceed about 1.2 times the average activity of the neuron's map to be a candidate for expansion.

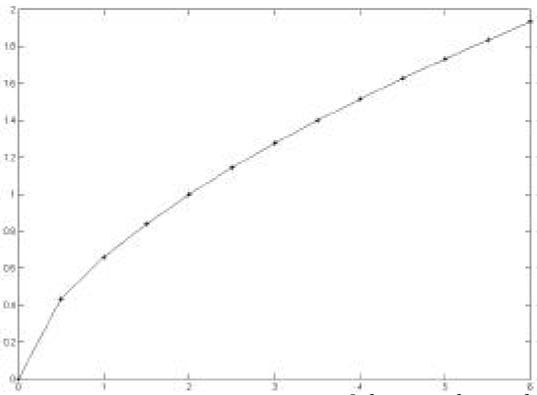


Fig. 1: Example of an attenuation / amplification factor for the heuristic rule criterions.

3.3. Determination of the Size to the Sub-maps

After the decision of that a neuron will be expanded to a child map it becomes necessary the determination of the sub-map size. It can be defined as a function of the percentage of the data the neuron represents relative to the total size of the data used to train the map. The following equation defines the size of the child map:

$$M_f = \left(\frac{N_f}{N_p} \right)^b M_p \qquad (4)$$

where N_f is the number patterns associated to the neuron that will be expanded, N_p is the number of patterns used to train the map, b is a constant (0.3 in our simulations) and M_p is the size of the parent map.

3.4. Mechanism of Pruning

The process known as "pruning" is present in DHSOM algorithm of the following way: for a given sub-map, if the quantization error of the parent neuron is reduced at least by 20% the expansion is considered valid. Otherwise the sub-map is excluded from the hierarchy. The purpose of this mechanism is to prevent the time wastefulness in that it says respect to the expansion of the tree and in the phase of signal and image codification. Only maps that significantly reduce the quantization error, regarding its parent neuron, are preserved in the tree.

4. DHSOM algorithm

DHSOM is an extension of the conventional SOM. The map in the first level, or root map, is trained with the full data set and according the quantization factors associated to each neuron new sub-maps are generated upon the data associated to them. Geometrically, the cells of the Voronoi diagram of the map in a level k are partitioned in a level $k + 1$ increasing of the resolution (detailing) of the reconstructed image. Figure 2 illustrates a simplified configuration of the structure generated by DHSOM.

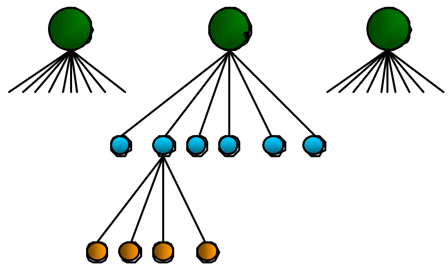


Fig. 2. Structure generated by the DHSOM

Due the dynamic characteristic of the network, after the convergence of the algorithm, we will have an unbalanced tree that will reflect the variability of the data set. The codebook will be generated to represent the regions with great accumulations of data.

The training of the DHSOM is performed sequentially, i.e., top to down, and the basic algorithm is described below:

1. Set current level = 1.
2. Train the map(s) of the current level.
3. Partition the data set used to train the map in the current level generating subgroups. Each subgroup S_i represent data mapped to neuron i of the current level.
4. For each neuron j that to satisfy the three conditions of growth, generated a sub-network with size according to eq. 4 that will be trained with the SOM algorithm, using the subgroup S_j mapped to its parent node.
5. Set level = level + 1 and train the map(s) of the current level.
6. Test for the pruning mechanism. Eliminate tree nodes (sub-maps) that achieve low decrease of quantization error regarding its parent node.
7. If there are yet maps to train back to step 2 otherwise end.

5. Results

The platform used was a personal computer with processor Intel Pentium III-800 MHz. All tests were performed in the Matlab® environment. Some functions of SOM Toolbox [12] were also utilized.

5.1. Synthetic data

Figure 3 shows the final configuration for a ten neurons one-dimensional map after training. The data set consists of four bi-dimensional clusters generated after Gaussian densities. According to the DHSOM algorithm most of the root map neurons expanded (only neuron 8 did not) to sub-maps and figure 4 shows the hierarchy obtained where *nf* is the number of sub-maps. Note that each node of the tree represents one entire map, and all the generated neurons can be seen in figure 5. The root map generated 9 sub-maps, only neuron 8 was not expanded. Figure 6 illustrates also the Voronoi diagram and links from data to their assigned neuron in the root map. Ths size of circles around root map nodes are related to their activity, $H(i)$.

As can be seen in figure 6, the generated sub-maps receive data only from its parent node and act to minimize the quantization error.

5.2. Image data

The algorithms used in our tests were the LBG, Hierarchical LBG, SOM, HSOM and DHSOM. Lena and Zelda images used had 512 x 512 pixels size and 256 gray levels, or 8 bits/pixel (bpp).

The algorithm used to train SOM maps was the batch one and the initial neighborhood was set 80% of the map size, decreasing to zero at the end of the training [7]. 3000 4×4 block samples were collected at random to form the training data.

The Hierarchical (or structured) LBG and the HSOM used had two levels; In both cases there were 10 codevectors at each level totalizing 100 leaves neurons in the structure that integrate the codebook.

The DHSOM was generated with 1-D output space maps. In the implementations with images we got structures with 4 hierarchical levels, being that the algorithm made use only of three initial levels because the last level was eliminated by the pruning mechanism.

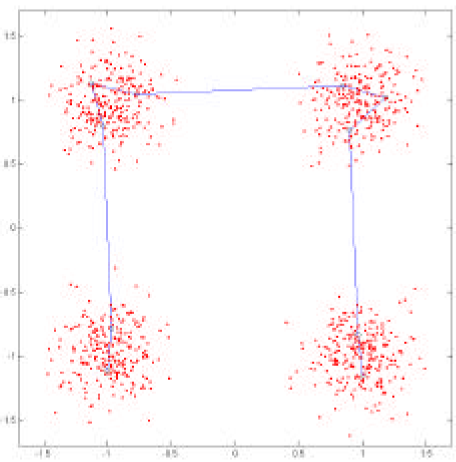


Fig. 3 – Data set (four clusters) and the grid (10 neurons 1-D map) after training.

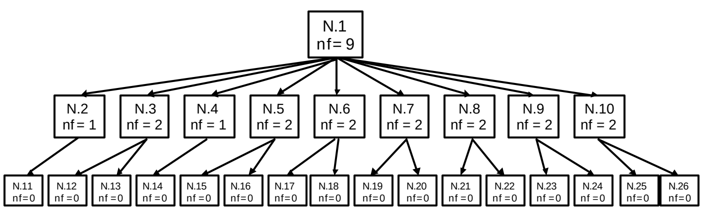


Fig. 4: Tree of maps obtained for data presented in fig. 3.

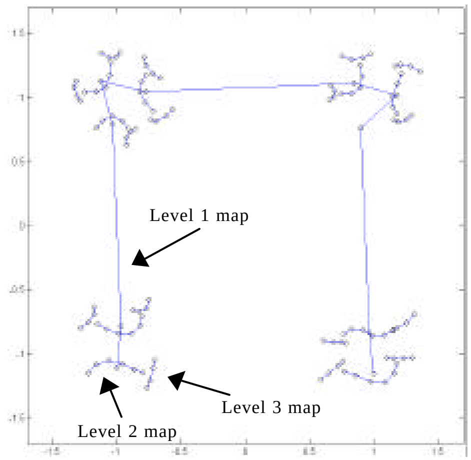


Fig. 5: All the generated neurons.

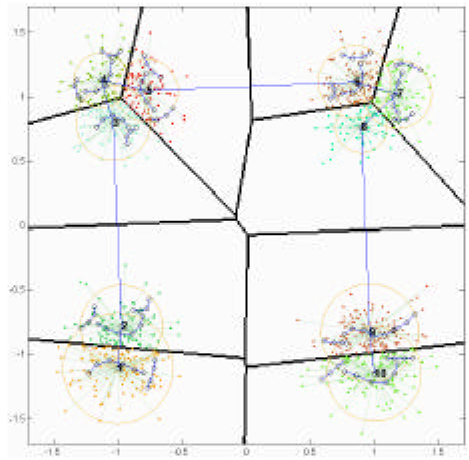


Fig. 6: All the generated neurons, data and Voronoi diagram for the root map.

Figure 7 and 8 show the original image of Lena and Zelda used in the simulations. It is not observed qualitative distinction between the reconstructed images with the Dynamic LBG and the DHSOM. Results for DHSOM are shown in figures 9 and 10. Then a quantitative analysis of the images becomes necessary. The measures of quality had been taken off in function of the mean square error (MSE) and the peak signal/noise relation (PSNR) given by:

$$MSE = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\left[\hat{I}(x,y) - I(x,y)\right]^2 \qquad (5)$$

$$PSNR = 10\log_{10}\left(\frac{255^2}{MSE}\right) \qquad (6)$$

where $I(x, y)$ represents the original image, $\hat{I}(x, y)$ is the reconstructed image and 255 is the maximum value for pixel.

Table 1 and 2 illustrates the results for the images of Lena and Zelda, respectively. In our case the images were scaled to the [0, 1] range, than the value 255 (see eq. 6) is replaced by 1 in results presented in table 1 and 2.

Fig. 7. Original Image of Lena

Fig. 8. Original Image of Zelda

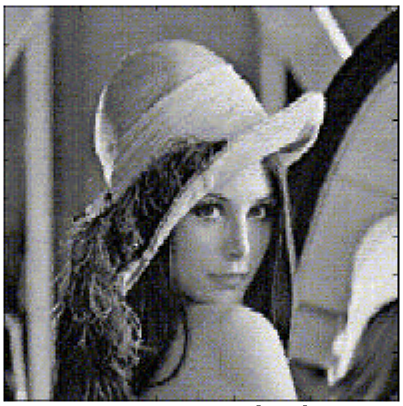


Fig. 9: Lena reconstructed with DHSOM

Fig. 10: Zelda reconstructed with DHSOM

The quantitative results show that the dynamic structures present a better performance to the hierarchical (fixed structure) methods. Both Hierarchical LBG and HSOM had 100 leaf nodes or centers, i.e., the size of the codebook. Its also seen that the dynamic structure of the algorithm SOM presents slight superior results to the dynamic structure of the LBG. It is important verify that the codebook generated by the traditional algorithms and the fixed hierarchical structures presents 100 vectors reference in codebook, while that the size of codebook generated by the hierarchical and dynamic structures is 79 and 83 vectors of reference for the images of Lena and Zelda, respectively. Even with less codevectors the quantitative results were better because the nodes were generated in regions of the input space were there a high quantization error.

TABLE 1
Quantitative analysis of the reconstructed Image of Lena

	MSE	PSNR
LBG (100 centers)	$1{,}8 \times 10^{-3}$	27,5
SOM (100 neurons)	$1{,}8 \times 10^{-3}$	27,53
Hierarchical LBG	$1{,}8 \times 10^{-3}$	27,37
HSOM	$2{,}2 \times 10^{-3}$	26,63
Dynamic LBG	$2{,}3 \times 10^{-3}$	26,33
DHSOM	$2{,}1 \times 10^{-3}$	26,7

TABLE 2
Quantitative analysis of the reconstructed Image of Zelda

	MSE	PSNR
LBG (100 centers)	$1{,}1 \times 10^{-3}$	29,76
SOM (100 neurons)	$1{,}1 \times 10^{-3}$	29,69
Hierarchical LBG	$1{,}1 \times 10^{-3}$	29,46
HSOM	$1{,}4 \times 10^{-3}$	28,68
Dynamic LBG	$1{,}3 \times 10^{-3}$	29,02
DHSOM	$1{,}2 \times 10^{-3}$	29,06

The small loss of quality of the dynamic structures when compared with the traditional algorithms of the LBG and the SOM it is compensated when we analyze the effective profit in the processing time. Table 3 shows to the training times for the algorithms using the image of Lena and Zelda. It is observed a reduction of 4,3 times the processing time compared with the conventional SOM.

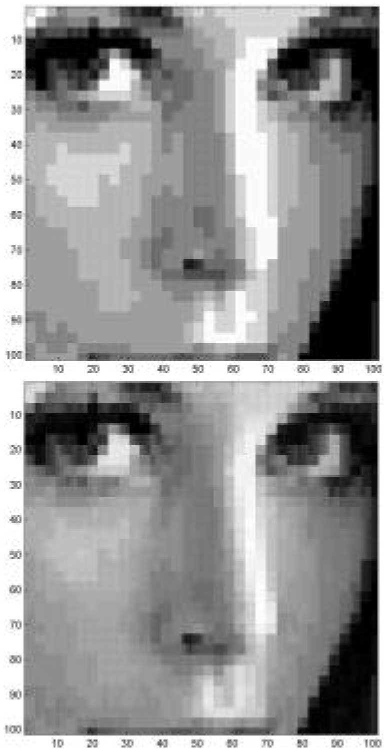
TABLE 3 - Training Time

	Lena	Zelda
LBG (100 centers)	207,57	208,49
SOM (100 nodes)	255,68	260,4
Hierarchical LBG	51,73	50,58
HSOM (100)	53,88	53,66
Dynamic LBG	60,64	60,31
DHSOM	57,49	58,82

Not only the training and coding time, other important possibility in using a hierarchical configuration is to progressively transmit the image, working with multi-resolution boarding. In applications where some distortion is acceptable we can work only with few levels of the tree enabling high compression rates and a faster image coding / decoding.

Regarding the compression rates, in the presented case we had got 0.437 bits for pixel, which corresponds about compress the data to 5,5% of the original size, even without considering compression techniques that eliminate code redundancy (Huffman, for example).

Figure 13 illustrates a portion of the Lena image after reconstruction. In fig. 13 (a) only the first level was used whereas fig. 13 (b) was reconstructed using the hierarchy obtained by the DHSOM. It is seen that the high level maps try to minimize the error by closing to the data.



(a)

(b)

Fig. 13 - Portion of Lena image: (a) reconstructed image using only first level map, and (b) reconstructed image using tree obtained with DHSOM

6. Conclusions

A new structured self-organizing map with dynamic growth was described. The main objective was to reduce the computational efforts both in training and coding/decoding phases, while keeping good image reconstruction.

The heuristic rules were defined with the aim to reduce the quantization error and to privilege nodes with high activity. The results were considered good that it was possible to keep better quantitative and similar qualitative figures using less codevectors (leaf neurons) and with less training and coding times.

Future works include the development of better rules in order to optimize these results and to explore the concept of parallel processing in the phase of generation of codebook. That is possible because the pertaining maps to one exactly hierarchical level are depended. Also, the edge degradations in the reconstructed image, horizontal growth and pruning, and applications of DHSOM in data mining and knowledge discovery in databases are also being considered.

Acknowledgements: This work was also in part conducted by J. M. Barbalho while pursuing his M.Sc. degree at UFRN.

References

[1] Erickson, D.S. and Thyagarajan, K.S., A neural network approach to image compression, *Proc. of the IEEE Intl. Symp. on Circuits and Systems*, vol. 6 , pp. 2921 –2924, 1992

[2] Linde, Y., Buzo, A. and Gray, R. M., An Algorithm for Vector Quantization Design, *IEEE Trans. Commun., Vol. 28, pp. 84-95, 1980*.

[3] Setiono, R. and Guojun, Lu, Image compression using a feedforward neural network, *In: Proc. of the IEEE World Congress on Computational Intelligence*, vol. 7, pp. 4761 -4765, 1994

[4] Qiu, G., Varley, M.R., and Terrell, T.J., Variable bit rate block truncation coding for image compression using Hopfield neural networks, In: *Proc. Third International Conference on Artificial Neural Networks*, pp. 233 –237, 1993

[5] Barbalho, J., Dória Neto, A.D., Costa, J.A.F. and Netto, M.L.A., Hierarchical SOM applied to Image Compression. *In: Proc. of the Intl. Joint Conf. on Neural Networks*, Washington, DC, July 2001, pp. 442-447.

[6] Gray, R. M. and Neuhoff, D. L. (1998). Quantization. *IEEE Trans. on Information Theory*, vol. 44, pp. 2325-2383.

[7] Kohonen, T., "*Self-Organizing Maps*," 2nd ed. Springer-Verlag: Berlim.1997.

[8] Nasrabadi, N. M. and King, Y., Vector Quantization of Images Based upon the Kohonen Self-Organizing Feature Maps. *In Proc. IEEE Int. Conf. Neural Netwoks, 1988*.

[9] Costa, J. A. F., *Automatic Classification and Data Analysis by Self-Organizing Neural Networks*. D.Sc.Thesis, State Univ. of Campinas, São Paulo, Brazil, 1999 (In Portuguese).

[10] Koikkalainen, P., Progress with the Tree-Structured Self-Organizing Map, *Proc. of the 11th European Conference on Artificial Intelligence, pp. 211-215, 1994*.

[11] Lampinen, J. and Oja, E., Clustering properties of hierarchical self-organizing maps. *Journal of Mathematical Imaging and Vision*, vol.. 2, pp. 261-272, 1992.

[12] SOM Toolbox Team. *SOM Toolbox*, URL: http://www.cis.hut.fi/projects/somtoolbox/ . 2000.

[13] Costa, J.A.F. and Netto, M.L.A., Estimating the Number of Clusters in Multivariate Data by Self-Organizing Maps. *Intl. Journal of Neural Systems*, vol. 9, pp. 195-202, 1999.

[14] Costa, J.A.F., & Netto, M. L. A., "Clustering of complex shaped data sets via Kohonen maps and mathematical morphology". In: *Proceedings of the SPIE, Data Mining and Knowledge Discovery*. B. Dasarathy (Ed.), Vol. 4384, pp. 16-27, 2001.

[15] Costa, J.A.F. and Netto, M.L.A., A new tree-structured self-organizing map for data analysis. In: *Proc. of the Intl. Joint Conf. on Neural Networks*, Washington, DC, July 2001, pp. 1931-1936.

[16] Vieira, F., Dória Neto, A.D., e Costa, J.A.F., An Efficient approach of the Salesman Travelling Problem Using Self-Organizing Maps, *Intl. Journal of Neural Systems*, vol. 13, 2003, (*in print*).

[17] Xavier, S., Dória Neto, A.D., Costa, J.A.F. and Netto, M.L.A., A neural hybrid system to large capacity memory association. In: *Proc. of the Intl. Joint Conf. on Neural Networks*, Washington, DC, July 2001, pp. 1174-1179.