ESANN'2009 proceedings, European Symposium on Artificial Neural Networks - Advances in Computational Intelligence and Learning. Bruges (Belgium), 22-24 April 2009, d-side publi., ISBN 2-930307-09-9.

Hardware Implementation Issues of the Neighborhood Mechanism in Kohonen Self Organized Feature Maps

Marta Kolasa[1] and Rafał Długosz[2,*]

1- University of Technology and Life Sciences, Institute of Electrical Engineering,
ul. Kaliskiego 7, 85-791, Bydgoszcz, Poland
2- Swiss Federal Institute of Technology in Lausanne, Institute of Microtechnology,
Rue A.-L. Breguet 2, CH-2000, Neuchâtel, Switzerland

Abstract. In this paper, we discuss an important problem of the selection of the neighborhood radius in the learning schemes of the Winner Takes Most Kohonen neural network. The optimization of this parameter is essential in case of hardware realization of the network given that the lower values of the radius can result in significant reduction of both the power dissipation and the chip area, even by 40-60% that is important in application of such networks in low power devices. The simulation studies reveal that using large initial values of the neighborhood radius usually is not the most optimal. For a wide range of the training parameters some optimal values, usually small, of the neighborhood radius may be indicated that allow for the minimization of the quantization error.

1 Introduction

Kohonen neural networks (KNN), often referred to as self organized feature maps (SOFM), belong to the group of the networks that are trained without supervision. They usually consist of a single layer of neurons that are organized in a map-like structure with different grids. KNNs have been broadly described in the literature [1, 2], along with various optimization techniques of the learning algorithm [3, 4].

KNNs usually are realized using software platform, however many attempts to realize them on transistor level were undertaken in the past [5]. Hardware implement-tations create some specific problems that are of second importance in the software realizations, e.g. the necessity of optimization of the energy consumption and the chip area. In this paper we show that the key parameters of the learning algorithm e.g. the neighborhood radius have a great impact on these parameters and need optimization.

The paper is organized as follows. In next section we present shortly the idea of SOFM. Next we present the hardware implementation issues of such network. Then we analyze simulation results. The conclusions are formulated at the end.

2 Kohonen Neural Network

The competitive learning in Kohonen networks is an iterative process, in which all training patterns of a given learning set are in particular iterations presented to the

[*] this work is supported by EU Marie Curie OIF fellowship, project No. 21926

ESANN'2009 proceedings, European Symposium on Artificial Neural Networks - Advances in Computational Intelligence and Learning. Bruges (Belgium), 22-24 April 2009, d-side publi., ISBN 2-930307-09-9.

network in a random order. For each new pattern the network calculates the Euclidean distance between this pattern and the weights vector in all neurons in the map. A neuron, whose weights are the most similar to the training pattern becomes a winner and is allowed to adapt its weights. Two learning algorithms are commonly distinguished, namely the Winner Takes All (WTA) and the Winner Takes Most (WTM). In the WTA approach only the winning neuron adapts its weights, while in the WTM algorithm also the neurons that belong to the winner's neighborhood are adapted.

In the WTA algorithm each neuron is independent from each other, which makes the WTA algorithm to have worse convergence properties than the WTM one. On the other hand in the WTM approach the neighborhood mechanism requires additional calculations, which makes this algorithm significantly more complex [1, 2, 6].

In this paper we focus on the WTM network with different topological neighborhoods, in which the initial value, R_{max}, of neighborhood radius, R, is one of the optimized parameters. In the WTM networks the adaptation is described as follows:

$$W_j(l+1) = W_j(l) + \eta(k)G(i, j, R)[X(l) - W_i(l)] \quad (1)$$

where η is the learning rate, W_j is the weights' vector of a given, j^{th}, neuron, and X is the input pattern in the l^{th} presentation. The neighborhood function, $G()$, introduces an additional factor to the adaptation process that depends on the neighborhood radius R. This factor may be interpreted here as a strength factor in the adaptation process of particular neurons that belong to the winner's neighborhood. One of the commonly used functions is the rectangular function, described as follows:

$$G(i,j,R) = \begin{cases} 1 & \text{for } d(i,j) \le R \\ 0 & \text{for } d(i,j) > R \end{cases} \quad (2)$$

The term, $d(i, j)$, is a distance from the winning, i^{th}, neuron to any other, j^{th}, neuron in the map. The neighborhood topology is defined as a grid of neurons in the map that determines which neurons belong to the neighborhood of the winner for a given value of R [1, 2]. Typical topologies that may be found in the literature are shown in Figure 1, i.e. the rectangular with 4 and 8 neighbors and the hexagonal grid. The topologies shown in Figure 1 are in this paper referred to as rect4, rect8 and hex, respectively.

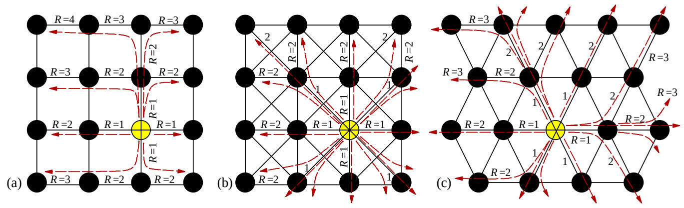

Fig. 1: The Kohonen map organized in different grids: (a), (b) the rectangular grid with four (rect4) and eight (rect8) neighbors and (c) the hexagonal grid (hex)

3 Hardware implementation issues of the WTM network

The CMOS implementation of the network topologies presented in Figure 1 has been proposed by the authors earlier. The idea of the neighborhood mechanism along with the CMOS circuit have been described in [6] for the rect8 topology, while in [7]

ESANN'2009 proceedings, European Symposium on Artificial Neural Networks - Advances in Computational Intelligence and Learning. Bruges (Belgium), 22-24 April 2009, d-side publi., ISBN 2-930307-09-9.

the authors proposed an additional mechanism that allows for easy reprogramming the network to both the rect4 and the hex topologies in a single chip. These circuits working asynchronously and in parallel for all neurons in the map are very efficient solution that makes the WTM network with any dimensions as fast as the WTA one. On the other hand, there are various implementation issues that have to be considered in case of the CMOS implementation of this mechanism. They may be boiled down to the necessity of reducing the number of logic gates used in the circuit. One of the parameters that have to be optimized is the initial value of neighborhood radius $R_{\max}$.

In the hardware implementation proposed in [6, 7] the parameter R is represented by a given number of bits, z, depending on the map dimensions. For example, in case of the map with 32x32 neurons z equals to 5 for the rect8 and the hex grids and to 6 for the rect4 grid. Particular neurons are linked only to the closest neighbors using $2(z+1)$ connecting paths. The factor 2 results from the necessity of sending the radius as well as another controlling signal, EN, in both directions between two neighboring neurons [6, 7]. Each bit in the radius R, in each neuron requires 6 logic gates. In the map with 32x32 neurons this makes ca. 30 000 logic gates (32·32·5·6) in total.

Some additional connecting lines, and corresponding logic gates, are required by the enable (EN) signal [6]. In the not programmable circuit this signal requires one logic gate per each connection. The smallest number of these logic gates is required in the rect4 topology, while the largest in the rect8 topology. For the 32x32 neurons in the map this makes about 4 000, 6 000 and 8 000 of gates, for particular topologies.

In general, this large number of the logic elements has a serious impact on both the chip area and the power dissipation of the KNNs implemented in hardware.

4 Optimization of the neighborhood mechanism

It is commonly assumed that the neighborhood radius R at the beginning of the learning process covers at least a half of the map [1, 2] and then decreases to zero according to the following formula:

$$R_k = 1.00001 + (R_{\max} - 1) \cdot (1 - k/L_{\max}) \tag{3}$$

where k is the number of a given iteration, while $L_{\max}$ is a total number of all iterations in the ordering phase. To verify this assumption the authors have implemented a software model for all network topologies described above, and made experiments for different network parameters. These experiments showed that this assumption may be relaxed in many cases, as the quantization error, which is defined as:

$$Q_{err} = \frac{\sum_{j=1}^{m} \sqrt{\sum_{l=1}^{n} (x_{jl} - w_{wl})^2}}{m} \tag{4}$$

where m is the number of the training patterns, does not decrease uniformly with the following iterations. Figure 2 (a) illustrates an example experiment, which starts with the radius $R_{\max} = 38$ for the rect4 topology with 20x20 neurons. It can be seen that the ordering in the map starts, in practice, only around the 700$^{\text{th}}$ iteration i.e. for $R = 11$, which is about ¼ of the map size. Further experiments with smaller starting values of the radius $R_{\max}$ gave better results, as the final value of the Q_{err} became smaller than in case (a), as shown in Figure 2 (b). For very small values of $R_{\max}$, e.g. 1 and 2, the error became large again, just like in the WTA algorithm. This is shown in Figure 3 (c). These simulations show that there exist some optimal settings of the $R_{\max}$ parameter, which have to be determined.

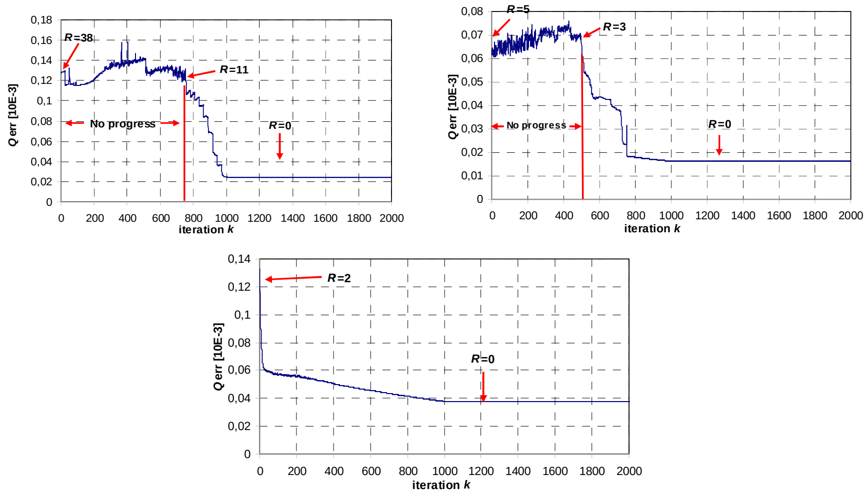

Fig. 2: Quantization error for the rect4 topology for 20x20 neurons in the map for: (a) $R_{\max}$= 38, (b) $R_{\max}$= 5 i.e. for 1/8 of the map dimensions, (c) $R_{\max}$= 2

To make these observations reasonable similar simulations have been performed for several hundreds different network parameters i.e. for different topologies, shown in Figure 1, different map dimensions (4x4 – 32x32), different numbers of inputs (2 and 3), different values of $R_{\max}$ and different training sets. We have used training sets, in which data are placed regularly in the data space, as well as those, in which centers representing different data classes are placed randomly in this space. These centers were surrounded by different amounts of training patterns with the maximal distance to these centers as an additional parameter. The example results have been collected in Figures 3 and 4, for two cases i.e. for data regularly placed in the 2-D data space and for 3-D space with the random distribution of these centers. Particular diagrams present the final quantization error as a function of the initial value of the radius $R_{\max}$. The 'final' means the value of this error after the training process is completed, i.e. for R equal to 0 and for the learning rate η equal to 0 as well. The presented results are for several example map dimensions i.e. for 4x4, 8x8, 16x16 and 20x20 neurons.

Several important conclusions may be drawn when analyzing these results. The first is that for most cases the rect4 topology brought either the best results or at least not worse than in case of other topologies. However, these optimum results in this case are for small values of the $R_{\max}$ parameter, while for larger values of $R_{\max}$ slightly better results are for other topologies, especially in case of larger maps. This is more visible in case of regularly placed data centers. For random data centers the rect4 topology in most cases was the most efficient even for wider range of $R_{\max}$, though in this case there is no such distinct optimum visible like for the regular data. In practice, the results for irregularly placed data centers have potentially a better meaning.

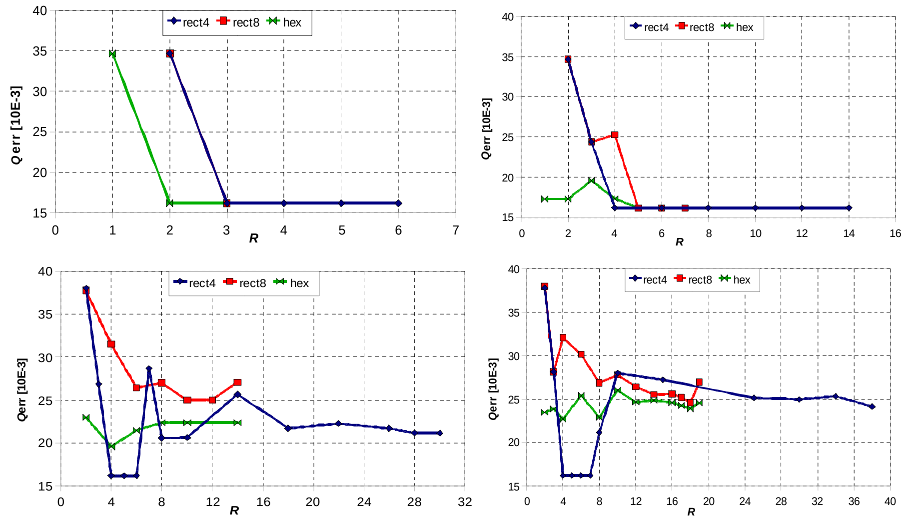

Fig. 3: The final quantization error as a function of the initial value of the radius, R_{max}, for different topologies, for (a) 4x4, (b) 8x8, (c) 16x16 (d) 20x20 neurons. The results are for 2-D input data space with regular distribution of data centers.

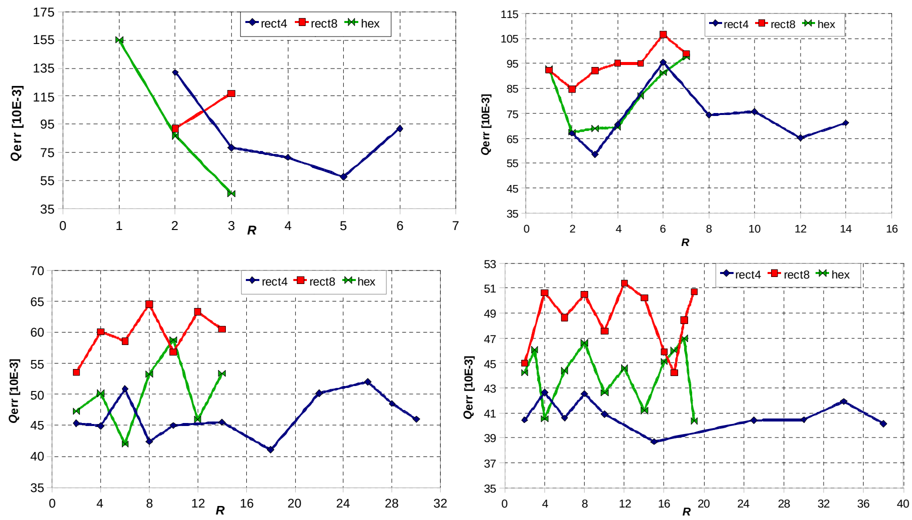

Fig. 4: The final quantization error as a function of the initial value of the radius, R_{max}, for different topologies, for (a) 4x4, (b) 8x8, (c) 16x16 (d) 20x20 neurons. The results are for 3-D input data space with random distribution of data centers.

The second observation is that in almost all cases a small optimal value of R_{max} exists, for which the Q_{err} parameter also reaches the low value and the map becomes ordered properly. This optimum usually is for R_{max} that is between 3 and 8, depending on the map dimensions. This is also better visible for the regular data, though for irregular data the Q_{err} parameter varies rather moderately. For example, in the case shown in Figure 4 (d) the optimum for the rect4 grid is for R_{max}=15, but in this case the error is smaller by about 4% only than in the cases of R_{max} = 3 or 5. This small difference, fluctuating between -5 and 4 % was observed for different input data files.

Analyzing the presented results the conclusions may be as follows: The neighborhood mechanism does not need to operate with large values of R_{max} – the values on the level of 10-20% of the map dimensions were sufficient in most cases. The best results in most cases have been achieved for the rect4 topology. These conclusions have the following practical meaning. In case of the rect4 topology an average number of gates associated with the EN signals equals to about $4 \cdot z$, so it is only a half of gates required by the rect8 topology. On the other hand the optimum value of R_{max}, which is close to 3 – 8 allows for representation of the radius R on 2 or 3 bits only. This is especially important in large networks with 32x32 neurons or larger, as this reduces the number of the logical gates associated with the radius signal even by 40-60 % i.e. to about 12500 for $z = 2$ and 18500 for $z = 3$, comparing to $z = 5$ i.e. when the radius R_{max} covers the half of the map in case of the rect4 topology. This additionally reduces the number of connecting paths in the map that minimizes the chip area.

5 Conclusions

The influence of the initial value the neighborhood radius on the training process in the WTM Kohonen network has been discussed. The presented results show that in case when the neighborhood mechanism is very strong i.e. when particular neurons have big number of neighbors (e.g. in the rect8 topology) and the initial value of the radius, R_{max}, is too large, the learning process is not optimal in many cases.

It has been also shown that seeking for small optimal values of the neighborhood radius is especially important in hardware implemented networks, since this allow for minimization of both the chip area and the power dissipation.

References

[1] T. Kohonen, *Self-Organizing Maps*, third ed. Springer , Berlin, 2001

[2] P. Boniecki, “The Kohonen neural network in classification problems solving in agricultural engineering”, *Journal of Research and Applications in Agricultural Engineering*, Vol. 50(1), Poznań, January 2005, pp. 37-40

[3] J.A. Lee, M. Verleysen , “Self-Organizing Maps with Recursive Neighborhood Adaptation”, *Neural Networks*, Vol. 15, Issues 8-9, October-November 2002, pp. 993-1003

[4] Jehan Zeb Shah, Naomie bt Salim, “A Fuzzy Kohonen SOM Implementation and Clustering of Bioactive Compound Structures for Drug Discovery”, *IEEE Symposium on Computational Intelligence and Bioinformatics and Computational Biology* (CIBCB), 28-29 September 2006, pp. 1-6

[5] D. M. Verleysen, P. Jespers, , and J-D Legat, “Analog Implementation of a Kohonen Map with On-Chip Learning”, *IEEE Transactions on Neural Networks*, Vol. 4, No. 3, May 1993, pp. 456-461

[6] M. Kolasa, R. Długosz, “Parallel Asynchronous Neighborhood Mechanism for WTM Kohonen Network Implemented in CMOS Technology”, *European Symposium on Artificial Neural Networks* (ESANN), Bruge, Belgium, April 2008, pp. 331-336

[7] R. Długosz,, M. Kolasa, “CMOS, Programmable, Asynchronous Neighborhood Mechanism For WTM Kohonen Neural Network”, *15th International Conference Mixed Design of Integrated Circuits and Systems* (MIXDES), Poznań, Poland, June 2008, pp. 197-201