Proc. Int. Conf. on Art. Neural Netw. ICANN '94
pp. 1125–1128
Springer Verlag 1994

Picture Encoding using Self-organized Cellular Neural Nets

Rüdiger W. Brause
J.W. Goethe-University, Frankfurt (FRG)
(brause@informatik.uni-frankfurt.de)

1 Introduction

The encoding of sensor information is a very important subject. Results are used in picture, speech and music encoding and compression which is very important in applications as telecommunication, sattelite data transmission, environmental and geographical image bases, music compression systems, high-resolution television transmission and storage and, therefore, in multi-media data bases.

An encoding compression of the n input lines of an input $\mathbf{x}$ can be done by a linear transformation $\mathbf{y}=\mathbf{W}\mathbf{x}$ using $m<n$ output lines $\mathbf{y}$. For linear systems, it is well known that the mean square error is minimized by selecting as base vectors the eigenvectors with the biggest eigenvalues. Thus, the eigenvector decomposition can be considered as an *optimal* transformation and should be preferred to all other current linear transformations as the descrete Walsh-Hadamard transformation, the descrete Fourier transformation or the descrete Cosinus transformation (Habibi et al., 1971) including the localized Fourier transform, the *Gabor* transformation (Daugman,1988). For pictures the base vectors are called *basis images*, and especially, *eigen images*.

The *transform coding* concept of (Habibi et al., 1971) reduces computational complexity of the transform by deviding the picture into subpictures, breaking the encoding process into parallel activities.

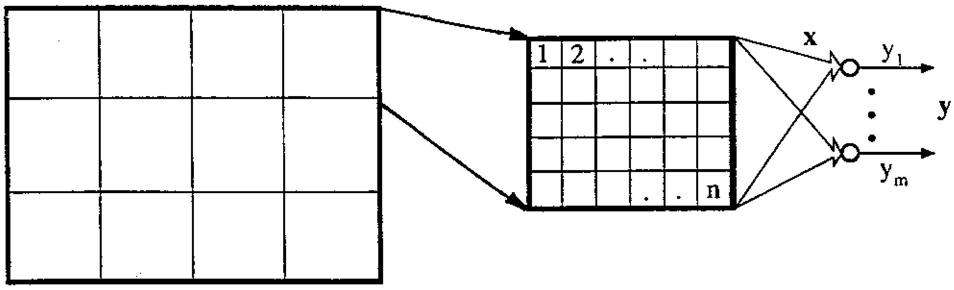

Fig. 1.1 The picture decomposition by parallel processing units

Now, let us identify each parallel processing unit as a subnet of m neurons. The parallel, distributed encoding of the whole picture is done by a neural network, where each subpicture of n pixels is coded by a local transformation process into m components by a subnetwork of m neurons.

Please note that the classical transform encoding process consists of two stages: a linear transformation and a vector quantization stage. Both stages contain non-linear operations and reduce the data stream; the linear transformation has a non-zero kernel and the vector quantization maps all data of the neighbourhood to only several class prototypes.

In this paper we use a neural model for the first encoding stage, although the second stage, the vector quantization, can also be modelled by neural nets (e.g. the Kohonen map or the maximum transinformation map (Brause,1994a)). Contrary to all conventional approaches, which either lead only to a eigenvector subspace with correlated coefficients, e.g. (Oja, 1989), (Williams,1985), (Földiak1989) or prescribes the formation order of the eigenvectors, e.g. (Sanger,1989), (Rubner et al.,1989), let us use the recent proposal (Brause, 1993) for a fully symmetrical network for PCA, construced by an objective function and implemented by a

→ 81%

biological plausible and in VLSI easily realizable network mechanism. A similar model was independently developed by (Freisleben, 1993).

2 The network model

Let us assume in a first step that we have *m* neurons which are laterally interconnected as shown in figure 2.1.

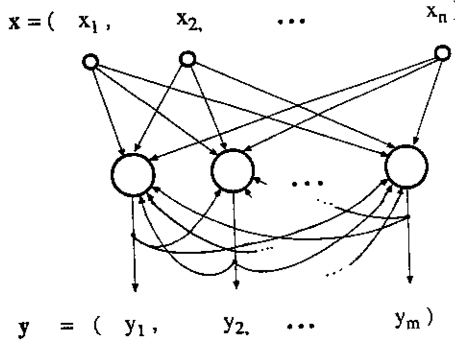

Fig. 2.1 The symmetric, lateral interconnected network model

Each neuron *i* has a randomly chosen weight vector $\mathbf{w}_i$. After we presented one input pattern $\mathbf{x}$ in parallel to each neuron of the linear system, the output of the neurons will result in

$$\mathbf{y} = \mathbf{W}\mathbf{x} + \mathbf{s} \qquad \mathbf{s} = \mathbf{U}\mathbf{y}, \quad u_{kk}=0 \qquad (2.1)$$

where s_i denotes the influence by the lateral connections which are weighted by the lateral weights u_{ij}. Rearranging (2.1) results in

$$\mathbf{y} = \mathbf{A}\mathbf{x} \qquad \text{with } \mathbf{A} = (\mathbf{I}-\mathbf{U})^{-1}\mathbf{W}$$

The input is assumed to be centered. If this is not the case, it can be made by introducing a special threshold weight learned with an Anti-Hebb-rule, see (Brause, 1993, 1994b). The learning rule for the weights $\mathbf{a}_i$ of the linear system is determined by the minimum of deterministic objective function

$$R(\mathbf{a}_1,..,\mathbf{a}_m) = 1/4\ \beta\Sigma_i\Sigma_{j\neq i}\ (\langle y_i y_j\rangle)^2 - 1/2\ \Sigma_i\ \langle y_i^2\rangle \qquad (2.2)$$

and is reached when the weight vectors become the eigenvectors of the autocorrelation matrix $\mathbf{C}$; the lateral inhibition weights become zero. To learn the weight vectors $\mathbf{a}_i$, a gradient descend may be used. This leads to complicated expressions for $\mathbf{w}_i$ and u_{ij}. Instead, we can use the stochastic algorithm

$$\mathbf{w}_i(t+1) = \mathbf{w}_i(t) + \gamma(t)\ \mathbf{x}\ (y_i + \beta\Sigma_{j\neq i}\ u_{ij}y_j) \qquad (2.3)$$

and for u_{ij} (t) the temporal floating average of the observed data with the constraints $\beta > 2/\lambda_{min}$ and $\gamma < 2/\lambda^2_{max}$. (2.4)

3 Self-organization of a cellular neural network

In this section a new self-organized, local formation of the PCA eigen images by the only locally interconnected network of section 2 is described. This approach is completely new: it combines the optimal PCA properties of the network in the input space with a kind of self-organization in the space of the physical input (and output) layout.

For the *activity phase*, a modular, localized organization of networks has been coined by Leon Chua and his coworkers by the term *cellular neural networks* (Chua et al., 1988) and has been adopted as a paradigm for a supercomputer for image processing, having a performance of 10^{12}=1000 GIPS (Giga instructions per second) in currently available technology (Roska, 1993). Here, the weights (*templates*) $\mathbf{W}^{ij}$ and $\mathbf{U}^{ij}$ of a neuronal cell (i,j) are set arbitrarily by the user and can be seen as a form of programming. A typical input layout for a 2-dim sensor field is shown in figure 3.1. Only the sensor elements (disks) and the neurons (rectangles), but no output lines are shown.

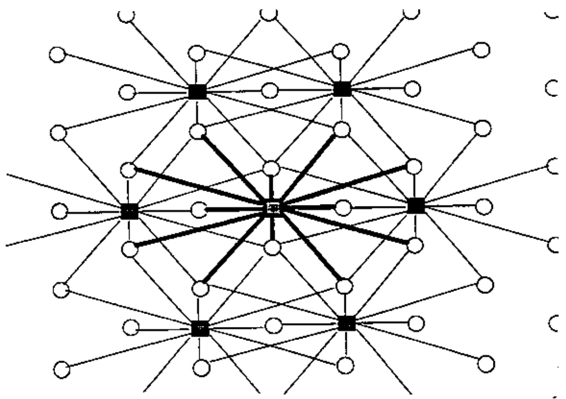

Fig. 3.1 An example of a 2-dim cellular neural network

For the *learning phase*, let us consider a symmetrical, lateral inhibited network as it has been introduced in section 2. Now, let us assume that we have only a limited radius *r* of inhibition influence as it is defined for CNN's. For a 1-dim neighbourhood network, as it used for instance for the coding of timed signals x(t) descretized by a tapped delay line, the interconnection schema for r=1 and r=2 is shown in figure 3.2.

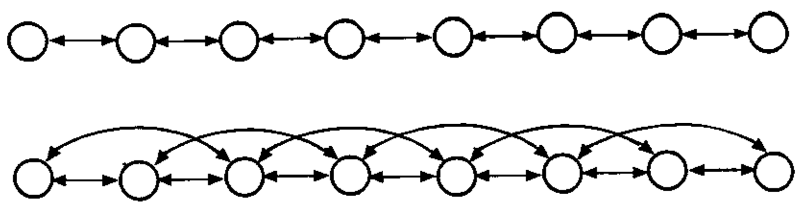

Fig. 3.2 1-dim lateral inhibition interconnections

The simulation used input patterns of n=36 components, each one set by Gaußean noise with different variance. The input weights for the m=8 neurons are randomly initilized with a fixed vector length $|w^i|=1$, the lateral weights are initialized with zero. The parameters β and γ_0 are set according to equation (2.4) with decreasing $\gamma(t)$. The results of the simulation is shown in table 3.1. Here, the index of the approximated eigenvector, denoted by the order of the corresponding eigenvalues, is listed for an inhibition radius of r=1 and r=2.

		neuron	1	2	3	4	5	6	7	8
r=1		eigenvector	1	2	1	2	1	2	1	2
r=2	run 1:	eigenvector	1	2	3	1	2	3	1	2
	run 2:	eigenvector	2	1	3	2	1	3	2	1

Table 3.1 The goal of convergence for r=1 and r=2

How can this result be explained? The eigenvector 1 is the one with the biggest eigenvalue $\lambda_1>\lambda_2$. Therefore, each neuron tries to converge to eigenvector 1 and will do this if no lateral connections exist. If two neurons have mutual lateral inhibition, the one having the initial weight vector most similar to eigenvector 1 will win the competition and converge to it, disabling all other neurons connected to it to converge to this eigenvector and leaving them only the possibility to converge to the one with the next smaller eigenvalue. This results in the alternating order of eigenvectors. Additionally, for r=2 we know that within the enlarged radius all other eigenvectors can exist except the one where the center neuron weight vector will converge to. Thus, different sets of eigenvectors can be observed; the PCA is performed by local groups of neuron. For example, in table 3.1 for run 1 the sets are {1,2,3}, {2,3,4}, {3,4,5}, {4,5,6}, {5,6,7}, {6,7,8}. This local base vectors can be termed *eigenvector-jets* analogously to the well known Gabor-jets of local Fourier transform.

Now, let us enlarge us this schema to the inportant case of two dimensions, for instance for image encoding on the sensor chip by cellular neural nets. In figure 3.3 definitions of neighbourhood for two 2-dim. networks of m=16 neurons, one with r=1 and the other with r=2, are shown.

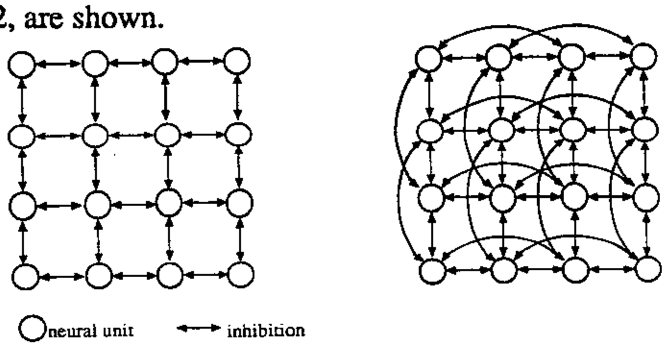

Fig. 3.3 2-dim neighbourhood inhibition definitions for r=1 and r=2

The results of the simulation runs for a 2-dim arrangement of m=16 neurons with r=1,2 are listed in table 3.2.

r=1	r=2, run1	run2	run3
1 2 1 2	1 2 3 1	2 3 1 2	2 1 3 2
2 1 2 1	3 1 2 3	3 1 2 3	1 3 2 1
1 2 1 2	2 3 1 2	1 2 3 1	3 2 1 3
2 1 2 1	1 2 3 1	2 3 1 2	2 1 3 2

Table 3.2 eigenvector convergence for r=1, r=2 and m=16 neurons

Although in the simulation the whole input is received by all neuronal units, the same results can be attended for systems with also localized input (local receptive fields) if the input statistics are translation-invariant. For most data like speech and image this is the case, because the neighboured data points are more correlated than ones with a longer distance, independent of the absolut position in time or picture coordinates.

References

R. Brause (1993) A Symmetrical Lateral Inhibited Network for PCA and Feature Decorrelation; Proc. ICANN-93, Springer Verlag, pp.486-489

R. Brause (1994a) An Approximation Network with maximal Transinformation; in this proceedings

R. Brause (1994b) Sensor Encoding using Lateral inhibited, Self-organized Cellular Neural Networks; University of Frankfurt, (Preprint)

L. O. Chua, L. Yang (1988) Cellular neural networks: Theory, Applications; IEEE Trans. Circuits Syst., Vol. 35, pp.1257-1290

J. Daugman (1988) Complete Descrete 2-D Gabor Transform by Neural Networks for Image Analysis and Compression; IEEE Trans. on Acoustics, Speech and Signal Processing Vol 36, No 7, pp. 1169-1179.

P. Földiák (1989) Adaptive Network for Optimal Linear Feature Extraction; IEEE Proc. Int. Conf. Neural Networks; pp. I/401-405 .

Freisleben (1993) PCA in a Network with Full Lateral Connections; Proc. Int. Conf. Art. Neural Networks ICANN-93, Springer Verlag

A. Habibi, P. Wintz (1971) Image Coding by Linear Transformation and Block Quantization; IEEE Trans. on Comm. Techn., Vol. COM-19, No 1, pp.50-62

N. S. Jayant, Peter Noll (1984) Digital Coding of waveforms, Prentice Hall .

E. Oja (1989) Neural Networks, Principal Components, and subspaces; Int. J. Neural Systems, Vol 1/1 pp. 61-68

T. Roska (1993) The analogic single-chip CCN visual supercomputer - a review; in: Chetverikov, Kropatsch (Eds.), Computer Analysis of Images and Patterns, Springer Verlag, pp. 813-821

J. Rubner, P. Tavan (1989) A Self-Organizing Network for Principal-Component Analysis, Europhys.Lett., 10(7), pp. 693-698.

Sanger (1989) Optimal unsupervised Learning in a Single-Layer Linear Feedforward Neural Network; Neural Networks Vol 2, pp.459-473

R.Williams (1985) Feature Discovery through Error-Correction Learning; ICS Report 8501, University of Cal. and San Diego