GENERALIZED CNN: POTENTIALS OF A CNN WITH NON-UNIFORM WEIGHTS

Marco Balsi

Dipartimento di Ingegneria Elettronica
Universita' di Roma "La Sapienza"
via Eudossiana 18, Roma, Italy I-00184

Abstract - *A generalization of the Cellular Neural Network paradigm is obtained by removing the uniformity constraint on weight values. Such Generalized CNNs are capable of new tasks, such as function approximation or associative memory. A stability analysis of these networks is presented. Adaptation and application of a gradient descent learning algorithm is then discussed.*

Introduction - The most important characteristic of Cellular Neural Networks (CNN) as defined by Chua & Yang [CHU88] is locality of connections, that greatly simplifies the layout problem for IC realization.

The said model, however, involves another significant constraint in the uniformity of weight values, so that processing consists of a spatial convolution with an operator defined by the cloning template. In this way, and also with the extension to non-linear and delay-type templates [ROS90], many image processing problems have been successfully solved [CNNA90], always using the network with strong enough self-feedback as to obtain saturated (i.e. ±1) steady-state outputs.

If the constraint on uniformity of weights is removed, new applications may be conceived for CNNs, such as Content-Addressable Memory (CAM) [TAN90], classification, function approximation, that cannot be implemented by traditional CNNs.

I shall call Generalized CNN (GCNN) a network which is identical to Chua & Yang's [CHU88] when parameters A(i,j;k,l), B(i,j;k,l), I(i,j) are allowed to vary arbitrarily while respecting the locality condition. GCNNs are included in the extended definition of CNN recently given by Roska [ROS92]. In the following, normalization $R_x = 1$, $C = 1$ is applied, and $B(i,j;k,l) = \delta^{kl}_{ij}$ without loss of generality. Voltages v_x, v_y, v_u in [CHU88] will be denoted x, y, u in the following. I shall always consider networks with clamped inputs and neglect specifying that sums over cell indices must be taken inside the significant neighborhood.

0-7803-0875-1/92 $3.00 ©1992IEEE

Therefore, GCNN neuron dynamics is written as follows:

$$\dot{x}_{ij}(t) = -x_{ij}(t) + \sum_{kl} A(i,j;k,l) f(x_{kl}(t)) + I(i,j) + u_{ij} \qquad (1)$$

Due to the peculiarities of the model, development of CNN theory has followed methods that are quite different from those usually applied in Neural Network (NN) theory and sometimes are more similar to Digital Signal Processing methodologies. This is most evident concerning one of the most important topics of NN theory: learning.

In this paper I shall discuss stability of GCNNs, based on known results of CNN and NN theory. On this basis I discuss possible applications of the paradigm. The learning issue is later confronted, by considering the possibility of application of standard NN algorithms, especially a gradient descent method.

Stability - It is possible to apply to GCNNs many known results from CNN and NN theory.

Theorem 1 *(bound on state values)*: If $\forall\ i,j\ \ |x_{ij}(0)| \leq 1$, $|u_{ij}| \leq 1$ and $|I_{ij}| \leq I$, then all states are bounded for all time $t > 0$ and the bound x_{max} is computed as follows:

$$x_{max} = 2 + I + \max_{i,j} \sum_{k,l} |A(i,j;k,l)| .$$

Proof: It is theorem 1 of [CHU88].

Theorem 2 *(stability of reciprocal nets)*: If $\forall\ i,j,k,l\ \ A(i,j;k,l) = A(k,l;i,j)$, then the network is globally asymptotically stable.

Proof: Follows immediately from theorems 2,3 and 4 of [CHU88].

Theorem 3 *(saturated steady-state output)*: If $\forall\ i,j\ \ A(i,j;i,j) > 1$ then magnitude of stable states must be grater than 1.

Proof: It is theorem 5 of [CHU88].

Theorem 4 *(stability of low-level-feedback nets)*: If $\forall\ i,j$

$$A(i,j;i,j) + 1/2 \sum_{k,l} \left(|A(i,j;k,l)| + |A(k,l;i,j)| \right) < 1$$

(where $A(i,j;i,j)$ may also take negative values) then the network is globally asymptotically stable.

Proof: Follows immediately from theorem 3 of [HIR89].

Theorem 5 *(stability of positive-cell-linking networks)*: If $\forall\ i,j,k,l$ $A(i,j;k,l) \geq 0$ and $\forall\ i,j,k,l$ there is a path on the network graph from i,j to k,l passing only through positive weights, then the network is almost everywhere asymptotically stable.

Proof: Follows from theorem 1 of [CHU90].

Theorem 6 *(stability of feed-forward processing GCNNs)*: If a GCNN can be decomposed into a cascade of globally asymptotically stable GCNNs, it is also globally asymptotically stable.

Proof: It is a corollary of theorem 5 of [HIR89].

Applications - Leaving away consideration of networks having periodic or chaotic attractors [BAR92], two modes of operation are interesting for neural computing: the first case is when, for every clamped input, the net is globally convergent [HIR89] regardless of initial conditions. In this case the net performs a classification of inputs when it has saturated final states, or otherwise it implements a continuous mapping from inputs to continuous-valued outputs. It need not be reset and can be cascaded by transferring the output of a stage to the input of the following one.

In the second mode inputs are clamped to a fixed value and the net has multiple attraction basins (it is convergent), depending on initial conditions. In this case classification or mapping is done between initial conditions and final outputs. This way of functioning is typical of Content-Addressable Memories (CAM).

In this paper I restrict consideration to networks operating in the first mode.

A task for such a network may be described in general by a function $\mathcal{F}: \mathcal{I} \rightarrow \mathcal{O}$ where input space $\mathcal{I}$ and output space $\mathcal{O}$ may be continuous or discrete. For instance, in a classification problem $\mathcal{I} \subseteq \mathbb{R}^n$ and $\mathcal{O} \subset \mathbb{Z}^m$ for suitable n and m. For this reason, I shall discuss learning problems as applied to function approximation tasks for boolean and continuous mappings.

Learning - All the work that is currently being done on CNN learning has been based on the fact that, unlike all other NN models, CNNs have very few parameters [HAR91] [SZO91]. This is not the case for GCNNs, and for this reason it is necessary to exploit classical learning algorithms from NN theory. The only example in literature may be found in [TAN90], where the Hebb rule was used for CAM purposes. Other possibilities include stochastic methods, such as simulated annealing, or gradient descent algorithms. In this paper I discuss application and modification of a gradient descent algorithm: Recurrent Back-Propagation (RBP) [PIN87], which is a

generalization of the well known Back-Propagation to non-feed-forward networks.

Recurrent Back-Propagation - Consider GCNN dynamics (1). A sigmoidal output function $f(x) = 2(1+\exp(-\beta x))-1$ was chosen, because the usual piecewise linear output function, having zero derivative outside the interval (-1,1), would give more problems of local minima during learning.

Denote γ^μ the desired steady state output matrix with input matrix u^μ. In the general case, u and γ take significant values over some units only (input and output neurons). Choose error measure E as follows:

$$E = 1/2 \sum_\mu \sum_{ij} \left(E^\mu_{ij}\right)^2; \quad E^\mu_{ij} = \alpha_{ij}[\gamma^\mu_{ij} - f(\underline{x}^\mu_{ij})]$$

where $\underline{x}^\mu$ is the stable state with input u^μ, α_{ij} is 1 for output neurons and 0 for neurons whose state is hidden to the external environment.

Gradient descent over error surface E yields the usual delta rule, which may be written as:

$$A_{k+1}(p,q;r,s) = A_k(p,q;r,s) + \Delta A_k(p,q;r,s); \quad I_{k+1}(i,j) = I_k(i,j) + \Delta I_k(i,j)$$

$$\Delta A(p,q;r,s) = -\eta \frac{\partial E}{\partial A(p,q;r,s)} = \eta \sum_\mu t^\mu_{pq} f(x^\mu_{rs}); \quad \Delta I(i,j) = \eta \sum_\mu E^\mu_{ij} f'(x^\mu_{ij})$$

where $t^\mu_{ij} = \left[f'(x^\mu_{ij})\right] z^\mu_{ij}$ and z^μ is the fixed point of the error back-propagation GCNN with dynamical equation

$$\dot{z}^\mu_{ij} = -z^\mu_{ij} + \sum_{kl} A(k,l;i,j) z^\mu_{kl} + E^\mu_{ij} \qquad (2)$$

This net has the same topology as the original one, with transposed weight tensor, a linear output function, and errors E_{ij} as biases.

System (2) has the same fixed points as (1), with the same eigenvalues [HER91]. However, convergence of the back-propagation network is not guaranteed by stability of fixed points of (1), because the output function is linear. For this reason, I added a piecewise linear output function $g(x) = 0.5(|x+K|-|x-K|)$; in this way we obtain the same back-propagation system while in the linear region, i.e. in a neighborhood of the solution (that can be made large for large enough K), but the network, whose dynamical equations are therefore written as

$$\dot{z}^\mu_{ij} = -z^\mu_{ij} + \sum_{kl} A(k,l;i,j) g(z^\mu_{kl}) + E^\mu_{ij} \qquad (2')$$

is now stable whenever the original one is.

When learning is accomplished on a sequential computer, RBP is rather slow, because it needs thousands of steps involving each the relaxation of two networks. However, this algorithm was chosen because of the success of ordinary back propagation in feed-forward nets. Thinking of real-life realizations, RBP may be implemented in hardware so that one iteration step of the algorithm lasts only a few time constants of the electronic circuit, so that learning time is actually governed only by the frequency of presentation of patterns.

Simulation results - Boolean and continuous function approximation was tried on 1-neighborhood planar networks (type 1, figure 1) and on layered systems of one-dimensional networks (i.e. planar networks with selected connections - type 2, fig. 2).

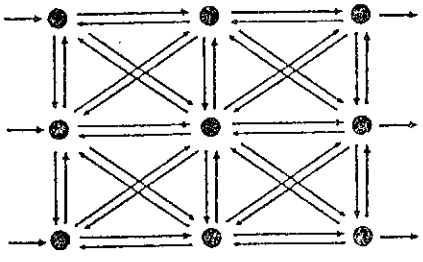
Learning was started with random weights satisfying theorem 4 so as to ensure stability, which was generally preserved during learning, provided that the learning rate was not too large, even if weights eventually violated the condition.

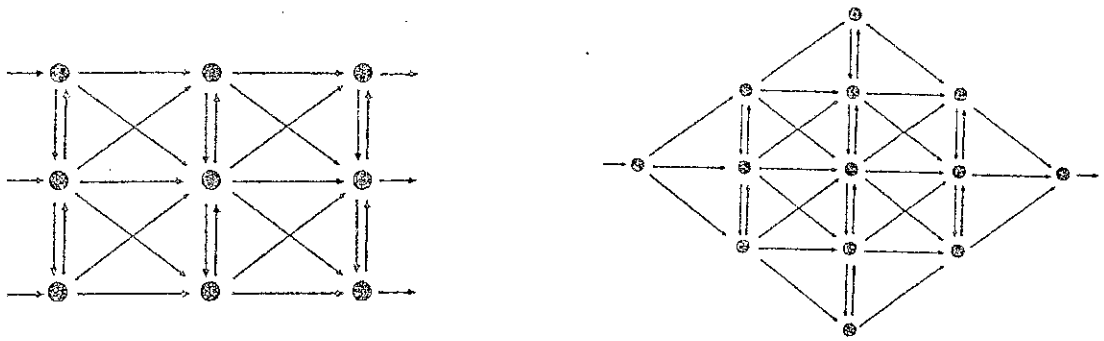
Type 1 nets were soon discarded, because they tend to oscillate very easily and therefore have very long settling times (hundreds of time-constants).

Type 2 networks, instead, proved capable of approximating boolean and continuous functions. A 2×2 network was taught to compute logical AND and XOR of its inputs (fig. 3); 3×3 nets with both topologies of fig. 2(a) and (b) were able to approximate sections of sinusoids.

Conclusions and perspectives - The results reported in this work are enough to say that new applications may open up for Cellular Neural Networks, in the fields of approximation, reconstruction of signals, classification. More extensive simulation is in progress in order to inquire into the performance of such networks when confronted with traditional fully connected networks.

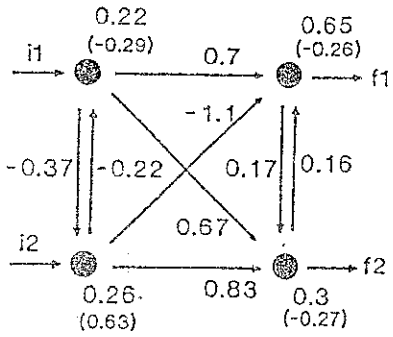
figure 1





(a) (b)

figure 2



i1	i2	f1 (XOR)	f2 (AND)
-1	-1	-0.99	-0.97
-1	1	0.99	-0.50
1	-1	0.99	-0.50
1	1	-0.96	0.94

figure 3 XOR/AND network: weights are written near connections and cells; biases I are in brackets.

References

[BAL92] M. Balsi, to appear in Int. J. Circ. Th. Appl.
[BAR92] A. Barone, M. Balsi, V. Cimagalli, in CNNA-92
[CNNA90] *Proceedings of IEEE International Workshop on Cellular Neural Networks and their Applications*, Budapest, Hungary, Dec.16-19, 1990
[CHU88] L.O. Chua, L. Yang, IEEE CAS-35(10), 1257 (1988)
[CHU90] L.O. Chua, T. Roska, IEEE CAS-37(12), 1520 (1990)
[HAR91] H. Harrer, J.A. Nossek, F. Zou, Technische Universität München, rep. TUM-LNS-TR-91-1
[HER91] J. Hertz, A. Krogh, R.G. Palmer, *Introduction to the Theory of Neural Computation*, Addison-Wesley, 1991
[HIR89] M.W. Hirsch, Neural Networks 2, 331 (1989)
[PIN87] F.J. Pineda, Phys. Rev. Lett. 59(19), 2229 (1987)
[ROS90] T. Roska: L.O. Chua, in [CNNA90]
[ROS92] T. Roska, Hungarian Academy of Sciences, rep. DNS-1-1992
[SZO91] P. Szolgay, T. Kozek, Hungarian Academy of Sciences, rep. DNS-10-1991
[TAN90] S. Tan, J. Hao, J. Vandewalle, in [CNNA90]
[ZOU91] F. Zou, J.A. Nossek, IEEE CAS-38(6), 675 (1991)