Complex Systems 4 (1990) 107–118

Learning by Choice of Internal Representations: An Energy Minimization Approach

D. Saad
E. Marom
Faculty of Engineering, Tel Aviv University,
Ramat Aviv 69978, Israel

Abstract. Learning by choice of internal representations (CHIR) is a learning algorithm for a multilayer neural network system, suggested by Grossman et al. [1,2] and based upon determining the internal representations of the system as well as its internal weights. In this paper, we propose an energy minimization approach whereby the internal representations (IR) as well as the weight matrix are allowed to be modified. Carrying out the analysis, consistency with the back propagation (BP) method [3] is demonstrated when a continuous valued system is considered, while a generalization of the CHIR learning procedure is obtained for the discrete case. Computer simulations show consistency with the results obtained by Grossman et al. for the restricted cases of parity, symmetry, and parity–symmetry problems.

1. Introduction

Learning by choice of internal representations (CHIR) [1,2] is a neural network learning algorithm based upon introduction of changes in both the IR of a discrete binary valued multilayer system, as well as its weights, for an ensemble of learned vectors. The changes are designed to improve the output value of each layer with respect to the current weights and the current IR of the preceding layer, so that a neural network-based recognition or classification system is obtained. Once the IR are redefined, the perceptron convergence rule [4] is applied for modifying the weights. The above mentioned method has been published [1] without a convergence theorem and provided quite successful simulation results. In this work, we concentrate in studying an energy minimization approach which seeks generating changes in the IR, while presenting an inherently consistent convergence mechanism. Two distinct cases of neuron representation, a continuous value case and a discrete one, are treated in this paper. The performance of the algorithm was tested using computer simulations of problems introduced in [1,2]. Several papers have been recently presented using IR changes in the training procedure [5,6], thus having certain resemblance with the method used in this work.

2. An energy minimization approach for a continuous valued net

We will define an energy function in a similar way as defined in the BP algorithm [3]:

$$E = \sum_{p=1}^{P} E^p = \sum_{p=1}^{P} \sum_{i=1}^{N^H} \left(v_i^{H,p} - \tau_i^p \right)^2 \tag{2.1}$$

where τ^p is the desired output vector related to the p vector out of P training vectors used in the training procedure; $v^{H,p}$ is the continuous value output vector of an H layer system, related to the p training vector; and N^H is the number of the output vector $v^{H,p}$ is obtained the the following equation:

$$v_i^{H,p} = f\left\{ \sum_{j=1}^{N^{H-1}} W_{ij}^H v_j^{H-1,p} \right\} = f\left\{ u_i^{H,p} \right\} \tag{2.2}$$

where

$$u_i^{H,p} = \sum_{j=1}^{N^{H-1}} W_{ij}^H v_j^{H-1,p}$$

The operator f, which represents the neural response, is considered to be a nonlinear operator acting on the product of the weight matrix W^H, connecting layers $H-1$ and H, and the IR of the preceding layer $v^{H-1,p}$.N^{H-1} is the number of neurons in the $H-1$ layer.

As in the BP algorithm, we will search for a procedure to minimize the energy E. However, we will allow at this time direct modifications of the IR $v^{H-1,p}$ as well as changes in the weights W^H.

The derivative of the energy function E is of the form

$$\frac{dE}{dt} = \frac{\partial E}{\partial W^H} \frac{dW^H}{dt} + \sum_{p=1}^{P} \frac{\partial E}{\partial v^{H-1,p}} \frac{dv^{H-1,p}}{dt} \tag{2.3}$$

where t is the training index and the derivatives are applied to each interconnection weight and each neuron of the IR vector. The following changes in W^H and $v^{H-1,p}$ will assure a negative contribution to the energy function:

$$\Delta W_{ij}^H = -\frac{\partial E}{\partial W_{ij}^H} = -\eta \left(v_i^{H,p} - \tau_i^p \right) f'\left[u_i^{H,p} \right] v_j^{H-1,p} \tag{2.4}$$

$$\Delta v_j^{H-1,p} = -\frac{\partial E}{\partial v_j^{H-1,p}} = -\sum_{i=1}^{N^H} \left(v_i^{H,p} - \tau_i^p \right) f'\left[u_i^{H,p} \right] W_{ij}^H \tag{2.5}$$

f' stands for the derivative of f with respect to the argument in the bracket, and η is a convergence coefficient. The modifications in the weights matrix and the IR can be performed for each training vector $v^{H,p}$ taken one at a time or any number of them taken in parallel. One should note that equation (2.4)

is identical to that derived for the weight matrix modifications in the BP algorithm.

Allowing these changes, to both the IR as well as to the weights, the energy function will decrease with each iteration and converge to a minimum value.

We will thereafter apply the same procedure to the former layer $(H-1)$, thus obtaining the required changes for the weight matrix W^{H-1} and the IR $v^{H-2,p}$.

The definition of the energy function will now be similar to the earlier definition (2.1). However, for an internal layer the target vector for the learning procedure will be the modified IR as computed from the previous correction (2.5):

$$E = \sum_{p=1}^{P} E^p = \sum_{p=1}^{P} \sum_{i=1}^{N^{H-1}} \left(v^{H-1,p} - v_{\text{new}}^{H-1,p} \right)_i^2 \tag{2.6}$$

One should note that the difference between the old IR $v^{H-1,p}$ and the new one $v_{\text{new}}^{H-1,p}$ is actually the former modification of the IR, i.e., $\Delta v^{H-1,p}$ (2.5). The output vector of the $H-1$ layer $v^{H-1,p}$ can now be expressed in terms of the previous weights W^{H-1} and IR $v^{H-2,p}$ as shown below:

$$v_i^{H-1,p} = f\left\{ \sum_{j=1}^{N^{H-2}} W_{ij}^{H-1} v_j^{H-2,p} \right\} \tag{2.7}$$

Applying the same procedure for obtaining the required modifications for the weight matrix and the IR, one gets

$$\Delta W_{ij}^{H-1} = -\frac{\partial E}{\partial W_{ij}^{H-1}} = -\eta \left(v^{H-1,p} - v_{\text{new}}^{H-1,p} \right)_i f' \left[u_i^{H-1,p} \right] v_j^{H-2,p} \tag{2.8}$$

$$\begin{aligned} \Delta v_j^{H-2,p} &= -\frac{\partial E}{\partial v_j^{H-2,p}} \\ &= -\sum_{i=1}^{N^{H-1}} \left(v^{H-1,p} - v_{\text{new}}^{H-1,p} \right)_i f' \left[u_i^{H-1,p} \right] W_{ij}^{H-1} \end{aligned} \tag{2.9}$$

where

$$u_i^{H-1,p} = \sum_{j=1}^{N^{H-2}} W_{ij}^{H-1} v_j^{H-2,p}$$

and the modifications are calculated for each vector $v^{H-1,p}$. As previously mentioned, the corrections for the weights can be made for each training vector one at a time or any number of them taken together.

We will now show that the expressions just derived lead directly to the known BP procedure. Inserting the explicit form of $\Delta v_i^{H-1,p}$ (2.5) into the brackets of (2.8) one indeed obtains:

$$\Delta W_{ij}^{H-1} = -\eta f' \left[u_i^{H-1,p}\right] v_j^{H-2,p} \sum_{k=1}^{N^H} \delta_k^{H,p} W_{ki}^H \tag{2.10}$$

where

$$\delta_k^{H,p} = \left(v_k^{H,p} - \tau_k^p\right) f' \left[\sum_{j=1}^{N^{H-1}} W_{kj}^H v_j^{H-1,p}\right]$$

which is identical to the modification of the weight matrix as derived from the BP algorithm [3]. Repeating the same procedure for all preceding layers, one obtains identical results to those obtained by the BP algorithm.

3. An energy minimization approach for a discrete valued net

After proving the consistency of the energy minimization approach for the continuous valued net with that of the BP algorithm, we apply the same theoretical tools to the case of a discrete valued net. We define the energy function in a similar way to the one used for the continuous valued net (2.1):

$$E = \sum_{p=1}^{P} E^p = \sum_{p=1}^{P} \sum_{i=1}^{N^H} \left(v_i^{H,p} - \tau_i^p\right)^2 \tag{3.1}$$

where τ^p, $v^{H,p}$ are defined ins equation (2.1–2.2) with one difference: the operator f, which represents the neural response, is now defined as the sgn function. The modifications for the IR and the weight matrix depend on the derivative of the neural response as shown earlier (2.4–2.5). Since the $\mathrm{sgn}(x)$ function has zero derivative for most of its range, we will approximate it by a function which has a small, almost constant, positive derivative along most of its dynamic range. An example of such a function is shown in figure 1. We will neglect the region near $x = 0$ by defining the width of this area to be smaller than our resolution. The IR and weight matrix modifications, expressed in equations (2.4) and (2.5) are thus applicable also for the *discrete IR case*. One notes, however, that the derivative function f', which is always positive, can be omitted in both equations: in the *discrete IR* modifications equation, we are interested only in sign changes, while in the weight matrix modification equation the contribution of the derivative f' can be included into the convergence coefficient η. As in the previous case, we will search for a procedure to minimize the energy E, by allowing modifications to both the IR and the weight matrix as described earlier in equations (2.4–2.5).

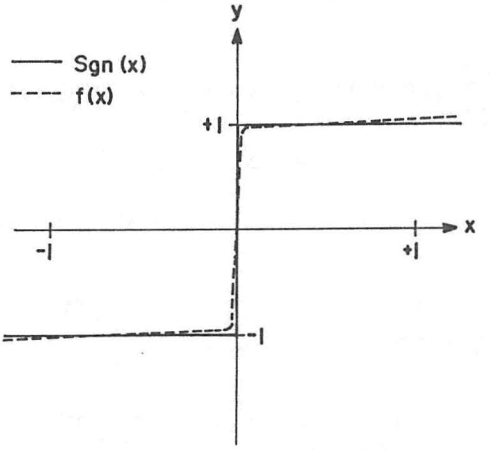

Figure 1: Typical function approximating $\mathrm{sgn}(x)$.

The required weight matrix modifications ΔW_{ij}^{H} resembles the perceptron learning rule:

$$\Delta W_{ij}^{H} = -\frac{\partial E}{\partial W_{ij}^{H}} = -\eta \sum_{p=1}^{P} \left(v_i^{H,p} - \tau_i^p \right) v_j^{H-1,p} \tag{3.2}$$

derived from equation (2.4) after eliminating the f' function as discussed above. Likewise, the modification $\Delta v^{H-1,p}$, assuring the convergence of the energy function, is

$$\Delta v_j^{H-1,p} = -\frac{\partial E}{\partial v_j^{H-1,p}} = -\sum_{i=1}^{N^H} \left(v_i^{H,p} - \tau_i^p \right) W_{ij}^{H} \tag{3.3}$$

Allowing discrete changes of the IR, the energy function decreases with each iteration and converges to a minimum value. A flip in the IR will be enforced whenever the modification $\Delta v_j^{H-1,p}$ is of opposite sign in relation to the value of $v_j^{H-1,p}$.

Due to the coefficient η, the weight changes (3.2) contribute much less to the energy function than those due to the IR changes (3.3). This might lead to local minima traps from which one can escape if the weights are updated several times in a row after each IR change, thus allowing the system to stabilize in a global minima. The number of times one should perform the weights change procedure for each IR modification is basically defined experimentally, but one can make a rough estimation, as shown below, by comparing the energy contributions of the two changes.

An energy change, generated by a bit flip in one of the output vectors, can be attributed to either an IR update in the $H-1$ layer or to an update of the W^H matrix:

$$v_i^{H,p} = f\left\{\sum_{j=1}^{N^{H-1}} W_{ij}^H v_j^{H-1,p} + \Delta W_{ij}^H v_j^{H-1,p} + W_{ij}^H \Delta v_j^{H-1,p} + \Delta W_{ij}^H \Delta v_j^{H-1,p}\right\} \tag{3.4}$$

Using equation (3.2) and comparing the contributions due to ΔW and Δv, one obtains:

$$\frac{\sum_{j=1}^{N^{H-1}} \Delta W_{ij}^H v_j^{H-1,p}}{\sum_{j=1}^{N^{H-1}} W_{ij}^H \Delta v_j^{H-1,p}} = \frac{\eta \sum_{p'=1}^{P} \sum_{j=1}^{N^{H-1}} \left(v_i^{H,p'} - \tau_i^{p'}\right) v_j^{H-1,p'} v_j^{H-1,p}}{\sum_{j=1}^{N^{H-1}} W_{ij}^H \Delta v_j^{H-1,p}} \tag{3.5}$$

Since the vectors are discrete, we can regard the summation over the output vector differences in the numerator of equation (3.5) as a "random walk." The IR modifications should be carried only over part of the IR neurons, "the most contributing ones," as will be explained later. This enables us to replace equation (3.5) by the expression

$$\frac{\sum_{j=1}^{N^{H-1}} \Delta W_{ij}^H v_j^{H-1,p}}{\sum_{j=1}^{N^{H-1}} W_{ij}^H \Delta v_j^{H-1,p}} = \frac{2\eta\sqrt{\alpha P N^{H-1}}}{2\beta N^{H-1} |W_{ij}^H|_{\max}} = \frac{\eta\sqrt{\alpha P}}{\beta\sqrt{N^{H-1}} |W_{ij}^H|_{\max}} \tag{3.6}$$

where $|W_{ij}^H|_{\max}$ is the value of the maximal weight (since we selected to modify the most significant neurons with the highest contributions), α is the percentage of erroneous output bits, and β is the fraction of IR bits that have been modified. Inserting practical values into equation (3.6), one obtains approximately a ratio of order 10 between the number of iterations carried out sequentially for implementing weight matrix modifications for each iteration modifying the IR.

The same procedure for defining the weight updates as well as the IR modifications will be then applied to the preceding layer $(H-1)$, thus obtaining the changes required for the IR $v^{H-2,p}$, and so on for the other preceding layers. To perform these changes in a specific layer h, the network is addressed with the same set of learning vectors. If the output vector differs from the desired one, we update the IR of layer h, and then the weight matrix W^h. If on the other hand the output is correct, we adopt the current IR as the proper one and go on to the other vectors.

The algorithm described until now was found to be too restrictive since it might lead to situations in which we would not be able to implement the IR changes by introducing modifications to the weights. Therefore, we considered modifying only those neurons that show maximal overall energy contribution and not *all* of them. To select such neurons, we examine the energy expression (2.1) that provides an estimate of the contribution resulting

from an IR bit flip:

$$\Delta E = \sum_{p=1}^{P}\sum_{i=1}^{N^H}\left[f\left(\sum_{j=1}^{N^{H-1}} W_{ij}^H\left(v_j^{H-1,p}+\Delta v_j^{H-1,p}\right)\right)-\tau_i^p\right]^2 - \left[f\left(\sum_{j=1}^{N^{H-1}} W_{ij}^H v_j^{H-1,p}\right)-\tau_i^p\right]^2 \tag{3.7}$$

Since the vectors are discrete, it can be easily shown that the energy decrease is

$$\frac{\Delta E}{2} = -\sum_{p=1}^{P}\sum_{i=1}^{N^H}\left[f\left\{\sum_{j=1}^{N^{H-1}} W_{ij}^H\left(v_j^{H-1,p}+\Delta v_j^{H-1,p}\right)\right\} - f\left\{\sum_{j=1}^{N^{H-1}} W_{ij}^H v_j^{H-1,p}\right\}\right]\tau_i^p \tag{3.8}$$

Expanding ΔE around $u^{H,p}$, one obtains

$$\Delta E \simeq \sum_{p=1}^{P}\sum_{i=1}^{N^H}\sum_{j=1}^{N^{H-1}} f'\left\{u_i^{H,p}\right\} W_{ij}^H \Delta v_j^{H-1,p}\tau_i^p \tag{3.9}$$

Since f' is a positive constant, as indicated earlier, equation (3.9) becomes

$$\Delta E \simeq -\sum_{p=1}^{P}\sum_{i=1}^{N^H}\sum_{j=1}^{N^{H-1}} w_{ij}^H \Delta v_j^{H-1,p}\tau_i^p \tag{3.10}$$

The decrease of the energy function is guaranteed by the change rule of Δv^{H-1} (3.3). The neural flips with maximal contribution will be those for which expression (3.10) is minimal; we will therefore modify only those neurons. These modifications of the weight matrix and the IR, as well as the total algorithm, show great similarity to the learning algorithm used by Grossman et al. [1,2] and are in fact a generalization of their algorithm to the multi-hidden-layer and multi-output neurons cases. The CHIR algorithm for a single output neuron [1] applies the following IR modification rule (for each neuron i):

$$\text{if} \quad W_{ij}^H v_j^{H-1,p}\tau_i^p < 0 \quad \text{then} \quad v_j^{H-1,p} \rightarrow -v_j^{H-1,p} \tag{3.11}$$

while the weight matrix modification is carried out by applying a modified perceptron learning rule:

$$\Delta W_{ij}^h = -\eta\sum_{p=1}^{P}\left(1-\sum_{k=1}^{N^H} v_k^{H,p}\tau_k^p\right) v_i^{h,p} v_j^{h-1,p} \tag{3.12}$$

where $v^{h,p}$ are the IR obtained in the former layer IR modification procedure and h represents any one of the layers. This rule regards the output error as a source for weight corrections for the inner layers as well as for the output layer.

In the CHIR generalization [2], the IR modification procedure is less restrictive and IR modifications are determined by their overall contribution to the output vector.

The two algorithms CHIR (equations 3.11–3.12) and Energy Minimization (equations 3.2–3.3) are very similar, almost identical, and provide similar results when testing various cases. However, there are two minor differences between these algorithms worth mentioning:

1. In the CHIR mechanism, the selection of the neurons that can be modified in the IR is determined stochastically. On the other hand, in the energy minimization approach those neurons are selected in a deterministic way, as shown above. This difference results in a faster convergence for the energy minimization approach, however with a lower percentage of successfully converging cases, thus leading to an overall performance similar to that of CHIR. By adding a stochastic process of some type, as for instance jogging the weights and thresholds, random modifications of the IR etc., it seems possible to increase the convergence rate.

2. An IR bit flip occurs in our energy minimization approach subsequent to the summation of the *total* contribution due to all of the next layer neurons (3.2). The CHIR mechanism [1] enables such a flip due to a single neuron contribution (3.11). In order to avoid multiple contradicting IR flips of the same neuron, Grossman [2] introduced a criterion for the IR modification, based upon its contribution to the whole output vector. The two criteria (Energy Minimization and CHIR) are similar but differ in probing the total contribution to the output vector. In our mechanism, due to the gradient descend procedure, we probe the contribution prior to the neural activation function, while the CHIR mechanism probes the output vector itself.

The generalization of the CHIR algorithm presented in this section, resulting from an energy minimization process, can be applied to multilayer and multi-output-neurons systems.

4. Computer simulations

In order to show the similarity between the performance of our procedure and the CHIR mechanism, we examined some of the the toy problems introduced by Grossman et al. [1,2] under the same conditions. The problems examined are: symmetry, parity, and parity–symmetry.

Symmetry. In this case the desired output is $+1$ if the input vector is symmetric around its center and -1 otherwise. The learning procedure

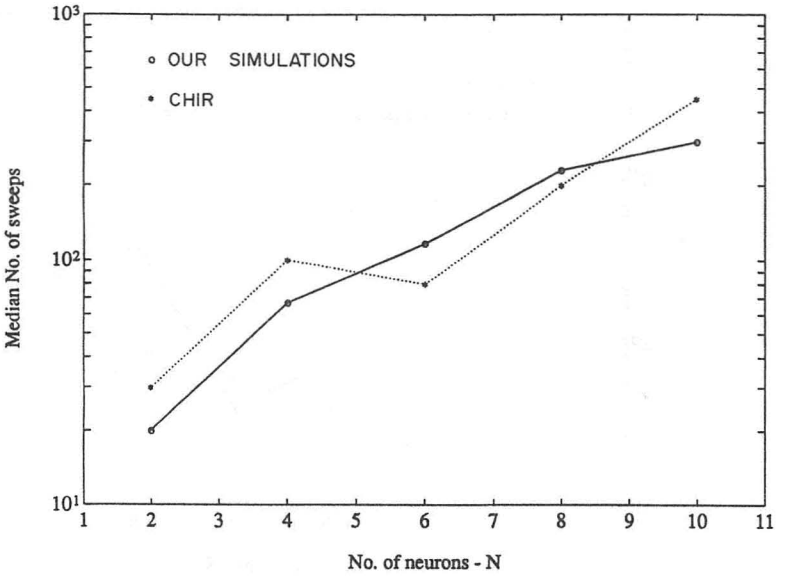

Figure 2: Symmetry. Median number of sweeps required for training a network of the form N:2:1 to solve the symmetry problem using an exhaustive training set.

was tested for $N = 2, 4, 6, 8, 10$ input neurons, 2 hidden neurons, and a single output neuron with the following parameters: $\eta = 0.1$ for the weights and $\eta = 0.05$ for the thresholds; the weight update repetition parameters (using Grossman's notation: I_{12} for the W^2 matrix and I_{23} for the W^3 matrix) are $I_{12} = 10$, $I_{23} = 5$. The maximum number of overall iterations was 200, each iteration carried over the exhaustive ensemble. Figure 2 compares the median number of pattern representations required for a fully successful learning based on this algorithm versus the results obtained by Grossman et al. [1].

Parity. The definition of the parity criterion is to provide an output 1 when the number of $+1$ bits in the input vector is even and -1 otherwise. In the simulations we used an $N = 3, 4, 5, 6, 7$ input neurons, a $2N$ neuron hidden layer, and a single output neuron. The algorithm parameters were $\eta = 1$ for both the weight updates and the thresholds; the repetition parameters were $I_{12} = 8, 9, 12, 12, 12$ and $I_{23} = 4, 3, 4, 4, 4$ corresponding to the number of input neurons (same parameters were used in [1]). The maximal number of overall iterations was 300, each iteration carried over the exhaustive ensemble. In order to compare the results with those introduced by Grossman et al. [1] and by Tesauro et

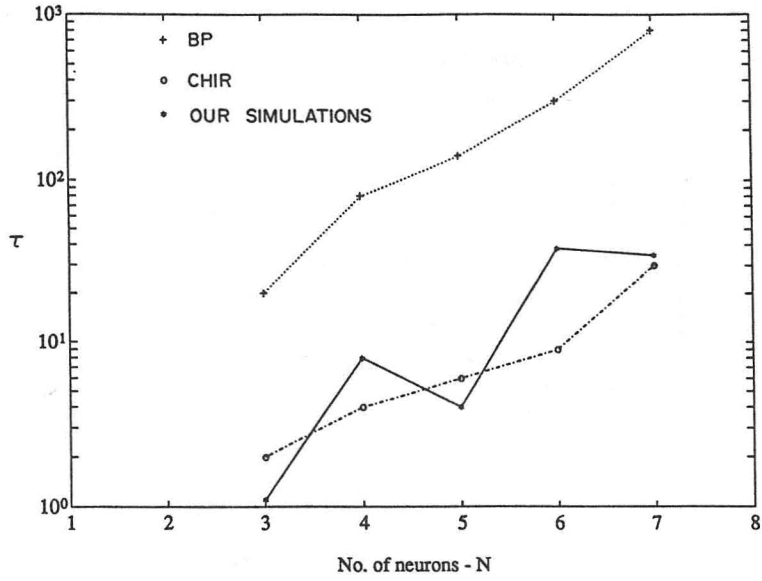

Figure 3: Parity. The inverse average of the number of sweeps required for training a network of the form N:2N:1 to solve the parity problem using an exhaustive training set.

al. [7] we introduced in figure 3 the inverse average parameter τ representing the number of iterations required for a fully successful learning procedure. The inverse average is defined as

$$\tau = \left[\frac{1}{n}\sum_{k=1}^{n}\frac{1}{t_k}\right]^{-1} \tag{4.1}$$

where t_k is the number of iterations required for a successful learning procedure with certain initial conditions (the summation includes only converging cases).

Parity–Symmetry. The parity–symmetry problem combines the two problems discussed above into one system, providing a two-neuron output: one represents the parity of the input vector and the other its symmetry. The simulations included $N = 4, 5, 6$ input neurons, $2N$ hidden neurons, and two output neurons. The parameters used were $\eta = 1$ for both weight updates and thresholds, and the repetition parameters $I_{12} = 12, 14, 18$ and $I_{23} = 8, 7, 9$ with respect to the number of input neurons (the same parameters as used in [2]). Figure 4 compares our results to those presented by Grossman [2].

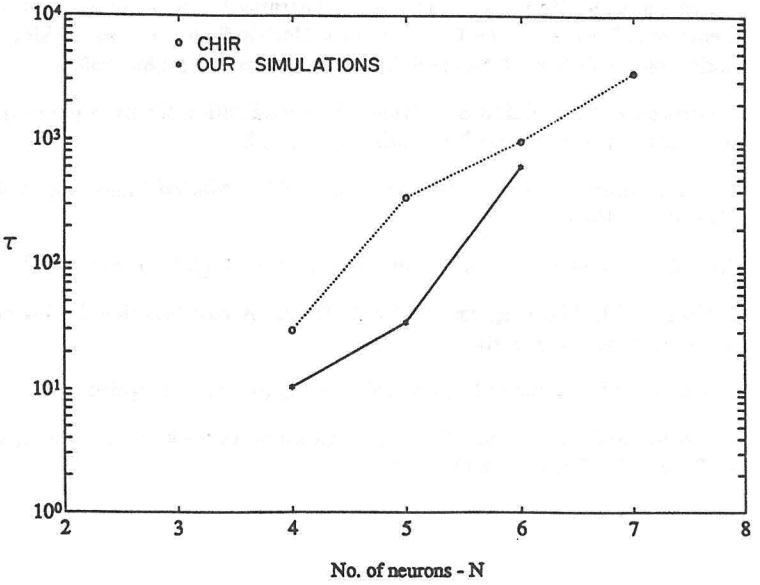

Figure 4: Parity–Symmetry. The inverse average of the number of sweeps required for training a network of the form N:2N:2 to solve the parity-symmetry problem using an exhaustive training set.

Conclusion

We have demonstrated that a continuously valued multilayer system, where *both the IR and the interconnection weights* are simultaneously modified according to energy minimization principles, behaves similarly to the BP learning procedure whereby *only the weights* are directly modified and the IR are only later indirectly changed. Applying a similar approach to a discrete system, we obtained a general version of the CHIR learning algorithm. The convergence procedure and the simulation results seem very promising, raising the possibility for a new, rapidly converging learning algorithm.

Acknowledgments

The authors would like to thank Tal Grossman for helpful discussions. The work of one author (D.S.) was supported by a stipend from the Buchmann fund which is gratefully acknowledged.

References

[1] T. Grossman, R. Meir, and E. Domany, "Learning by choice of internal representations," *Proc. of the Connectionist Models Summer School* (Morgan Kaufmann Publishers, 1988) and *Complex Systems*, **2** (1989) 555.

[2] T. Grossman, "The CHIR algorithm: A generalization for multiple output and multilayered networks," *Complex Systems*, **3**.

[3] D.E. Rumelhart and J.L. McClelland, *Parallel Distributed Processing*, Vol. 1 (MIT Press, 1986).

[4] M.L. Minsky and S.A. Papert, *Perceptrons*, 3rd ed. (MIT Press, 1988).

[5] A. Krogh, G.I. Thorbergsson, and J.A. Herz, "A cost function for internal representations," a preprint.

[6] R. Rohwer, "The moving targets training algorithm," a preprint.

[7] G. Tesauro and B. Janssen, "Scaling relationships in back propagation learning," *Complex Systems*, **2** (1988) 39.