The Hopfield Associative Memory (AM) can be understood as an extremely simple Error Backpropagation Network (EBN). This feature we use for the initialisation of the synaptic matrix. Motivated by findings of neural physiology we further consider networks of artificial neurons with multiplicative couplings. Two proposals for the self-organisation of the network architecture are given in the third section.

1. Introduction to the Error Backpropagation learning

The Error Backpropagation Network (EBN) [5] consists of neuron-like elements which are arranged in layers. Each element \( i \) is designated by its output state \( s_i \). For the first (input) layer the output is fixed by the environment. The elements of the following layers receive an activation \( E_i \) from the elements of its preceding layer. Due to this activation their own output state \( s_i \) is determined:

\[
s_i = f(E_i) \quad \text{with} \quad E_i = \sum_j w_{ij} s_j
\]

(1)

Here the index \( j \) runs over all elements of the preceding layer. Figure 1 gives the schema of an EBN.

The result of the network is given by the output states of the last (output) layer. As in the case of all supervised networks the wished (teaching) output states \( s_i^T \) for the output layer must be known at least for several examples of input patterns. Now the learning task is to change the couplings \( w_{ij} \) in such a way that an optimal fit of the networks output \( s_i \) to the \( s_i^T \) is found. In other words: the distance (ERROR) between \( s_i \) and \( s_i^T \) must be minimised.
Self-organisation of nonlinear networks

input layer
first hidden layer
last hidden layer
output layer

Figure 1. Error Backpropagation Network.

In the EBN this is achieved by a deterministic gradient search in the space of the couplings $w_{ij}$:

$$\text{ERROR} = 0.5 \sum_i (s_i - s_i^T)^2$$  \hspace{1cm} (2)

$$w_{ij} = w_{ij} + \Delta w_{ij} = w_{ij} - \varepsilon \frac{\partial \text{ERROR}}{\partial w_{ij}}$$  \hspace{1cm} (3)

The sum in (2) runs over all output elements; $\langle \rangle$ means averaging over several learning patterns. In the case of the layered EBN the calculation of $\Delta w_{ij}$ can be done using the delta rule which allows us to determine the $\Delta w_{ij}$ straightforwardly, backpropagating the error derivation from the last layer back to the input one.

The delta rule works for all transfer functions $f(x)$ but in the case of

$$f(x) = \frac{1}{1 - e^{-x}}$$ \hspace{1cm} (4)

it becomes very simple (Fig. 2). In section 2 we want to show that it is also possible to obtain the Hebb rule [3] as a special case of the delta rule. In the third section we investigate the influence of multiplicative couplings to an EBN. In the fourth section we want to discuss several possibilities for the self-structuring of networks by means of evolution ideas [1].

2. Relation to the Associative Memory

2.1. The Associative Memory as a special case of Error Backpropagation Network

Assuming the network consists of two layers (input and output) only, the learning rule becomes very simple. One obtains:

$$\Delta w_{ij} = -\varepsilon (s_i - s_i^T) f'(s_i) s_j$$ \hspace{1cm} (5)

In the case of an initialised network ($w_{ij} = 0, \forall i, j$) with the linear transfer function $f(x)=x$ this yields:

$$\Delta w_{ij} = \varepsilon s_i s_j$$ \hspace{1cm} (6)

for one pattern. For the usual task of the Associative Memory to reproduce the input at the output layer one obtains:

$$w_{ij} = \varepsilon \langle s_i s_j^T \rangle = P = \varepsilon \frac{1}{P} \sum_{p=1}^{P} s_i^T(p) s_j^T(p),$$ \hspace{1cm} (7)

for all patterns after one iteration. This is actually the commonly used Hebb rule [3] for the Associative Memory with $s_j \in \{-1, +1\}$. For the case that $w_{ij} \neq 0$, equation (6) can be generalised:

$$\Delta w_{ij} = \varepsilon \langle s_i s_j^T \rangle - \varepsilon \langle s_j^T w_{ij} s_j^T \rangle$$ \hspace{1cm} (8)

The second term corresponds to a learning pattern dependent external field. Similar systems have been investigated in [2].

2.2. Hebb-like initialisation of the EBN

Now we want to investigate the learning in an EBN. We have $s_j = 0.1$ and $f(x)=1/(1+e^{-x})$. For two layers it turns out:

$$\Delta w_{ij} = 0.25 \varepsilon (s_i^T - 0.5) s_j$$ \hspace{1cm} (9)

For an EBN with 3 or more layers a new problem arises. It depends on the fact that the values of the hidden nodes are not known at the beginning. So it is necessary to make some
assumptions about it. If the number of elements of the hidden layer is equal to those of the output layer, i.e. each element of the hidden layer is expected to have the same output value as one element of the output layer.

To demonstrate this possibility we have made simulations with a 3-layered EBN of the type shown in Figure 3. Here it is necessary that the number of output elements is less than, or equal to the number of hidden elements. The $w_{jk}$ were determined according to eq. (10):

$$ w_{jk} = - \Delta w_{jk} = e \frac{1}{p} \sum_{p=1}^{P} \left( s_{j}(p) - 0.5 \right) s_{k}(p), $$

(10)

Because of the fact that in the EBN learning task the input is different from the required output we have marked it by $s_k$. The $w_{ij}$ have been initialised randomly:

$$ w_{ij} \in \{-w_{jk}, +w_{jk}\} \quad \forall \ j \neq i $$

$$ w_{ii} = \text{const.}, \text{const.} \times \text{abs}(w_{jk}) $$

(11)

This method was applied to a 4-4-4 encoder problem and to the comparison of two binary numbers of length $8$ each (Figs. 4 and 5). For the random initialisation the $w_{ij}$ were out of the interval $(-w_{ij}, +w_{ij})$, where $w_{ij}$ is the averaged value of the Hebb initialised links.

It can be expected that the method described above will not work for all learning tasks because of the fact that the hidden layers will have other codings for the patterns. Here it is possible to assume that each type of output pattern corresponds to a certain internal structure within the hidden layer(s). Because this internal structure is unknown one can try to generate a random internal structure for each of the possible output patterns.

3. Multiplicative couplings

Several results of neural physiology suggest the introduction of nonlinear couplings in the EBN. As a special simple model of nonlinear neuronal coupling we use the activation function

$$ E_i = \prod_{j=1}^{N} w_{ij} \theta_j, $$

(12)
Self-organisation of nonlinear networks

Figure 5. ERROR and reliability for 2x8 digits comparison for a) random initialisation and b) Hebb initialisation.

This assumption replaces the standard additive couplings (see eq. (1)). From biology it is known that special neurons with multiplicative interactions exist. We want to demonstrate here that a network of multiplicative neurons shows also learning behaviour. With the assumption $s_0 = 0.5$ and $\Theta_j = -1d w_{10}$ we can write

$$E_i = \prod_{j=0}^{N} s_{ij}. \quad (13)$$

Because of $E_i \geq 0$, $f(w_{ij})$ the activation function $s_i = f(E_i)$ has to be varied (Fig. 6):

$$s_i = \frac{1}{\pi} \arctan E_i \quad (14)$$

Figure 6. The activation function according to eq. (14).

Figure 7. ERROR for the 4-2-4 encoder with multiplicative couplings.

With the ERROR value

$$\text{ERROR} = \frac{1}{2} \sum_i (s_i - s_i^T)^2, \quad (15)$$

and the learning rule

$$w_{ij} = w_{ij} + \Delta w_{ij} = w_{ij} - \varepsilon \frac{\Delta \text{ERROR/}}{\Delta w_{ij}}, \quad (16)$$

as in the case of the standard EBN for a 3 layer network it turns out:

$$\Delta w_{ij} = -\varepsilon \left( s_i - s_j^T \right) f'(E_i) E_j \ln s_j, \quad (17)$$

for $i \in$ output layer, $j \in$ hidden layer

$$\Delta w_{jk} = -\varepsilon \left( s_k - s_j^T \right) f'(E_j) w_{kj} \frac{1}{s_j} f'(E_j) E_j \ln s_k, \quad (17)$$

for $k \in$ input layer

For the example of the 4-2-4 Encoder Fig. 7 shows the ERROR value as a function of the learning cycles.

We should remark here that we are convinced that both the
use the example of a simple pattern recognition task. For our example we used a network of 150 input, 20 inner and 26 output elements. The input elements were arranged in a $10 \times 15$ matrix. The machine was taught with 12 alphabets from different authors. One of these alphabets is shown in Fig. 8.

In Fig. 9 the error and the reliability of correct assignment of the net is drawn. After learning the network recognised more than 99% of the shown letters correctly. Not shown (unknown) letters were recognised with a reliability of about 70%.

In Figure 10 the number of link values falling into the interval $(x, x+\Delta x)$ dependent on $x$ is shown. Most of the link values are very small. Setting these values (about 70%) to 0, i.e. disconnecting the corresponding neurons did not worsen the

4. Self-organisation of the network architecture

A major problem of EHN is the structure of layers, nodes and couplings. The delta rule does not influence this structure; that means it must already be given at the beginning of the learning procedure. Up to now we have only a limited knowledge about the demands to the network structure in dependence of the problem. So it would be very nice to have an algorithm which changes the structure of the network during learning. In the following we discuss two possible examples for algorithms acting on the structure of the EHN.

4.1. Link method

Let us first consider a more link-oriented method for the self-organisation of a network architecture. To explain the method we
Self-organisation of nonlinear networks

performance of the network, e.g. the reliability for the correct assignment of patterns. So the motivation for an evolution rule is given:
(i) Connect every hidden element with a certain number of input and output elements. For our example we used 20 hidden and 10 output ones.
(ii) Perform a number of learning cycles.
(iii) Disconnect the neurons connected by a small link value.
(iv) Connect the hidden elements with other (randomly chosen) input and output elements using the free links originating from the last step.
(v) Continue with (ii).
Performing the evolution algorithm we need much more learning cycles but the calculation time for one cycle is much shorter, because the sum to evaluate the $E_j$ runs over 20 (10) elements instead of 150 (26). So the net calculation time remains approximately constant. The advantage consists in the lower memory demand (about 20% of the standard EBN algorithm) to store the $w_{ij}$ values.

4.2. Node method

The idea of the node method consists in doubling an "overcharged" element $j$ into two elements $j_1$ and $j_2$. Because of the deterministic character of the learning rule a symmetry break between the elements $j_1$ and $j_2$ must be introduced. This can be done in at least two ways. Either there is introduced an additional weighted link directed from element $j_1$ to $j_2$ or the $w_{ij}$ are divided nonsymmetrically.

For the first method see Fig. 11:

Figure 11. Doubling the $j$-th element.

As a result of this method a network with a completely new structure (i.e. its adjacency matrix changes) is achieved (Fig. 12). This can be considered to be a disadvantage because of the fact that new layers will arise. But it can be shown that the delta rule works also in this case. Problems will occur if one tries to run the algorithm on a parallel computer. For basic research on sequential computers this method might be used without any decrease in performance. As a result one can obtain interesting network structures which perhaps show new features.

The other method is more in line with usual EBN and we should discuss it here in more detail. It is clear that only the hidden elements can be doubled because of the fact that input and output are fixed by the environment. As a result of doubling the performance of the EBN should be changed only a little. Therefore both new elements $j_1$ and $j_2$ should act in the same way as the old one; that means they must obtain the same activation $E_j$ or in other words their connections to the preceding layer must be equal ($w_{jk} = w_{j_2k} = w_{jk}$). Otherwise it must be guaranteed that also the elements of the succeeding layer obtain the correct activation. This can be achieved by splitting the weight $w_{ij}$ into two values $w_{ij_1} = w_{ij_2}$ with $w_{ij_1} = w_{ij_1} + w_{ij_2}$. This splitting might be symmetrical or not. A symmetry break can also be achieved by adding small random numbers to the weights of the newly created links.

Up to now we have not yet discussed which elements are "overcharged". Perhaps it cannot be decided whether an element is
Self-organisation of nonlinear networks

overcharged or not in the unchanged network. But it is possible to
double any element and to look for the development of its two
"children" after some iterations. If they show a very similar
behaviour the doubling was not necessary and they can be merged
again; otherwise the doubling was useful. The charge $C_{ij}$ of two
elements $i$ and $j$ can be doubled by the following equation:

$$C_{ij} = \langle (s_{i} - s_{j})^2 \rangle_p$$  \hspace{1cm} (18)

$\langle \rangle$ means averaging over all patterns. If the correlation $C_{11, 12}$
is greater than a critical value $C_{\text{crit}}$, the two nodes will remain
within the network, otherwise not. In this way the number of nodes
will increase more and more. But the total number of elements
should not exceed some predefined number so it is necessary also
to remove nodes. Therefore one must decide whether a node $j'$ is
yet necessary or not. Again one can determine the charge $C_{jj'}$. If
it is sufficiently small the two nodes $j'$ and $j$ can be merged;
i.e. the element $j'$ has to be removed and its outgoing couplings
$w_{ij}$ are added to $w_{ij}$ ($w_{ij} = w_{ij} + w_{ij'}$). Other elements $j'$ which
are not necessary have only very small outgoing couplings $\sum_{i=1}^{n} w_{ij}$
less 1. These elements can be removed without any changes in the
remaining network.

5. Conclusion

The aim of our work was to show some new ways to increase the
performance of Error Backpropagation networks. It turns out that
the Hebb rule of Associative Memory can be regarded as a special
type of Backpropagation learning. Using the corresponding Hebb
rule for the initialisation of the EBN one can speed up the
learning rate of such a network.

The use of multiplicative couplings may bring new features to an
EBN.

In the last part of our work we have discussed two methods for
the self-structuring of EBN. Both methods use ideas from evolution
theory such as mutation and selection. In one case the node
structure remains constant but the network is not totally linked
and the link-structure makes an evolution. In the second case also
the node-structure is effected by evolution. So it is possible that
the network can find an optimal structure for each problem by
itself.

We hope that the ideas described here will help a little to
improve the performance of neural networks.

Acknowledgements

The authors are grateful to R. Feistel and A.A. Vedenov for
fruitful discussions.

References

Complex systems, Deutscher Verlag der Wissenschaften,
Berlin.
Proceedings of the Conference Dynamics of Networks, Eisenach
Distributed Processing: Explorations in the Microstructure of
This fully indexed proceedings contains the invited lectures and some selected papers, and is published in two volumes:


Volume II: connectionism and neurocomputers contains papers that deal with the behaviour of more abstract model neurones and aspects of the design of neurocomputer architectures.

Neurocomputing as developed in the Soviet Union has strong connections with both neuronal electrophysiology and mathematical statistical physics. The papers in these volumes provide an overview of Soviet approaches to the theory of neural networks. Together with the papers from non-Soviet contributors, the volumes present a viewpoint that emphasises coherent behaviour in networks of dynamical systems as the basis for the focus of attention.

The volumes will be of interest to all researchers in neural networks, both those working on real nervous systems (cognitive and neuroscientists) and physicists, mathematicians and engineers working on the design and fabrication of neural nets.

Arun V. Holden is Reader in General Physiology at the University of Leeds; Vitaly I. Kryukov is Head of the Statistical Methods Laboratory at the Research Computing Centre of the USSR Academy of Sciences, Pushchino.