R. Valk (Hrsg.)

GI – 18. Jahrestagung II

Vernetzte und komplexe Informatik-Systeme

Hamburg, 17.-19. Oktober 1988

Proceedings

Springer-Verlag
Berlin Heidelberg New York
London Paris Tokyo

in: R.Valk (Hrsg.), Vernetzte und komplexe Informatik-Systeme
Informatik-Fachberichte 188, Springer Verlag 1988

Fault-Tolerance in Non-linear Neural Networks

Dr. R. Brause, J.-W. Goethe University
FB 20, VSFT,Postfach 111932
D- 6000 Frankfurt 11, West-Germany

ABSTRACT:

Among the models of parallel computing architectures of neural networks, the model of distributed associative memory has very promising features including fault tolerance.
Fault tolerance is here not merely an artificial addition to the existing architecture but intrinsically tied to the basic functions. By addition of a threshold to the linear connection matrix the resulting model is fault-tolerant for errors in input data by providing in one function-cycle a complete pattern recognition process.
The properties of this inherent fault-tolerant process are analytically analyzed and the optimal threshold is calculated.
Furthermore, a hardware model is presented and its fault-tolerance properties are evaluated.

1. Introduction

> *... I don't think we ever debugged our machine completely, but that didn't matter. By having this crazy random design it was almost sure to work no matter how you built it.*
>
> *Marvin Minsky about his learning machine*

In modern parallel computer architectures a new generation of highly parallel, real-time oriented architecture for artificial intelligence is at the horizon. These attempts favorize computers made by many, small processing elements of very low complexety and therefore very limited computing power, connected directly together contrary to a relative small number of complex processors, communicating with a high amount of overhead. An example of these attempts is the connection machine [HILL].
One important class of highly functional parallel models are those proposed since 30 years by neurological and cybernetical scientists for modelling brain functions. The models are based on the function of simple elements, the neurons, connected extensively in a specific manner (*Neural Networks*). Every connection is assigned a specific weight.
These weights may represent special events or relations, therefore implementing directly semantic nets in hardware [FELD]. The connection models with dedicated connections have only a small degree of fault-tolerance because the failure of a node erases the whole associated event.
However, if the event is assigned to a specific state of the whole set of connections the inherent fault-tolerance properties are much more promising. Even in the early papers the surprising fault-tolerance, error-correcting and pattern completion properties are mentioned [WOOD],[KOH2], but never evaluated. Because the recall of the stored patterns are quite good, even when portions of the storage matrix (weights) are erased, the patterns seem to be stored in a distributed manner like a holographic picture. The model was therefore termed *hololgic memory* in the beginning [WILL2], [LONG]. Since the pattern completion effect can be used to construct a very fast and simple inference engine [HIN2] neural network models like holographic memories are very promising candidates for the computer architecture for high-level AI-functions.

For example, in many models of artificial intelligence the problem is divided into subproblems in a layered manner [BRA]. In figure 1 the layers of computer vision and speech recognition systems are shown.
On each level the layer has to provide some basic fault-tolerant abilities like recognition of varied, noise-disturbed and incomplete patterns.

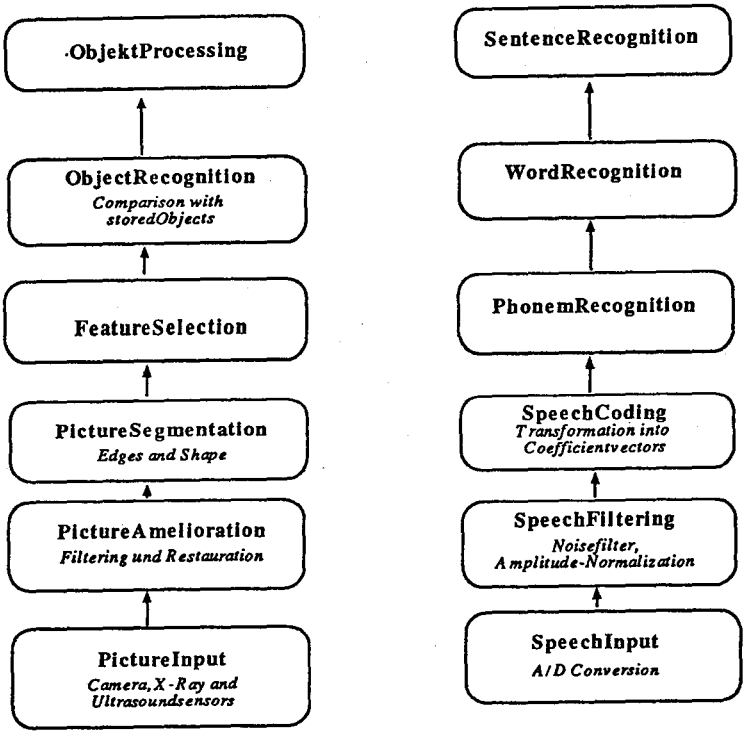


Fig 1 processing layers in computer vision and speech recognition

In the neural network approach each layer has the nearly the same homogen structure of interconnected, very simple processing elements, e.g. [FUK].
In part 2 of this paper the network for one layer is introduced as it was formulated by Kohonen [KOH1] and McCulloch and Pitts [McCUL] and some necessary conditions for pattern recognition and memory recall in the presence of disturbed input data will be given.

In part 3 it is shown that the recall process of stored data can be viewed as a pattern recognition and error correction process which is controlled by a threshold. The optimal thresholds for two pattern similarity measures are evaluated and the optimal coding of input data is discussed.

A simple hardware model and its corresponding fault model is proposed in part 4 and the maximal number of connections whose failure or insufficient fabrication do not impede the proper recall process is derived.

Part 5 draws the conclusions of this paper.

2.0 The functional model

Let us first consider the formalization of the network concept.
Assume that we have m processing elements which have a processing function f(.), and n input lines which can be connected to the processing elements. The set of links (weights) between the input lines and the processing elements can than be described by a matrix $\mathbf{W} = (w_{ij})$. The output z_i of the interconnection network to the processing element i is a linear combination of the values of the input lines $x = (x_1,...,x_n)^T$where T denotes the transpose. In a vector product $vw^T$the T will be omitted. The output y_i of the processing element i is therefore

$$y_i = f(z_i) = f(\sum_j w_{ij}x_j) \quad z = (z_1,...,z_m)^T \tag{2.0a}$$

or $$y = f(z) = f(W\,x)$$

In Figure 2 the hardware model of the basic input-output configuration is shown.

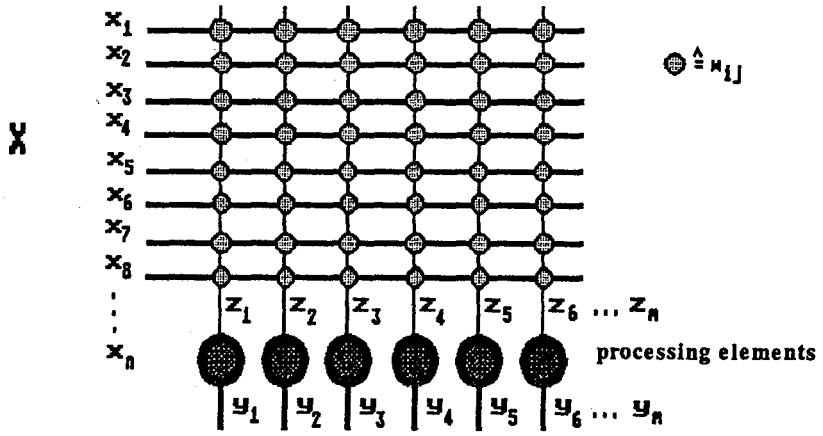


Fig. 2 Hardware model of the neural network

The network is used as an associative memory in two different modes: the imprinting (storage) mode and the memory recall (readout) mode.

In the *storage mode* each time one of the sequentially presented input patterns $x^1..x^p$ (the class prototypes) with their associated output patterns $y^1..y^p$ appear, the weights are locally augmented by the correlation of input x^k and output data y^k (Hebb's rule) with a proportional constant c_i.

$$\Delta w_{ij} = c_i^k\, y_i^k\, x_j^k \tag{2.0b}$$

After the presentation of p patterns the imprinting storage is complete:

$$w_{ij} = c_i^0 + \sum_{k=1}^{p} c_i^k y_i^k x_j^k$$

In the *memory recall mode* the recall is done by presenting an event to the input lines x and reading out the output at y

$$y_i = f(W\,x)_i = f\Big(\sum_j w_{ij}x_j\Big) = f\Big(c_i^0 + \sum_j \sum_k c_i^k\, y_i^k x_j^k x_j\Big) = f\Big(c_i^0 + \sum_k c_i^k y_i^k x^k x\Big) \qquad (2.0c)$$

Linear Projection Model

Let us consider the processing elements as simple analog amplifiers: $f(z_i) =: z_i$.
If we present a specific event x^r which was stored before, the recalled output is

$$y_i = z_i = y_i^r x^r x^r c_i^r + \sum_{k,\,k\#r} c_i^k y_i^k x^k x^r + c_i^0 \qquad (2.0d)$$

response + *cross-talk*

It can be interpretated as the proper response y_i^r and additionally evoked crosstalk from other stored patterns.
For stored events which are orthogonally coded the product $x^k x^r$ is 0. If we additionally normalize (2.0b) with the factor $c_i^k := (x^k x^k)^{-1}$ and $c_i^0 = 0$ the equation (2.0d) becomes

$$z_i = y_i^r x^r x^r c_i^r = y_i^r$$

and the proper response y_i^r is derived without any threshold involved. Since the memory model stores simply cross-correlations it was termed *correlation memory* [KOH1].
This is the model for which Kohonen demonstrated good pattern completion abilities [KOH2].
It should be noted that the linear model needs the orthogonalization of the input patterns to store them correctly. If a non-orthogonal (faulty) pattern is input, due to the linearity of (2.0d) the output will be faulty, too.

Orthogonal Projections

Since activity patterns of sensors of the real world do not provide orthogonal coded patterns generally, let us make two less restricting, but efficient assumptions:

1) Only the output is orthogonally coded: $y^k y^r = 0$ for k#r
2) The activity (spike rate!) is only positive: $x_i, y_i \geq 0$

From these two assumptions we can conclude that for every component i there exists at most one y^{k_i} which has $y_i^{k_i} \neq 0$.
So equation (2.0c) reduces to

$$y_i = f(c_i^0 + c_i^{k_i} y_i^{k_i} x^{k_i} x) \qquad (2.0e)$$

In the case of $x = x^r$, $y_i^r = 0$, $k_i \# r$, the non-zero correlation $z_i = c_i^0 + c_i^{k_i} y_i^{k_i} x^{k_i} x$ is the crosstalk in the memory recall. Certainly, if we normalize the weights and use only orthogonal input vectors (see above) we will get a proper recall.
Instead of using orthogonal input vectors, which is a strong restriction, and normalizing the weights let us try to get the proper response by introducing a *threshold function*

$$T(x) = \begin{cases} 0 & x \le 0 \\ 1 & x > 0 \end{cases} \qquad (2.0f)$$

which should suppress the cross-talk in (2.0e).
Before we explicitely choose the function f(.) and the constants c_i let us take a closer look to the meaning of "suppressing crosstalk".

2.1 Similarity Measures and Fault-tolerant Memory Recall

Let us look at some arbitrary input data pattern vector **x** which can be interpreted as a disturbed, faulty version of a class prototype $\mathbf{x}^r$. Our understanding of this fact is, that among all class prototypes $\mathbf{x}^k$ the input **x** mostly resembles to $\mathbf{x}^r$. The error correction of **x** and the recall of the pattern $\mathbf{y}^r$ becomes now an ordinary pattern recognition problem: we have to assign an unknown pattern **x** to the appropriate class which is represented by the class prototype $\mathbf{x}^r$. The classification rule "*take the most similar class prototype*" can be mathematically interpreted

1) by the demand for the *maximal cross-correlation*

$$\mathbf{x}\mathbf{x}^r = \max_k \mathbf{x}\mathbf{x}^k \qquad (2.1a)$$

or

2) by the demand for the *minimal distance*

$$|\mathbf{x}-\mathbf{x}^r| = \min_k |\mathbf{x}-\mathbf{x}^k| \qquad (2.1b)$$

Certainly, the two measures are coupled : $|\mathbf{x}-\mathbf{x}^r|^2 = |\mathbf{x}|^2 - 2\mathbf{x}\mathbf{x}^r + |\mathbf{x}^r|^2$
Let us now try to understand the different meanings of the two measures for our problem by a geometric interpretation.

Geometrical Interpretation
The demand for maximal cross-correlation means that we choose as classprototye for **x** the pattern $\mathbf{x}^r$ which satisfy

$$\mathbf{x}\mathbf{x}^r > \mathbf{x}\mathbf{x}^k \qquad \text{for every class } k \# r \qquad (2.1c)$$

The boundary between two classes $\mathbf{x}^r$ and $\mathbf{x}^k$ is given with $\{\mathbf{x}^* | \mathbf{x}^*\mathbf{x}^r = \mathbf{x}^*\mathbf{x}^k\}$.
With the distance $\mathbf{d}^{rk} := \mathbf{x}^r - \mathbf{x}^k$ the equation $\mathbf{x}^*(\mathbf{x}^r - \mathbf{x}^k) = 0$ of the boundary becomes $\mathbf{x}^*\mathbf{d}^{rk} = 0$ and the boundary $\{\mathbf{x}^* | \mathbf{x}^*\mathbf{d}^{rk} = 0\}$ consists of the hyperplane which is orthogonal to the distance vector $\mathbf{d}^{rk}$ between the two class prototypes.
In figure 3 the situation is illustrated.
This boundary means for all **x** on the right hand side of the plane, that the angle α between **x** and $\mathbf{d}^{rk}$ is $-90^0 < \alpha < +90^0$ and so $\cos(\alpha) > 0$.
Thus

$$0 < \cos(\alpha)\, |\mathbf{x}||\mathbf{d}^{rk}| = \mathbf{x}\,\mathbf{d}^{rk} = \mathbf{x}\,(\mathbf{x}^r - \mathbf{x}^k)$$

or $\mathbf{x}\mathbf{x}^r > \mathbf{x}\mathbf{x}^k$ which is the condition (2.1c) for maximal cross-correlation.

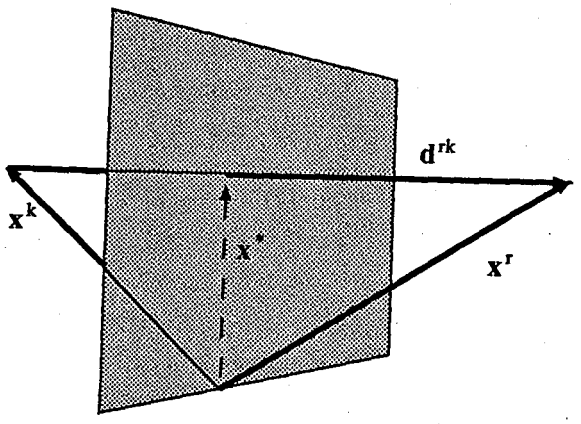


Fig.3 The class boundary between two classes for n=3

As we can see, the classification works quite good in our illustration. What are the problems of this classification scheme?
For a correct classification of $x=x^r$ into the class r the relation $x^r x^r > x^r x^k$ must hold. Therefore, our whole pattern space is divided again in two sets by a hyperplane with

$$\{x^* | x^r(x^r - x^*) = 0\}$$

i.e. the difference vector $d^{rk}:=x^r-x^k$ is orthogonal to x^r. To allow a correct classification of the prototype vector itself all other prototype vectors should not be in the area bounded by the hyperplane orthogonal to x^r. In figure 4 the "forbidden areas" are shown gray-shaded for three class prototypes in the 2-dim case.

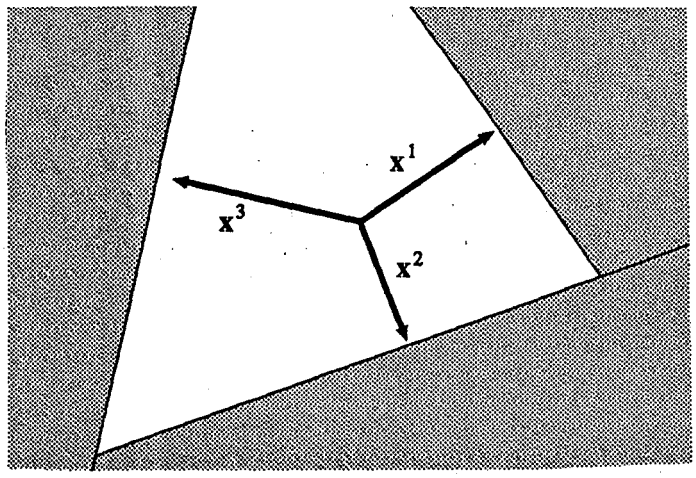


Fig.4 restrictions for correct recognition

Let us now look at the other similarity measure, the distance to the class prototypes. The classification schema (2.1b) is eqivalent with a tesselation of the pattern space; the boundary between two classes is a hyperplane which intersects orthogonally the difference vector d^{rk} at $d^{rk}/2$, see figure 5. The proof is in appendix A.
It should be noted that the set of class prototype vectors in figure 5 cannot correctly be

recognized by the classification rule (2.1a).

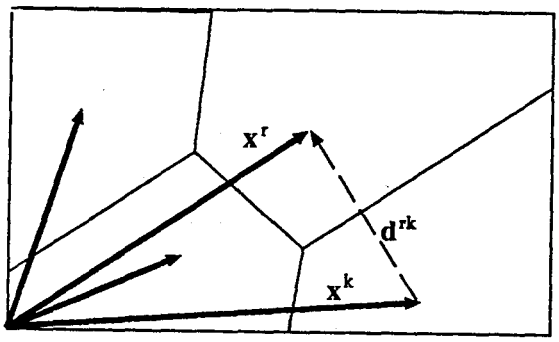


Fig.5 tesselation of the pattern space

It is interesting to note that the set of class protypes can be seen as a state in Kohonens *Topologie conserving mapping* algorithm (see [KOH3]). This can be used for instance to implement the optimal mapping by choosing the appropriate class prototypes (e.g. a set of equally-spaced vectors). As we can see in part 3.2, the mapping can only approximately implemented by the associative memory device.

2.2 Fault Tolerance and Interprocessor Communication

In section 2.1 we derived as class boundarys hyperplanes in the pattern space. Since we intend to realize the fault-tolerant memory recall (i.e.pattern recognition operation) by a threshold function T(.) the class boundary for the classification rules (2.1a) and (2.1b) may serve us as threshold.
Let us now determine the function f(.) to suppress the cross-talk by the use of a threshold.
Since we want to use the results in binary, noise-suppressing computers we restrict our view to binary vectors, i.e. x_i, y_i,are out of {0,1} and w_{ij} is of $\mathbf{Z}^+$, the natural numbers including 0. This model was first given by McCulloch and Pitts [McCUL].
Our processing elements are now pure, simple threshold elements. Each one is controlled by its threshold t_i such that the output

$$y_i = f(z_i,t_i) =: T(z_i - t_i) \qquad (2.2a)$$

results.
A stored response $\mathbf{y}^r$ is only then properly recalled by the associated prototype $\mathbf{x}^r$ if for every component of $\mathbf{y}^r$ which has

$$y_i^r = 0 \quad \text{it holds} \quad z_i \le t_i^r \qquad (2.2b)$$
and
$$y_i^r = 1 \quad \text{it holds} \quad z_i > t_i^r$$

With the "natural" choice $c_i^0 := 0$ and $c_i^k := 1$ in (2.0e) and (2.2a) the relations (2.2b) become the *maximal correlation classification rule*

$$xx^r \le t_i^r \Rightarrow y_i^r = 0$$
and
$$xx^r > t_i^r \Rightarrow y_i^r = 1 \qquad (2.2c)$$

It should be noted that for orthogonal input class prototypes formula (2.2c) reduces to

$$0 \le t_i^r < x^r x^r \qquad (2.2d)$$

which allows to set the threshold to zero $t_i^r = t = 0$. In this case the linear and non-linear models are equivalent for the recall of binary prototype vectors.
With the threshold of (2.2c) every processing element can compute whether the correlation is strong enough ($t_i^r(k) < z_i$) to belong to class r or is crosstalk ($t_i^r(k) \ge z_i$) due to class k.
Two problems arise. *First,* it is only valid for two classes r and k. If we have more classes, we will have more possible thresholds; to distinguish between more similar protypes the border demands higher correlations. For the necessary threshold decision we have to know all the other correlations obtained at the other processing elements. And this is the *second* problem: the proposed parallel network model does not contain communication between the processing elements.
As solution to this problem we have to choose a threshold t_i^r, which do not depend on the other classes k, thus preventing the communication.
On the one hand we have to consider the worst case and choose the highest of all thresholds to guarantee the correct class-membership of the recognized patterns, our pattern recognition process will assign on the other hand a certain number of patterns of class r to the null vector class and recognize only the most similar ones.
As we can see, in this model without communication we can not make the best possible classification but only a sufficient one.

3.0 Fault-tolerant Memory Recall

In the previous section 2 we have seen that the introduction of a threshold function for the linear matrix model results in a pattern classification operation. The correct classification is based on a similarity measure between the input and the class prototype. In the case of orthogonal projection the similarity measure determines a specific threshold for every processing unit.
In this section we will compute first the necessary and sufficient threshold between two classes and then generalize the result.
Then we describe a pattern completion operation, used for relatonal database requests, as a special case of the bulid-in fault tolerance.
At last the input pattern coding for optimal fault-tolerant operations is considered.

3.1 Optimal Thresholds

Let us determine the threshold t_i^r which is sufficient for a proper recall of all prototypes.
So (2.2b) becomes for the recognized prototype $\mathbf{x} = \mathbf{x}^r$ (classification restriction, cf. fig.4)

$$\max_{k,\, k\#r} x^k x^r \le t_i^r < x^r x^r \qquad (3.1a)$$

This relation guarantees us a suppression of the crosstalk and a proper memory recall for a prototype x^r by a processing unit i if the length of x^r (which is the number of ones) is greater than the greatest overlap of x^r with another class prototype. This can be interpreted as a kind of majority voting system for fault suppression.

Maximal Crosscorrelation

The sufficient threshold condition for x is by (3.1a)

$$\max_{k,\, k\#r} \; xx^k \le t_i^r < xx^r \qquad (3.1b)$$

With some geometrical considerations (see appendix B) we get the threshold between class r and k.

With $K^r_{max} := \max_k x^k x^r$ and normal activity $|x^r|^2 = |x^k|^2 =: a$

and relation (3.0a) we get from appendix B

$$t_i^r = |x| \left(1/2(a+K^r_{max}) \right)^{1/2} \qquad (3.1c)$$

Minimal Distance

The cross-correlation gives for two binary vectors v and w the number of common components having the value '1'. Let us now consider another measure of similarity : the **Hamming distance** $d_H(v,w)$, defined as the number of components which are different between the two vectors v and w. What relations hold between the two measures?

The number of non-zero components in the distance vector (v-w) is just the number of components which are different between the two binary vectors. Since the number of '1' in a binary vector x is $|x|^2$, the quadratic Euklidean distance for binary vectors is the Hamming distance:

$$d_H(v,w) = d_E(v,w)^2 = (v-w)^2 = v^2 - 2vw + w^2 \qquad (3.1d)$$

Further calculations (see appendix C) gives us the sufficient threshold with the minimal Hamming distance $d_H^r := \min_k d_H(x^r,x^k)$

$$t_i^r := 1/2 \left(|x^r|^2 + |x|^2 - d_H^r/2 \right) \qquad (3.1e)$$

As we can see a good threshold is determined by the shortest distance of x^r to its class boarder. This results in the classification strategy of assigning only those x to class r which are in a secure neighbourhood of x^r. The boarder of these neighbourhoods correspond to the cicles in figure 6.

Certainly, there are many patterns which are in no circle and are therefore projected to the null vector. We can compensate this effect by a sufficient enlargement of the circles, eleminating the space between the circles and the boundaries. The resulting mapping of the input patterns is no more an exact orthogonal projection but only approximately. The recalled output patterns of patterns belonging to class r will be very close (very short distance) to the

output pattern associated to the class prototype, but not necessary the same.

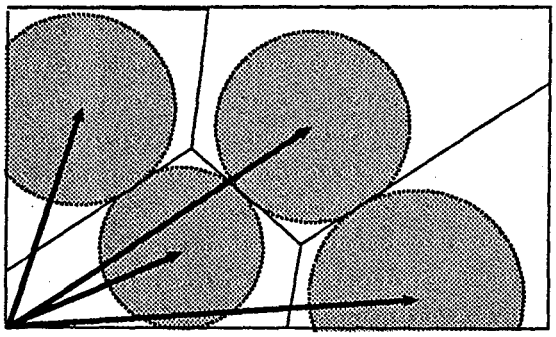


Abb. 6 Classification regions

Both thresholds (3.1c) and (3.1e) use the pattern strength $|x|^2$ to set up the threshold. This can be accomplished by a simple hardware addition to our hardware model of figure 2. This is shown in figure 7 .

The whole device can be implemented as a VLSI-chip of very regular structures. The summation for every output component y_i can be easily done by the superposition of the currents caused by different input lines. The connections are then realized as diode/resistor combinations. The modifiable resistor can be a physical device like a EEPROM connection [GOS] or a binary counter. If the resulting current in the column is greater than the threhold value t_i, which in turn is set by some internal constants and the external (see square elements in figure 6) generated $x=|x|^2$; the threshold element sets the output y_i from 0 to 1. This is done immediately, so the whole search and pattern recognition process takes only one cycle which is with the technology of today in the range of nanoseconds.

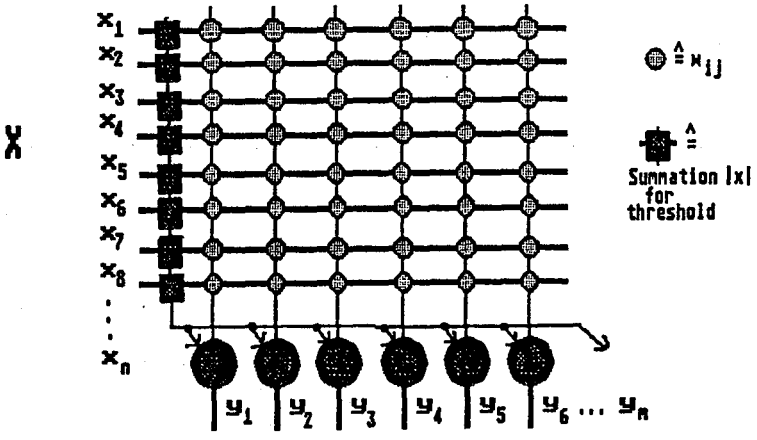


Fig. 7 Modified hardware model

It should be noted that in the binary case we can get another threshold without the need for changing our hardware model of figure 2.

The threshold relation of the class-boarder $d_H(x,x^r) < d_H^r/2$ (see appendix C) can be expressed as

$$|x^r|^2 - d_H^r/2 < 2xx^r - |x|^2$$

With $c^0 := -1$ and $c_i^k := 2$ we get from equation (2.0e) $z_i = 2xx^r - \Sigma_j x_j$ which is in the binary case $z_i = 2xx^r - |x|^2$. According to the threshold conditions (2.2b) the threshold can therefore be chosen as

$$t_i^r := |x^r|^2 - d_H^r/2 \qquad (3.1f)$$

In the binary case at normal activity $a := |x^r|^2$ the optimal threshold is determined by the minimal Hamming distance between the stored input pattern class prototypes. If all class prototypes have the same Hamming distance d then the decision boarder between two classes is given by d/2.
It is interesting that the above classification rule (3.1f) coincidences well with the result of coding theory, which states that error-correction in block codes can only be obtained if the disturbed codeword has a Hamming distance lower than half of the minimal Hamming distance between two codewords.

3.2 Fault Tolerance, Pattern Completion and Relational Database

Let us now consider a special case of fault tolerance in the memory recall: the pattern completion operation.
Pattern completion is obtained when one of the class prototype pattern vectors is only partially filled. The sparse vector is treated like any other faulty input data: by the threshold mechanism it is mapped into an appropriate class. If the input and output coding are the same, the classification and fault-correction of the incomplete input data results in the output of the completed input pattern.
This can be seen as a very fast request to a relational data bank. For example, a relational tuple *(relation, object1, object2)* can be coded by the concatenation of the codes for relation, *object1* and object2, s.[HIN]. The resulting long vector can be stored in the memory, associated with itself so that $y^k = x^k$. If we want have only the incomplete tuple, for instance *(relation, object1, -)* and we are searching for the rest, all we have to do is to present the incomplete, x^k which have some '1' lacking as an input pattern to the memory device. If the tuple was properly coded and the Hamming distance to the other stored tuples is big enough, then the complete relation will be output. Thus the basic functional proportions and the fault-tolerance abilities are tied intrinsically together.

3.3 Optimal coding of input data

In most of the papers dealing with associative memory, the coding of the input and output vectors are not treated, in spite of the fact that the memory recall is very sensible to the overlap, i.e. to the Hamming distance of the stored patterns. For the optimal fault-tolerant, error-correcting memory recall in a real implementation of an associative memory it is important to obtain some guide lines for the optimal coding of the input events which yields maximal error-correction. Since the optimal coding is dependant on the requirements of the input data, two different attempts are presented.

a) Suppose, we want the maximal possible fault-tolerance. This is obtained by the maximal possible Hamming distance d.

$$\max_{r,k} d(x^r,x^k) = \max_{r,k} |x^r|^2 + |x^k|^2 - 2x^r x^k = 2a \qquad |x^r|^2 = |x^k|^2 =: a$$

This is obtained for $x^r x^k = 0$, i.e. all class prototypes are orthogonal. The number N(a) of possible prototypes is then quite small:

$$N(a) = \lfloor n/a \rfloor$$

Example : With n=10 and a=3 we have d=6 and only N(a) = 3 class prototypes.

b) Suppose, we want as many events coded randomly with $|x^k|^2=a$ as possible and want to have the maximal expected Hamming distance *d* between the events.
What is the optimal *a* ?

$$E(d(x^r,x^k)) = 2a - 2E(x^r x^k) = 2a - \sum_{i=1}^{n} E(x_i^r)E(x_i^r)$$

with the expectation function E(x).

With

$$E(x_i) = 0\,P(x_i=0) + 1\,P(x_i=1) = a/n$$

we have

$$E(d(x^r,x^k)) = E(d(a)) = 2a - 2n\, a/n\, a/n = 2(a - a^2/n)$$

The expectation value is maximized at a*

$$\frac{\partial}{\partial a} E(d(a))\Big|_{a=a^*} = 2(1-2a^*/n) = 0$$

and therefore the optimal length or "activity" of a prototype vector is $a^* = n/2$ with the maximal expected Hamming distance $d = n/2$.

The number of possible different prototypes is

$$N(a^*) = \binom{n}{a^*} = \binom{n}{n/2}$$

It is interesting to consider the question
What is the value of a *which maximizes the number* N *of possible prototypes ?*
It is

$$N(a) = \binom{n}{a} = \binom{n}{n-a} = \binom{n}{s} \quad \text{with } s := n-a$$

Since N(a) is monotonically increasing with a = 1, 2, ... a<<n, and this goes also with increasing s (i.e. decreasing a) for a = n, n-1, ... the function has a maximum at a=s and therefore

$$a^* = s^* = n-a^* \quad \Rightarrow \quad a^* = n/2$$

The optimal length of a vector with the maximal expected Hamming distance yields also the maximal number of possible vectors.

Example : For n=10 we have a*=5, d=5 and N(a*)=252 different prototype patterns.

4.0 Hardware Fault Tolerance

In the former part of this paper we took a closer look to the fault-tolerance properties of the treatment of input data. Beside this software-aspect we want to know now: what is the hardware fault tolerance typical for this kind of design? To what extent of degradation does the device continue to function properly ?
In comparison with its biological counterparts the matrix model with its complete connected units has too much connections. Are they all necessary? Under what circumstances do the device continue to function, even in the presence of failures or lacking connections?

4.1 The Linear Model

As we can see in part 2.0 the input of faulty data yields faulty output, too. Certainly, this is also true when we apply valid data to a matrix of randomly failed connections. Kohonen calculated in [KOH1] the mean and variance of the output patterns. He also showed in [KOH3], p.165, that in the case of orthogonal output patterns (orthogonal projection) an uniformly distributed random error $e_x := |x - x^r|$ in the input data is attenuated to the projection $e_y := |y - y^r|$ of the output by

$$\mathrm{var}(e_y) = p/m\, e_x^2$$

When the number of classes p is smaller than m= dim(y), the noise is diminuished.
By a memory matrix with failed connections. pattern recognition can be successfuly made if only the maximal component is taken.
Even for the operation of the linear model Kohonen found [KOH3,p114] that not all connections must be made; a relation of 40 between the number of input lines and the number of connections should be sufficient. Neither this nor other fault-tolerant statements [WOOD] are justified analytically.
Let us do this now for our threshold model of section 3.2.

4.2 The Fault Model

As a hardware model let us assume the functional model of figure 7 with the threshold function of (3.1e).
To explore the maximal fault-tolerance capabilities of our model let us assume that all class prototypes $x^1 \ldots x^P$ are maximal fault-tolerant coded, i.e. the x^i are orthogonal (cf. section 3.3). Since the output y^i is orthogonal, too, the weights reduce for $y_j^P = 1$

$$w_{ij} = \sum_k y_j^k x_i^k = \sum_k x_i^k = x_i^k$$

So the weights can only have the two values, 0 or 1.
Then for the ease of the model, let us consider *only two kind of fault events* of the hardware elements: stuck_at_one and stuck_at_zero. This is one of the most simple assumption which

are possible, but it will already show us some interesting fault tolerance proportions of the model.
More complicated fault models should be set up with a concrete hardware implementation on hand.
The components which can be faulty are the connection elements, the threshold elements and the summation elements (square elements in figure 7) of |x|. If we have for example 1000 input lines and 1000 output lines the number of connections are 10^6. In this example the threshold and summation elements constitute (with the same hardware complexity) only 0.2% of the hardware elements. If they fail, the output will be erroneous, of course, and must be corrected in the next stage by the next matrix device.
The main problem is the amount of connections: *what will be if they fail?*
For the failure of connections w_{ij} with stuck_at_1 we must distinguish two kinds of failures: **active failures** which produce constantly a '1' (e.g. binary counters) and **passive failures** which will cause a '1' only if the corresponding input line is activated (e.g. EEPROM connections).
In a large number of hardware independent connections we can neither assume that all faults are on the same input line (which will just cause an input error of one bit) nor that they are all on the same output line (which will be corrected in the following layer). Instead we will assume in the following evaluation that the faults are equally distributed in the whole connection matrix.

4.3 Tolerable Hardware Faults

Let us denote the failure probabilities

P_0 := P(connection defect and stuck_at_0)
P_1 := P(connection defect and stuck_at_1)

and assume that the faults occur independently.

Our question of 4.0 in this context is now:
How many connections can fail without producing erroneous output when a proper input pattern is presented?

When a class prototype is applied the erroneous activity results in an erroneous z_i:

$$z_i \rightarrow \text{error}(z_i)$$

and we have two situations where a faulty output is produced:

1) A column signal y_i^r is turned from 0 to 1 if too many connection weights are stuck_at_1. With (2.2b) this is equivalent to

$$\text{error}(z_i(x)) > t \qquad \text{with } x = x^k \dotplus x^r$$

2) A column signal y_i^r which should be 1 is turned to 0 if too many connections are

stuck_at_0. With (2.2b) this means

$$\text{error}(z_i(x)) \leq t \qquad \text{with } x = x^r$$

Let N_0 denote the number of weights stuck_at_0 and N_1 the number of weights stuck_at_1 which influence the summation in the output line z_i.
No error will occur and the prototype pattern x^r will invoke a correct output pattern if the inverse of the two error conditions hold

$$\begin{array}{lll} \text{For} & y_i^k=0 & \text{error}(z_i(x)) = z_i(x^k)+N_1-N_0 \leq t \\ \text{and} & y_i^r=1 & \text{error}(z_i(x)) = z_i(x^r)+N_1-N_0 > t \end{array} \qquad (4.3a)$$

With the Hamming distance (3.1d) and the optimal threshold of (3.1e) the two conditions for correct memory recall become for $|x^k|^2 = |x^r|^2 = a$

$$\begin{array}{lll} \text{For} & y_i^r=0 & 1/2\,(2a-d(x^k,x^r))+N_1-N_0 \leq 1/2\,(2a-d/2) \\ \text{and} & y_i^r=1 & 1/2\,(2a-d(x^r,x^r))+N_1-N_0 > 1/2\,(2a-d/2) \end{array} \qquad (4.3b)$$

The relations are for all x^k valid if $d(x^k,x^r)$ is minimal, i.e. $d(x^k,x^r) = d$. With $d(x^r,x^r) = 0$ we get

$$\text{For} \quad y_i^r=0 \quad N_1-N_0 \leq d/4 \qquad (4.3c)$$
$$\text{and} \quad y_i^r=1 \quad N_0-N_1 < d/4 \qquad (4.3d)$$

We should notify that the number N_0 and N_1 in (4.3c) are different to those defined for (4.3d) because the situations for the occurence of the faults are different.
In the *active failure model* N_1 is the number of weights active stuck_at_1 and in *passive failures* N_1 represents the number of only those weights passive stuck_at_1 which are connected to an active input line of x^r.
Let us now evaluate $N_0$and N_1 for the two conditions.

Too much stuck_at_1 faults

When we apply $x = x^k$ to the input lines, there are no connections $w_{ij} \neq 0$ which are connected to an active input line because $x^k x^r=0$; therefore there are no stuck_at_0 faults effective and N_0 becomes 0.
In the *active fault model* we know from appendix D that out of n elements with the failure probability P_1 there will be

$$N_1 = nP_1 \qquad (4.3e)$$

faulty.
In the *passive fault model* there are *a* active lines which can cause signals stuck_at_1 at the weights w_{ij} =0. Therefore, the first condition (4.3c) becomes with d=2a (orthogonal patterns)

$$\text{For} \quad y_i^r=0 \quad P_1 \leq a/2n \qquad \textit{active fault model} \qquad (4.3f)$$
$$\text{For} \quad y_i^r=0 \quad P_1 \leq 1/2 \qquad \textit{passive fault model} \qquad (4.3g)$$

Too much stuck_at_0 faults

The second relation (4.3d) is determined by a different situation. When we apply x^r to the

input lines, the maximum number of connection with weights $w_{ij} \neq 0$ and stuck_at_0 which can cause an error is a; the expected number is therefore

$$N_0 = a\, P_0 \qquad (4.3h)$$

In the *active fault model* there are n weights in a column which can be stuck_at_1 and can compensate the absence of signals, so (4.3e) holds here, too.
In the *passive fault model* all the activated lines have weigths $w_{ij} \neq 0$ and therefore the stuck_at_1 faults are not observable: $N_1 = 0$.
With (4.3e) and (4.3h) the relation (4.3d) becomes with d=2a

$$\text{For} \quad y_i^r = 1 \qquad \begin{array}{ll} P_0 < 1/2 + P_1\, n/a \leq 1 & \textit{active fault model} \\ P_0 < 1/2 & \textit{passive fault model} \end{array} \qquad (4.2i)$$

When P_1 reaches the tolerable upper limit for active faults of a/2n then P_0 have to be $P_0 < 1$.
If we consider only lacking connections ($P_1 = 0$), we can conclude that the device tolerates up to half of the connections beeing left out.

Example: Let n=100, a=10. A proper memory recall of the prototypes is only ensured if P_1 is at most 0.05 .

4.4 Discussion

The previous computations in part 4.3 are intended to demonstrate the fault tolerance power , inherent to the non-linear neural network models.
As we can see, the hardware model is very sensitive for active stuck_at_1 faults, i.e. faulty activity, but very robust and fault-tolerant for lacking connections. Thus the fabrication process can be made very easy or the number of connections can be reduced in the design.
Nevertheless, we should be still aware of the assumptions under those we have concluded the results above:

- the hardware model is very simple. More possible faults will yield a more adequate model. This is especially interesting when you regard a concrete implementation chip.

- the hardware faults are assumed to be independant. This can be when , for instance, stuck_at_0 means the interruption of a connection and stuck_at_1 means the shortcut of an output driver of the connection.
  Generally, the two kinds of faults are stochastically dependant due to the different internal failures of a connection, leading to the same syndromes stuck_at_0 or stuck_at_1. Without reasonable assumptions for the implementation of a connection this can hardly be quantified.

- The threshold condition (3.1e) implies as goal the class with the minimal hamming distance. Other goals (like that of the class with the maximal cross-correlation) and other models lead to other threshold conditions (e.g.

(3.1f)) and therefore to other restrictions for P_0 and P_1.

♣ As input patterns we chose orthogonal class prototypes. If we input instead only similar patterns, it can be shown that then the tolerable degree of failures (i.e. P_1 and P_0) will be less.
The limit is reached for the patterns on the class boarder with the distance d/2 to the class prototype. In this case no defect can be more tolerated.

5.0 Conclusion

In this paper we have investigated the necessary and sufficient conditions for memory recall in a non-linear neural network model of associative memory. By the nature of an included threshold mechanism the device gives a proper response even in the presence of erroneous input data. Thus, the memory recall operation includes a pattern recognition process which can be used for pattern search and pattern completion problems in the field of artificial intelligence. Since the whole operation is done in one clock cycle, the device can be regarded as a *very fast, parallel processor for high-level instructions with inherent fault-tolerance.*
The analysis of the hardware model reveals that the model is quite sensibel for erroneous activity due to active, faulty connections, but very robust and fault-tolerant for the failure or lack of connections. If the implementation of the model chooses only passive connections the resulting design promises to tolerate many faults not only in the normal life cycle but even in the fabrication process.

This work was supported by the Stiftung Volkswagenwerk.

References

[BRA] R. Brause, Fehlertoleranz in intelligenten Benutzerschnittstellen, Informatik Technologie 3, Oldenbourg Verlag 1988, in press

[FELD] J.A.Feldman, D.H.Ballard
Computing with connections , University of Rochester, Computer Science Department, TR72, 1980

[FUK] K. Fukushima, A Neural Network Model for selective Attention in Visual Pattern Recognition, Biologigal Cybernetics 55, p.5-15, Springer-Verlag 1986

[GOS] K.Goser, C. Foelster, U.Rueckert, Intelligent memories in VLSI. Information Sciences 34, p61-82, 1984

[HILL] D.Hillis, The Connection Machine
MIT Press, Cambridge, Massachusetts, 1985

[HIN1] Hinton, Anderson, Parallel Models of Associative Memory, Lawrence Erlbaum associates, Hillsdale 1981

[HIN2] Hinton, Implementing Semantic Networks in Parallel Hardware, in /HIN1/

[HOPF] J.J.Hopfield, Neural Networks and physical systems with emergent collective computational abilities, Proc.Natl.Acad.Sci.USA, Vol 79, pp.2554-2558, April 1982

[KOH1] T. Kohonen, Correlation Matrix Memories
IEEE Transactions on Computers C21 1972

[KOH2] Kohonen et alii, A demonstration of pattern processing properties of the optimal associative mapping, Proc. Int. Conf. Cybernetics and Society, Washington DC 1977

[KOH3] T.Kohonen, Self-Organisation and Associative Memory, Springer-Verlag Berlin,New York, Tokyo 1984

[LONG] H.C.Longuet-Higgins, Holographic model of temporal recall, Nature 217, 1968, p.104

[McCUL] W.S.McCulloch, W.H.Pitts, A Logical Calculus of the Ideas Imminent in Neural Nets, Bulletin of Mathematical Biophysics Vol 5,1943,pp.115-133

[WIL1] D. Willshaw, Models of distributed associative memory, Unpublished doctoral dissertion, Edinburgh University 1971

[WIL2] D.Willshaw, Holography, Association and Induction, in /HIN1/

[WIL3] D.Willshaw, O.P.Bunemann, H.C.Longuet-Higgins Non-holographic associative memory
Nature, 1969, pp.960-962

[WOOD] Variations on a Theme by Lashley:
Lesion Experiments on the Neural Model.
Psychological Review 85, 1978

Appendix A The class boundary of minimal distance

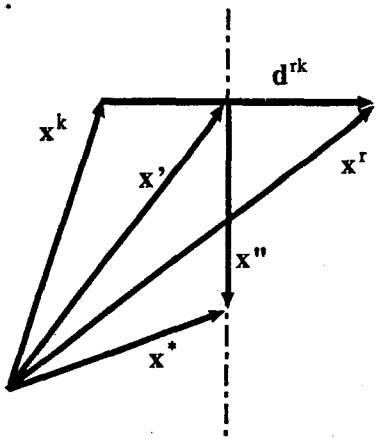
Let $\{x^*\}$ the boundary between two classes r and k which are formed by the classification criterium

$d(x,x^k) > d(x,x^r)$ then x of class r *classification by*
$d(x,x^k) < d(x,x^r)$ then x of class k *minimal distance*

Theorem:

a) $\{x^*\}$ is a hyperplane which
b) is orthogonal to the distance vector $\mathbf{d}^{rk} := (x^r - x^k)$ and
c) intersects at $\mathbf{d}^{rk}/2$.

Proof:



At the boarder the equation

$$d(x^*,x^r) = d(x^*,x^k)$$

holds.
With $d^2(x^*,x^r) = (x^*-x^r)^2$ we get

$$(x^r)^2 - (x^k)^2 + 2x^*(x^k-x^r) = 0$$

and with (see left figure)

$$x' := 1/2\,(x^r-x^k) + x^k$$
$$x^* := x" + x'$$

we get

$$(x^r)^2 - (x^k)^2 + 2x'(x^k-x^r) + 2x"(x^k-x^r) = 0$$
$$(x^r)^2 + (x^k)^2 - 2x^kx^r - (x^r-x^k)^2 - 2x"d^{rk} = 0$$

$$\underbrace{\qquad\qquad\qquad\qquad}_{0}$$

and so

$x"d^{rk} = 0$ or $x" \perp d^{rk}$ which prooves a) and b).

For $x^* = x'$ which is the common point of the hyperplane and the distance vector we know that $d(x',x^r) + d(x',x^k) = d(x^r,x^k)$. Because on the boarder the equation $d(x',x^r) = d(x',x^k)$ is also valid, we get $d(x',x^r) = |d^{rk}|/2$ which is part c) of the theorem.

Appendix B The threshold for maximal correlation

The classification rule (2.1a) is

$$xx^r = \max_k \; xx^k$$

Let us now regard the seperation of two classes r and k by the classification decision. We know from part 2.1 that the decision boarder is orthogonal to the distance vector $d^{rk} = |x^r - x^k|$. For all x on the boarder we have

$$x^* x^r = x^* x^k := x^* c^{rk}$$

with c^{rk} parallel to x^* (and therefore orthogonal to d^{rk}) as illustrated in figure B.

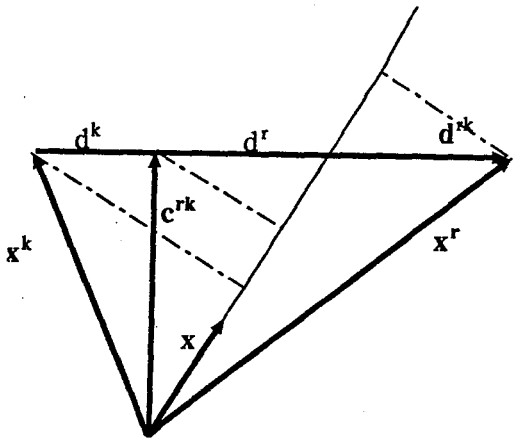


Fig. B boundary of classification with maximal correlation

For a pattern x we see from figure B that for the projections on x the relation holds

$$xx^r > x^* c^{rk} > xc^{rk} > xx^k \qquad \text{(B.1)}$$

The classification is

$$\begin{aligned} xx^r &> t_{cross} && \text{then x is of class r} \\ xx^r &\leq t_{cross} && \text{then x is not of class r} \end{aligned} \qquad \text{(B.2)}$$

Certainly, the basic decision criterion for class r is $xx^r > xx^k$. Since our hardware mechanism supports only one comparison without communication between processing elements, the algorithm implied by (2.1a) can not be implemented directly. Instead, we have to choose a threshold t_{cross} which makes a correct decision in one comparison, using only available parameters. For this reason we choose one of the intermediate values of ralation (B.1) as threshold, bearing in mind that this narrows the set of patterns belonging to class r. Some patterns of class r are projected on the null vector.

So we choose

$$t^{rk}_{cross} := x^* c^{rk} = |x||c^{rk}| \qquad \text{(B.3)}$$

By basic geometric proportions (see fig. B) we have with $|c^{rk}| =: c$

$$|x^r|^2 = c^2 + (d^r)^2 \qquad (d^{rk})^2 = (d^r + d^k)^2$$
$$|x^k|^2 = c^2 + (d^k)^2$$

and by combining and substitution we get

$$|c^{rk}| = \left(|x^r|^2 - \left((|x^r|^2 - |x^k|^2 + |d^{rk}|^2) / 2d^{rk} \right)^2 \right)^{1/2}$$

or in correlation terms

$$|c^{rk}| = \left(|x^r|^2 - (|x^r|^2 - x^r x^k)^2 / (|x^r|^2 + |x^k|^2 - 2x^r x^k) \right)^{1/2}$$

The threshold must be valid for all classes

$$t^r_{cross} = \max_k t^{rk}_{cross} = |x||c^{rk}|$$

Since the threshold part $|c_r|$ can be calculated once before the pattern recall process and stored in the processing element, the resulting threshold can be build up at recall time by calculation of $|x|$ which can be done very easily as shown in figure 6.

For normalized prototypes ($|x^r|^2 = |x^k|^2 =: a$) we get with the maximal cross-correlation K_r

$$t^r_{cross} = \max_k |x| \left(1/2(a + x^r x^k) \right)^{1/2} = |x| \left(1/2(a + K^r) \right)^{1/2} \qquad \text{(B.4)}$$

Another threshold may be taken from the distance measure, which has the same decision criterium in the case of normalized prototypes (see C.3).

With $(d^r)^2 = \min_k (x^r - x^k)^2 = \min_k 2(a - x^k x^r) = 2(a - K^r)$

we get by (C.6)

$$t^r_{cross} = 1/4 \, (2|x|^2 + a + K^r) \qquad \text{(B.5)}$$

Appendix C The threshold for minimal distance classification

The classification of a pattern x, to the class of the most resembling prototype x^k is determined by the rule of (2.1b)

$$|x - x^r| = \min_k |x - x^k|$$

With $d(x,x^k) := |x - x^r|$ we have for all classes $k \neq r$

$$d(x,x^k) > d(x,x^r)$$

$$(x - x^k)^2 = d^2(x,x^k) > d^2(x,x^r) = (x - x^r)^2$$

and so

$$x x^r > x x^k - 1/2 \, |x^r|^2 + 1/2 \, |x^k|^2$$

The classification rule is then

$$xx^r > t^{rk}_{dist} \quad \text{then x is of class r} \qquad (C.1)$$
$$xx^r \leq t^{rk}_{dist} \quad \text{then x is not of class r}$$

with the threshold

$$t^{rk}_{dist} = xx^k - 1/2\,|x^r|^2 + 1/2\,|x^k|^2 \qquad (C.2)$$

For normalized prototypes ($|x^r|^2=|x^k|^2=:a$) this becomes

$$t^{rk}_{dist} = xx^k \qquad (C.3)$$

which is essentially the cross-correlation criterium.
Let us now calculate a threshold which implements the demand of (2.1b). As it is already indicated in appendix B, the threshold which is a decision boarder to all classes will not assign all patterns of class r to the classprototype but some to the null vector.
From appendix A we know that the boundary between two classes r and k is at $d(x^r,x^k)/2$.

Thus for x of class r we have

$$d(x^r,x) < d(x^r,x^k)/2 < d(x,x^k)$$

and with condition (2.1b) we get

$$d(x^r,x) < \min_k\ d(x^r,x^k)/2 < \min_k d(x,x^k)$$

With $d^r := \min_k d(x^r,x^k)$ we have with positive d(.)

$$(x-x^r)^2 = d^2(x^r,x) < (d^r/2)^2$$
$$xx^r > 1/2\,(|x|^2 + |x^r|^2 - (d^r/2)^2)$$

The classification rule becomes

$$\text{If } xx^r > t^r_{dist} \quad \text{then x is of class r} \qquad (C.4)$$
$$\text{If } xx^r \leq t^r_{dist} \quad \text{then x is not of class r}$$

with the threshold

$$t^r_{dist} = 1/2\,(|x|^2 + |x^r|^2 - (d^r/2)^2) \qquad (C.5)$$

For normalized protypes this is

$$t^r_{dist} = 1/2\,(|x|^2 + a - (d^r/2)^2) \qquad (C.6)$$

In the binary case for the Hamming distance we get

$$t^r_{dist} = 1/2\,(|x^r|^2 + |x|^2 - d_H{}^r/2) \qquad (C.7)$$

and

$$t^r_{dist} = 1/2\,(|x|^2 + a - d_H{}^r/2) \qquad (C.8)$$

Appendix D Evaluation of the failure probability

Suppose we have n connections in one column (cf. fig.7) and a probability of failure of each connection. The probability that among *n* independent elements just *j* are faulty is

$$P(\text{fault})^j\,(1-P(\text{fault}))^{n-j}$$

Since there are $\binom{n}{j}$ such faulty tuples of j elements, the probability of j faults in n elements is

$$P(j \text{ faults}) = \binom{n}{j} P(\text{fault})^j\,(1-P(\text{fault}))^{n-j} \qquad (D.1)$$

This is the binomial distribution. The expected number of faulty elements is therefore

$$N = E_{(\text{number of faulty elements})} = \sum_{j=0}^{n} j\,P(j \text{ faults}) = \sum_{j=0}^{n} j \binom{n}{j} P(\text{fault})^j\,(1-P(\text{fault}))^{n-j}$$

With P:=P(fault) and Q:=1-P is

$$N = \sum_{j=0}^{n} j \binom{n}{j} P^j\,Q^{n-j} = P\frac{\partial}{\partial P}\sum_{j=0}^{n}\binom{n}{j}P^j\,Q^{n-j} = P\,n\,(P+Q)^{n-1} = nP \qquad (D.2)$$