J.A. Ritcey, L.E. Atlas, A. Somani, D. Nguyen, F. Holt and R.J. Marks II, "A signal space interpretation of neural networks", Proceedings of the IEEE International Symposium on Circuits and Systems, pp.370-376, Philadelphia, May 1987.

1987 IEEE INTERNATIONAL SYMPOSIUM ON CIRCUITS AND SYSTEMS

Volume **2** of **3**

Dunfey City Line Hotel
Philadelphia, PA
May 4-7, 1987

87CH2394-5

J.A. Ritcey, L.E. Atlas, A. Somani, D. Nguyen, F. Holt and R.J. Marks II, "A signal space interpretation of neural networks", Proceedings of the IEEE International Symposium on Circuits and Systems, pp.370-376, Philadelphia, May 1987.

A SIGNAL SPACE INTERPRETATION OF NEURAL NETS

J.A.Ritcey^, L.E.Atlas^, A.Somani^, D.Nguyen*, F.Holt* and R.J.Marks, II^

^ Dept. of Electrical Engineering
University of Washington
Seattle, WA 98195

* Boeing High Technology Center
Bellevue, WA 98125

ABSTRACT

Hopfield neural net processors (NNP) have been shown to be an interesting class of fault tolerant, parallel computers for pattern recognition. In this paper, we give some limited simulation results that contrast the performance of the Hopfield NNP, whose T-matrix is in sum-of-outer-products form, and the Projection NNP, which uses an orthogonal projection onto the linear space spanned by the library elements. A Compact NNP is introduced which promises good recall ability with a low density of neuron interconnections.

I. INTRODUCTION

Since the introduction of neural networks (NN) to the engineering community by Hopfield [1], a number of applications and variaties of the basic net have been proposed. In this paper, we present both the Hopfield and the Projection neural net processors (NNP) and compare performance based on some limited simulation results. The Projection NN (PNN), in which Hopfields T-matrix, T_H, is replaced by the projection matrix, T_P, onto the linear subspace spanned by the library elements, is suggested by signal space concepts. In addition to these baseline performance comparisons, we present a reduced complexity neural net, the Compact NN (CNN), whose T-matrix is obtained from T_H by quantizing far off-diagonal terms to zero. Reorganization of the library elements is a key point in the development of efficient nets of this type.

The research at the University of Washington is supported by the Boeing Electronics Company, High Technology Center under contract LD2709.

II. NONLINEAR NEURAL NET PROCESSORS

The usual setting is that we are given L library elements $F_L=\{f_1,\ldots,f_L\}$, in which each library element f_μ is an N-vector whose elements are chosen from $\{-1,+1\}$. Defining the set $V_N = \{v=(v_1,\ldots,v_N)^T \mid v(k) \in \{-1,+1\}, k=1,2,\ldots,N\}$ we see that $f_\mu \in V_N$, for all μ. We describe V_N as the set of hypercube vertices in R^N with $\mathrm{card}(V_N)=2^N$. Next, assume we are given a probe vector $p \in R^N$. The NNP seeks to find that $f_\mu \in F_L$ such that the L_2 distance, $||f_\mu - p||^2$, is a minimum. The Hopfield NNP forms the matrix

$$T_H = -NI_N + \sum_{\mu=1}^{L} f_\mu f_\mu^T \qquad (1)$$

where I_N is the NxN identity matrix. Notice that T_H is the sum of outer products of library elements, with the diagonal elements set to zero. The Hopfield iteration is to set $v_0=p$ and iterate according to

$$v_{n+1} = \mathrm{sgn}(\, T_H v_n \,) \qquad (2)$$

until a fixed point v, satisfying $v=\mathrm{sgn}(T_H v)$, is reached. Since $v \in V_N$, we select v as our estimate of which library element is closest to the probe p in Hamming (or L_2) distance. This NNP works well when $L < C_N = N/(2\log N) << N$, that is we operate the net below its capacity C_N [2]. In general, the net may take many iterations to converge--if it does so at all. In addition, we are not guaranteed that $v \in F_L$. Fixed points that are not members of the original set of library elements F_L are known as <u>false memories</u>. Also, it is not assured that every library element f_μ is a fixed point. A NNP for which every library element is a fixed point is said to have the <u>input verification property</u>. Note that the Hopfield neural net is not optimal in the sense that if we assume $\Pr(\mu=1) = \ldots = \Pr(\mu=L) = 1/L$, $p=f_\mu + n$, $n \sim MVN(0,\sigma^2 I_N)$, then the minimal probability of error classifier implements

$$\min_{\mu} ||p-f_{\mu}||^2 \text{ or when } f_i f_j^T = \delta_{ij},$$

$$\max_{\mu} p^T f_{\mu}.$$

This is the well-known matched filter receiver [3]. The advantages of a NNP lie in areas other than optimality under a min P_E criterion and strict assumptions on the input noise. The HNN is fault tolerant in the sense that if one interconnection is broken or if 1 neuron is "stuck at" a given value, the net is remarkably resilant. Also, the NNP solution does not require detailed assumptions about the noise distribution. For these reasons, and because a neural net uses a large number of simple processors, NNPs are an important class of parallel computers.

Of course, if given the probe p, we only wanted to find the nearest library element in S_L = span($f_1,...,f_L$), the linear subspace spanned by the library elements, we would set $v=T_P p$ where $T_P = F(F^TF)^{-1}F^T$ is the projection onto the subspace S_L and $F=[f_1|...|f_L]$ is the NxL library element matrix. This suggests the projection NN (PNN) whose iteration is defined by

$$v_{n+1} = \text{sgn}(\ T_P v_n\ ) \qquad (3)$$

with $v_0=p$. This neural net has been suggested by [4] and is discussed in Marks and Atlas [5]. No analytical results such as [2] are available for the capacity of the PNN, but limited simulation studies show that the PNN usually converges to a fixed point in fewer iterations than a HNN.

One might measure the overall performance of a neural net processor by its performance in five basic areas:

(1) Input verification
Are all library elements fixed points?

(2) False memories
How likely is it that we converge to an element not in F_L?

(3) Speed of convergence
How many iterations does it take to reach a fixed point?

(4) Fault tolerance

(5) Implementation complexity.

Areas (4) and (5) are addressed in section IV. In section III, we compare the HNN and PNN based on (1)-(3).

III. SIMULATION RESULTS ON THE HNN AND PNN.

Based on 780 independent simulation trials with N=100, and L=10, we find that the HNN and PNN both have their merits, but in different areas. The results are summarized in Tables 1 and 2. In Table 1, we see a histogram of the number of iterations required to reach a fixed point for the PNN and the HNN. Our experience in this and other simulation studies is that the PNN often exhibits single-step convergence. Table 1 shows that on the average, the HNN can take many more iterations than the PNN to converge to a fixed point. It is not clear from this data whether a majority of the neurons were still changing after the first iteration or if it is only a small subset of the N neurons that take a protracted number of iterations to converge.

No. of Iterations	HNN	PNN
10	0	0
9	5	0
8	25	0
7	14	0
6	40	10
5	101	19
4	139	51
3	204	161
2,1	252	539

TABLE 1 Histogram of the number of iterations required for a fixed pt.

	No. of False Fixed Points
HNN	16
PNN	110

TABLE 2 False Memories

Note that although the speed of convergence of the PNN is much faster than that of the HNN, the probability of landing at a false fixed point is increased. In Table 2, we show that of the 100^2=10,000 possible states, the HNN exhibits convergence to some 16+10=26 while the PNN converges to 110+10=120 (in 780 trials). The PNN has the input verification property, while this is not guaranteed in the formulation of T_H;i.e., that every library element is a fixed point. Library elements are always fixed points in the PNN by construction of T_P. However, when L << N and the net is operated below capacity, this will usually be the case for T_H as well. In all of our simulations, the neural net is operated synchronously.

IV. THE COMPACT NEURAL NET

An important issue in NNP design is the implementation complexity. Electronic, optical, and hybrid net architectures have been proposed. An important complexity measure, regardless of implementation technology, is the

$$\min_{\mu} ||p-f_\mu||^2 \text{ or when } f_i f_j^T = \delta_{ij},$$

$$\max_{\mu} p^T f_\mu .$$

This is the well-known matched filter receiver [3]. The advantages of a NNP lie in areas other than optimality under a min P_E criterion and strict assumptions on the input noise. The HNN is fault tolerant in the sense that if one interconnection is broken or if 1 neuron is "stuck at" a given value, the net is remarkably resilant. Also, the NNP solution does not require detailed assumptions about the noise distribution. For these reasons, and because a neural net uses a large number of simple processors, NNPs are an important class of parallel computers.

Of course, if given the probe p, we only wanted to find the nearest library element in $S_L = \text{span}(f_1,\ldots,f_L)$, the linear subspace spanned by the library elements, we would set $v=T_P p$ where $T_P = F(F^TF)^{-1}F^T$ is the projection onto the subspace S_L and $F=[f_1 | \ldots | f_L]$ is the NxL library element matrix. This suggests the projection NN (PNN) whose iteration is defined by

$$v_{n+1} = \text{sgn}(\ T_P v_n\ ) \qquad (3)$$

with $v_0=p$. This neural net has been suggested by [4] and is discussed in Marks and Atlas [5]. No analytical results such as [2] are available for the capacity of the PNN, but limited simulation studies show that the PNN usually converges to a fixed point in fewer iterations than a HNN.

One might measure the overall performance of a neural net processor by its performance in five basic areas:

(1) Input verification
Are all library elements fixed points?
(2) False memories
How likely is it that we converge to an element not in F_L?
(3) Speed of convergence
How many iterations does it take to reach a fixed point?
(4) Fault tolerance
(5) Implementation complexity.

Areas (4) and (5) are addressed in section IV. In section III, we compare the HNN and PNN based on (1)-(3).

III. SIMULATION RESULTS ON THE HNN AND PNN.

Based on 780 independent simulation trials with N=100, and L=10, we find that the HNN and PNN both have their merits, but in different areas. The results are summarized in Tables 1 and 2. In Table 1, we see a histogram of the number of iterations required to reach a fixed point for the PNN and the HNN. Our experience in this and other simulation studies is that the PNN often exhibits single-step convergence. Table 1 shows that on the average, the HNN can take many more iterations than the PNN to converge to a fixed point. It is not clear from this data whether a majority of the neurons were still changing after the first iteration or if it is only a small subset of the N neurons that take a protracted number of iterations to converge.

No. of Iterations	HNN	PNN
10	0	0
9	5	0
8	25	0
7	14	0
6	40	10
5	101	19
4	139	51
3	204	161
2,1	252	539

TABLE 1 Histogram of the number of iterations required for a fixed pt.

	No. of False Fixed Points
HNN	16
PNN	110

TABLE 2 False Memories

Note that although the speed of convergence of the PNN is much faster than that of the HNN, the probability of landing at a false fixed point is increased. In Table 2, we show that of the 100^2=10,000 possible states, the HNN exhibits convergence to some 16+10=26 while the PNN converges to 110+10=120 (in 780 trials). The PNN has the input verification property, while this is not guaranteed in the formulation of T_H;i.e., that every library element is a fixed point. Library elements are always fixed points in the PNN by construction of T_P. However, when L << N and the net is operated below capacity, this will usually be the case for T_H as well. In all of our simulations, the neural net is operated synchronously.

IV. THE COMPACT NEURAL NET

An important issue in NNP design is the implementation complexity. Electronic, optical, and hybrid net architectures have been proposed. An important complexity measure, regardless of implementation technology, is the connectivity of the NNP. Specifically, both T_H and T_P previously defined are, in general, dense NxN matrices. T_H, of course, has zero diagonal elements--that is, the Hopfield NNP uses no auto (or self) interconnections. As N increases, it becomes increasingly difficult to lay out a densely interconnected net. For this reason, we have investigated Compact Neural Nets (CNN) in which elements of the T-matrix far from the diagonal are set to zero. In this section, and in all of our studies of the CNN to date, we restrict ourselves to the Hopfield formulation and let $T=T_H$.

A HNN whose T-matrix is all zero except for the 1 upper and lower diagonals, can be efficiently implemented using a ring architecture. This is illustrated in equation (4) where X signifies a nonzero element and N=4.

$$Tv = \begin{bmatrix} 0 & X & 0 & X \\ X & 0 & X & 0 \\ 0 & X & 0 & X \\ X & 0 & X & 0 \end{bmatrix} \begin{bmatrix} v(1) \\ v(2) \\ v(3) \\ v(4) \end{bmatrix} . \quad (4)$$

Notice that the design wraps around so that neuron i is connected to neuron (i+1)mod(N). A ring architecture corresponding to the T-matrix shown in (4) is shown in Figure 1.

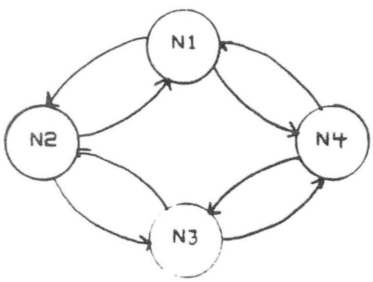

Figure 1 Ring Architecture

The question arises : Which matrix T is least affected by quantization of outer elements to zero? Although we have no analytical proof (in the sense of least performance degradation), we seek T matrices whose elements are largest near the main diagonal and smallest far from the the diagonal. At least in this way $\|T-T_j\|$ is maximized, when T_j is the original $T=T_H$ with all but the j upper and lower sub (super) diagonals set to zero. But what freedom do we have in the design of T, given that we follow the Hopfield recipe (1)? The answer is that we can rearrange, or permute, the elements of every f_μ to obtain g_μ where $g(i) = f_\mu(\pi(i))$ and $(\pi(1),\ldots,\pi(N))$ is any permutation of $(1,\ldots,N)$. Any probe vector that we receive would be permuted upon arrival, input to the modified or compact NNP, a fixed point reached (or, possibly, we allow only a fixed number of iterations for ease of implementation), and the output sent through the inverse permutation (if really necessary) to obtain our best estimate of which library element was transmitted, given the received (distorted) probe vector as data.

In all of our studies, we have found that a single iteration of $v_{n+1} = \mathrm{sgn}(T_C v_n)$, where T_C is the CNN T-matrix, is sufficient for convergence of a majority of the neurons. Thus, for probe p, we decide that $v=\mathrm{sgn}(T_C p)$ was transmitted. More iterations may increase recall performance, at the expense of implementation complexity.

An example of the library element matrix F and its rearrangement G is shown in (5). Here, L=3 and N=10. THese values are used for illustration and were not used in any of our simulations. Notice that -1,+1 components appear randomly distributed.

F	G	
---	---	
--+	---	
--+	---	
-+-	--+	
+--	--+	(5)
+++	-++	
-++	+++	
---	-+-	
---	+--	
+--	+--	

Notice that +1 or -1 components now appear in bursts. We have used a greedy algorithm to determine this rearrangement. The algorithm starts at row i=1 and makes 1 pass through the data to i=64. Most of the rows of G are close in a Hamming sense, except for i=1 and i=64. This affects recall ability of these components, but is an artifact of our rearrangement algorithm. No attempt has been made at this time to utilize the bursty structure of the library elements in order to develop a more fault tolerant compact neural net. Remember that the Hopfield Neural Net is already fault tolerant by design. In the CNN, we expect that error correcting code techniques can be applied. Some simulation results for N=64 and L=3 and 1 iteration are shown in Table 3.

D	No.	Converged
0	3	3
1	192	192
2	300	289
3	300	276
4	300	235
5	300	204

Table 3 Convergence of probe vectors at distance D from library

Notice that as the Hamming distance (D) between the probe and the nearest library element increases, the percentage of elements that converge correctly decreases. Although the quantization of T_H to T_C has some effect, we feel that the primary reason is because only a single iteration is allowed. In any event, the degradation is gradual and no threshold effect is visible, at least in these limited simulations. In Table 4, we list a summary of some further results on the percentage of probes that correctly converged, parametric in N,L and the number of neighbors (diagonals) used in the CNN.

N	L	No. of Neighbors	% Converge
32	3	1	0.99
32	3	2	0.99
32	4	1	0.77
32	4	2	0.60
64	3	1	1.00
64	3	2	1.00
64	4	1	0.92
64	4	2	0.89
64	5	1	0.63
64	5	2	0.35

Table 4 Percentage of Correct Convergence for the Compact Neural Net

For example, when the number of neighbors is 2, a total of 4 off-diagonals is used in T_C. These results are also taken after a single iteration. We find a strong relationship with the capacity results of [2].

V. CONCLUSIONS

In conclusion, we have present some limited simulation results that contrast the differences between the Hopfield NN and that suggested by matched filter theory, the Projection NN. The fact that the PNN exhibits a large number of false memories is a great disadvantage in many applications. However, simulations of NNP that use large numbers of neurons, say, N=10,000, may yet show the importance of the speed advantages of the PNN.

VI. REFERENCES

1. J.J.Hopfield, Proc. Natl. Acad. Sci., USA, vol. 79, pp.2554-2558, 1982.

2. R.J.McEliece, E.C.Posner, E.R.Rodemich, and S.S.Venkatesh, " The Capacity of the Hopfield Associative Memory", submitted to IEEE Trans. on Information Theory.

3. H.L.Van Trees, Detection, Estimation, and Modulation Theory, Part I, Wiley: New York, pp.257-271.

4. L.Personnaz, I.Guyon, G.Dreyfus, "Information Storage and Retrieval in Spin-Glass Like Neural Networks", J.Physique Lett., vol.46 (1985) L-359-L-365.

5. R.J.Marks, II and L.E.Atlas, "Content Addressable Memories: A Relationship Between Hopfield's Neural Net and AN Iterative Matched Filter", submitted to IEEE Trans. on Circuits and Systems.