$$\frac{2^{n\ k}!}{(2^{n-m}!)^{2^{m-k}}}$$

ways, so the remaining bits can be assigned in

$$\left(\frac{2^{n-k}!}{(2^{n-m}!)^{2^{m-k}}}\right)^{2^k}$$

ways. Thus the number of regular s-boxes in which the first k output co-ordinates are linear functions is given by

$$R(n,m,k) = 2^k \left(\frac{2^{n-k}!}{(2^{n-m}!)^{2^{m-k}}}\right)^{2^k} \prod_{i=0}^{k-1}(2^n - 2^i)$$

Consider the function Φ: $Z_2^n \rightarrow Z_2^{2^m-1}$ constructed from every nonzero linear combination of the output co-ordinates of f. Counting the number of nonlinear regular s-boxes corresponds to counting the number of functions Φ with no affine co-ordinates.

The number of ways we can choose l co-ordinates of Φ such that k of them are linearly independent is equivalent to the number of $l \times m$ binary matrices (without taking the order of rows into account, i.e. two matrices with the same rows but in different orders are counted once) with nonzero distinct rows which have rank k. This is given by [4, 5]

$$LI(m,l,k) = \binom{m}{k}_2 \sum_{j=0}^{k}(-1)^j\, 2^{\binom{j}{2}} \binom{2^{(k-j)}-1}{l} \binom{k}{j}_2$$

where

$$\binom{m}{k}_2 = \begin{cases} 1 & k = 0 \\ \dfrac{\prod_{i=0}^{k-1}(2^m - 2^i)}{\prod_{i=0}^{k-1}(2^k - 2^i)} & n \geq k > 0. \end{cases}$$

It is clear that for every k linearly independent co-ordinates of Φ, denoted by (ϕi_1, ϕi_2,...,ϕi_k), we can find $(m - k)$ co-ordinates of Φ, denoted by (ϕi_{k+1}, ϕi_{k+2},...,ϕi_m), such that

$$(\phi_{i_1}\ \phi_{i_2}\ .\ .\ \phi_{i_m})^t = A(f_1\ f_2\ .\ .\ f_m)^t$$

where A is an $m \times m$ invertible binary matrix and $(f_1\ f_2\ .\ .\ f_m)$ denotes the output co-ordinates of f. This means that as f varies over all the set of distinct regular s-boxes, (ϕi_1, ϕi_2 . . ϕi_m) scans the whole set but in a different order. From the above argument, it is clear that the number of ways of constructing certain k linearly independent co-ordinates of Φ from affine functions is also given by $R(n, m, k)$.

Using the inclusion-exclusion principle, the number of linear regular s-boxes, i.e. regular s-boxes with the property that one or more of the nonzero linear combinations of their output co-ordinates are affine, is given by

$$RL(n,m) = \sum_{l=1}^{2^m-1}(-1)^{l-1} \sum_{k=1}^{min(l,m)} LI(m,l,k)\ R(n,m,k).$$

To express the above count as a fraction of the total number of regular s-boxes, denoted by $FRL(n, m)$, we divide by the total number of $n \times m$ regular s-boxes

$$\frac{2^n!}{(2^{n-m}!)^{2^m}} \qquad n \geq m$$

To give a numerical example, for $n = 6$ and $m = 4$, which is the size of DES s-boxes, $FRL(6,4) = 2.46 \times 10^{-16}$. We can easily get an upper bound for $FRL(n, m)$ by noting that

$$RL(n,m) < (2^m - 1)\ R(n,m,1)$$

and hence that

$$FRL(n,m) < \frac{2(2^n-1)(2^m-1)(2^{n-1}!)^2}{2^n!} = O\left(\frac{2^{5n/2}}{2^{2^n}}\right).$$

Conclusion: We have derived an exact expression for the number of regular s-boxes with the property that one or more of the nonzero linear combinations of their output co-ordinates are affine. From the above, it is clear that this fraction decreases dramatically with the number of inputs.

15 June 1995
Electronics Letters Online No: 19951086

A.M. Youssef and S.E. Tavares (*Department of Electrical and Computer Engineering, Queen's University, Kingston, Ontario, K7L 3N6, Canada*)

S.E. Tavares: corresponding author

E-mail: tavares@ee.queensu.ca

References

1 GORDON, J., and RETKIN, H.: 'Are big S-boxes best?'. Lecture Notes in Computer Science: Proc. Workshop on Cryptography, (Springer-Verlag, 1982), pp. 257–262
2 BIHAM, E., and SHAMIR, A.: 'Differential cryptanalysis of DES-like cryptosystems'. Advances in Cryptology: Proc. Crypto '90, (Springer-Verlag, 1991), pp. 1–21
3 MATSUI, M.: 'Linear cryptanalysis method for DES cipher'. Advances in Cryptology: Proc. Eurocrypt '93, 1994, (Springer-Verlag), pp. 366–397
4 NYBERG, K.: 'Perfect nonlinear S-boxes', Advances in Cryptology: Proc. Eurocrypt '91, (Springer-Verlag, 1992), pp. 378–386
5 STRONG, R.: Private communication
6 GOLDMAN, J., and ROTA, G-C: 'On the foundation of combinatorial theory IV. Finite vector spaces and Eulerian generating functions', *Stud. Appl. Math.*, 1970, **XLIX**, (3), pp. 239–258

Reduced-complexity circuit for neural networks

S.S. Watkins and P.M. Chau

Indexing terms: Neural networks, Reduced instruction set computing

The Letter demonstrates that a 10 bit reduced-complexity VLSI circuit can be used in place of a 32 bit floating-point processor to speed up some neural network applications, reducing circuit area and power consumption by 88% with a negligible increase in RMS error. Applications were executed on a radial basis function neurocomputer using the reduced-complexity circuit implemented with FPGA technology. One application produced better results than had been previously obtained for a NASA data set using either neural network or non-neural network approaches.

Introduction: Today's hardware capabilities are limiting the development of neural network research. Neural networks learn by adjusting weights on input and internal signals by very small increments until the network has converged on a solution that is satisfactory for all training patterns, and this process can take days, weeks or months on a modern workstation. A neural network usually exhibits a significant amount of potential parallelism, and hardware accelerators can reduce the learning time by orders of magnitude by exploiting this parallelism. Many applications require less than 32 bits of floating-point precision [1], and this fact can be used to reduce the cost of the accelerator circuits in terms of area and power. For one application described below, the use of a unique 10 bit reduced-complexity multiply/accumulate circuit resulted in an area and power saving of 88% over a full 32 bit floating-point circuit, while the learning results as measured by RMS error were within 0.03% of the 32 bit results.

Radial basis function neural networks: Our research has focused on implementing radial basis function (RBF) neural networks with reduced-complexity VLSI circuits as a means to accelerate learning while minimising costs in terms of area and power. The RBF network uses a radial basis function (usually a Gaussian) as the transfer function of a neuron rather than the traditional sigmoid function. Radial basis functions have been used to solve mapping and function estimation problems with positive results [2]. The equations describing an RBF neuron's output x_j in terms of an input vector and stored weights are:

$$z_j = \sum_k (C_{jk} - I_k)^2 \qquad (1)$$

$$x_j = \exp(-z_j/w_j) \qquad (2)$$

where C_{jk} is the stored centre for the input element k of neuron j; I_k is element k of the input vector; w_j is the width of the Gaussian for neuron j.

We have implemented an RBF neurocomputer with the 10 bit reduced-complexity circuit, a PC and software written in C. The subtraction, squaring and summing of the input layer are performed by the reduced-complexity circuit. The results of these operations are passed to a lookup table to produce a Gaussian x_j. While the reduced-complexity circuit was applied to an RBF network, it can also be easily applied to a back-propagation neural network, since this type of network also requires the multiply/accumulate operation.

Application results: Two applications have been executed on the custom RBF neurocomputer with very positive results: (i) a remote sensing application that has important implications for both short-term and long-term climate modeling [3]; (ii) a Mackey-Glass time series estimation application [4]. The execution of the remote sensing application produced better results than had been previously obtained for the data set supplied by NASA. Fig. 1 shows the learning curves for the remote sensing application. The Y-axis represents the RMS error of the network after X passes through the training set, and is a measure of how well the network is learning. The Figure shows that results using 10 bits of precision are nearly indistinguishable from results using 32 bits. The most important measure of the quality of the reduced-complexity approach is how well the network does on the test patterns after it has been trained. On the remote sensing test patterns, the 10 bit reduced-complexity approach produced results that are within 0.03% of the 32 bit results.

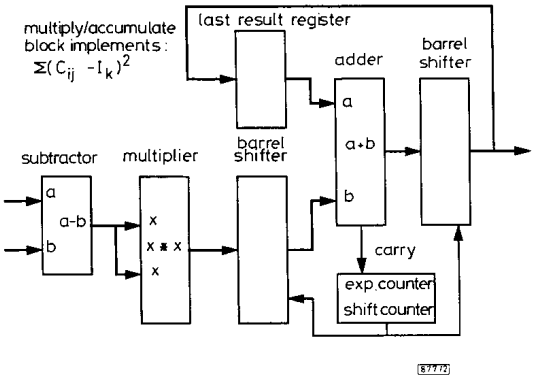

Fig. 1 *Comparison of learning curves for 8, 10 and 32 bit precision*

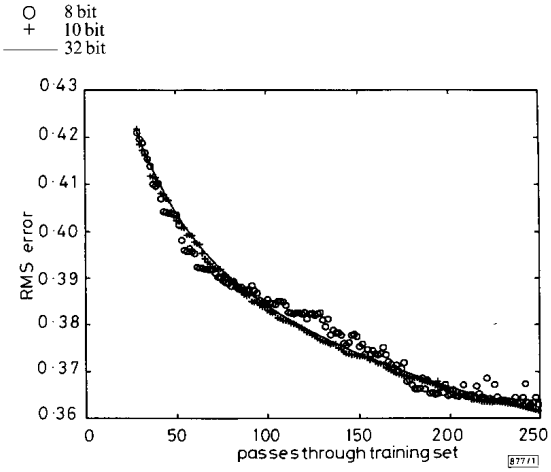

Fig. 2 *Block diagram of reduced-complexity multiply/accumulate circuit (input layer for RBF network)*

Reduced-complexity circuit: Fig. 2 shows the reduced-complexity circuit. It consists of subtraction and multiplication operations with 10 bits of precision, and accumulation with 20 bits of precision. Two 20 bit-wide barrel shifters, a counter to represent an exponent and a small control block complete the circuit. The barrel shifters and counter create a true floating-point capability. The circuit scales better than O(Nlog$N$) in terms of time and area (where N is the number of network inputs). The N term accounts for time, because an additional cycle is required for each additional input. The log$N$ term accounts for area. The area scales better than log$N$ because, as the number of inputs grows, the additional bits needed to represent the sum are in the exponent (i.e. storage area scales as log(log$N$)).

Table 1: Comparison of different circuit implementations for RBF network input layer

Processor	Number of transistors	Power [mW]	Cycle time [ns]	Mult./ accum. ops. per µs	Power per ops. per µs
10 bit mult. 20 bit accum. custom CMOS	6860	103	20	50	2.1
Texas Inst. TMS320C50	> 50000	500	25	40	12.5
10 bit mult. 40 bit accum. FPGA	41400 (180 CLB's)	375	40	25	15.0
32 bit mult. 40 bit accum. custom CMOS	60000	892	20	50	17.8
Texas Inst. TMS320C30 floating-pt. DSP	> 100000	1000	40	25	40.0

Order is based on most efficient implementation in terms of power per operation

Analysis of different implementations: The circuit was implemented with an FPGA to demonstrate feasibility. Others have also implemented neural network accelerators with SRAM FPGAs, but they have not compared FPGAs with other approaches [5]. Table 1 shows the area (number of transistors) and power comparisons for both custom and FPGA implementations of the circuit in order of efficiency (as measured by power per operation). Also shown for purposes of comparison are commercial fixed-point and floating-point digital signal processing chips (DSPs). As can be seen from the Table, the 10 bit fully custom approach is the most efficient and uses 88% less area and power than a 32 bit fully custom approach. The DSP approach is second. Because of the FPGA's static-RAM-based block structure, it cannot compete with either a fully custom VLSI circuit or a fixed-point DSP in terms of area and power consumption. A typical FPGA five-input logic block uses 32 5 bit static RAM cells, nine multiplexers and two registers. The SRAM cells alone account for 160 transistors, enough to implement 40 complementary two-input custom logic gates. In contrast, only 32 bits of DSP onboard memory are required to represent the RBF network input layer (two 16 bit words for the subtraction and multiply/accumulate operations).

Table 2: Comparison of different RBF network accelerators

Accelerator system	Number of processors	Mult./accum. ops per second (millions)
RBF network 50 MHz custom array	64	3200
TI TMS320C50 40 MHz fixed-pt. DSP	64	2560
RBF network 25 MHz FPGA array	64	1600
TI TMS320C30 20 MHz floating-pt. DSP	64	1600
Adaptive Solutions CNAPS-64	64	1280
HNC SNAP-64	64	302

Order is based on performance in terms of multiply/accumulate operations per second

System comparison results: Table 2 shows a performance comparison of systems using both the custom CMOS and FPGA reduced-complexity circuit implementations (with 10 bits of multiply and 20 bits of accumulate precision) with two commercially available neural network accelerators, the Adaptive Solutions CNAPS-64 and the Hecht-Nielsen SNAP-64, and with two DSP-based systems. Performance numbers for the commercial accelerators were

obtained from product data sheets. If local memory weight storage were used on the RBF networks, then the RBF network systems could support the numbers shown in Table 2. As Table 2 shows, the custom CMOS implementation is the clear winner, while the fixed-point DSP system is second.

Conclusions: Low cost neural network accelerators can be created using reduced-complexity VLSI circuits that greatly improve learning and pattern matching performance. These circuits provide the benefit of using much less area and having lower power consumption than complex 32 bit floating-point circuits, with negligible loss in the quality of results.

27 June 1995
Electronics Letters Online No: 19951113

S.S. Watkins and P.M. Chau (*UCSD, Department of Electrical and Computer Engineering, La Jolla, CA 92093, USA*)

References

1 EBERHARDT, S., DUONG, T., and THAKOOR, A.: 'Design of parallel hardware neural network systems from custom analog VLSI building block chips'. Proc. IJCNN, Washington DC, June 1989, pp. 148–155
2 LIPPMANN, R.P.: 'An overview of neural network pattern classifiers'. Proc. IEEE Neural Networks for Signal Processing Workshop, Princeton, NJ, 1991, pp. 266–275
3 WATKINS, S., CHAU, P., TAWEL, R., LAMBRIGTSEN, B., and PLUTOWSKI, M.: 'A hybrid radial basis function neurocomputer and its applications'. Proc. Neural Information Processing Systems 6 Conf., Denver, CO, 29 November - 2 December 1993, pp. 850–857
4 PLATT, J.: 'A resource-allocating neural network for function interpolation', *Neural Comput.*, 1991, **3**, (2), pp. 213–225
5 ELDREDGE, J., and HUTCHINGS, B.: 'RRANN: A hardware implementation of the backpropagation algorithm using reconfigurable FPGAs'. IEEE Int. Conf. Neural Networks, Orlando, FL, 28 June - 2 July 1994, pp. 2097–2102

Electromagnetic scattering from cylindrical arrays of infinitely long thin wires

R. Vescovo

Indexing terms: Electromagnetic wave scattering, Discrete Fourier transforms

Plane wave scattering by a cylindrical array of equally spaced infinitely long perfectly conducting thin wires is examined. The symmetry of the structure is exploited to derive a closed form expression for the currents excited on the wires, using the concept of circulant matrix and, alternatively, using a DFT (discrete Fourier transform) approach.

Introduction: The electromagnetic behaviour of arrays of parallel cylinders has been investigated by many workers [1 – 4]. In this Letter we consider N perfectly conducting infinitely long wires, each of radius a, arranged around the z-axis of a Cartesian system $O(x,y,z)$ to form an equally spaced cylindrical array of radius R. The plane containing the z-axis and the ith wire of the array forms an angle ϕ_i with the xz-plane, where $\phi_i = \alpha + 2\pi N^{-1}(i-1)$, $i = 1, 2, ..., N$, and $0 \leq \alpha < 2\pi N^{-1}$. The structure is illuminated by a plane wave whose propagation vector $\mathbf{k}$ lies in the xz-plane and forms an angle θ_0 ($-\pi/2 < \theta_0 < \pi/2$) with the x-axis. The incident electric field vector $\mathbf{E}^i$ lies in the xz-plane. A time dependence $\exp(j\omega t)$ is assumed and suppressed throughout. Under the assumption that the radius a of the wires is small compared to the wavelength ($a << \lambda$), only the z-component of the electric field will excite currents on the wires. This component is given by

$$E_z^i = E_0 \cos\theta_0 \exp[-jk(x\cos\theta_0 + z\sin\theta_0)] \quad (1)$$

where $k = 2\pi\lambda^{-1}$. Furthermore, the current excited on the ith wire has the form $I'_n \exp(-jkz \sin\theta_0)$, where I'_n is to be determined [1]. In the thin wire approximation, we neglect other current contributions, which produce fields close to the wires. For this reason, we assume that the distance between adjacent wires is at least several wire radii [1].

An exact expression for the currents excited on the wires was derived by Wilson [2], expanding the incident field in terms of cylindrical waves, then determining the currents induced by each cylindrical mode and summing over all modes. This procedure leads to each current being expressed as an infinite sum. In the following, we express each current exactly as a finite sum, following two different procedures.

Closed form expression for currents: In cylindrical co-ordinates the components E_ρ^s and E_z^s of the electric field at a point $P(\rho, \phi, z)$ due to the currents $I_n' \exp(-jkz \sin\theta_0)$ on the wires is given by [2]

$$E_\rho^s = -jE_0 \sin\theta_0 \exp(-jkz\sin\theta_0) \sum_{n=1}^{N} I_n H_1^{(2)}(k\rho_n \cos\theta_0) \quad (2)$$

$$E_z^s = -E_0 \cos\theta_0 \exp(-jkz\sin\theta_0) \sum_{n=1}^{N} I_n H_0^{(2)}(k\rho_n \cos\theta_0) \quad (3)$$

where

$$I_n = I'_n \frac{\mu_0 \omega \cos\theta_0}{4E_0} \quad (4)$$

In eqns. 2 and 3, ρ_n is the distance between the nth wire and the observation point, that is, $\rho_n^2 = \rho^2 + R^2 - 2\rho R\cos(\phi_n - \phi)$, while $H_0^{(2)}$ and $H_1^{(2)}$ are Hankel functions. In eqn. 4, μ_0 is the permeability of free space, and I_n is a dimensionless current. The z-component of the total electric field is $E_z = E_z^i + E_z^s$ where E_z^i and E_z^s are given by eqns. 1 and 3, respectively. Imposing the condition of vanishing E_z at the centre of each wire, we obtain the following linear system for the currents I_n:

$$\sum_{n=1}^{N} I_n H_0^{(2)}(k\rho_{mn}\cos\theta_0) = b_m \qquad m = 1, 2, ..., N \quad (5)$$

where $b_m = \exp(-jkx_m\cos\theta_0)$ and $x_m = R\cos(\alpha + 2\pi N^{-1}(m-1))$, while ρ_{mn} is the distance between the axes of the wires m and n if $m \neq n$, and $\rho_m = a$. Eqn. 5 can be expressed in the matrix form $Ax = b$, where $x = [I_1, ..., I_N]^T$ (T denotes transposition), $A = [a_{mn}]$ with $a_{mn} = H_0^{(2)}(k\rho_{mn}\cos\theta_0)$, and $b = [b_1, ..., b_N]^T$. We observe that a_{mn} depends on $n-m$, that is, $a_{mn} = c_{n-m}$. Furthermore, $c_p = c_{p+N}$. Therefore, the matrix A is circulant. Hence, the solution $x = A^{-1}b$ of the above equation can be obtained using the formulas reported in [5] for the inversion of a circulant matrix. After some manipulations we obtain

$$I_n = \sum_{p=1}^{N} I^p \exp(j2\pi N^{-1}(p-1)(n-1)) \quad (6)$$

where

$$I^p = N^{-1} \frac{\sum_{q=1}^{N} \exp(-jkx_q\cos\theta_0) \exp(-j2\pi N^{-1}(p-1)(q-1))}{\sum_{q=1}^{N} H_0^{(2)}(k\rho_{q1}\cos\theta_0) \exp(j2\pi N^{-1}(p-1)(q-1))} \quad (7)$$

Eqns. 6 and 7 are next derived following an alternative procedure. We introduce the DFT-transformed currents $I^1, I^2, ..., I^N$, related to $I_1, I_2, ..., I_N$ by eqn. 6. Substituting into eqn. 5, we obtain

$$\sum_{p=1}^{N} T_{mp} I^p = b_m \qquad m = 1, 2, ..., N \quad (8)$$

where

$$T_{mp} = \sum_{n=1}^{N} H_0^{(2)}(k\rho_{mn}\cos\theta_0) \exp(j2\pi N^{-1}(p-1)(n-1)) \quad (9)$$

It is easy to verify that $T_{mp} = T_{1p}\exp(j2\pi N^{-1}(p-1)(m-1))$. Substituting the latter equation into eqn. 8 and expressing the coefficients b_m in terms of the corresponding DFT-transformed coefficients $b^1, b^2, ..., b^N$, we obtain, for each $m = 1, 2, ..., N$: