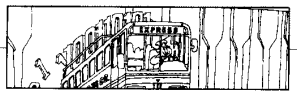


Hardware Requirements for Neural Network Pattern Classifiers

A Case Study and Implementation

A special-purpose chip, optimized for computational needs of neural networks, performs over 2,000 multiplications and additions simultaneously. Its data path is suitable particularly for the convolutional architectures typical in pattern classification networks but can also be configured for fully connected or feedback topologies. A development system permits rapid prototyping of new applications and analysis of the impact of the specialized hardware on system performance. We demonstrate the power and flexibility of the processor with a neural network for handwritten character recognition containing over 133,000 connections.

Bernhard E. Boser

Eduard Sackinger

Jane Bromley

Yann leCun

Lawrence D. Jackel

AT&T Bell Laboratories

Neural networks are rapidly gaining acceptance as powerful and versatile tools for pattern classification.[1,2] However, the widespread use of neural network classifiers remains contingent on the availability of powerful hardware to provide adequate speed. Such hardware is particularly important as the computational requirements of neural network algorithms are quite different from the high-precision processing for which general-purpose computers are optimized. Typical neural network problems involve huge amounts of low-resolution and possibly redundant data and require a correspondingly high number of low-precision arithmetic operations to be performed.

Special-purpose VLSI (very large-scale integration) processors let us overcome neural network implementation problems. With their regular structure and the small number of well-defined arithmetic operations, these networks are well matched to integrated circuit technology. The high density of modern technologies lets us implement a large number of identical, concurrently operating processors on one chip, thus exploiting the inherent parallelism of neural networks. The regularity of neural networks and the small number of well-defined arithmetic operations used by neural algorithms greatly simplify the design and layout of VLSI circuits.

But processing speed is not the only constraint on neural network hardware design; neural network classifiers benefit from a highly structured topology with local receptive fields. Of particular importance are convolutional architectures in which neurons with identical weights process different parts of the input or internal state. This topology builds into the network knowledge about locality of data, improving recognition performance. At the same time it lets us multiplex neurons with identical sizes and realize the large networks required for difficult classification tasks within the density limitations of current VLSI technology. The high speed of VLSI technology—five orders of magnitude greater than that of

0272-1732/92/0200-0032$03.00 © 1992 IEEE

natural neurons—compensates for the loss of computing speed resulting from partial serial processing in such an implementation.

Matching the arithmetic precision of the hardware to the requirements of neural networks is crucial for an efficient hardware implementation. Neural networks are often quoted for their low-resolution requirement, which lends itself well to analog implementation. Experiments, however, show that the precision requirements of neurons within a single network vary. Specifically, research indicates that higher accuracy is often needed in the output layer, for example, for selective rejection of ambiguous or otherwise unclassifiable patterns. This situation can be handled with a hybrid architecture, which evaluates the bulk of the network with low-resolution analog hardware but implements selected connections on a digital processor with higher accuracy.

Based on these considerations, we designed and fabricated ANNA (Artificial Neural Network ALU), a special-purpose chip for neural network pattern classification.[3] The chip features 4,096 individual synapses that can be multiplexed to implement networks with several hundred thousand connections. We can program the number of synapses per neuron to values between 16 and 256; the number of neurons varies accordingly between 256 and 16. Implementable network topologies include fully connected architectures with or without feedback, local receptive fields, and TDNNs (time-delay neural networks) or higher order convolutional connections. Depending on the network topology, the chip can sustain a performance of up to 5×10^9 connections per second (C/s). Arithmetic operations take place with 6-bit resolution on the weights and 3-bit resolution on the states.

Because of the high speed, parallelism, low resolution, and other characteristics of ANNA, which differ considerably from those of general-purpose computers, this hardware must be readily accessible to the network designer, so new applications can be prototyped easily. A development system, consisting of a PC or VME board containing an ANNA chip and a digital signal processor (DSP), addresses these needs. We download the topology and weights of a network into the VME board, and the DSP issues control commands to ANNA. The DSP also preprocesses and trains the networks and processes operations that require higher precision than that of ANNA.

We selected an optical character recognition neural network to test and demonstrate the flexibility and power of the neural network chip.[1] This network identifies handwritten digits from a 20×20-pixel input image and employs neurons with local receptive fields as well as a fully connected layer. The network with over 133,000 connections fits on one ANNA and is evaluated at a rate in excess of 1,000 characters per second, which constitutes a speedup of two orders of magnitude over a DSP-based implementation. Despite the low resolution of the chip, the error rates of the neural network processor and DSP implementation are very similar at 5.3 percent and 4.9 percent. For comparison, the measured human performance on the same database is 2.5 percent errors.

Our OCR neural network with 133,000 connections identifies handwritten digits at 1,000 cps and fits on one ANNA chip.

Neural network hardware

Artificial neural networks that solve difficult problems in areas such as speech recognition and synthesis, or pattern classification, consist of thousands of neurons with tens or hundreds of inputs each. Every neuron computes a weighted sum of its inputs and applies a nonlinear function to its result. Architectural parameters, such as the number of inputs per neuron, and each neuron's connectivity vary considerably within a network, and from application to application. A special-purpose neural network processor must be flexible and powerful enough to accommodate a wide range of applications. At the same time, the requirements must be carefully balanced and the special nature of the task exploited to bring an efficient implementation within reach of today's technology.

We can distinguish two phases of operation in many neural network applications. During the learning phase, the topology and weights of the network are determined from a labeled set of examples using a rule such as backpropagation,[4] or a network-growing algorithm.[5] In the subsequent retrieval or classification phase, the network parameters are fixed.

The network recognizes patterns based on information stored in the architecture and weights during training. Since the computational and infrastructure requirements (training database) during the learning phase are considerably more complex than those for classification, efficiency considerations call for separate hardware for learning and retrieval. Network parameters determined during learning are downloaded into processors specialized for the classification task. This approach, which we focus on here, contrasts with implementations of neural network processors with on-chip learning.[6,7] Those circuits are not suitable for the pattern recognition problems we investigate here, because of limitations of the training algorithms implemented on these chips or because of the limited size of the network that can be trained.

The basic operation performed by a neuron during classi-

fication is a weighted sum, followed by a nonlinear squashing function f, typically a hyperbolic tangent or approximation thereof:

$$y = f(\Sigma_i x_i w_i + b).$$

We generally refer to the inputs x_i of the neuron as connections and the w_i parameters as weights. Each input is either tied to the output y of another neuron or to an external input. Optionally, a bias b may be added to the weighted sum.

The total number of connections in neural networks for applications such as handwritten character recognition may amount to 10,000 to several hundred thousands.[8] Networks that solve more general problems, such as recognition of entire words instead of isolated characters, require even larger numbers of connections. The speed requirements of typical applications call for a few tens to several thousands of classifications per second. For each classification, the network must evaluate one multiplication and one addition for every connection, which translates to a few billion multiply-add operations per second. Only parallel implementations, in which several connections are evaluated concurrently, achieve such computational power.

ANNA evaluates dot products of state and weight vectors and applies a nonlinear squashing function to results.

The most general network topology permits connections between any two neurons. Such a high degree of (possible) connectivity, combined with the need for parallel processing, results in enormous hardware requirements, and therefore calls for a compromise. Usually, the neurons in a network are arranged in layers, each of which receives inputs only from neurons in the previous layer. Layers may be fully connected; that is, each neuron may be connected to every neuron in the preceding layer. Often, however, we use local connectivity to express knowledge about the problem (geometric relations such as the neighborhood of pixels in an image) in the network architecture and thus improve the recognition performance.[1]

For example, the fact that some pixels in an image are adjacent to each other can be built into the network architecture by constraining neurons to receive inputs only from neighboring pixels. In a fully connected topology, such information must be derived from the training set during the learning phase, usually meeting with only partial success.

A neural network processor could be designed to implement only networks with fully connected topology. Local connectivity would then be realized by simply setting the weights of unused connections to zero. Since, in typical neural networks, the ratio of such unused connections to actual connections is easily 100, such an implementation is unacceptably inefficient. The added complexity of the hardware required to support local connectivity is no match for the millions of connections saved.

Another challenge for a compact hardware implementation of a classifier is the amount of memory needed for storing several tens or hundreds of thousands of weights. Fortunately, the weights of many neurons in important connection topologies, including time-delay or feature extraction neural networks,[1,2,9] are identical. In these architectures the connection topology corresponds to a one- or higher dimensional convolution, followed by the nonlinear squashing function, as is illustrated. We can realize such a structure with a single, time-multiplexed neuron with a corresponding saving of storage and computing devices.

We can further optimize the hardware complexity by matching the computational accuracy of the processor to the requirements of typical neural networks. Both experience and theory[10] indicate that neural network classifiers can be designed to be insensitive to low-resolution arithmetic. Experiments with character recognizers show that the recognition performance remains virtually unchanged when the inputs and outputs of the neurons are quantized to 3 bits, and the weights to approximately 5 bits. Higher resolution is required in the last layer for the rejection of ambiguous or unclassifiable patterns. Since in typical neural networks the output layer contains only a small fraction of the total number of connections, we reduce system complexity by evaluating those connections on a different processor with higher accuracy.

ANNA

Figure 1 shows the building blocks of ANNA, a neural network chip that implements the concepts just outlined. It concurrently evaluates several dot products of state and weight vectors and applies a nonlinear squashing function to the results. Data enters the chip through a shift register, which reads up to four values at a time. A file with 16 vector registers stores intermediate results when multilayer networks are evaluated.

The 64-word-wide (3 bits per word) shifter reads up to four inputs in each cycle. In this process, the current shifter contents shift left one to four word positions. The use of a shifter limits the number of pins required and supports convolutional network topologies and multiplexing of neurons with identical weights. The shifter alone handles one-dimensional convolutions, while an external data formatter; for example, a line

delay register,[11] is needed for two- or higher dimensional computations. Data loaded into the chip can be buffered temporarily in a file of 16 vector registers to reduce the required input bandwidth, to evaluate neurons with more than 64 inputs, or to store intermediate results.

Eight banks of vector multipliers perform the actual computation. Each bank consists of a latch to hold the state vector plus eight vector ALUs with 64 synapses each. A multiplexer that can be configured to combine the contributions from one to four vector multipliers connects the outputs from the vector multipliers to the neuron bodies. When the latches of several vector multiplier banks hold different data, the network evaluates neurons with up to 256 inputs. The number of neurons depends on the number of inputs: Extremes of 16 neurons with 256 inputs each, or 256 neurons with 16 inputs, as well as many intermediate arrangements are possible.[12] The topology can be rearranged on a per-instruction basis to permit evaluation of several layers of a network with different architectures on a single chip without performance penalty.

The neuron bodies first scale the output from the vector multipliers by a factor that can be set in the range 1/16 to 1/2 in eight levels to optimize the useful dynamic range of the circuit. Then the neuron bodies evaluate the squashing function and convert the result to the same 3-bit, signed magnitude representation used at the input of the chip. The weights in the vector multipliers are stored as charge packets on capacitors and must be refreshed periodically. Two on-chip digital/analog converters (DACs) update the values of two different synapses in each clock cycle for a refresh speed of 110 μs for the entire array.

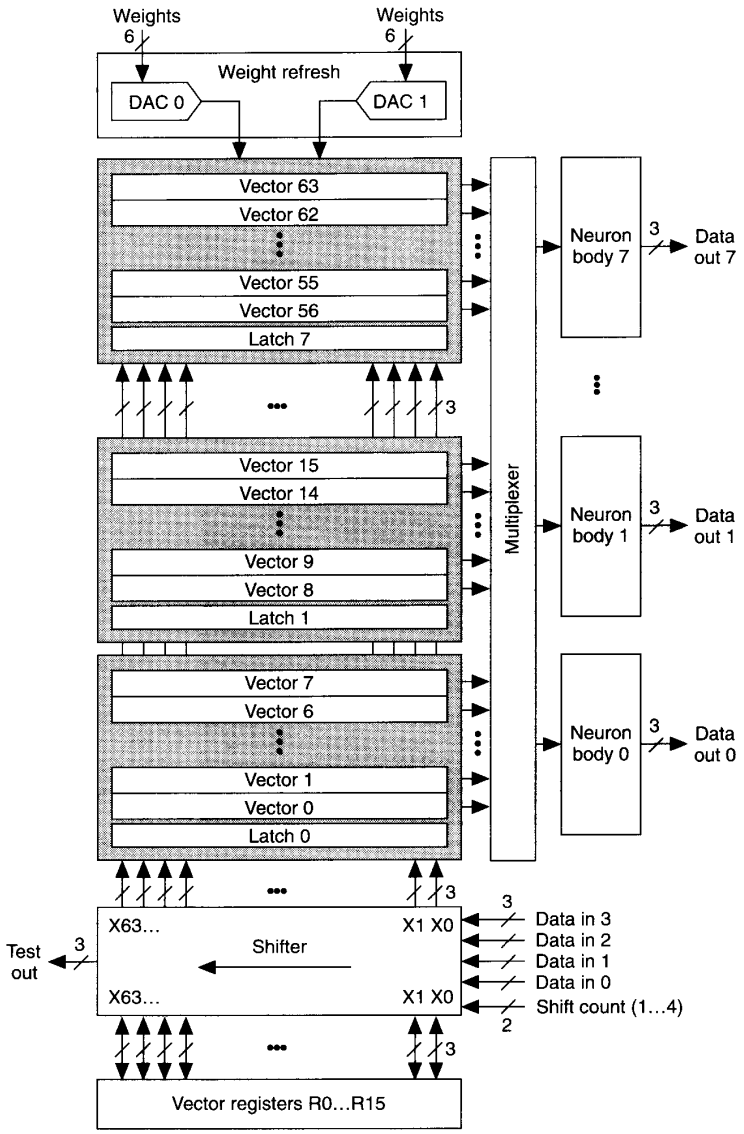


Figure 1. Block diagram of the neural network chip, ANNA.

The chip is programmed with three instructions, CALC, SHIFT, and STORE, to perform computations, load data from an external data source, and transfer data between the shifter, register file, and vector multiplier banks. Parameters for each instruction determine shift count, source of data, number of inputs per neuron, or neuron gain. The CALC instruction executes in four 50-ns cycles, and the network evaluates the other two operations in one clock cycle concurrently with an ongoing CALC instruction. In 200 ns the chip can, for example, load eight states and store them in a register and two latches, and evaluate the dot product and nonlinear function of eight vectors with 256 components each. The weight refresh takes place simultaneously transparent to

Table 1. System features.

Characteristic	Value
Synapses	4,096
Bias units	256
Synapses per neuron	16 to 256
Weight accuracy	6 bits
State accuracy	3 bits
Input rate	120 Mbps
Output rate	120 Mbps
On-chip data buffers	4.6 Kbits
Computation rate (sustained)	5 Gcps
Refresh (all weights)	110 μs
Clock rate	20 MHz

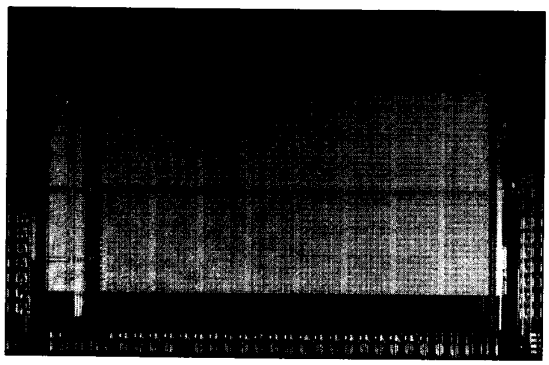


Figure 2. Die photograph. The synapse array can be seen in the center, the shifter and register file on the left, the neuron bodies at the top, and the weight refresh DACs on the right.

the user. Table 1 summarizes the features of the chip.

The chip contains 180,000 transistors and measures 4.5 × 7 mm^2 (see Figure 2). It was fabricated in single-polysilicon, double-metal, 0.9-μm CMOS technology with a 5V power supply. The current drawn by the chip reaches 250 mA when all weights are programmed to their maximum value but is less than 100 mA in typical operation.

Programmability is one of the key features of the neural network chip. Table 2 lists a selection of network topologies that can be implemented and the achieved performance in each case. The chip processes networks with full or sparse connection patterns of selectable size, as well as networks with feedback at a sustained rate of over 10^9 connections per second.

Table 2. Sample network architectures and performance.

Network topology	Average performance (GC/s)
Fully connected (one layer)	
64 inputs, 64 outputs	2.1
128 inputs, 32 outputs	1.2
32 inputs, 128 outputs	1.2
Local receptive fields	
64 × 1, 64 features	2.3
16 x16, 16 features	4.7
16 × 8, 32 features	3.6
Multilayer network	
64 inputs, 32 hidden, 32 hidden, 32 outputs	0.8
Hopfield neural network	
64 neurons	2.1

Of particular importance for neural network pattern classifiers are neurons with local receptive fields and weight sharing, such as TDNNs.[9] The neural network chip supports weight sharing in several ways. The shifter and register file enable loading of data and the computation to go on in parallel. Also, data that has been loaded onto the chip once can be buffered and reused in a later computation. Finally, rather than requiring separate hardware for all weights, neurons with identical parameters are stored only once.

Development system

As mentioned before, the characteristics of the ANNA chip—high speed, parallel computation, limited instruction set, and low resolution—differ considerably from those of general-purpose computers. Efficient algorithms that derive optimal benefit from the special processor can be designed only if the processor is available in the early design stages. A development system consisting of an ANNA, a workstation, and appropriate software addresses this requirement. Figures 3 and 4 illustrate the hardware setup, which includes an ANNA and a 20-Mflops DSP32C with 1-Mbyte fast static RAM.

A DMA interface that directly maps the SRAM into the address space of the PC bus or VMEbus exchanges data with the host computer. The DSP, which is also used for pre- and postprocessing and for computations that require higher precision than that of ANNA, generates instructions for ANNA. The entire system is controlled by a program running on the workstation that calls routines and exchanges data with the DSP transparently to the user. The software for the system is written in the high-level language C++, with the exception of a few time-critical routines that are handcoded in DSP assem-

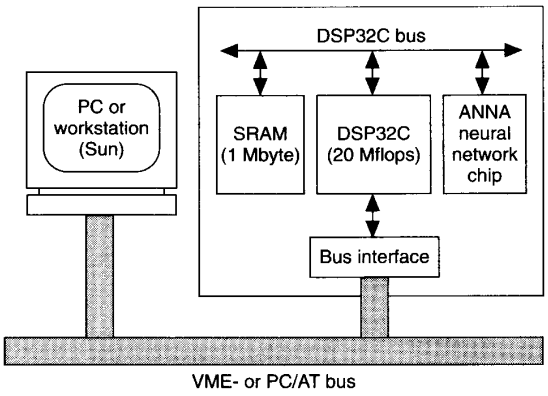


Figure 3. Block diagram of the neural network accelerator board.

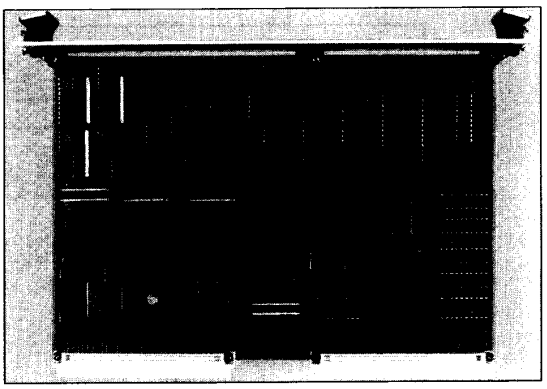


Figure 4. Neural network accelerator with ANNA and the DSP32C.

bly language.

Networks are trained on the workstation or the DSP. The neural network chip can also be included in the training process, for example, to adjust the network to ANNA's low-resolution processing. Training of individual chips is not necessary, however, because of the good matching between individual devices. Once trained, the network topology and weight values are downloaded into the DSP for execution on ANNA.

Character recognition

Speed, capacity, and programmability are important aspects of neural network hardware. Their practical relevance, however, must be proven on a real-world application, such as the implementation of an optical digit recognizer[1] on the neural network chip we describe here. This network has been trained with the backpropagation algorithm[4] to recognize handwritten digits from a 20 × 20-pixel image. The classification error rate on a test set consisting of 2,000 handwritten digits is 4.9 percent miss classifications, compared to a human performance of 2.5 percent on the same data.

Figure 5 illustrates the architecture of the network; Table 3 lists statistical information about each layer. The more than 3,500 neurons with a total of over 133,000 connections are arranged in five layers. The first four layers employ a 2D convolutional topology with various kernel sizes and subsampling factors. Because of weight sharing, the number of weights (free parameters) in these layers is much smaller than the number of connections. The last layer is fully connected. We chose this topology to maximize recognition performance and classification speed of an implementation on a floating-point DSP32C digital signal processor.[13]

Special steps are necessary to adapt the network to the low resolution of the chip. Simple quantization of all weight values

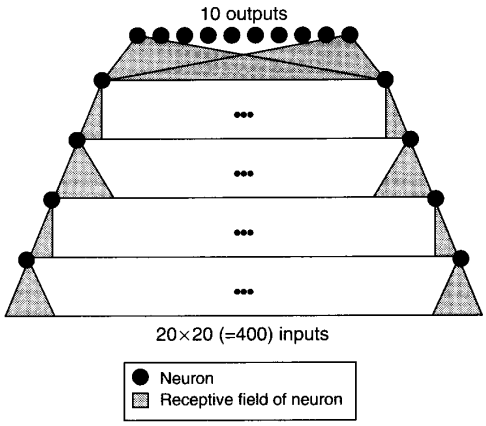


Figure 5. Architecture of the character recognition network.

Table 3. Connectivity of character recognizer neural network.

Layer	Neurons	C/s	Weights
5	10	3,000	3,000
4	300	1,200	12
3	1,200	50,000	500
2	784	3,136	4
1	3,136	78,400	100

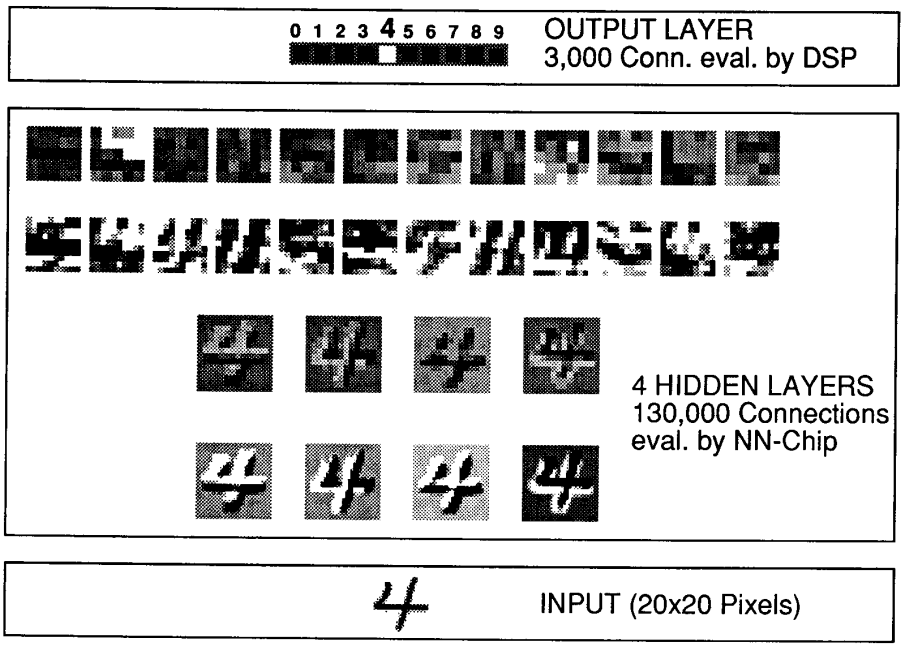


Figure 6. Sample chip output for optical character recognition. The gray levels encode the neuron state.

results in an unacceptable loss of accuracy. However, experiments reveal that the computational accuracy provided by the chip is adequate for all but the 3,000 weights in the last layer of the network. This last layer is retrained with quantized data obtained from the chip to eliminate performance degradation. After retraining, the classification error rate on the test set is 5.3 percent, compared to the original 4.9 percent. This result is obtained consistently with different chips for which the last layer has not been retrained individually. Figure 6 shows the input, output, and internal states of the neural network for a sample input that has been processed by the neural network chip.

The first four layers of the network with 97 percent of the connections but only 616 weights fit on a single neural network chip. The remaining 3,000 connections of the last layer are evaluated on the DSP32C. The throughput of the chip is more than 1,000 characters per second or 130,000 connections per second. This figure is considerably lower than the peak performance of the chip (5G connections per second), a consequence of the small number of inputs of most neurons in the network for which the chip cannot fully exploit its parallelism. Nevertheless, the chip's performance compares favorably to the 20 characters per second that are achieved when the entire network is evaluated on the DSP32C. The recognition rate of the chip is far higher than the throughput of the preprocessor, which relies on conventional hardware. Improvements of both the recognition rate and accuracy can be expected when the network architecture is tuned to take full advantage of the parallelism of the ANNA chip.

NEURAL NETWORKS ARE ATTRACTIVE FOR PATTERN classification applications but suffer in practice from the limited speed that can be achieved with implementations based on classical processors. This problem can be overcome with highly parallel special-purpose VLSI circuits. While a fully parallel implementation of sufficiently large networks is currently not feasible, we can achieve adequately high performance with an architecture that exploits the limited connectivity and weight sharing that are typical for pattern classifiers. We demonstrated this performance with a neural network classifier with over 133,000 connections that has been implemented on a single

neural network chip performing over 1,000 classifications per second. This result eliminates throughput from the constraints faced by network designers. The availability of fast special-purpose hardware for large applications sets the conditions to explore new neural network algorithms and problems of a scale that would not be feasible with conventional processors.

We expect further advances when the architecture of the network is modified to fully take advantage of the chip's parallelism. While the size of the current network has been constrained by the speed of conventional hardware, such issues vanish because of the high speed of the chip. The price for this throughput is the specialization of the circuit, specifically its low resolution, and its focus on neural network algorithms. Future research will benefit from the speed of the novel hardware but must also address questions regarding the limitations of special-purpose hardware. Furthermore, it appears attractive to implement larger tasks (to include image location, segmentation, and scaling into the recognition process) with neural networks to benefit from the powerful hardware.

References

1. Y. leCun et al., "Handwritten Digit Recognition with a Back-Propagation Network," in *Neural Information Processing Systems,* Vol. 2, D.S. Touretzky, ed., Morgan Kaufmann Publishers, San Mateo, Calif., 1990.
2. I. Guyon et al., "Design of a Neural Network Character Recognizer for a Touch Terminal," *Pattern Recognition,* Vol. 24, No. 2, 1991, pp. 105-119.
3. B.E. Boser and E. Sackinger, "An Analog Neural Network Processor with Programmable Network Topology," *ISSCC Dig. Tech. Papers, Proc. IEEE Int'l Solid-State Circuits Conf.,* Vol. 34, Feb. 1991, pp. 184-185.
4. D.E. Rumelhart and J.L. McClelland, *Parallel Distributed Processing—Explorations in the Microstructure of Cognition,* Vol. 1, MIT Press, Cambridge, Mass., 1986.
5. M. Mexard and J.P. Nadal, "Learning in Feed-Forward Layered Networks: The Tiling Algorithm," *J. Phys.,* Vol. A-22, 1989, pp. 2191-2203.
6. Y. Arima et al., "A 336-Neuron, 28K-Synapse, Self-Learning Neural Network Chip with Branch-Neuron Architecture," *ISSCC Dig. Tech. Papers, Proc. IEEE Int'l Solid-State Circuits Conf.,* 1991, pp. 182-183.
7. J. Alspector, R. Allen, and A. Jayakumar, "Relaxation Networks for Large Supervised Learning Problems," *Neural Information Processing Systems,* Vol. 3, R. Lippmann, J. Moody, and D. Touretzky, eds., Morgan Kaufmann Publishers, 1991, pp. 1015-1021.
8. L.D. Jackel et al., "VLSI Implementations of Electronic Neural Networks: An Example in Character Recognition," *Proc. IEEE Int'l Conf. Systems, Man, and Cybernetics,* 1990, pp. 320-322.
9. A. Waibel et al., "Phoneme Recognition Using Time-Delay Neural Networks," *IEEE Trans. Acoustics, Speech, and Signal Processing,* Vol. 37, No. 3, Mar. 1989, pp. 328-339.
10. M. Stevenson, R. Winter, and B. Widrow, "Sensitivity of Feedforward Neural Networks to Weight Errors," *IEEE Trans. Neural Networks,* Vol. 1, No. 1, Mar. 1990, pp. 71-80.
11. P.A. Ruetz, "The Architectures and Design of a 20-MHz Real-Time DSP Chip Set," *IEEE J. Solid-State Circuits,* Vol 24, No. 2, Apr. 1989, pp. 338-348.
12. H.P. Graf and D. Henderson, "A Reconfigurable CMOS Neural Network," *ISSCC Dig. Tech. Papers, Proc. IEEE Int'l Solid-State Circuits Conf.,* Vol. 33, 1990.
13. M.L. Fuccio et al., "The DSP32C: AT&T's Second-Generation Floating-Point Digital Signal Processor," *IEEE Micro,* Vol. 8, No. 6, Dec. 1988, pp. 30-48.

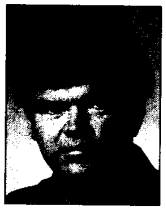
Bernhard E. Boser is an assistant professor in electrical engineering at the University of California, Berkeley. While writing this article, he was working on algorithms and parallel hardware for artificial neural networks at AT&T Bell Laboratories.

Boser received a diploma in electrical engineering from the Swiss Federal Institute of Technology in Zurich, Switzerland, and MS and PhD degrees from Stanford University. He is a member of the IEEE and the Computer Society.

Eduard Sackinger is a member of the technical staff at AT&T Bell Laboratories in Holmdel, New Jersey, on artificial neural-network hardware and its applications to character recognition. He previously worked at the Electronics Laboratory of the Swiss Federal Institute of Technology where he investigated analog applications to floating-gate devices. His research interests include analog circuit design and parallel distributed processing.

Sackinger received the MS and PhD degrees in electrical engineering from the Swiss Federal Institute of Technology (ETH), Zurich, Switzerland. He is a member of the IEEE.

Jane Bromley, a consultant at AT&T Bell Laboratories, works on the application of artificial neural networks to human perception tasks, including the reading of handwritten text. She received the BSc degree in physics and the PhD degree in biophysics from Imperial College, London University. She is a member of the American Psychological Society and the Association for Research in Vision and Opthalmology.



Yann leCun is a member of the technical staff at AT&T Bell Laboratories. He previously worked as a research associate in the Department of Computer Science of the University of Toronto, Canada. His current interests include neural networks and connectionist models, learning theory, pattern recognition, and VLSI implementations of massively parallel systems.

LeCun obtained an engineering diploma in electrical engineering from the School of Electronic and Electrotechnic Engineering, Paris. He holds a PhD degree in computer science from the Pierre and Marie Curie University, Paris, during which time he introduced an early version of the backpropagation algorithm. He served as session chair on the organizing committees of several conferences, including the International Joint Conference on Neural Networks, the Neural Information Processing Systems Conference, and the International Neural Network Conference.

Lawrence D. Jackel heads the Adaptive Systems Research Department at AT&T Bell Laboratories in Holmdel, New Jersey, where he pursues research into theory, applications, and hardware for machine learning and machine perception. His initial research was in microfabrication and microscience.

Jackel received a BS degree in physics from Brandeis University in Waltham, Massachusetts. He was awarded the PhD degree in experimental physics from Cornell University in Ithaca, New York. He is a member of the IEEE, the Computer Society, and the American Physical Society.

Direct questions concerning this article to Bernhard E. Boser, EECS Dept., Cory Hall, University of California, Berkeley, CA 94702.