ELECTRONIC NEURAL NETWORKS

Richard E. Howard, Lawrence D. Jackel, and Hans P. Graf

We are exploring electronic implementations of fine-grained parallel computing models that are loosely drawn from models of biological neural function. Experimental custom chips that combine a new mix of analog and digital processing with standard fabrication technology have shown the feasibility of the neural network approach. Early results on transforming aspects of biological computing to electronic hardware suggest that networks of highly-interconnected, simple, low-precision processors may give us new tools for tackling problems that have been difficult for standard computers.

Richard E. Howard *is head of the Microelectronics Research Department of AT&T Bell Laboratories in Holmdel, New Jersey. He joined AT&T in 1978 and is working on electronic neural networks, high-temperature superconductors, and microscience research. Mr. Howard has a B.Sc. in physics from the California Institute of Technology and a Ph.D. in applied physics from Stanford University.* ***Lawrence D. Jackel*** *and* ***Hans P. Graf*** *are, respectively, department head and member of the technical staff in the Device Structure Research Department of Bell Laboratories in Holmdel. Mr. Jackel joined AT&T in 1975 and is responsible for investigating the potential of electronic neural network algorithms and hardware. He has a B.A. in physics from Brandeis University and a Ph.D. in experimental physics from*
(continued on page 64)

Introduction

Today's computers excel at high-precision arithmetical calculations, but they appear to be poorly matched to problems such as those encountered in machine perception where a computer must make sense out of a flood of low-accuracy, low-information-content data. Researchers have now begun to use some of the hard-won knowledge gained by biologists about the nature of the brain as a guideline for reshaping our thinking about electronic computers. Biological computers have a completely different architecture from electronic computers, giving relatively simple animals the ability to perform functions of learning, complex pattern recognition, and motor control that far exceed that possible using current semiconductor electronics in existing architectures.

One of the most basic, yet most powerful, functions of biological computers is pattern matching. Imagine, for example, a simple line drawing of the face of a cat next to a photograph of a cat. In spite of the large differences between the images, the brain can easily and quickly associate them together as representing the same general class of object. Although it is not known precisely how this is done, a key element is thought to be a rapid search through a massive database of images and features for the best match between the perceived object and a stored memory. Such a database must be enormous if it is to organize the visual world into useful categories and subcategories. Yet the search through this vast storehouse of data occurs in a fraction of a

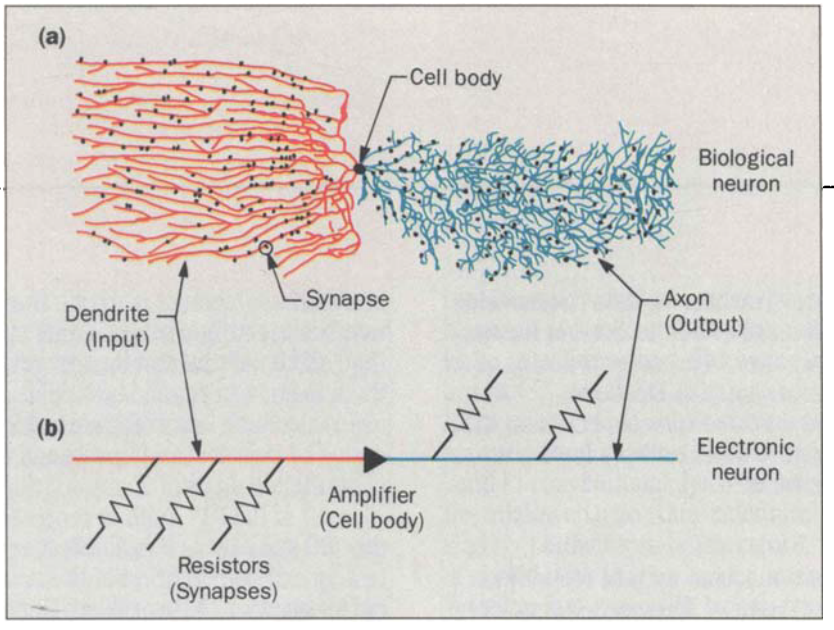


Figure 1. (a) Schematic of a biological neuron. The cell body receives input from other neurons through synaptic connections to the dendrites (red). Output signals from the cell travel along axons (green) to synaptic connections at the input structures of other neurons. (b) Electronic analog that models some of the functional aspects of a biological neuron. The electronic amplifier receives signals on its input through resistors connected to the output structures of other amplifiers. Similarly, its output is sent to other amplifiers through their input resistors.

second using biological hardware with only millisecond time constants. If we had electronic hardware to conduct a fast search for close matches, we would be on our way to duplicating a key portion of the processing necessary for pattern recognition.

A Neural Model

Figure 1(a) is a drawing of a neuron, or nerve cell, showing the cell body connected to its input and output structures. There are about 10^{11} neurons in a human brain, each with about 10^4 input and output connections. The input structure is a branching tree of fibers, called *dendrites*, that receive input from other neurons through adjustable connections called *synapses*. The output of the cell is sent along another set of fibers, called *axons*, to the synapses of other neurons. When a neuron is excited at its input synapses, it produces a train of pulses that travel along its axon. Input at excitatory synapses increases the pulse rate, while input at inhibitory synapses reduces the pulse rate. The output pulse rate depends on both the strength of the input signals and strength, or weight, of the synaptic connections. The memory and processing ability of this network are thought to lie in the pattern of these connections and their weights.

In a simple neural model, the pulse rate is determined by a sigmoid function of a weighted sum of the input signals.[1] A model of this function can be built using traditional electronic components in an *analog sum-of-products* circuit [Figure 1(b)]. In this circuit, the variable pulse rate of the neuron is replaced by the variable output voltage of the electronic amplifier and the synaptic connections are replaced by conductances connecting the input lead of the amplifier to the output leads of other amplifiers. Input voltage signals supply current into the wire dendrite in proportion to the product of the input voltage and the synapse conductance. The sigmoid transfer function is naturally provided by the saturating characteristics of the amplifier. This circuit, which can be used as a building block for both memories and processors, is called a *perceptron*[2] or *adeline*,[3] and was proposed over 20 years ago. However, building (or even simulating) large networks of perceptrons was not possible until recently. Furthermore, it has taken 20 years to develop learning rules for the layered perceptrons[4] and feedback networks[5,6] needed

to do complex tasks. (Here "learning" means determining the synaptic weights, which can either be done in the network or in an external computer.) For a discussion of other issues in neural network learning, see Denker.[7]

The networks we describe can also be made with optics; readers are referred to other authors for discussions of optical neural networks.[8]

Pattern Classification

A first step toward machine pattern recognition is deciding how to represent the data. Researchers in electronic neural networks use a long vector to represent the various attributes of the object being scrutinized. For example, in the vector representing the image of a cat face, one component might represent

is the image basically round

while another might indicate

are features in about the right place to represent ears.

In the simplest case, this vector can be binary (only "1"s and "0"s), with each representing the presence or absence of a feature in the image. In a more complex system, these could be analog representations of the strength of each feature in the image.

These features would first have to be found by a neural network preprocessor that measures the matches between different parts of the image and a set of stored features. The electronic problem then reduces to searching through a large database of binary numbers, each a "memory" of a particular object, and finding the memory that is the best match to the test image. In the binary case, the best match would be determined by counting the number of matching bits between the test vector and all memories and keeping the highest score. In a more sophisticated scheme, the matches might be weighted by, for example, counting the presence of eyes more heavily than the presence of fur.

One way to achieve this function of matching bits between two vectors is to use the `exclusive or` function between the corresponding bits of the two vectors and then add up all the matches to get a number representing the number of common bits. This same function can be used in a much less complex fashion using the analog processing of our electronic neuron in Figure 1(b) instead of the usual digital circuit.

If the "1" state is represented by a voltage, V, and the "0" state by $-V$ in the test vector, then the bitwise testing and computation of the sum can be done using the circuit shown in Figure 1(b). The bits of the stored vector are represented by resistors (synapses) connecting the amplifier input wire (dendrite) with the output wires of other amplifiers. When the voltages associated with the test vector are applied to these resistors, the resulting currents are automatically summed on the input wire giving an input signal to the amplifier proportional to the match between the two. This simple circuit can test all the bits simultaneously and, using an analog threshold, produce an output voltage indicating whether more than a specified number of bits match between the two vectors.

Each of these amplifier circuits with its associated resistors stores one memory vector, and large chips containing many of them can be used as a massively parallel associative memory (i.e., a memory in which data are recalled by a partial match to a key). The array of amplifiers and the associated matrix of synaptic conductances that interconnect them form the distributed parallel processor. In use, a test vector would be presented to all these memories in parallel, and each amplifier would simultaneously generate an output voltage proportional to the quality of the match. Using positive feedback, analogous to that used in a simple flip-flop circuit, it is easy to make the circuit select the best match in about the same processing time. Such a circuit is efficient because all electronic components on the chip are simultaneously operating at their maximum speed; nothing is waiting for anything else.

Learning

In addition to making fast, parallel searches, some electronic neural networks can learn by experience how to

organize categories and subcategories in a database.

If the elements of the synaptic matrix are adjusted to improve the match for known sets of training data, the system will be adaptable. This capability is important in areas such as speech recognition when the proper categories may not be known ahead of time. It is also important for ensuring fault tolerance in large electronic networks.

Learning rules were developed for picking the elements of the resistive matrix so that certain classes of input patterns could cause particular output patterns. The simplest such network is the "two-layer perceptron" introduced by Rosenblatt[2] in 1961. This circuit consists of a single layer of sum-of-products circuits that calculate the overlap between the test pattern and the array of stored patterns. The hope was that perceptrons would be useful for pattern recognition in which the input pattern might be a bit map of an image and the output pattern would be a code that identified a particular image. Minsky and Papert[9] showed that, unfortunately, the number of useful mappings that could be done with a two-layer perceptron is limited.

Nevertheless, even with two-layer networks of limited size, it was possible to make machines that did a credible job of complex tasks such as weather prediction or even motor control.[10] With modern electronics, we can now economically make networks hundreds or thousands of times larger than was possible in the early 1970's. Thus, it is important to reexamine this technology.

A multilayered network in which neuron layers are sandwiched between input and output layers can produce even more useful processing.[11] The first layer, for example, can be used to translate the problem into a more suitable representation so that the second layer can make the desired mapping. Recently, learning rules were developed for multilayered networks.[4] These rules compare the actual output of a network with the desired output. The synapses are adjusted to bring the two closer together.

Although no hardware has been built for this learning technique to date, the theory has been studied extensively. The problem with using these powerful learning algorithms in semiconductor hardware lies in making continuously variable synaptic conductances using only the standard processing technology that is essential for the large amplifier arrays. Recent work by Schwartz and Howard has shown the feasibility of storing charge on standard MOS (metal oxide semiconductor) capacitors to control the synaptic conductances.[12] This technique, similar to that used in conventional dynamic silicon memories, can hold the analog charge state indefinitely if the chip is cooled slightly below room temperature. Chips are now being designed with this technique as the first step in making systems that can profit by experience.

CMOS Programmable Networks

A neural network chip will have arrays of electronic neurons connected to each other through a matrix of synapses. There are two basic approaches to such a network. The first is to determine ahead of time the values of the resistors needed in the matrix. That is, do the learning in an external computer and fix the results on the chip. Because of the small size of resistors, this network can be extremely compact; densities as high as one billion "synapses" /cm^2 have been shown. (See Panel 1.) However, for many applications, it is impractical or even impossible to specify in advance the strength and location of synaptic connections from one neuron to another. Even if the learning is done externally, changing situations will require changes in stored memories. For this case, the synapses must be able to change. Numerous methods for making programmable synapses are possible; they all require much more chip area than fixed resistive synapses.[13,14]

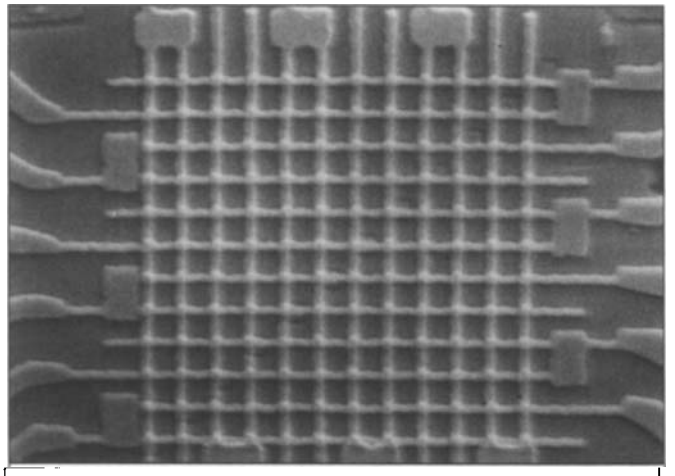
When designing a neural network, we also need to make large, reliable circuits containing tens of thousands of transistors on a single chip. This means that, at least in the initial phase, we must make maximum use of standard VLSI silicon technology. (VLSI stands for very-large-scale integration.) In our laboratory, we have used novel CMOS (complementary metal oxide semiconductor) circuitry combined with standard static RAM cells (random access memory) to make a programmable, neural network chip with 54 electronic "neurons" and about 3,000 synapses.[14] The chip contained about 75,000 transistors, was 6 mm on a side,

and was made using completely standard fabrication technology. To simplify the design (and reduce the size of the circuit) the synapses only had a limited set of values. The basic synaptic unit, shown in Figure 2, is about 100 μm on a side. The unit is designed so that when the input from the axon rises to the "1" state, transistor switches allow connection of the output dendrite to either a current source (excitation) or a current sink (inhibition). The connection is determined by the states of the two RAM cells contained in the unit:

- A 1 stored in one of the cells specifies an excitatory connection.
- A 1 stored in the other specifies an inhibitory connection.
- When 0s are stored in both cells, there is no connection between the axon and the dendrite.
- The state with both cells storing a 1 is not allowed.

Among other uses, the chip can be configured to function as an associative memory. This means that the chip determines the closest match of a test word to members of a list of stored words. Here "closest" means the most number of "1" bits in common (the "and" function). The associative memory function is often needed for pattern classification tasks such as our "cat recognizer." Although we still do not have enough information to deal with complex subjects such as cats, this chip can be used to find key features in handwritten characters. This is the first step toward building a reading machine. The chip executes the function about 1,000 times faster than a VAX™ 11/750 computer programmed to do the same task. The usefulness of the chip is somewhat limited because it can only store about 50 memory words or patterns of 50 bits each. We can solve this problem by combining many chips in a hierarchy so that the best match is done by a tree search.[15] Advances in technology may increase density by an order of magnitude on a chip; wafer scale integration should lead to even larger memories.

Although the match found by this method may not be absolute, it is still close to optimal. We showed that such a hierarchy could be used for the bandwidth-compression scheme known as vector quantization.[15]



Panel 1. Thin-film Synapse Arrays

A major component of an electronic neural network is the array of connections between neurons, the synapses. In the most general circuit, every neuron is potentially connected to every other, so for N neurons, we need an $N \times N$ array of connections. (For many neural network architectures, the connections are sparser, and a smaller array, or set of arrays, will do.) The simplest synapse array consists of fixed resistors; the "programming" that determines the synapse values must be done before the array is made. (See the article by Jones[1] in this issue, p. 65) One virtue of resistive synapses is that they can be packed very densely. The physics of resistors allows them to be made much smaller than transistors, and the number of lithographic features needed to define a resistor is far fewer than needed to define a transistor.

The photograph above shows a demonstration 12×12 synapse array made by e-beam (electron-beam) lithography with four synapses/μm^2. The entire array fits into an area about the same as that required to store a few bits on a conventional silicon RAM chip. At this density, a 1-cm^2 chip would contain 400 million such connections.

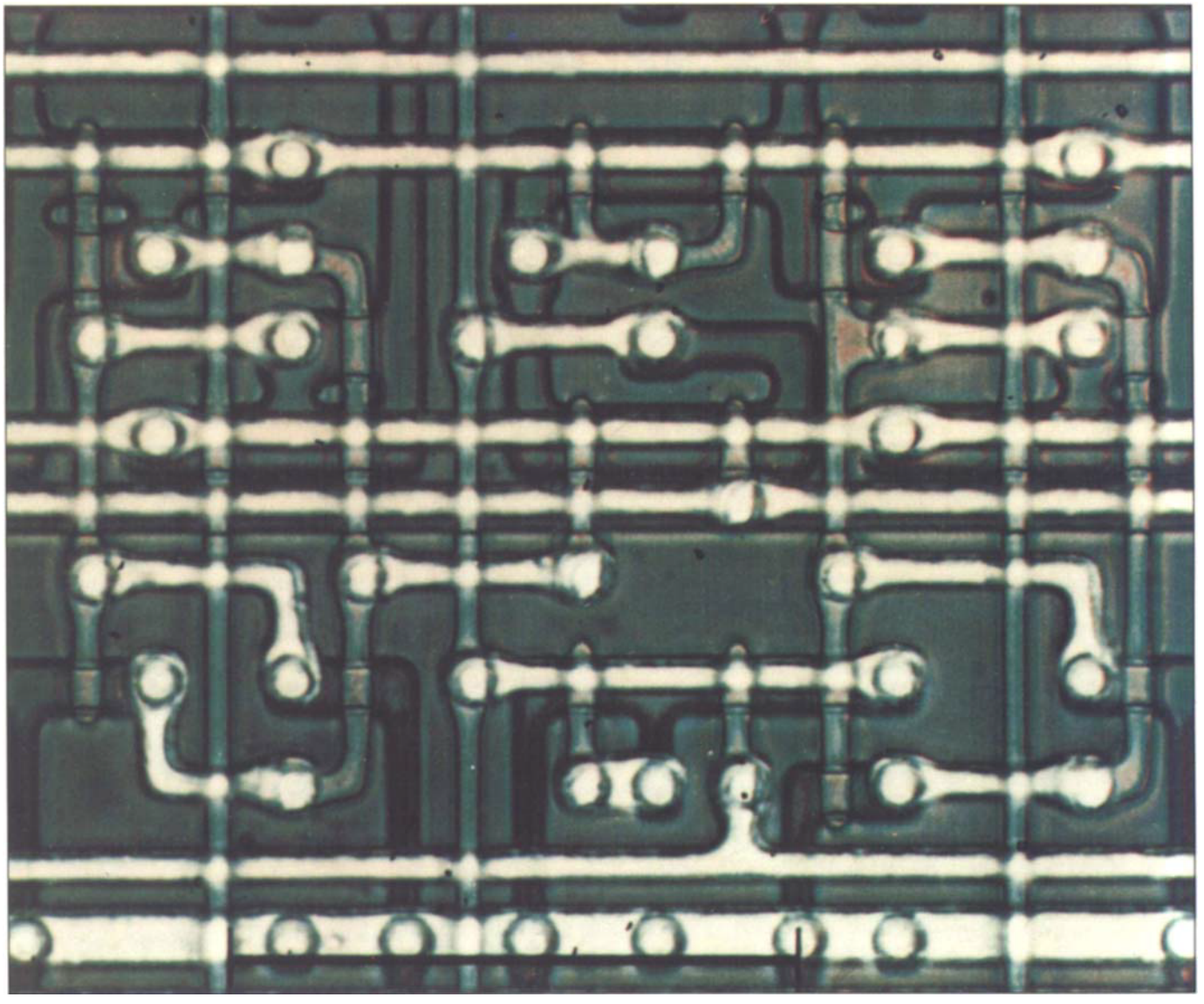


Figure 2. Micrograph of an electronic synapse fabricated using a conventional 2.5-μm silicon CMOS process.

Conclusion

Neural networks provide a new way of looking at some classes of complex problems. Special-purpose VLSI chips are now being made that implement neural network algorithms, providing a new tool for computationally difficult tasks in machine perception.

Acknowledgment

We thank the Bell Laboratories Holmdel Neural Network Group for their crucial contributions to the research described here.

References

1. M. A. Jones, "Programming Connectionist Architectures," *AT&T Technical Journal*, Vol. 67, No. 1, January/February 1988, pp. 65-68.
2. F. Rosenblatt, *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*, Spartan Press, Washington, D.C., 1961.
3. B. Widrow, "Generalization and Information Storage in Networks of Adaline 'Neurons'," in *Self Organizing Systems*, M. C. Yovits et al., eds., Spartan Press, Washington, D.C., 1962, pp. 435-461.
4. *Parallel Distributed Processing*, D. E. Rumelhart and J. L. McClelland, eds., MIT Press, Cambridge, Massachusetts, 1986.
5. J. J. Hopfield, "Human Memory, Error Correction Codes and Spin Glasses," *Proceedings of the National Academy of Sciences USA*, Vol. 79, 1982, p. 2554.
6. J. J. Hopfield, "Neurons with Graded Response have Collective Computational Properties Like those of Two State Neurons," *Proceedings of the National Academy of Sciences USA*, Vol. 81, 1984, p. 3088.
7. J. Denker et al., "Large Automatic Learning, Rule Extraction, and Generalization," *Complex Systems*, No. 1, 1987, p. 877.
8. "Neural Networks for Computing," J. S. Denker, ed., *AIP Conference Proceedings No. 151*, American Institute of Physics, New York, 1986.
9. M. Minsky and S. Papert, *Perceptrons*, MIT Press, Cambridge, Massachusetts, 1968.
10. B. Widrow, *IEEE Transactions on Applications in Industry*, September 1964, pp. 269-277.
11. R. P. Lippmann, "An Introduction to Computing with Neural Nets," *IEEE ASSP Magazine*, Vol. 4, No. 2, 1987, p. 4.
12. D. B. Schwartz and R. E. Howard, "A Programmable Analog Neural Network Chip," *Proceedings of the Custom Integrated Circuits Conference*, May 1988.
13. J. L. Lamb et al., "Resistive Synaptic Interconnects for Electronic Neural Networks," *Journal of Vacuum Science and Technology*, Vol. A5, 1987, p. 1407.
14. H. P. Graf and P. DeVegvar, "A CMOS Associative Memory Chip Based on Neural Networks," *Digest of the Technical Papers of the 1987 IEEE International Solid State Circuits Conference*, L. Winner, ed., 87CH-2367, IEEE Press, 1987, p. 304.
15. L. D. Jackel et al., "Building a Hierarchy with Neural Networks: An Example—Image Vector Quantization," *Applied Optics*, Vol. 26, No. 23, December 1, 1987.

Biographies (continued)

Cornell University. Mr. Graf joined AT&T in 1983 and is working on VLSI chips for neural network models and the application of these chips to pattern recognition problems. He has a diploma in physics and a Ph.D. in physics, both from the Swiss Federal Institute of Technology, Zurich.

(Manuscript received October 5, 1987)