Application of the ANNA Neural Network Chip to High-Speed Character Recognition

Eduard Säckinger, *Member, IEEE,* Bernhard E. Boser, *Member, IEEE,* Jane Bromley, Yann LeCun, *Member, IEEE,* and Lawrence D. Jackel, *Senior Member, IEEE*

Abstract— **A neural network with 136 000 connections for recognition of handwritten digits has been implemented using a mixed analog/digital neural network ship. The neural network chip is capable of processing 1000 characters per second. The recognition system has essentially the same error rate (5%) as a simulation of the network with 32-bit floating-point precision.**

I. Introduction

We have designed, fabricated, and tested a reconfigurable neural network chip, called the ANNA chip (for analog neural network arithmetic and logic unit). The chip is optimized for locally connected, weight-sharing networks and time-delay neural networks (TDNN's), but can also be used for a wide variety of other network architectures, such as fully connected and recurrent networks. Synaptic weights are learned off-chip, quantized to the chip's resolution, and then down-loaded into the chip's weight memory. A detailed description of the chip has been published in [1] and [2].

Locally connected, weight-sharing neural networks have a wide range of application. They have been successfully used for optical character recognition (OCR) of both digits and letters [3], character recognition from touch screens [4], image segmentation [5], and speech recognition [6]. LeCun *et al.* [3] have described a neural network for optical recognition of digits. The network performs particularly well in recognizing noisy, handwritten characters and in estimating the confidence of the classification. The high performance is due to a technically advanced architecture featuring local connections and weight sharing. This architecture is well suited for implementation on the ANNA chip. An extended version of the network described in [3] has been chosen as an application example for the ANNA chip and is described in this paper.

Many neural network chips have been developed to date, but demonstration of large real-world applications for them is often neglected. The implementation of the OCR network with a total of 136 000 connections using one ANNA chip clearly shows its practical significance.

Important questions, such as the speed advantage gained through this special-purpose hardware and the impact of low-resolution arithmetic on the classification accuracy, are discussed in this paper.

Manuscript received July 17, 1991; revised October 22, 1991. This work was supported in part by the U.S. Army SDC.

The authors are with AT&T Bell Laboratories, Holmdel, NJ 07733.

IEEE Log Number 9105810.

II. The ANNA Neural Network Chip Architecture

The ANNA neural network chip, implemented in a 0.9 μm CMOS technology, contains 180 000 transistors on a 4.5 × 7 mm^2 die (see Fig. 1). The chip implements 4096 *physical* synapses, which can be time multiplexed in order to realize networks with many more than 4096 connections. The resolution of the synaptic weights is 6 bits and that of the states (input/output of the neurons) is 3 bits. Additionally, a 4-bit scaling factor can be programmed for each neuron to extend the dynamic range of the weights. The weight values are stored as charge packets on capacitors and are periodically refreshed by two on-chip 6-bit D/A converters. The synapses are realized by multiplying 3-bit D/A converters (analog weight times digital state). The analog results of this multiplication are added by means of current summing and then converted back to digital by a saturating 3-bit A/D converter. Although the chip uses analog computing internally, all input/output is digital. This combines the advantages of high synaptic density, high speed, low power of analog, and easy interfacing to a digital system such as a digital signal processor (DSP).

In the following, a simplified discussion of the chip's architecture is presented (see Fig. 2). This description leaves out many details but is sufficient to describe the implementation of the network on the chip. More detailed descriptions can be found in [1] and [2].

The chip evaluates eight dot products of state vector $\boldsymbol{x}$ and weight vectors $\boldsymbol{w_i}$ in parallel. The state vector is supplied by a barrel shifter and the eight weight vectors are selected from a large (4096) on-chip weight memory by a multiplexer. The eight scalar results $\boldsymbol{w_i} \cdot \boldsymbol{x}$ are passed through a squashing function $f(\cdot)$ yielding eight scalar neuron outputs z_i. The whole neuron-function evaluation process takes 200 ns, or four clock cycles. The chip can be reconfigured for weight and state vector sizes of 64, 128, and 256. These figures also correspond to the number of synapses per neuron.

The input state vector $\boldsymbol{x}$, is provided by a shift register which can be shifted by one, two, three, or four positions in two clock cycles (100 ns). Correspondingly, one, two, three, or four new data values are read into the left end of the shift register. This barrel shifter serves two purposes: (i) Because of pin limitations, it is not possible to load the whole state vector (up to 256 × 3 bits) in parallel onto the chip; therefore sequential loading is imperative. (ii) A barrel shifter is the ideal preprocessor for networks with local receptive fields and weight sharing (convolutional networks) as well as time-delay neural networks (the reason for this will be clarified in the

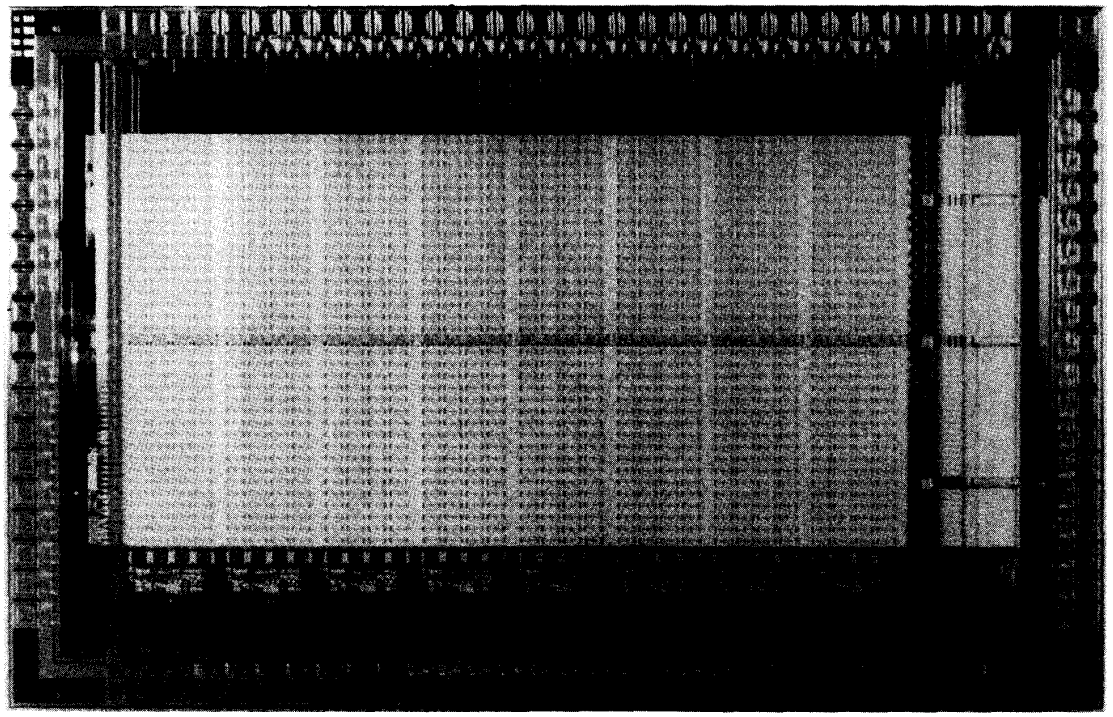


Fig. 1. Micrograph of the ANNA chip.

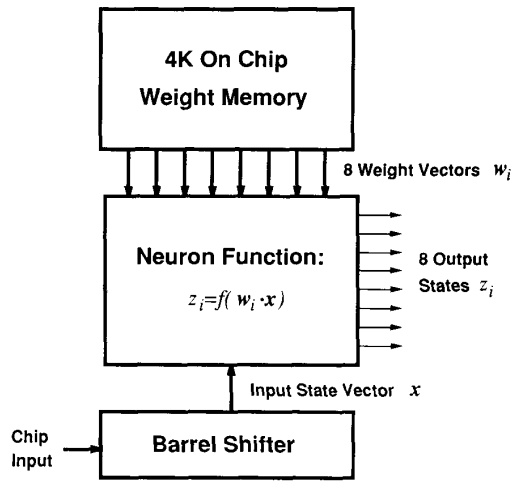


Fig. 2. Simplified architecture of the ANNA chip. The input state vector, which can be of size 64, 128, or 256 (×3 bit), is loaded into the barrel shifter. This vector is multiplied with eight weight vectors of the same size selected from the weight memory. The result of the dot product is passed through a squashing function (neuron function) which appears at the output of the chip.

network implementation section). The barrel shifter on the chip has length 64, but can be extended to larger sizes by means of an associated vector-register file. The barrel shifter can be operated in parallel to the neuron-function unit, such that the new state vector is available as soon as a new calculation cycle starts.

A total of 4096 analog weight values are stored on the chip. These values can be grouped into vectors of size 64, 128, and 256 in a flexible way. For instance, it is possible to have simultaneously 32 weight vectors of size 64, eight vectors of size 128, and four vectors of size 256 on the same chip. Of the many weight vectors stored on chip, eight $(\boldsymbol{w}_1, \cdots, \boldsymbol{w}_8)$ are selected in each calculation cycle and multiplied with the content of the barrel shifter $(\boldsymbol{x})$.

If all neurons are configured for the maximum size of 256 synapses, the chip can evaluate a maximum of $10 \cdot 10^9$ connections per second (10 GC/s), corresponding to 8 neurons × 256 synapses/200 ns ≈10 GC/s. In practical applications the speed may be lower for the following reasons: (i) the application does not make full use of the chips parallelism, i.e., the neurons have fewer than 256 synapses or fewer than eight neurons use the same input data, and (ii) the neuron-function unit has to wait for the barrel shifter to prepare the state vector for the next calculation. For example a network with a 16 × 16 local receptive field, weight sharing, and 16 features performs at a rate of 5 GC/s.

For practical use, the chip has to be integrated into a system. The hardware around the chip has to perform three major tasks: (i) supply and store the state data to and from the chip (memory controller), (ii) generate microcode words corresponding to the network topology to be evaluated (sequencer), and (iii) refresh the dynamic on-chip weight storage (refresh controller). A VME board (see Fig. 3) has been built that contains the ANNA chip and, for maximum flexibility, a digital signal processor (DSP32C). On the one hand the

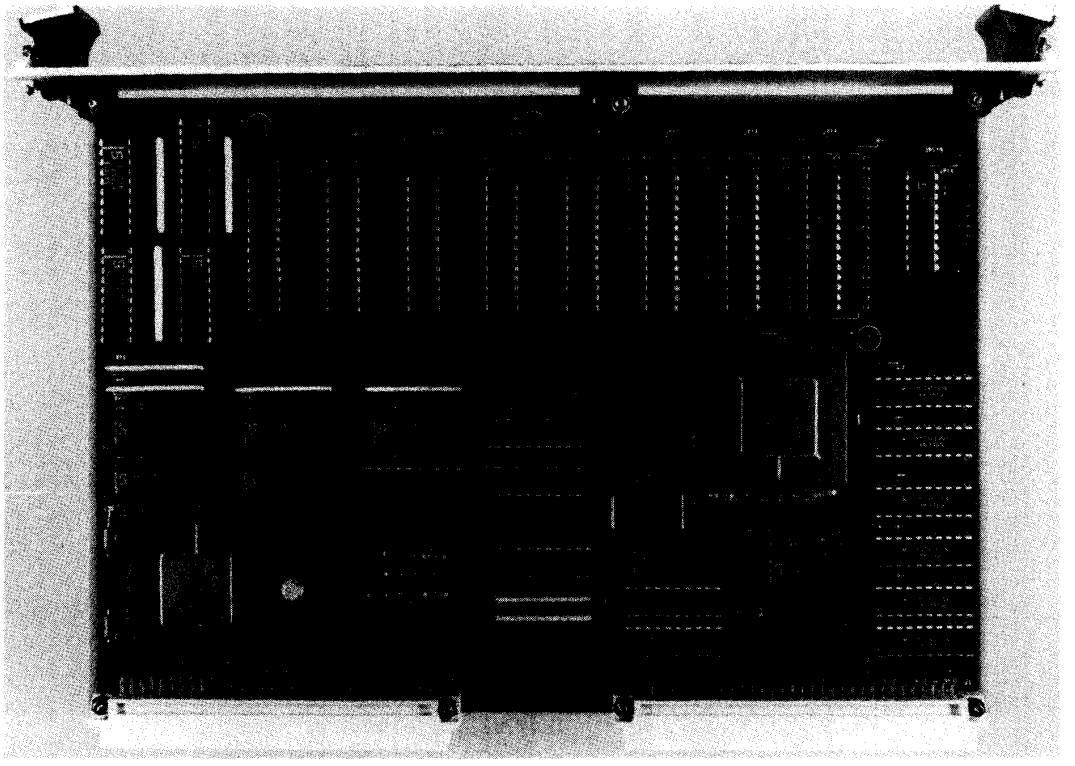
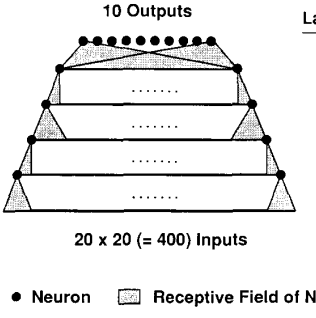


Fig. 3. VME board containing the ANNA chip and a DSP32C digital signal processor.

DSP acts as a memory controller, sequencer, and refresh controller for the ANNA chip, while on the other it can be used to preprocess the input to the neural network (e.g. size normalization), postprocess the results from the network (e.g. by implementing a classification algorithm), and run learning algorithms (e.g. back-propagation). In this way a complete task can be performed on the board with a minimum of data exchange with the host.

III. Optical Character Recognition (OCR) Network

The general structure of the OCR digit recognizer is a five-layer feedforward network (see Fig. 4). Since there is no feedback the network can be evaluated in a single pass. The net has 400 inputs, corresponding directly to the 20 × 20 pixel image; i.e., no preprocessing, such as feature extraction, is done. The ten outputs of the network code the ten digits in a "1 out of 10" code. Since the outputs of the neurons are real valued (as opposed to thresholded outputs), the output contains information not only about the classification result (the most active output), but also about the *confidence* of this classification. Actually, the difference between the most active and the second most active neuron is an accurate measure of the confidence which can be used to reject ambiguous digits. An example showing all the states of the network for the case of a handwritten 6 is shown in Fig. 5. The states of the input and the first four layers are represented as gray levels; the



Layer	Neurons	Synapses
5	10	3,000
4	300	1,200
3	1,200	50,000
2	784	3,136
1	3,136	78,400

Fig. 4. General structure of the OCR network.

states of the output layer are proportional to the size of the black (negative) and the white (positive) squares.

Of the five layers of the network, only the last layer is fully connected, with all weights being independent. The first four layers are carefully constrained to improve the capability of the network to generalize well for patterns the network has not been trained on [3]. These constraints are symbolized by the local receptive fields shown in Fig. 4 and are discussed in greater detail below.

Each neuron in the first layer has 25 inputs and connects to a 5 × 5 neighborhood in the pixel image (see Fig. 6). This neighborhood is called the receptive field of the neuron. Two adjacent neurons, belonging to the same feature map, have local receptive fields that are displaced by *one* pixel. The

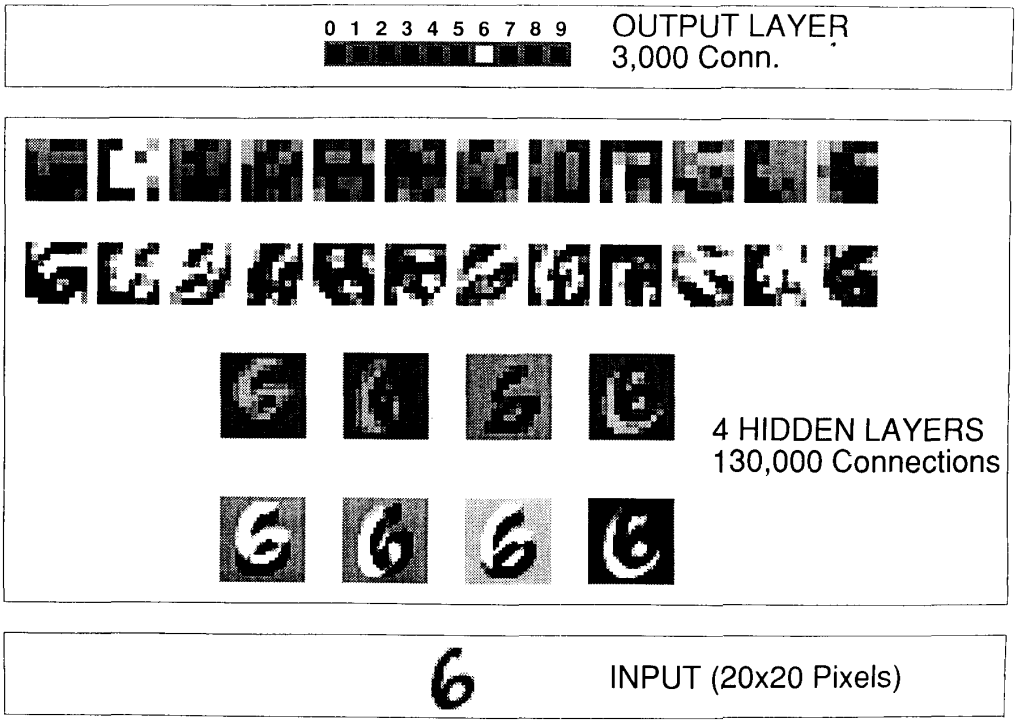


Fig. 5. Example for the states of the OCR network.

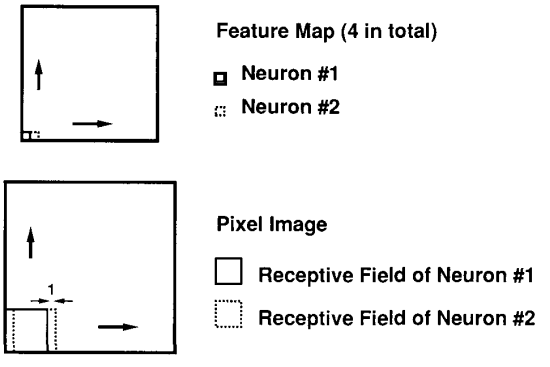


Fig. 6. Architecture of the first layer of the OCR network.

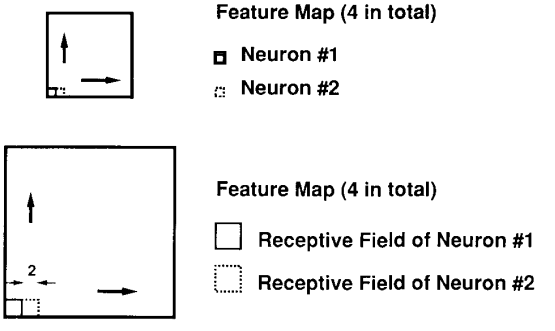


Fig. 7. Architecture of the second layer of the OCR network.

neurons are grouped into four feature maps, each organized as described above. All neurons within one feature map have identical weights (weight sharing); therefore the whole layer is determined by just 4 × 25 parameters (plus four bias values). Another way to interpret the operation of the first layer is to view it as four separate two-dimensional, nonlinear convolutions of the pixel image.

The second layer reduces the spatial resolution of the four feature maps generated by the first layer, resulting in another four feature maps a quarter of the original size. The purpose of this layer is to provide some degree of translational and rotational invariance. This operation is implemented by neurons with four synapses, each neuron averaging four inputs. Again, the architecture of weight sharing and local receptive fields is used, but now in order to reduce the spatial resolution, the local receptive fields of adjacent neurons do *not* overlap, but are displaced by *two* input units (see Fig. 7).

The third layer is similar to the first. It performs feature extraction using a 5 × 5 receptive field. The new aspect of this layer is that inputs from one or two feature maps are *combined*. Most neurons in layer 3 have 50 inputs, 25 of which connect to one feature map, with the other 25 connecting to the same spatial area in another feature map (see Fig. 8).

The fourth layer performs the same averaging and subsampling function as explained for the second layer.

The fifth and last layer has 300 inputs and ten outputs and is fully connected; i.e., it contains 3000 independent connections.

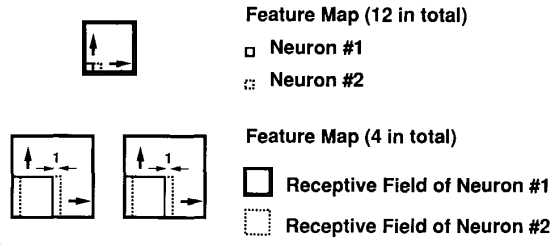


Fig. 8. Architecture of the third layer of the OCR network.

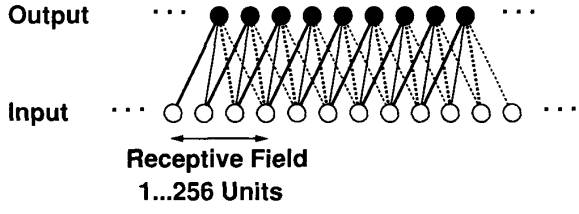


Fig. 9. A one-dimensional nonlinear convolution suitable for the ANNA chip.

This layer classifies the patterns by using ten hyperplanes in the 300-dimensional feature space generated by the first four layers of the network.

IV. Implementation of the OCR Network on the ANNA Chip

To demonstrate the practical usefulness of the chip for real-world applications, the OCR network has been implemented using the ANNA chip.

The computational precision required for each layer in a multilayer feedforward network typically *increases* from the input layer to the output layer [7]. At the same time the number of connections and thus the computational load *decrease* from the input to the output. The optimal hardware implementation of the OCR network is therefore to put the first n_c layers on the ANNA chip and the remaining $5 - n_c$ layers on a floating point processor such as the DSP 32C. Implementations with both $n_c = 3$ and $n_c = 4$ have been realized and studied. In the first case, 96.9% of the connections of the network are evaluated by the ANNA chip; in the second case this figure increases to 97.8%. In the following, the implementation of the first four layers on the ANNA chip will be discussed in detail.

The ANNA chip can directly perform a one-dimensional nonlinear convolution. The corresponding network for this operation is illustrated in Fig. 9. All neurons have the same weight vector (indicated by corresponding line types) and are displaced by one input unit. The straightforward implementation on the ANNA chip is to realize only one neuron on the chip and to time multiplex it between the various locations. As can be seen from Fig. 9, the barrel shifter set to shift-count = 1 performs the correct multiplexing for this network. The evaluation of such a network with n neurons takes n calculation cycles on the ANNA chip; however if the network contains more than one neuron with identical receptive fields, then up to eight neurons can be evaluated in parallel.

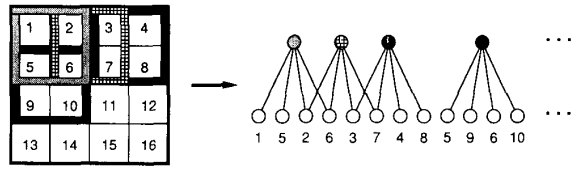


Fig. 10. Implementation principle of the first layer on the ANNA chip.

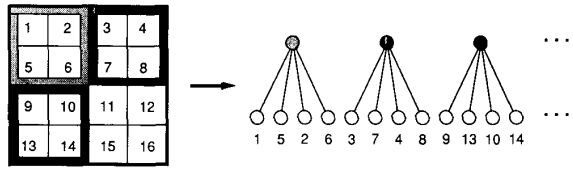


Fig. 11. Implementation principle of the second layer on the ANNA chip.

As explained previously, the first layer of the OCR network carries out a two-dimensional nonlinear convolution. How can this 2-D convolution be mapped to the 1-D convolution suitable for the chip? Fig. 10 illustrates the principle for the example of a 4 × 4 image and a 2 × 2 receptive field: (i) The two-dimensional image data is reformatted into a one-dimensional stream as illustrated by the figure. This reformatting transforms the two-dimensional receptive fields into one-dimensional receptive fields. (ii) Although this reformatting transforms all two-dimensional receptive fields into one-dimensional ones, not all of the one-dimensional receptive fields are used; for example, the field 5–2–6–3 is not used. These fields can easily be skipped by using a shift count larger than 1 (2 and 4 is the example) or by using multiple shift instructions. Up to eight new state values (two shift instructions with shift count = 4) can be loaded while a calculation takes place and thus the execution speed remains the same as in the simple one-dimensional case.

The second layer and the fourth layer are mapped to a 1-D convolution as illustrated in Fig. 11. In contrast to the first layer, subsampling causes more of the 1-D receptive fields to be skipped. This is carried out by increasing the shift count or using multiple shift instructions, as previously mentioned for the first layer.

The third layer is different from the first one in that it extracts features from two feature maps simultaneously. Fig. 12 shows how this type of feature extraction can be mapped to an ordinary 2-D convolution with a subsampling factor of 2. By interleaving the columns of the feature maps as well as the columns of the kernel data, the two disjoint local receptive fields become one contiguous field. The implementation of this 2-D convolution on the chip is straightforward.

The ANNA chip's programmable weight range is $-0.5 \cdots 0.5$; the OCR network, however, requires weights up to the value of 1.0 in layers 1 and 3. These large weights are realized on the chip by using two connections fed by the same input value. Neurons with many synapses typically have small weights, which means that in larger networks this procedure will be unnecessary.

In order to facilitate the implementation of neural networks

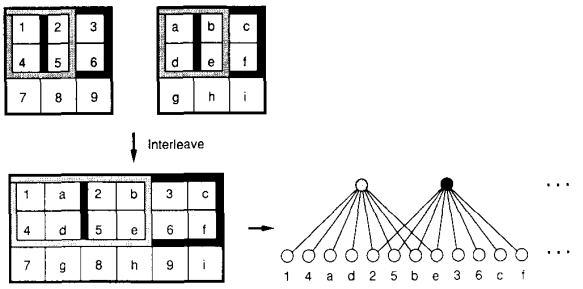


Fig. 12. Implementation principle of the third layer on the ANNA chip.

on the ANNA chip, a LISP program has been developed (called NECTAR) which generates assembly code for the ANNA chip. Convolutional networks with subsampling and extraction from multiple feature maps are covered. Furthermore, NECTAR supports configuration of the chip for the appropriate neuron sizes (64, 128, and 256), quantization of the real-valued weights (and biases) from the simulation to 6 bits, and selection of the optimal scale factor, as well as realization of large (>0.5) weight values. The output of NECTAR (symbolic assembly code) is then passed through ANNANAS (ANNA Assembler), which generates microcode that can be run on the ANNA chip.

The percentage of on-chip weight-memory used for layers 1 to 4 is 69%. The first and second layers each occupy four vectors of length 64; the third layer requires 12 vectors of length 128 since each neuron has $2 \times (25 + 25)$ synapses (the factor 2 is because the weights here are larger than 0.5); and the fourth layer uses 12 vectors of length 64. Note, that many weight values are programmed to 0; for instance in the second layer only four of the 64 weight values are nonzero. This means that much larger networks can be implemented on a single ANNA chip if the number of synapses per neuron is larger.

V. Performance

The execution time of the network on the chip is an important parameter. High speed is necessary when either the patterns appear at a high rate, or when each pattern is classified several times using different scales, translations, and rotations in order to recognize it independent of distortions.

In Table I the speed of the ANNA chip is compared with the corresponding figures of a DSP 32C (40 MHz) and a SUN SPARC.[1] The execution time for the ANNA chip is specified for a 20 MHz clock rate and for the microcode generated by the NECTAR/ANNANAS system.[2] The numbers also include the time necessary to refresh the dynamic weight storage. It is assumed that the formatter can supply and store the state data at the rate the ANNA chip consumes and produces them. The total is calculated for the case where the four layers are executed sequentially on the same chip. If a separate chip is assigned to each layer and the chips operate as a pipeline, the total time decreases to 330 μs.

TABLE I
Execution Time of the OCR Network on Various Hardware

Layer	ANNA Chip	DSP 32C	SUN SPARC 1+
1	330 μs	29 ms	290 ms
2	210 μs	1 ms	10 ms
3	320 μs	19 ms	190 ms
4	100 μs	0.5 ms	5 ms
5	—	0.5 ms	5 ms
Total	960 μs	$\approx$ 50 ms	$\approx$ 0.5 s

The average speed measured in connections per second of the network on the chip is 140 MC/s. The chip, however, is capable of performing 5000 MC/s. The reason for this low utilization is that only a small fraction of the chip's parallelism is used by this particular network. The chip can multiply vectors of size 256 in one calculation cycle, but layers 2 and 4 of the network require only the multiplication of vectors of size 4. Larger networks (more synapses per neuron) will execute more efficiently on the ANNA chip than the current one.

The figures of Table I show that a classification rate of 1000 characters/second can be achieved by a pipelined system using the ANNA chip and a DSP. This rate corresponds to a speedup factor of 50 over the DSP and a factor of 500 over the SUN implementation.

The *error rate* measures how many misclassification the network makes per 100 patterns when tested on a test set it has never seen before. Another even more important performance parameter is the *reject rate*, which specifies the number of patterns, with low classification confidence, that have to be rejected in order to achieve a desired error rate, for example 1%. This figure is important since errors are usually related to a cost which must be kept under a certain limit. The rejected patterns can for instance be classified manually.

The network running on the SUN using 32-bit floating point arithmetic achieves an error rate of 4.9% and a reject rate of 9.1% (for an error rate of 1%) on a test set taken from segmented, handwritten ZIP codes that appeared on real U.S. mail [3].[3] These performance figures have also been measured for three ANNA chips and for two implementations, one with three layers ($n_c = 3$) and one with four layers on the chip ($n_c = 4$), and are shown in Fig. 13. The degradation with respect to error rate is small and in the case of $n_c = 3$ is less than 1%. The reject rate is affected more seriously by the chip's low-resolution arithmetic. This can be explained by the fact that the error rate depends only on the most active output of the network while the reject rate depends on the precise value of the most active and the second most active output, which requires more precision for its calculation.

The observed degradation is due to the following chip non-idealities, presented here in order of decreasing importance:

- quantization of the states to 3 bits;
- quantization of the weights to 6 bits;
- imprecision in the analog computation;

[1] The figures for the DSP 32C and SUN SPARC are estimates assuming efficient code.

[2] A handcrafted implementation of the first three layers using more weight storage than the one described here runs at the following speed: first layer 200 μs, second layer 100 μs, and third layer 130 μs.

[3] The test set is rather difficult; i.e., human performance on this set is 2.5% raw error rate and 3.5% reject rate (for 1% error rate).

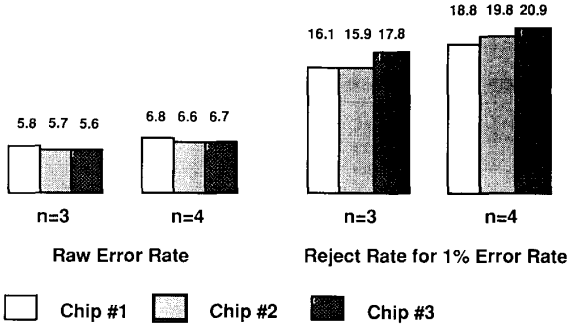


Fig. 13. Recognition accuracy of the ANNA chip. The error and reject rates are given in percent for three different chips and for the implementation of three and four layers on the chip.

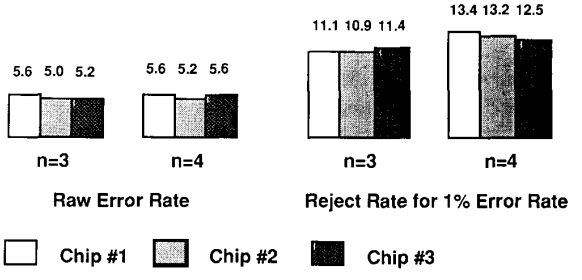


Fig. 14. Recognition accuracy with the ANNA chip after retraining. The error and reject rates are given in percent for three different chips and for the implementation of three and four layers on the chip.

- approximation of the tanh nonlinearity by a piece wise-linear function.

A simulation using full precision but a piecewise-linear function shows only little performance degradation: 5.0%/10.3% (raw error rate/reject rate for 1% error rate). If quantization effects are included in the simulation, the performance drops to 5.5%/14.2%. The actual chip implementation performs at an average of 5.7%/16.6%, showing the effects of the analog computation.

A simple and effective way to improve the performance of the network is to *retrain the last layer.* During retraining with back-propagation, the chip is used for the forward pass, and the backward pass affects only the last layer which is off-chip; the weights on the chip are frozen. This method is very effective since back-propagation is used only for one layer and thus reduces to the simple delta rule. Back-propagating further into the networks is difficult because of the weight quantization (large steps in weight space) and state quantization (derivatives are 0 or infinite). The results (Fig. 14) obtained by retraining just the last layer are encouraging.

After retraining, not only the error rate, but also the reject rate, is very close to the original performance of the network without quantization. It is very important for practical purposes that retraining *not* be done for each individual chip. The above figures were obtained by using chip 1 as a distortion model to retrain the last layer. After that, chip 1 was replaced by other chips without degradation of performance. This is due to good chip-to-chip matching.

Instead of back-propagation the "optimal hyperplane" algorithm [8, p. 353] has been tried to train the last layer. The algorithm adjust the weights of the last layer such that they correspond to hyperplanes optimally separating the training examples of one class from the rest. With this algorithm, the error rate is reduced to 4.9% in the case of $n_c = 3$. This is the same value as for the original network evaluated with floating-point precision.

The effect of retraining the last layer can be illustrated as follows: By quantizing the weights and the states of the first four layers, the feature representation of each pattern in the second to last layer is shifted slightly in the feature space. The decision hyperplanes in the same feature space as given by the last layer, however, stay in the same place. By retraining the last layer, the corresponding hyperplanes are moved such that they take into account the systematic deviations introduced by the chip.

VI. Conclusions

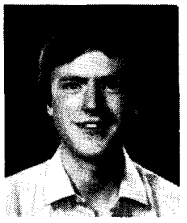
It has been demonstrated that a large and practical neural network application can be implemented with a single ANNA chip. A speed advantage of 50 to 500 over conventional hardware is gained, despite the fact that the network has not been specifically tailored to take advantage of the chip's resources.

The recognition accuracy achieved with the chip's 6 bit/3 bit arithmetic compares favorably with the accuracy of the network evaluated with floating-point precision. This accuracy has been achieved by retraining the last layer to adapt to the chip's low-resolution arithmetic. Chip-to-chip matching is sufficiently good that one chip can be replaced with another without adversely affecting performance.

The ANNA chip is well suited for networks larger than the one described. Networks with more synapses per neuron will take even better advantage of the chip: (i) The chip's parallelism is utilized better and more connections per second are evaluated. (ii) The impact of the state quantization will be less because each neuron takes into account, and thus averages, the quantization errors of more state values.

References

[1] B. Boser and E. Säckinger, "An analog neural network processor with programmable network topology," in *ISSCC Dig. Tech. Papers,* 1991, pp. 184–185.
[2] B. Boser, E. Säckinger, J. Bromley, Y. LeCun, and L. D. Jackel, "An analog neural network processor with programmable topology," *IEEE J. Solid-State Circuits,* vol. 26, pp. 2017–2025, Dec. 1991.
[3] Y. LeCun *et al.*, "Handwritten digit recognition with a back-propagation network," in *Neural Information Processing Systems,* D. S. Touretzky, Ed. San Mateo, CA: Morgan Kaufmann, 1990, pp. 396–404.
[4] I. Guyon, P. Albrecht, Y. LeCun, J. S. Denker, and W. Hubbard, "A time delay neural network character recognizer for a touch terminal," *Pattern Recognition,* 1990.
[5] H. P. Graf, R. Janow, C. R. Nohl, and J. Ben, "A neural-net board system for machine vision applications," in *Proc. Int. Joint Conf. Neural Networks,* 1991, pp. I-481–I-486.
[6] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. Lang, "Phoneme recognition using time-delay neural networks," *IEEE Trans. Acoust., Speech, Signal Process.*, pp. 328–339, Mar. 1989.
[7] I. Kanter and S. Solla, private communication, 1990.
[8] V. Vapnik, *Estimation of Dependences Based on Empirical Data.* New York: Springer Verlag, 1982.



Eduard Säckinger (S'84–M'91) was born in Basel, Switzerland, on August 13, 1959. He received the M.S. and Ph.D. degrees in electrical engineering from the Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, in 1983 and 1989, respectively. In the fall of 1983 he joined the Electronics Laboratory of the Swiss Federal Institute of Technology, where he investigated analog applications to floating-gate devices. His doctoral work was on the theory and CMOS integration of a differential difference amplifier.

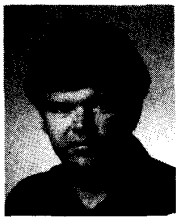
Since September of 1989 he has been with AT&T Bell Laboratories, Holmdel, NJ, working on artificial neural-network hardware and its applications to character recognition. His research interests include analog circuit design and parallel distributed processing.

Bernhard E. Boser (S'83–M'88) received a diploma in electrical engineering in 1984 from the Swiss Federal Institute of Technology, Zurich, Switzerland, and the M.S. and Ph.D. degrees from Stanford University, Stanford, CA, in 1985 and 1988, respectively. His thesis was on the design and implementation of oversampled analog-to-digital converters.

Since 1988 he has been with AT&T Bell Laboratories, Holmdel, NJ, where he works on VLSI implementations of artificial neural networks for pattern recognition applications.

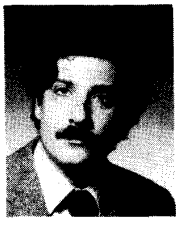
Jane Bromley received the B.Sc. honors degree in physics in 1983 and the Ph.D. degree in biophysics in 1987, both from Imperial College, London University, London, U.K. Her Ph.D. thesis and postdoctoral work involved the study of human vision and visual dysfunction caused by brain lesions using psychophysical techniques.

In 1990 she joined AT&T Bell Laboratories, Holmdel, NJ, where she has been working on the application of artificial neural networks to human perception tasks, including the reading of handwritten text.



Yann LeCun (S'87–M'90) was born in France in 1960. He obtained the Diplôme d'Ingénieur degree in electrical engineering from the Ecole Supérieure d'Ingénieurs en Electrotechnique et Electronique, Paris, in 1983. In 1987, he obtained the Ph.D. degree in computer science from the Université Pierre et Marie Curie, Paris. While working for the Ph.D., he introduced an early version of the back-propagation algorithm. From August 1987 to October 1988 he was a research associate in the Department of Computer Science of the University of Toronto, Canada. Since October 1988, he has been a Member of Technical staff at AT&T Bell Laboratories, Holmdel, NJ.

Dr. LeCun has served as session chairman on the organizing committees of several conferences, including the IJCNN'89 (Washington DC), the NIPS'90 Conference (Denver, CO), and the INNC (Paris) in 1990. His current interests include neural networks and connectionist models, learning theory, pattern recognition, and VLSI implementations of massively parallel systems.



Lawrence D. Jackel (M'82–SM'90), a native of New York City, received the B.S. degree in physics from Brandeis University, Waltham, MA, in 1969. He was awarded the Ph.D. degree in experimental physics in 1976 from Cornell University, Ithaca, NY. His thesis topic was in the field of superconducting devices.

Since 1975, he has been at AT&T Bell Laboratories, Holmdel, NJ, where his initial research was in microfabrication and microscience. In 1984 he became head of the Device Structures Research Department and began studies in artificial neural networks. In 1990 his department was renamed the Adaptive Systems Research Department and pursues theory, applications, and hardware for machine learning and machine perception.

Dr. Jackel is a member of the American Physical Society.