AD-A236 270

(2)

AD

AD-E402 198

Technical Report ARFSD-TR-91005

NEURAL NETWORKS: AN OVERVIEW

Thomas Brady

May 1991

US ARMY
ARMAMENT MUNITIONS
& CHEMICAL COMMAND
ARMAMENT RDE CENTER

U.S. ARMY ARMAMENT RESEARCH, DEVELOPMENT AND ENGINEERING CENTER

Fire Support Armaments Center

Picatinny Arsenal, New Jersey

Approved for public release; distribution is unlimited.

91 5 4 064

REPORT DOCUMENTATION PAGE

Form Approved
OMB NO. 0704-0188

1. AGENCY USE ONLY ***(Leave blank)***

2. REPORT DATE
May 1991

3. REPORT TYPE AND DATES COVERED

4. TITLE AND SUBTITLE
NEURAL NETWORKS: AN OVERVIEW

5. FUNDING NUMBERS

6. AUTHOR(S)
Thomas Brady

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
ARDEC, FSAC
Fire Control Division (SMCAR-FSF-RC)
Picatinny Arsenal, NJ 07806-5000

8. PERFORMING ORGANIZATION REPORT NUMBER
Technical Report ARFSD-TR-91005

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
ARDEC, IMD
STINFO Br
ATTN: SMCAR-IMI-I
Picatinny Arsenal, NJ 07806-5000

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 words)***

This report provides an introduction to the field of neural networks, what they are, how they came about, how they work, how they are used, and their current status and potentialities.

14. SUBJECT TERMS
Neural networks Pattern classification Perceptrons Computers

15. NUMBER OF PAGES
43

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	SAR

CONTENTS

	Page
Introduction	1
History	3
Examples of Neural Networks	5
The Perceptron	5
Backpropagation	10
The Adaptive Resonance Model	15
The Neocognitron	17
Experimental Work	17
New Directions	20
Society of Mind	20
Neural Darwinism	21
Conclusions	23
References	37
Distribution List	39

FIGURES

		Page
1	The physiological neuron	25
2	An object recognition problem	26
3	The simplest neural network	27
4	Neural separation	27
5	Geometry for two-layer perceptron	27
6	Two-layer perceptron	27
7	Geometry for three-layer perceptron	28
8	Three-layer perceptron	28
9	Geometry for exclusive "Or"	29
10	Network for backpropagation	29
11	More complicated backpropagation network	30
12	Feedback for adaptive resonance	30
13	Neocognitron arrangement	31
14	Aircraft shapes for testing	31
15	Listing of training output	32
16	Segmented input image used for training (default)	33
17	Listing of test output	34
18	Labeled test image (scene 2)	36
19	Darwin II model	36

INTRODUCTION

Neurons are brain cells. There are approximately ten billion (10^{10}) brain cells of varying types in the human brain. These are interconnected by axons and dendrites, the axon being the output fiber of a neuron which branches off into dendrites which in turn connect to other neurons. A typical structure is shown in figure 1. The interconnectivity can be quite dense with one neuron receiving inputs from up to thousands of others. The action of the neuron is to generate an electrical pulse or pulse train if the summation of its inputs exceeds a threshold associated with a particular neuron. The collection is thus similar to a vast highly interconnected network of threshold elements. It is this network which constitutes the human thinking machine.

This thinking machine has capabilities that, to date, have frustrated duplication by large scale digital computers. These capabilities are not in the area of esoteric high level mathematics but in what are generally considered simple normal processes, that is processes that could be performed even by children. As an example consider the children's puzzle shown in figure 2. A picture is given and scattered throughout the picture are a certain number of given objects. The goal is to find all these objects. Now imagine programming a computer to do the same task. Assuming the picture to be black on white it could be presented to the computer as a rectangular array of pixels that are either activated or not with a resolution compatible with the scene and objects being represented. Algorithms must now be devised that can recognize the sought for objects regardless of their locations in the scene, their orientations or sizes. It must also be able to recognize them if they are partly obscured. It must perform this recognition while ignoring other objects and separating them from the desired objects. A child can do this without much trouble. A program to do this, if it could be written, would probably be very large, slow, and fragile. The difficulty in writing such programs arises from the fact that we really don't know how we do the recognition process. We don't know how the brain does it. Recognition of voices over the telephone is another example. Even with poor connections there are usually a number of voices that we can immediately recognize over the phone. Yet if we were asked to write out the steps how this recognition was made, that is, the recognition algorithm that could be used by somebody else, most would admit defeat while the remainder would probably become embroiled in Fourier analysis, linguistics analysis, information theory, etc. Again the fact is that we don't know how the recognition is made, how the brain does it.

The children's puzzle referred to could be considered a target recognition problem where now the desired objects are tactical targets in a military environment that are being sought by some fire control system. Or it could be considered the scene presented to a robot which must move to the desired objects and collect them. In other words this is a problem of strategic pertinence. There is thus more than just scientific curiosity to motivate research into how the brain works and how it solves such problems. Conventional computing techniques are based on sequential operation. The algorithm is constructed as a list of steps that are to be executed sequentially. A machine that performs these sequential operations is referred to generically as a Von Neumann machine after the mathematician who introduced the stored program concept in digital computer design. The networks in the brain however are highly parallel. Instead of one sophisticated central process unit (CPU) working sequentially we have, or seem to have, billions of relatively unsophisticated CPU's - the neurons - working for the most part in parallel fashion. And somehow this parallel arrangement has a power that can't be approached by the conventional Von Neumann machine.

A field of research has grown directed to understanding and exploiting the power of brain-like neural nets. The physiological neurons are replaced by relatively simple threshold elements together with a specified or random interconnection topology. The interconnections do not usually, if ever, retain the physiological characteristics of axons or dendrites but are lossless, instantaneous transmission lines which can, however, weight or modify by a constant multiplier the signal being transmitted. (Sometimes a delay is introduced. And sometimes a degree of physiological realism is introduced as for example, a refractory period. For the most part however the abstraction is rather severe.) Thus from the structures of the brain is abstracted, via Occams razor, networks that are referred to as synthetic neural networks, artificial neural networks or - when it is understood that the physiology is not relevant - simply as neural networks. Such simplified networks can be arranged to exhibit learning, pattern recognition, control, problem solving, and other activities reminiscent of human behavior. The trick however is in the arranging of the model or how it is constructed. A great deal of research has been and is being spent to develop models that are versatile, robust, fast, and reliable. A number of diverse disciplines are involved in this research such as psychology, biology, mathematics, computer science, physics, and engineering. At least three journals are now devoted to the subject, such as IEEE Transactions on Neural Networks, Neural Networks, and Neural Computation. In September and October 1990 the Proceedings of the IEEE were devoted exclusively to the subject. Many other journals regularly publish papers on the subject.

In February 1990 an ILIR was authorized to review the current status of neural networks technology with particular attention to its use in vision systems for automatic target recognition (ATR) and robotic systems. The results of that review are summarized in this report.

HISTORY

Work on artificial neurons can be traced back to the late 1930's and early 1940's in the work of N. Rashevsky and his colleagues, notably Landahl and Householder (refs 1 and 2). Their work has not motivated by physiological realism - as they admit - but instead was concerned with the capabilities of idealized networks describes by linear first order differential equations. They were able to show by such methods that a number of psychological functions could be simulated.

At about the same time McCulloch and Pitts (ref 3) were developing the model of neural nets as logic nets with the neurons as all or nothing logical devices. This was felt to be a representation that was much closer to the biological networks. In fact it seems to have discouraged the Rashevsky group from proceeding with their differential equation models on the ground that they were too unrealistic. The McCulloch-Pitts model showed promise for a while. It was shown mathematically by Kleene (refs 4 and 5) that it was capable of representing a very wide variety of events and into the 1950's a great deal of research was spent in exploring the capabilities of such models. But then progress dropped rapidly. Among the reasons given for the failure of the McCulloch-Pitts approach to a realistic model were (ref 6); a full knowledge of input-output relations which is required but is not available for any biological species; a precision of connection is required which is not present in the brain; the number of neurons often exceeds those in actual nervous systems; there was no adaptive behavior; and a nonvolatile memory seemed to be impossible in such a model.

In the late 1950's a new phase of neural network research started with the formulation of F. Rosenblatt of neural nets he called perceptrons (ref 6). These were neural nets wherein the neurons were threshold elements and if the sum of the inputs exceed a threshold associated with that neuron then the neuron would switch its state. Each input line to the neuron has a weight associated with it that corresponds to synaptic efficiency. These weights would be initially set to random values and then adjusted during a training period. Effectively, these networks learned to classify the present inputs. Moreover, Rosenblatt was able to prove mathematically that if a set of weights existed for a given classification of inputs then the training procedure would converge to a set of weights in a finite number of steps that will allow the perceptron to make the same classification. (This will be presented in more detail in the next section.) This proof was for a particular type of perceptron. Much interest was generated by Rosenblatt's work and many investigators became involved in perceptron-type research. It was found however that while there were some problems the perceptron did very well with, there were many others that it didn't and no one knew why. Then in 1969 Minsky and Papert published their study of perceptrons (ref 7) in which they showed the limitations of perceptrons, fundamental limitations as it turned out. This work is accused by some of stifling further research in this area although the authors disagree.

At the same time Rosenblatt and his colleagues were developing the perceptron, Widrow and his colleagues were developing a similar type of neural net that used a least-mean-squares algorithm for adjusting the weights (refs 8 and 9). Their work was directed elsewhere however when they met with a lack of success in devising methods for training multilayer networks which are required for most problems. This was essentially the same problem that the perceptron investigators encountered.

The 1970's were a time of relative inactivity for neural networks although some significant research was done. Werbos in 1974 (ref 10) developed a method whereby multilayer nets could be trained, a problem which has proved frustrating to the development of the perceptron and Madeline (the name associated with the nets of Widrow, et al.) His work however was to be unknown to the scientific community until the 1980's when it was discovered. Also at this time Grossberg (ref 11) and Fukushima (ref 12) were developing models that involved feedback and new types of elements into the neural net. Both continued their research and development in the 1980's.

The field was rejuvenated in the 1980's with the rediscovery of Werbos's method by Rumelhart, Hinton, and Williams (ref 13). The method was now called the method of backpropagation. Currently this is the most popular method in neural network activities but it is not without it critics. A 1988 edition of the Minsky-Papert book (ref 7) includes a critique of the method. Its major shortcoming is that it is essentially an optimization technique that is subject to the trap of local extrema. Also it can be very slow in converging. Much of the more recent work in neural networks has been devoted to the search for methods to eliminate or reduce these shortcomings of the backpropagation method. To avoid hangups at local extrema the so-called simulated annealing method (ref 14) has been introduced which effectively, on a probabilistic basis, allows the search for an extrema to start in a coarse manner, with even some steps away from the extrema, and then to gradually refine the search. Methods of speeding up convergence are discussed by Werbos in his 1990 article (ref 15).

In 1986 Minsky (ref 16) proposed a different approach which he referred to as a society of mind. Rather than going to ever more complicated networks such as elaborate multilayer perceptrons with backpropagation or the feedback structures of Grossberg or Fukushima he raised the possibility of going to simpler structures. An organization of such simple structures would then be managed to perform the task at hand. While his book does not get down to the neuronal level it is persuasive in its illustrations of how various mind-like behaviors can be reduced to the operations of a group or society of simple components. The approach could be called psychological in the way it goes from behavior to components that could account for such behavior. No implementations or simulations are presented so that much remains to be done with these concepts.

Another different approach was made in 1987 by Edelman (ref 17) with his theory of neuronal group selection which he called neural darwinism. Most, if not all, approaches to the brain problem treat it as a computer or information processor of some kind. To quote from Edelman and his colleagues (ref 18), "The understanding of brain function is widely viewed as requiring an abstract approach based on theories of information, logic, computation, and cybernetic control. We present . . . an analysis which suggests that the application of many of these abstract principles to animal behavior is inconsistent with what is known about the nervous system . . . As biologists we emphasize that a computational formalism applied in isolation to behavior can give only a very incomplete and potentially misleading picture of the nature of mind." Whereas the society of mind approach is a top down approach, neural darwinism is a bottom up approach proceeding from very detailed physiological analysis. It has similarities to Minsky's ideas that it considers organizations of networks rather than one general network. Computer models have been developed of increasing degrees of sophistication, the Darwin I, Darwin II, and Darwin III programs. Edelman's theory and presentation of it are quite difficult however and do not seem to have reached an audience in the engineering community. The publication of reference 18 in an engineering journal may change this and the neural network community may become more interested in his work.

The above history has not mentioned a number of contributions to this subject. This is because the purpose was not to give a detailed history but rather a general history that will convey the overall picture of the activity in a modest space. More detail may be found in the references.

EXAMPLES OF NEURAL NETWORKS

The Perceptron

An example of a simple neural net is shown in figure 3. There are two variable inputs, x_1 and x_2. There is one constant θ. The inputs can be analog. Associated with the variable inputs x_1 and x_2 are the weights w_1 and w_2. Associated with the input θ, called the threshold, is the weight -1. The circle represents the neuron function f where

$$y = f(w_1x_1 + w_2x_2 - \theta)$$

and

$$f(x) = \begin{cases} 1, & x \geq 0 \\ -1 & x < 0 \end{cases}$$

and is referred to as a hard limiter. The equation

$$w_1x_1 + w_2x_2 = \theta$$

is the equation of a straight line in the x_1 - x_2 plane as illustrated in figure 4 with slope $-w_1/w_2$ and x_2 - intercept θ/w_2. If input values x_1, x_2 are such that

$$x_2 \geq \frac{\theta}{w_2} - \frac{x_1 w_1}{w_2} \text{ or } w_1x_1 + w_2x_2 - \theta \geq 0$$

then

$$y = 1$$

otherwise $y = -1$. Thus this network classifies a point (x_1, x_2) into one of the two categories divided by the straight line. In this sense it recognizes the input. If there were n inputs we would have

$$y = f(w_1x_1 + w_2x_2 + \ldots + w_nx_n - \theta)$$

The expression in the parentheses now represents a hyperplane in n-space which divides the n-space into two categories. If the input n-dimensional point is on one side of the hyperplane the output polarity is plus. If it is on the other side the output is negative.

What a perceptron does is to start with arbitrary (random) values for the w_i and θ and by training adjust the w_i so that the network learns to properly classify the input. The training proceeds as follows:

A set of training points $\underline{x}^{(j)} = (x_1^{(j)}, x_2^{(j)}, \ldots, x_n^{(j)})$ is made available. Some of these points are in region A and the rest are in region B where A and B are separated by a hyperplane. If a point is in region A the output is to be +1 and if it is in B it is to be -1. When an $\underline{x}^{(j)}$ is presented to the net the output will be plus or minus one. If $\underline{x}^{(j)} \varepsilon A$ and the output is positive the response is correct and no adjustment is required. If the response is negative however the response is incorrect and the weights are adjusted in accordance with

$$w_i(t + \Delta t) = w_i(t) + x_i^{(j)}, \; i = 1, 2, \ldots, n$$

Similarly if the output is supposed to be negative, $\underline{x}^{(j)} \varepsilon$ B, and instead it is positive then adjustment is

$$w_i(t + \Delta t) = w_i(t) - x_i^{(j)}, \; i = 1, 2, \ldots, n$$

The training consists of repeating this process for all the training points over and over until no mistakes are made, i.e., it has converged.

Rosenblatt was able to show (ref 6) that if the sets A and B are separable by a hyperplane then the above process will converge in a finite number of steps. In other words if a solution exists then the training process will find a solution though nct necessarily the same one. Depending on the data used the separating plane that it is converged to could be displaced a bit from the one guaranteeing existence.

Since it is one of the few things of consequence in neural network theory that can be proved and since the proof is simple it will be presented here to illustrate the concept of perceptron convergence.

We consider a single layer perception as shown in figure 3 but with n inputs, that is it accepts vector inputs $\underline{x}^{(i)}$ where the $\underline{x}^{(i)}$ have n components. Thus we have n weights $w_1, w_2, \ldots, w_n$ and a threshold θ. We will use the inner product notation

$$<\underline{w}^{(k)}, \underline{w}^{(i)}> = w_1^{(k)} w_1^{(i)} + w_2^{(k)} w_2^{(i)} + \ldots + w_n^{(k)} w_n^{(i)}$$

The classification will be binary which means the input vector space is to be divided into two parts A and B and the perceptron is to be trained to recognize the vectors which are in A. Let us take the threshold to be zero, i.e., $\theta = 0$. Then the training procedure is as follows.

Initialize $\underline{w}$ to an arbitrary one of the vectors $\underline{x}$, i.e., let

$$\underline{w}^{(o)} = \underline{x}^{(i)}$$

where the $\underline{x}$ are the training vectors. Let the first vector presented to the perceptron be $\underline{x}'$. We know if this vector belongs to A or B. If $\underline{x}' \varepsilon$ A and y = 1 this is a correct response and we proceed to the next input vector. If $\underline{x}' \varepsilon$ A and y = -1 this is an incorrect response and we make the substitution

$$w_i \leftarrow w_i + x_i' \qquad i = 1, 2, \ldots, n$$

and proceed to the next vector. If $\underline{x}' \varepsilon$ B and y = -1 this is a correct response. If $\underline{x} \varepsilon$ B and y = + 1 the response is incorrect and the w_i are updated to

$$w_i \leftarrow w_i - x_i' \qquad i = 1, 2, \ldots, n$$

and the next vector is input.

This procedure can be simplified by changing the sign of the vectors belonging to B. That is, if $\underline{x}^{(k)} \varepsilon$ B we make the replacement

$$\underline{x}^{(k)} \leftarrow - \underline{x}^{(k)}$$

Then we only have to make the test

$$\langle \underline{w}, x^{(k)} \rangle > 0$$

and, if required,

$$\underline{w} \leftarrow \underline{w} + \underline{x}^{(k)}$$

We further simplify by letting the $\underline{x}^{(i)}$ be unit vectors.

The theorem then states the following: if there exists a unit vector $\underline{w}^*$ and a $\delta > 0$ such that for all

$$\langle \underline{w}^*, x \rangle > \delta$$

then $\underline{w}$ will be updated only a finite number of times by the above procedure. Put another way, if $\underline{w}^{(0)}$ is the initial weight vector and $\underline{w}^1$ $\underline{w}^2 \ldots$, the sequence of updated weights then there exists M such that after $\underline{w}^M$ no further changes are made in $\underline{w}$, it has converged.

We follow here the proof of Minsky and Papert (ref 7). A similar proof has been gi. by Novikoff (ref 19).

Let

$$G(\underline{w}^{(i)}) = \frac{<\underline{w}^*,\underline{w}^{(i)}>}{|\underline{w}^{(i)}|} \leq 1$$

because, since $|w^*| = 1$, this is just the cosine of the angle between $\underline{w}^*$ and $\underline{w}^{(i)}$.

Consider now the sequence of $<\underline{w}^*,\underline{w}^{(i)}>$ remembering that the $\underline{w}^{(i)}$ involved are only those arising in an update since otherwise no change was made in $\underline{w}$.

$$<\underline{w}^*,\underline{w}^1> = <\underline{w}^*,\underline{w}^0 + \underline{x}^1> = <\underline{w}^*,\underline{w}^0> + <\underline{w}^*,\underline{x}^1> > <\underline{w}^*,\underline{w}^0> + \delta$$

$$<\underline{w}^*,\underline{w}^2> = <\underline{w}^*,\underline{w}^1 + \underline{x}^2> = <\underline{w}^*,\underline{w}^1> + <\underline{w}^*,\underline{x}^2> > <\underline{w}^*,\underline{w}^0> + 2\delta$$

$$\cdots$$

$$<\underline{w}^*,\underline{w}^N> > <\underline{w}^*,\underline{w}^0> + N\delta \geq N\delta,$$

so that the numerator of G <u>increases</u> linearly with the number of corrections to the weighting vector $\underline{w}$.

For the denominator consider the sequence of

$$|\underline{w}^1|^2 = <\underline{w}^1,\underline{w}^1> = <\underline{w}^0 + \underline{x}^1,\underline{w}^0 + \underline{x}^1>$$

$$= |\underline{w}^0|^2 + 2<\underline{w}^0,\underline{x}^1> + |\underline{x}^1|^2 < |\underline{w}^0|^2 + 1$$

since $<\underline{w}^0\ \underline{x}^1> < 0$. Similarly

$$|\underline{w}^2|^2 = <\underline{w}^1 + \underline{x}^2,\underline{w}^1 + \underline{x}^2> = |\underline{w}^1|^2 + 2<\underline{w}^1,\underline{x}^2> + |\underline{x}^2|^2 < |\underline{w}^0|^1 + 1$$

$$\cdots$$

$$|\underline{w}^N|^2 < N$$

Thus

$$G(\underline{w}^N) > N\delta/\sqrt{N}$$

Since

$$G(\underline{w}) \leq 1$$

it follows that

$$\sqrt{N}\,\delta \leq 1$$

or

$$N \leq 1/\delta^2$$

so that the number of corrections is bounded. The process converges.

A division of the x_1 -x_2 plane that is not separable (although it is easy to set up a neural network for it) is shown in figure 5. A neural network that will perform the classification is shown in figure 6. This is an example of a 2-layer network or perceptron. It can be seen that it consists of three duplicates of the figure 3 network, one for each side of the triangular region. The last neuron will fire and generate a +1 only if all three preceding neurons are generating +1's.

It follows simply from this how a three-layer network would arise. In figure 7 the region A consists of two disjoint sets. Thus for each one a network as in figure 8 is required. The last layer is then a single neuron which fires if either one of the preceding neurons fire.

For binary inputs, i.e., $x = \pm 1$, the geometry reduces to discrete points which can be taken as the vertices of n-dimensional cubes. In figure 9 this is shown for the x_1 -x_2 plane. It is obvious from this figure and the preceding discussion how a network would require two layers to perform an exclusive OR function. The points (1,-1) and (-1,1) must be separated from the others and this requires two lines. A possibility is shown in the figure.

Backpropagation

A few words about gradient or steepest descent methods will be useful preliminary to the method of backpropagation for neural nets. More details can be found in reference 20.

We consider a function of n variables

$$f(x_1, x_2, \dots, x_n)$$

which we wish to minimize by varying the x_i in some systematic manner. (It is assumed that for one reason or another the process of solving the set of simultaneous nonlinear equations

$$\frac{\partial f}{\partial x_i} = 0, \; i = 1,2,\dots,n$$

is impractical.) The x_i may be thought of as functions of time, $x_i(t)$, such that

$$x_i(t), x_2(t), \dots, x_n(t)$$

represents a curve in n-dimensional euclidean space. Of all the possible curves the one of fastest change is sought for, i.e., a curve of steepest descent. We can also parameterize this curve by it's arc length s and consider x_i as $x_i(s)$. Starting at some arbitrary point $p = (x_1^0, x_2^0, \dots, x_n^0)$ we let t and s = 0. Then arc length will be measured relative to this point as t increases. We have

$$\frac{ds}{dt} = \left[\sum_{i=1}^{n}\left(\frac{dx_i}{dt}\right)^2\right]^{1/2}$$

in analogy with two- and three-dimensional spaces.

Now

$$\frac{df}{ds} = \sum_{i=1}^{n} \frac{\partial f}{\partial x_i}\frac{dx_i}{ds}$$

Since

$$\frac{dx_i}{ds} = \frac{dx_i}{dt}\frac{dt}{ds} = \frac{dx_i}{dt} \Big/ \left[\sum_{j=1}^{n}\left(\frac{dx_j}{dt}\right)^2\right]^{1/2}$$

we may write

$$\frac{df}{ds} = \sum_{i=1}^{n} \frac{\partial f}{\partial x_i}\frac{dx_i}{dt} \Big/ \left[\sum_{j=1}^{n}\left(\frac{dx_j}{dt}\right)^2\right]^{1/2}$$

Let the dx_i/dt be denoted u_i. Now we wish to find a curve such that the rate of change along this curve, df/ds, will be a maximum. That is, we wish to solve

$$\frac{\partial}{\partial u_i}\left(\frac{df}{ds}\right) = 0, \; i = 1, 2, \ldots, n$$

for the u_i. This solution specifies differential equations for the curve. Carrying this through gives

$$\frac{\partial}{\partial u_i}\left\{\sum_{i=1}^{n} \frac{\partial f}{\partial x_j} u_j \Big/ \left[\sum_{K=1}^{n} u_K^2\right]^{1/2}\right\} = \frac{\partial f}{\partial x_i}\left[\sum_{K=1}^{n} u_K^2\right]^{1/2} - \frac{1}{2}\left[\sum_{K=1}^{n} u_K^2\right]^{3/2} 2u_i \sum_{j=1}^{n} \frac{\partial f}{\partial x_j} u_j = 0$$

which reduces to

$$\frac{\partial f}{\partial x_i} \sum_{K=1}^{n} u_K^2 - u_i \sum_{j=1}^{n} \frac{\partial f}{\partial x_j} u_j = 0$$

or

$$\frac{dx_i}{dt} = \alpha \frac{\partial f}{\partial x_i} \qquad i = 1, 2, \ldots, n$$

where

$$\alpha = \sum_{K=1}^{n} u_K^2 \Big/ \sum_{K=1}^{n} \frac{\partial f}{\partial x_K} \frac{dx_K}{dt}$$

is not a function of the index i. Thus the direction of steepest descent (or ascent) as given by the dx_j/dt is proportional to the gradient of f.

To illustrate backpropagation as simply as possible consider the network shown in figure 10. During the training session we have associated with each input vector $\underline{x} = (x_1, x_2)$ the desired output vector $\hat{\underline{y}} = (\hat{y}_1, \hat{y}_2)$. Then we can define an error

$$E = \sum_{i=1}^{2} (y_i - \hat{y}_i)^2$$

This is now the function we wish to minimize as a function of the weights. The independent variables are the weights w_{ij}. Therefore we require the derivatives (gradient)

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial y_j}\frac{\partial y_j}{\partial w_{ij}} = (y_j - \hat{y}_j)\frac{\partial y_j}{\partial w_{ij}}$$

The last differential requires the y's to be differentiable functions of their inputs so that the hard limiter used previously will no longer suffice. For this reason the sigmoid function s(x) is introduced as

$$s(x) = 1/(1 + \varepsilon^{-x})$$

which is, approximately, 1 when x>>0 and 0 when x<<0. The derivative of this function is given by

$$\frac{ds}{dx} = \frac{\varepsilon^{-x}}{(1 + \varepsilon^{-x})^2} = s(1 - s)$$

We can now write

$$y_j = s(w_{1j}x_1 + w_{2j}x_2) = s(sum_j)$$

$$sum_j = w_{1j}x_1 + w_{2j}x_2$$

so that

$$\frac{\partial y_j}{\partial w_{ij}} = \frac{\partial s}{\partial sum_j}\frac{\partial sum_j}{\partial w_{ij}} = s(1 - s)x_i$$

to give

$$\frac{\partial E}{\partial w_{ij}} = (y_j - \hat{y}_j)s(1 - s)x_i$$

as the gradient term. The differential equation for the weight change is then proportional to this.

$$\frac{dw_{ij}}{dt} = \alpha(y_j - \hat{y}_j)s(1 - s)x_i = \alpha\delta_j x_i$$

where

$$\delta_j = (y_j - \hat{y}_j)s(1-s)$$

In discrete form

$$w_{ij}(t + \Delta t) = w_{ij}(t) + \Delta t \, \alpha \, \delta_j x_i$$

Lumping $\Delta t\alpha$ together as one proportionality term β gives the w_{ij} corrections as

$$w_{ij}(t + \Delta t) = w_{ij}(t) + \beta\delta_j x_i$$

Consider now an additional layer as shown in figure 11. We now require additional sensitivities -or gradients - for the w'_{ij} terms. Take as a specific case

$$\frac{\partial E}{\partial w'_{11}} = \frac{\partial E}{\partial y_1}\frac{\partial y_1}{\partial w'_{11}} + \frac{\partial E}{\partial y_2}\frac{\partial y_2}{\partial w'_{11}} = (y_1 - \hat{y}_1)\frac{\partial y_1}{\partial w'_{11}} + (y_2 - \hat{y}_2)\frac{\partial y_2}{\partial w'_{11}}$$

Since

$$y_i = s\left[y_1^1 w_{1i} + y_2^1 w_{2i}\right] = s(\text{sum}_i)$$

$$\frac{\partial y_i}{\partial w'_{11}} = s(\text{sum}_i)[1 - s(\text{sum}_i)]\, w_{1i} \frac{\partial y'_1}{\partial w'_{11}}$$

Since

$$y'_1 = s(w'_{11} x_1 + w'_{21} x_2) = s(\text{sum}_i)$$

$$\frac{\partial y'_1}{\partial w'_{11}} = s(\text{sum}_i)[1 - s(\text{sum}_i)]\, x_1$$

The gradient can be written

$$\frac{\partial E}{\partial w'_{11}} = (y_1 \underbrace{- \hat{y}_1) s(\text{sum}_1)[1 - s(\text{su}}_{\delta_1}m_1)]\, w_{11} s(\text{sum}'_1)[1 - s(\text{sum}'_1)]\, x_1$$

$$+ (y_2 \underbrace{- \hat{y}_2) s(\text{sum}_2)[1 - s(\text{sum}_2}_{\delta_2})]\, w_{12} s(\text{sum}_i)[1 - s(\text{sum}_i)]\, x_1$$

$$= (\delta_1 w_{11} + \delta_2 w_{12}) s(\text{sum}_i)[1 - s(\text{sum}_i)] x_1$$

This gives in discrete form for the correction to w'_{11}

$$w'_{11}(t + \Delta t) = w'_{11}(t) + \beta[\delta_1 w_{11} + \delta_2 w_{12}]\, s(sum_i)\,[1 - s(sum_i)]\, x_1$$

The corrections for w'_{12}, w'_{21}, w'_{22} would be found in a similar manner. This illustrates how propagation is made back into the network layer by layer. If a third layer was added, say with weights w_{ij}'', then the chain of partial derivatives would be extended in a similar manner to include these terms.

The Adaptive Resonance Model

The proceeding examples were based on supervised learning. That is, the training required an element external to the network that controlled the adjustment of the weights depending on the correctness of the response. A classification of inputs was established before the training and then this predetermined classification was taught to the network by systematically adjusting the weights. A number of models have been developed which do not use a predetermined classification and are referred as unsupervised learning models. The adaptive resonance model of S. Grossberg, more specifically the ART1 model, which is for binary input shall be discussed here.

Let the inputs be n-component binary vectors $\underline{x} = (x_1, x_2, \ldots, x_n)$ where x_i is either 0 or 1. Let the output be an M-component vector $\underline{y} = (y_1, \ldots, y_m)$ which is also binary. This will be capable of an M-way classification if for each of M different inputs one y_i of the M y-components is 1 and the others are zero. Let $\underline{x}'$ be the first of a sequence of input vectors. And let the y response be $\underline{y}' = (1, 0, \bar{0}, \ldots, 0)$. Associated with each of the output nodes is a summer

$$\mu_j = \sum_{i=1}^{n} w_{ij} x_i$$

where w_{ij} is the line weight from input i to the summer for output node j. Initially

$$w_{ij} = 1/(1 + n) \qquad j \neq 1$$

$$w_{i1} = x_i'/\left[-5 + \sum_{i=1}^{n} x_i'\right]$$

We have

$$w_{i1} \geq w_{ij} \text{ or } w_{i1} = 0$$

In other words the lines for the $\underline{x}'$ classification have the weights enhanced to reflect that particular input. If the same vector is presented again and the μi calculated, i = 1,...,M then

$$\mu_1 > \mu_i \qquad i \neq 1$$

so that node 1 is selected. Now let another vector $\underline{x}^2$, $\underline{x}^2 \neq \underline{x}^1$, be presented as input. If upon calculating the μ_i μ_1 turns out to be greater than the rest then this would have to be put into the same category as $\underline{x}^1$. If μ_1 is not the maximum then all the other μ_i will be the same and the second node can be taken as the new category and the weighting on the lines to this node enhanced accordingly. And we can proceed in this manner until there is no further variation in the input, or the output nodes are exhausted.

For some input distributions this procedure works fine. Unfortunately there are many input distributions where it doesn't. The learning is unstable. As Carpenter and Grossberg (ref 21) put it, "... the networks adaptability, or plasticity, enables prior learning to be washed away by more recent learning in response to a wide variety of input environments." To overcome this problem they introduced the theory of adaptive resonance to building into the competitive learning process a self-regulating control structure for stability and efficiency. This introduces feedback into the network.

The feedback involves a comparison of input and output and an action based on this comparison. Now the output is an excited node. This had been excited by a particular pattern which can be remembered on feedback lines. That is, let there now be lines from the output node that are weighted to the pattern of the input that excited it. Figure 12 is an attempt to visualize this. Suppose the input $\underline{x} = (1,0,1)$ excited $\underline{y} = (1,0)$. Then this could be remembered by setting

$$t_{11} = 1,\ t_{12} = 0,\ t_{13} = 1$$

which is a copy of $\underline{x}$. Now suppose another input also excites the same $\underline{y}$. Then this input should be close in some sense to the previous input. Let $\underline{t}_1 = (t_{11}, t_{12}, t_{13})$, sometimes called the expectation. Then

$$< \underline{t}_1, \underline{x}^1 > / | \underline{x}^1 |$$

is a measure of the match between the input and the expectation. Since the only values are 0 and 1 it is the fraction of x values that agree with the expectation. If this is above some prescribed value (usually called the vigilance term) they are said to match. Otherwise it is a mismatch. If a mismatch occurs then this node is eliminated from consideration and the maximum is sought over all the output nodes except this one. A match is referred as a resonant condition between input and expectation and the input weights, that is the weights on the input side of the network, will be strengthened accordingly.

The adaptive resonance model, ART1, thus checks input patterns to see if they are consistent with what has been previously established within some prescribed tolerance -the vigilance parameter. In this way new patterns will not wipe out what has been learned but if they are close to a previous pattern they will be incorporated into that category with perhaps a corresponding expectation adjustment and if they are not close a new category will be set up provided a node is available.

The Neocognitron

This is another example of a network that uses more than just elementary neurons and feedforward propagation as in the perceptron and the backpropagation models. The adaptive resonance model used feedback lines for a form of memory and a vigilance parameter to measure the closeness of patterns. In the neocognitron, as developed by K. Fukushima (ref 22) different types of neuron models are used. In the original version there is only feedforward propagation. In a more recent version, described in reference 23, feedback is employed. The feedback model is to allow recognition of patterns that were shifted in position or distorted in shape. These are multilayer networks that employ unsupervised learning. The feedback model resembles the adoptive resonance model in the interplay of feedforward and feedback signals but in the neocognitron the interplay can occur at all layers instead of just the input layer.

There are three types of neurons employed in the neocognitron. The first is for feature extraction, the second to allow for position errors in the input and the third, which is inhibitory, is used to enhance the feature selectivity of the first type. How these neurons are arranged is shown in figure 13 which was adapted from reference 23.

The same types of neurons are used in the feedback paths. Here they are used however to provide gain controls for the forward flow cells.

EXPERIMENTAL WORK

Because so little can be predicted from theory about the performance of any particular neural net, simulation is a necessity in the evaluation of proposed models. Even for the perceptron as described earlier it is not known in general how to specify the number of layers required, the number of neurons per layer, or the thresholds. That is, in general we will not know the hyperplanes required and how or if they separate the n-space. For more complicated models, such as those for adaptive resonance and the neocognitron, even the weak aid of n-dimensional geometry becomes more remote and insight can be obtained only by running computer simulations.

Grogan and Johnson (ref 24) have made such a study of the ART1 and the neocognitron models in the recognition of aircraft shapes. For the ART1 study three aircraft were represented by overhead silhouettes - actually binary signals on a 16 x 16 pixel array. Six output category nodes were used. The vigilance parameter was varied. The input underwent translation and rotation. The shapes were clean, i.e., there was no noise. Taken from reference 24 the shapes of the aircraft on the 16 x 16 field are shown in figure 14.

For a vigilance factor of 0.9 all three aircraft were recognized (without any translation or rotation). For a vigilance factor of 0.7 however the three aircraft are categorized into only two categories, that is, this value was too low for sharp discrimination. This is the problem with this type of discrimination. A high value is fine for clean images but if there is some noise then a high value will tend to take noisy patterns as new patterns. A low value tends to blur the boundaries of the categories. Thus while ART1 is presented as an example of unsupervised training it most likely will require supervision on an operators part to select a value of the vigilance parameter that is suitable for the particular task at hand.

When the input shapes were presented also in translated and rotated positions a high value (0.9) of the vigilance parameter caused the translated and rotated shapes to be taken as new patterns. This led to the authors' conclusion that an ART1 network should have a preprocessor to make the input pattern invariant to rigid body transformations. They did not discuss sensitivity to the vigilance parameter other than for the two values mentioned above.

Also studied in the same report was a neocognitron model. This seems to be the version without feedback - the authors refer to a multilayer feedforward network. The same images and sizes were used for the ART1 model. Four cell layers were used. Training is done layer by layer, that is the input layer is trained first, then the layer adjacent to this and so on until all the weights have stabilized. The training is of a competitive type, i.e., winner-take-all. We quote the authors; "Considerable effort was required to adjust the network parameters to assign unique categories to the three aircraft and provide some invariance to translation and noise. . .." An further, ". . .even with our numerous attempts at adjusting the network parameters we were unable to uniquely categorize the shapes the same in both the untranslated and translated images. . . .it becomes evident that there is a tradeoff in the networks ability to discriminate and to provide translational invariance." The authors conclude, ". . .for the task of aircraft identification and orientation estimation unsupervised learning does not offer the required performance." According to Fukushima (ref 23), "The neocognitron (without feedback) has the ability to recognize stimulus patterns correctly, even if the patterns are shifted in position or distorted in shape." The apparent conflict here is most likely an indication of the fine tuning required and the patterns being used.

The above study was based on direct images being presented to the network. An alternative method is to use model-based techniques. These are techniques wherein abstract representations of the object are used for characterization. For example the shape of the aircraft could be considered to be given as a function of two variables by f(x,y) where the range of f could be binary or analog and the domain the boundary and interior points. The moments of this function are

$$m_{pg} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x^p y^q f(x,y)dxdy$$

and the feature vector

$$\underline{y} = (m_{00}, m_{01}, m_{10}, \ldots, m_{nn})$$

can be used to characterize the particular aircraft. Similarly, if the contour is taken as t he complex function z(t) the characterization may be expressed as Fourier coefficients given by

$$F(n) = \frac{\pi}{2} \int_{-\infty}^{\infty} z(t)\varepsilon^{-int} dt$$

Other transformations have also been used, e.g., Walsh, Fourier-Merllin. (Taking moments of course can also be considered a transformation.) The use of such transforms often allows for easy manipulation of the object mathematically in the feature vector space so that invariant representations may be established, i.e., normalization may be made with respect to size, translation, and rotation. Object recognition is now equivalent to recognizing a point, i.e., the feature vector in an N-dimensional space where N is the number of components in the feature vector, e.g., the number of Fourier coefficients.

Reeves and Prokop (ref 25) have developed an object recognition system that uses neural nets to identify objects represented by feature vectors. The system also performs the segmentation required in the feature vector preprocessing. Given an image the system will segment it using either a threshold type of segmentation or a Markov field based segmentor as determined by the user. A feature vector will then be set up using either moments or Fourier descriptors again as determined by the user as well as the number of terms. The user then specifies the neural network topology, i.e., the number of layers and the number of neurons per layer. The model currently set up is the backpropagation model although this could be replaced - via some programming -

with other models. The system can then be trained by a given set and then tested against other image sets. Output is both graphic and tabular. A typical training and test sequence is shown in figure 15 through 19.

The system has been implemented at the ARDEC facility and its capabilities are being studied. It will also be used to study the Markov field segmentor which is a recent development that employs a neural network type structure.

NEW DIRECTIONS

Society of Mind

The sequence of neural networks described above, perceptron, backpropagation, adaptive resonance and neocognitron, shows a trend of increasing complexity with the multilayer neocognitron with feedback being the most complex. It is also - at least based upon the investigation of Grogan and Johnson - the most difficult to tune, that is, the adjustment of the network parameters for satisfactory operation. Thus while the training is unsupervised a significant amount of outside intervention is required. This trend to large and more complicated multifunctional networks (the neocognitron is claimed to effect not only pattern recognition under position shifts and distortion but also to segment, restore imperfect patterns and eliminate noise), has recently been criticized and questioned as the proper and most promising direction for such research to take. Minsky and Papert (ref 7) have suggested that the study of how the human brain works suggests that it is not with large multifunctional neural nets but rather with small simple nets - even, perhaps, at the level of simple perceptrons. They propose the concept of a society of small agents which individually are extremely limited but when organized are capable of more than the sum of their parts. Minsky refers to this organization as a society of mind and in his book (ref 16) explores the subject in detail. Minsky's presentation is based upon what is referred as a top-down approach. That is, starting with a particular type of human activity a hierarchy of simple agents will be synthesized - on paper - that could organize to perform that activity. An example he uses is that of a child building a tower out of blocks, and reduces this to a tree-like structure of agents where each agent is of extremely limited capability but together they become a builder. This sounds simplistic but it is only the introduction. More complicated behavior is considered by degrees along with the structures and agents that would be needed to effect such behaviors. In this way, for example, memory is introduced in a way that is somewhat reminiscent of the adaptive resonance expectation lines. Minsky speaks of K-lines as activators of the agents that had been involved in the past for some particular task. When the task comes up again the proper agents are brought into play by that particular K-line that connects to them. Levels of memory are then brought in by hierarchies of K-lines, that is, a K-line activating other K-lines and so on. Another concept introduced is that of frames. The idea here is that a perceptual experience activates some structures in the brain - some collection of agents that is - that have been acquired previously. By this is meant they represent some interaction with the environment that has occurred in the past history of the individual. The frame represents a general form of this experience or ". . . somewhat like an application form with many blanks or slots to be filled." The blanks are called the terminals and are used as connection points for which present experience can attach information. In this way assumptions are aroused. The activation of the frame carries along with it the points that must be assumed - filled by default values - until information for the specific points is available. For example, under the stimulus of a desert battle environment, loud mechanical noise and a large shape, the frame corresponding to a tank could be activated. With this framework the details of friend or foe, armaments, etc. would be the terminal values to be filled in. Again there can be interactions between frames with more than one frame being activated. The society of mind, as presented by Minsky, does not get down to wiring details. It is not a blueprint of brain behavior. Rather it is a theory of how the mind may work based a good deal on the learning behavior of children and psychological analysis of behavior. It is not intended to be a final work on the subject but instead a possible stimulus to new avenues of research.

Neural Darwinism

This is a bottom-up approach. Whereas the top-down approach goes from behavior to structure, the bottom-up approach goes from structure to behavior. That is, starting with the basics of physiological knowledge a model is developed in accord with biological principles which hopefully will explain or duplicate human behavior.

G. Edelman (ref 17) has attempted to take what is known about the brain and demonstrated how it leads to the functioning of the brain as in perception and memory. He has argued that this must be done without an a priori categorization of the world since this is not the way the world is presented to the developing brain. The brain, he argues, develops as a selective system as in the Darwinian system governed by the principles of selection and that in reality a form of selection occurs rather than what is called learning. The categorization comes about through selection for the organism's survival.

His theory is based on three fundamental claims (ref 26). "(1) During the development of the brain in embryo a highly variable and individual pattern of connections is formed between neurons. (2) After birth a pattern of neural connections is fixed in each individual but certain combinations of connections are selected over others as a result of the stimuli the brain receives through the senses. (3) Such selection would occur particularly in groups of brain cells that are connected in sheets, or "maps," and these maps "speak" to one another back and forth to create categories of things and events."

In this approach the basic unit of selection is the neuronal group, a set of interconnected neurons that may number in the thousands, that in some sense function together. The critical aspect is the formation of maps and the communication between such maps, which Edelman refers to as "re-entry." "The purpose of the maps is to create perceptual categorizations. . ." (ref 17).

Darwin II is a simulation of recognition developed in the early 80's by Edelman and his colleagues. To again quote Rosenfield (ref 26): "The automaton is the first man-made device that strictly adheres to the most sophisticated neurophysiological knowledge of our day, and an examination of its functioning provides a sharp contrast to the computational and PDP approaches. . ." The over all structure is shown schematically in figure 20. The left channel is a two-layer structure for feature detection both on the local and non-local level. The right channel is also a two-layer structure but designed to respond to correlations of features that are relatively invariant to translations and rotations of the input. Re-entry between the channels as shown at the bottom permits integration of the two types of responses.

Darwin II was followed by Darwin III which incorporates motor control into the model so that its response can be evaluated by observation of its motor acts. (Again this is all simulated. It is a simulated robot not an actual piece of robotic hardware.) Both models, especially Darwin III, are described in some detail in reference 18).

The two channels in the models have been referred to as visual and tactile branches (ref 18) so that the circuit is similar to proposals of sensor fusion that have been made in automatic target recognition investigations. It should be pointed out that the use of two channels was not to be interpreted to mean that more than two channels could not be used. The interaction could be multichannel.

Neural Darwinism or the theory of neuronal group selection as it is also called is a difficult theory based very strongly on the most advanced neurophysiological knowledge. The organizations it leads to are quite complicated as can be inferred from figure 20. This complication is somewhat to be expected since so little is permitted a priori in the theory, e.g., categorizations or a training supervisor. The goal however is a rigorous understanding of neural behavior on a physiological basis. This is not necessarily the goal of all researchers in neural networks. Many are problem solvers looking for new tools to aid them in problem solving. If some of the brains ability can be duplicated to solve a problem there will be no guilt feelings because a strict adherence to neurophysiological principles have not been maintained.

CONCLUSIONS

Much has been done in the field of neural networks in the last 50 years or so. It is currently an extremely active field. Progress, however, still leaves something to be desired. In a recent review of the technology with respect to its applications to automatic target recognition (ref 27) it is still spoken of in terms of its potential, what could be rather than what is now. Similarly in a review with respect to biomedical implications (ref 28) the conclusion is that it is potentially promising. Minsky and Papert (ref 7) made the claim (in 1988) that "little of significance had changed since 1969" (1969 saw the first edition of their book "Perceptrons"). They take a very dim view of backpropagation methods because they are essentially hill-climbing methods with the attendant local-extrema problems, and such difficulties increase when the scale of application is increased. This is another complaint about current techniques; that they have only been applied to small problems, toy problems as some call them, and are not practical for realistically sized problems. These considerations probably helped in the motivation towards a society of mind. But while a society of mind is an attempt to simplify it seems to lead to complicated structures of agents. Eventually wiring diagrams will have to be produced and these may well be as complicated and troublesome as anything we have now.

The fact remains, however, that neural networks do exist that are fast, compact, robust, capable of learning, generalizing and much more. These are, of course our individual brains. The existence proof is very substantial. The solution is there but it has yet to be expressed in our mathematics - Von Neumann (ref 29) has said that ". . . the brain does a new kind of mathematics. . ." - or captured by our analytic or synthesis techniques. It is encouraging that different approaches have similarities that indicate some common ground. An example of this is what some call fusion techniques or multisensor techniques. It has been mentioned above briefly how Edelman was led to such multiple channels in his neural Darwinism models. Others are attempting such an integration using Markov field techniques (ref 30). It is, at least, implicit in Minsky's work. In other words, Edelman recognizes the need for it working up from basic biological principles; Minsky recognizes the need for it working down from psychological analysis and Bildro, et al. recognizes the need for it on engineering grounds. Similarly we have the correspondence between the K-lines of society of mind and the expectation channels of adaptive resonance. Thus while we have investigators going their own way there are points of convergence between the investigations which may be indicative of some underlying fundamental principles. Thus while from one aspect progress may appear to be slow, a broader view is not discouraging. While a complete understanding of how the brain works may be a golden grail of technology there are still many benefits to be obtained from a partial understanding, particularly with respect to the problems of pattern and image recognition.

At ARDEC the emphasis is on automatic target recognition (ATR) and robotics so the question is how should neural network research be directed here to obtain maximum benefits for these endeavors? The answer is multifaceted. First surveillance and study of the literature should continue to keep abreast of the latest developments and to be able to evaluate them. Second, experimental capability should be developed since, as explained in the section on experimental work, theoretical analysis in this field requires significant support from simulation. As mentioned in that section an object recognition system has been implemented at ARDEC. This can be tied in to real world images through a VICOM system and will allow testing of segmentation methods and neural network classifiers. At present the system uses backpropagation method. Since this architecture is modular though, coding for different methods could be programmed into it. Work has started on using this system. It should be emphasized however that this is not a parallel system. It is a serial simulation of a parallel system. Third, research should be started into fusion and integration methods pertinent to target recognition to see how much can be squeezed from such approaches as those of Minsky, Edelman, and others. Special attention should be given to communication between networks. This would also involve some simulation and probably a separate simulator from that mentioned above. And fourth, should be a blank space to allow for any breakthrough that may occur in the field. More seriously, it should be recognized that the field is still wide open. An altogether different approach could happen at any time - or has happened and hasn't been noticed. Nothing is a priori locked out. There should be enough flexibility to be able to respond to any new techniques or ideas that might occur.

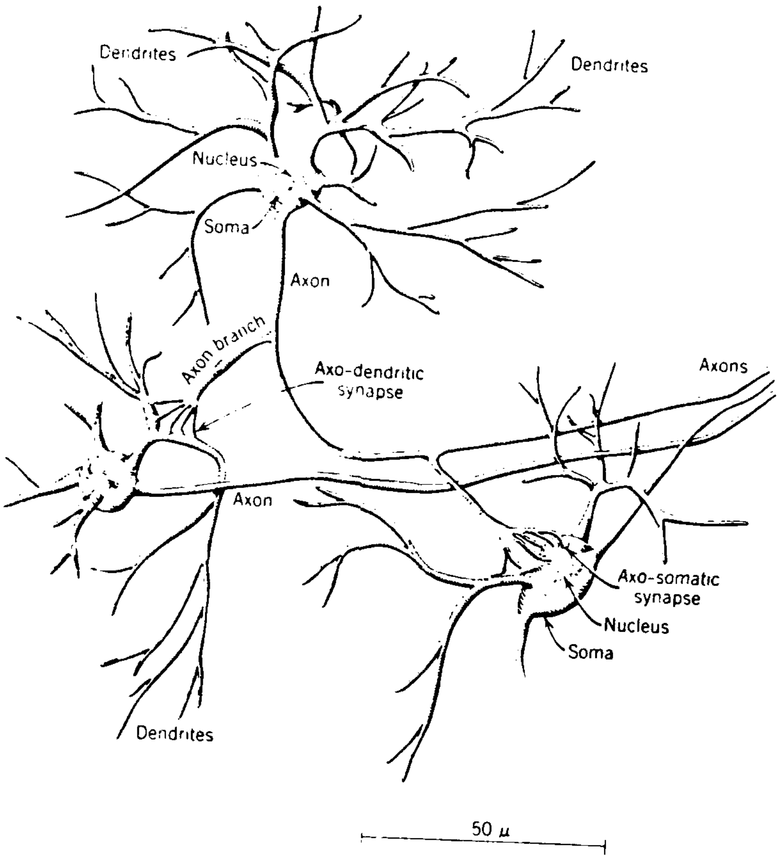

Figure 1. The physiological neuron

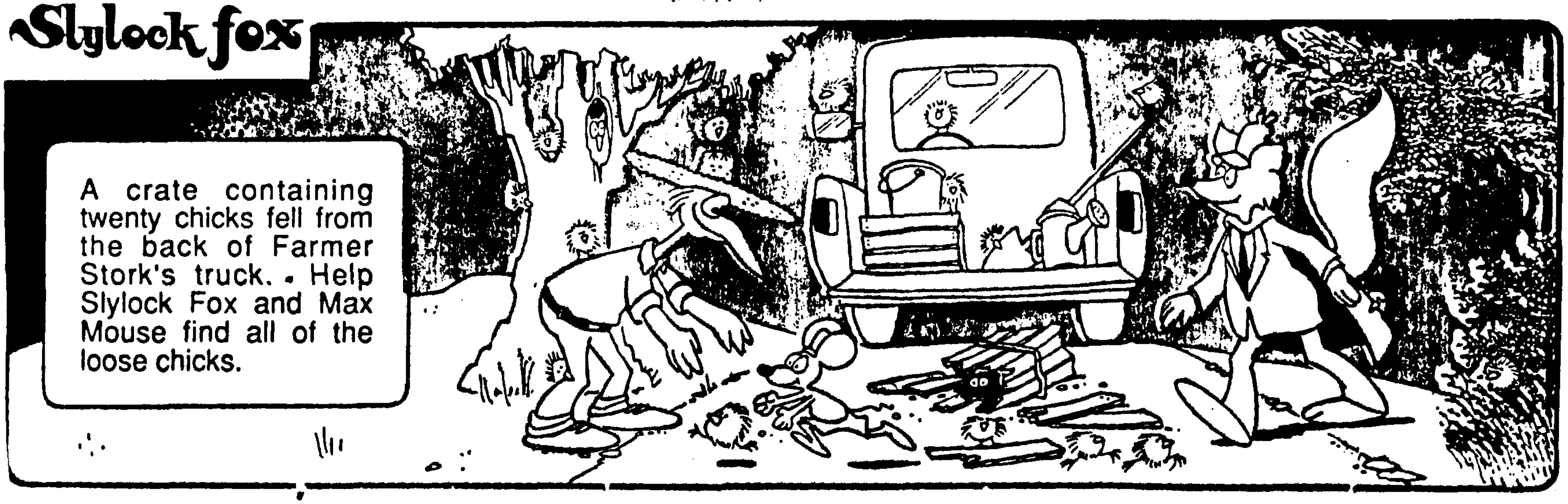

Figure 2. An object recognition problem

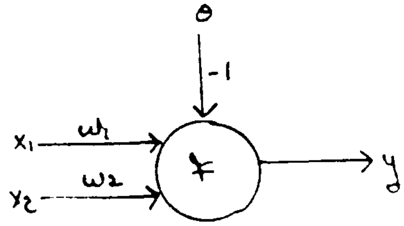

Figure 3. The simplest neural network

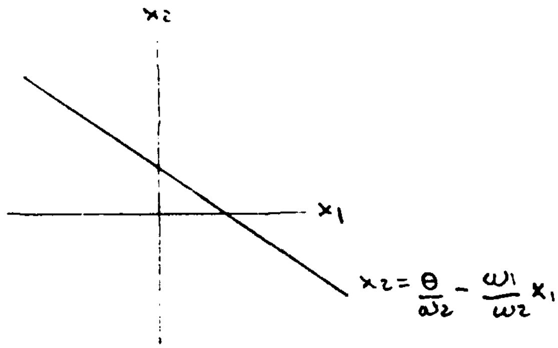

Figure 4. Neural separation

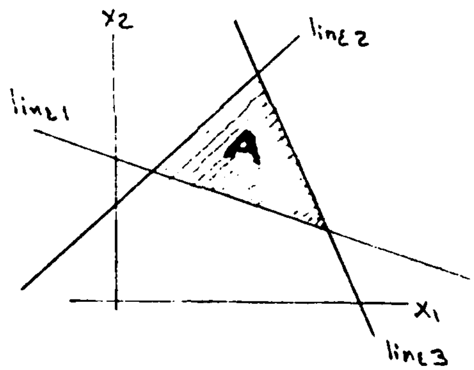

Figure 5. Geometry for two-layer perceptron

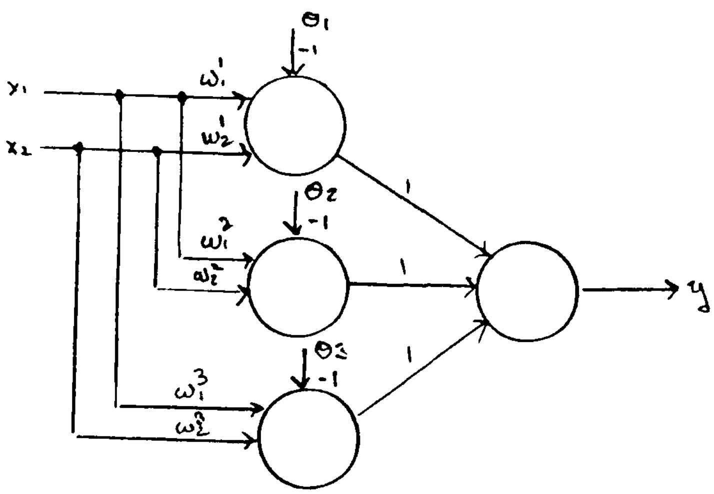

Figure 6. Two-layer perceptron

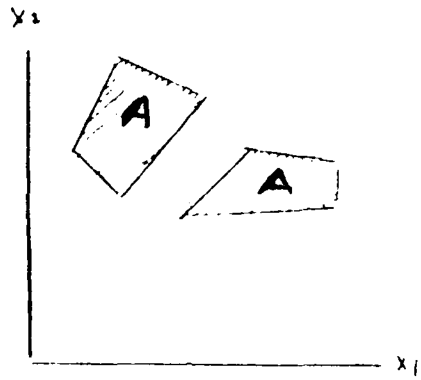

Figure 7. Geometry for three-layer perceptron

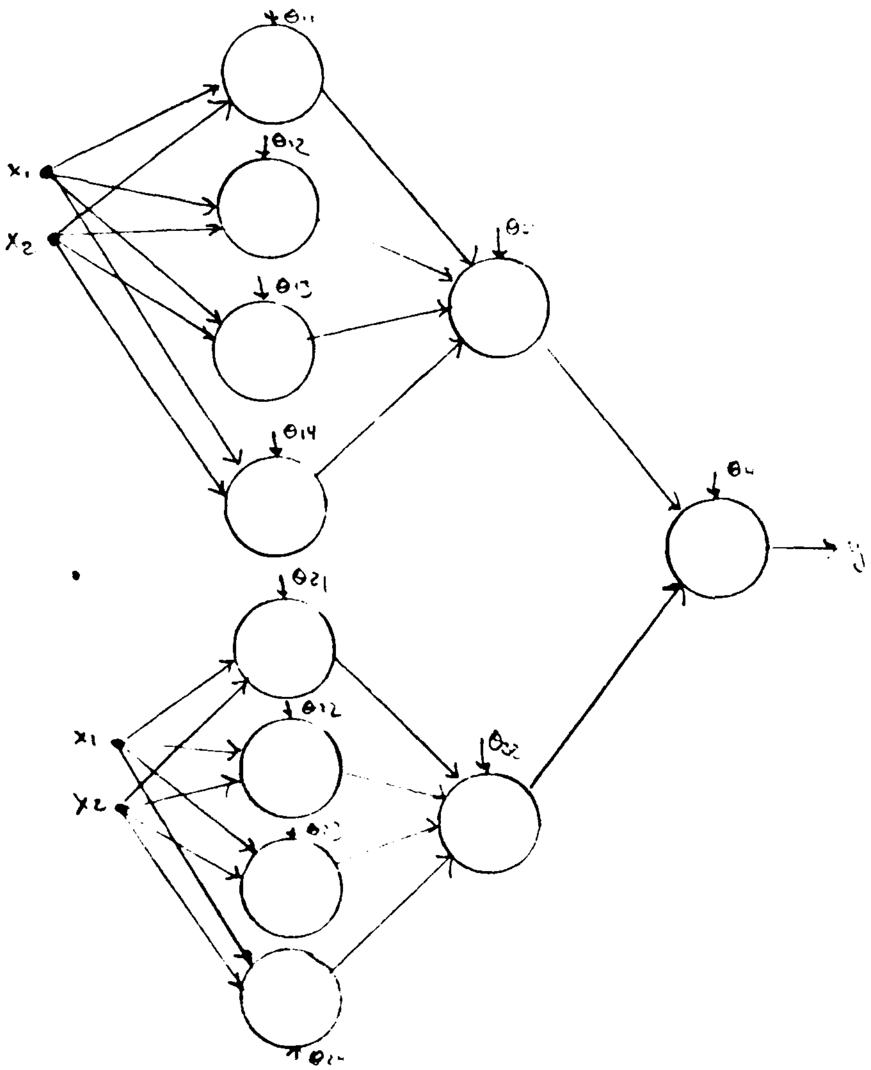

Figure 8. Three-layer perceptron

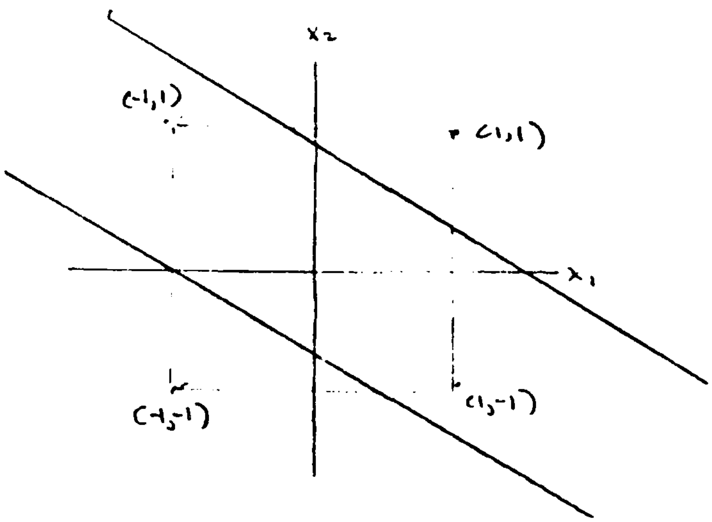

Figure 9. Geometry for exclusive "Or"

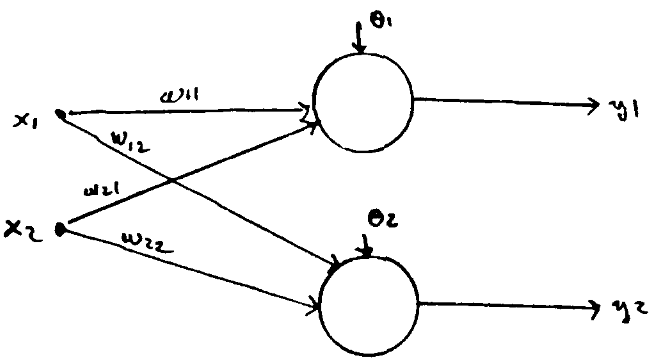

Figure 10. Network for backpropagation

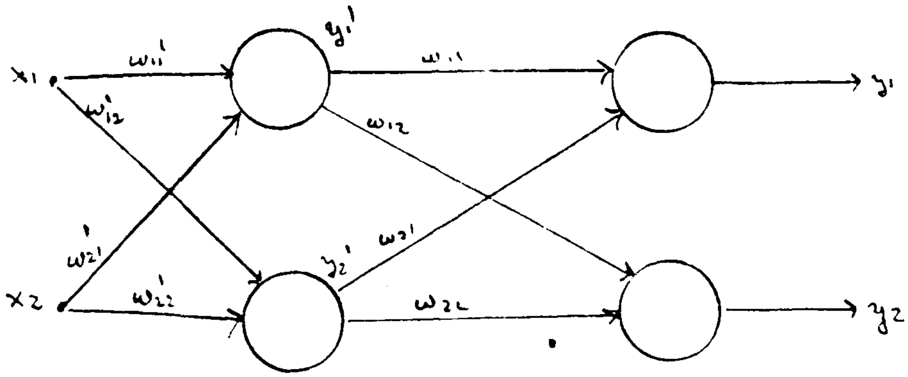

Figure 11. More complicated backpropagation network

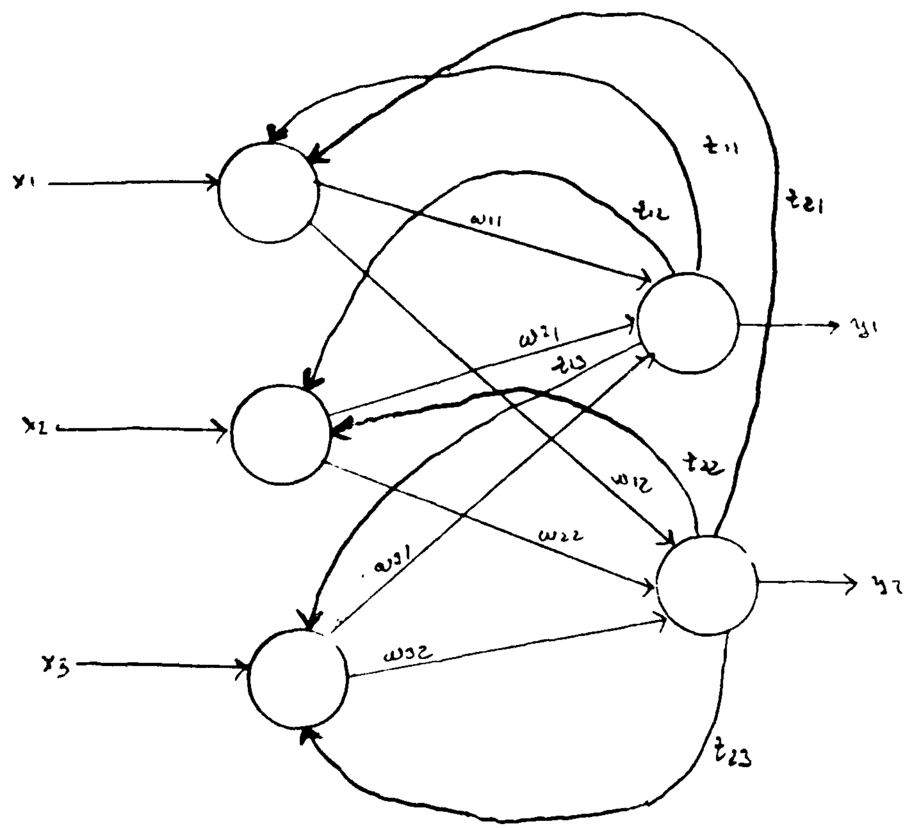

Figure 12. Feedback for adaptive resonance

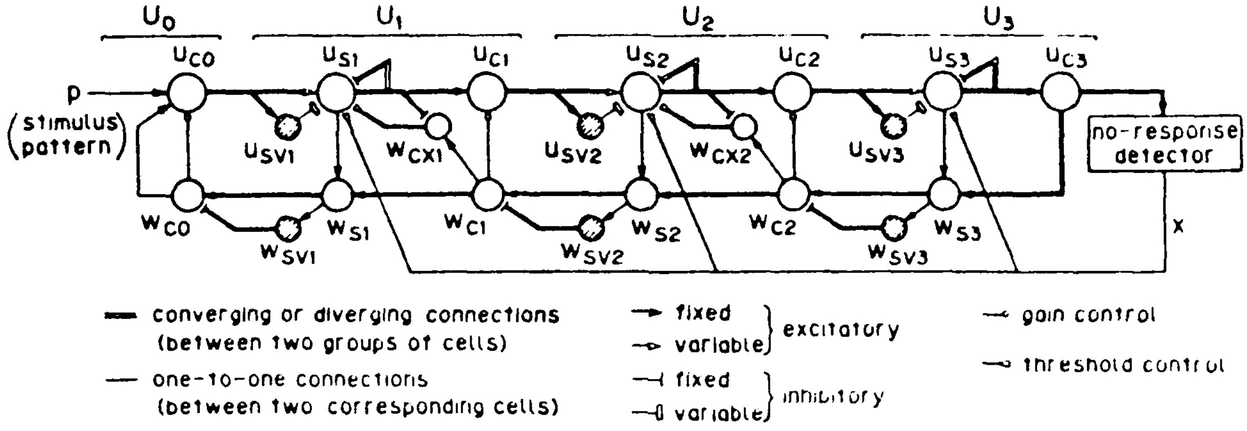

Figure 13. Neocognitron arrangement

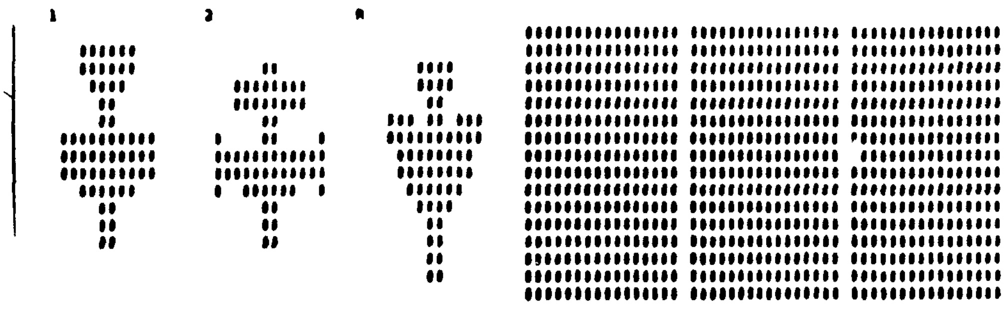

Figure 14. Aircraft shapes for testing

```
    Display Type .......................... SUNGREY
    Number of Classes ..................... 3
    Neural Network Configuration .......... net1.mts
    Neural Network last TRAINED with ...... default
    Segmentation script ................... nn_seg

UU.VISIX: Standard Moments order 4

1        The program version(network type)
1        The number of networks (layers)
11       The number of inputs to each neuron
3        The number of neurons
1        The interconnection pattern, 1=Full
1        The neuron type: 1=threshold, 2=ramp, 3=sigmoid




-----------------------------------------------------------
--- PREPROCESSING input to create network input vector
-----------------------------------------------------------

    Input .................................. default.im
    Creating a segmented byte image ........ default.seg
    Creating a bitplane version ............ default.bit
    Creating an image set .................. default.ims
    Labeling the images in the set ......... default.ims

              [--- see figure 5      (ed.) ---]
    --
    --   Enter a list of class identifiers
    --   for each numbered segment in sequence.
    --   Terminate the list with a return.
    --   Enter list : 3 2 1 3
    --   Is this labeling correct? (y/n) : y
              [--- see  figure 6     (ed.) ---]
    --
    Creating moment feature vector file .... default.mts

    Creating the network input data ........ input.mts

-----------------------------------------------------------
--- TRAINING the Neural Network
-----------------------------------------------------------

    Training data : default

    Neural Network Log :

UU.VISIX:Neural Network Log File

687      - the number of iterations for convergence.
```

Figure 15. Listing of training output

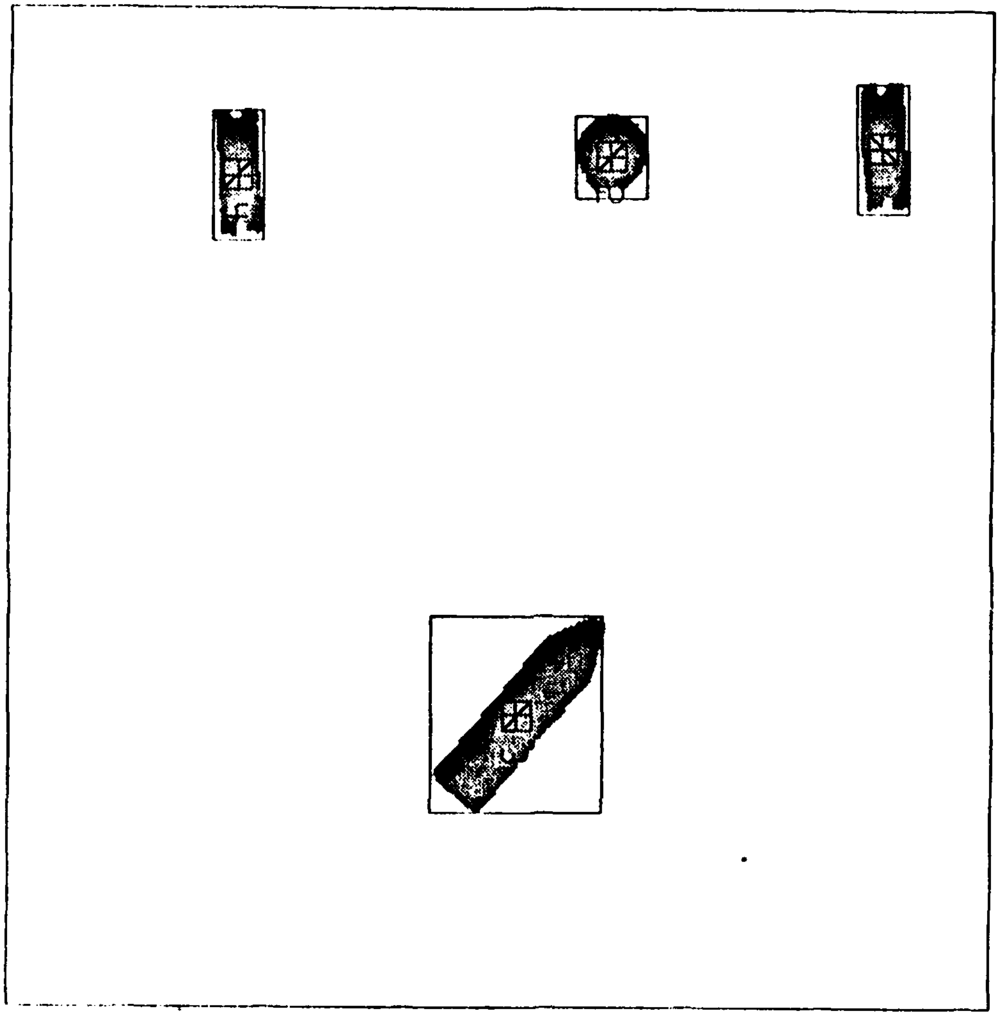

Figure 16. Segmented input image used for training (default)

```
Display Type ........................... SUNGREY
Number of Classes ...................... 3
Neural Network Configuration ........... net1.mts
Neural Network last TRAINED with ....... default
Segmentation script .................... nn_seg

UU.VISIX: Standard Moments order 4

1       The program version(network type)
1       The number of networks (layers)
11      The number of inputs to each neuron
3       The number of neurons
1       The interconnction pattern, 1=Full
1       The neuron type: 1=threshold, 2=ramp, 3=sigmoid
```

```
------------------------------------------------------------
--- PREPROCESSING input to create network input vector
------------------------------------------------------------

  Input ................................. scene.im
  Creating a segmented byte image ........ scene.seg
  Creating a bitplane version ............ scene.bit
  Creating an image set .................. scene.ims
  Creating moment feature vector file .... scene.mts

  Input ................................. scene2.im
  Creating a segmented byte image ........ scene2.seg
  Creating a bitplane version ............ scene2.bit
  Creating an image set .................. scene2.ims
  Creating moment feature vector file .... scene2.mts

  Input ................................. fulrak.im
  Creating a segmented byte image ........ fulrak.seg
  Creating a bitplane version ............ fulrak.bit
  Creating an image set .................. fulrak.ims
  Creating moment feature vector file .... fulrak.mts

  Creating the network input data ........ input.mts
```

Figure 17. Listing of test output

```
------------------------------------------------------------
--- TESTING the Neural Network
------------------------------------------------------------

    Object   Class Found Misfire

       1       1     1      0
       2       2     1      0
       3       3     1      0
       4       4     1      0
       5       5     3      0
       6       1     1      0
       7       2     2      0
       8       3     2      0
       9       4     1      0
      10       5     1      0
      11       6     3      0
      12       1     1      0
      13       2     2      0
      14       3     2      0
      15       4     2      0
      16       5     3      0

------------------------------------------------------------
--- Classification Complete.
------------------------------------------------------------

    To display labeled images enter > netpic mts

        [-----   see figures 8 through 11 (ed.)  ----]
```

Figure 17. (cont)

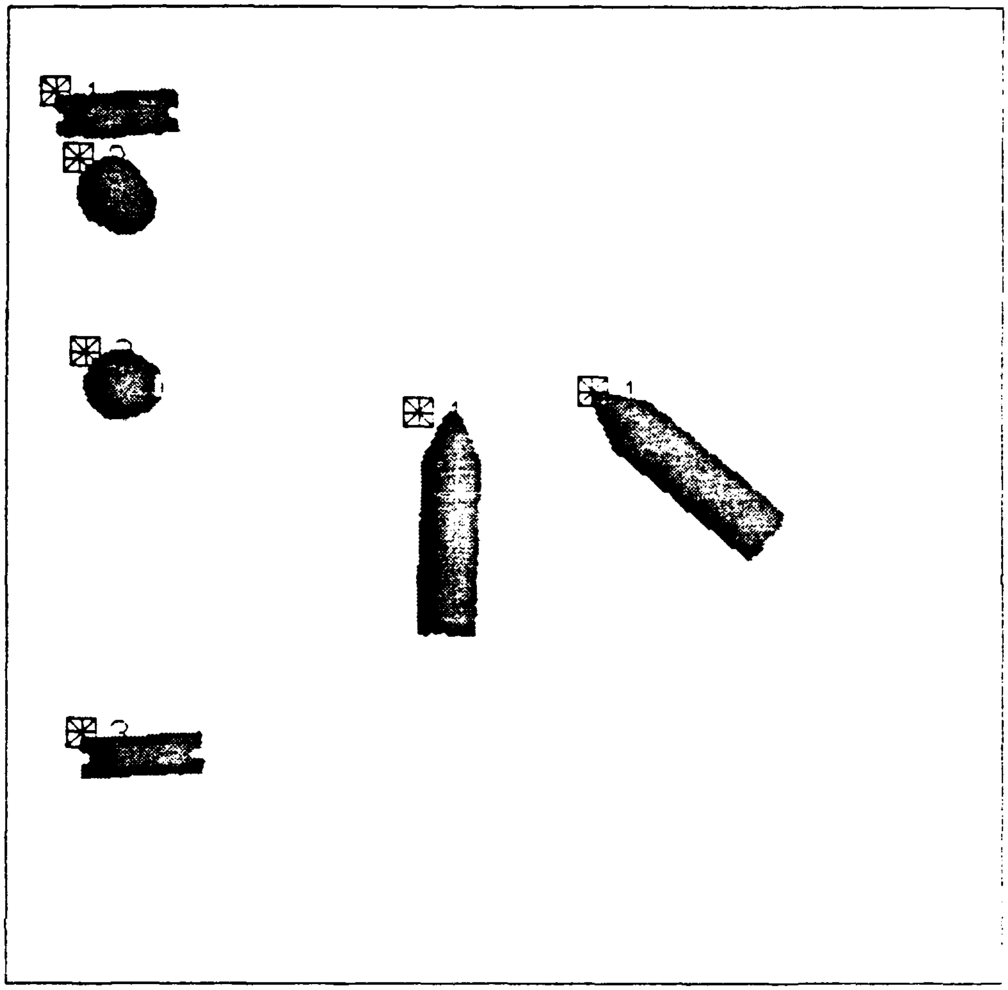

Figure 18. Labeled test image (scene 2)

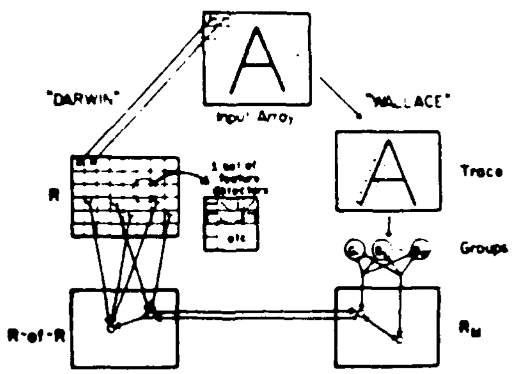

Figure 19. Darwin II model

REFERENCES

1. Rashevsky, N., Mathematical Biophysics, rev. ed. Dover, 1960.

2. Householder, A. S. and Landahl, H. D., Mathematical Biophysics of the Central Nervous System, Principia Press, Bloomington, IN, 1945.

3. McCulloch, W. S. and Pitts, W., A Logical Calculus of the Ideas Immanent in Nervous Activity, Bulletin of Math, Biophysics, 1943.

4. Kleene, S. C., Representation of Events in Nerve Nets and Finite Automata, in Shannon and McCathy (Ed.) Automata Studies, Princeton University Press, 1956.

5. Minsky, M., Computation: Finite and Infinite Machines, Prentice-Hall, 1967.

6. Rosenblatt, F., Principles of Neurodynamics, Spartan Books, 1962.

7. Minsky, M. and Papert, S., Perceptrons, MIT Press, Revised Edition, 1988.

8. Widrow, B., Generalization and Information Storage in Networks of Adeline Neurons, Self-Organizing Systems 1962, Spartan Books, 1962.

9. Widrow, B. and Lehr, M., 30 Years of Adaptive Neural Networks: Perceptron, Madeline and Back Propagation, Proceedings of the IEEE, Sep 1990.

10. Werbos, P., Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences, PhD thesis, Harvard, Cambridge, MA, 1974.

11. Grossberg, S., Adaptive Pattern Classification and Universal Recoding, Biological Cybernetics, Vol. 23, 1976.

12. Fukushima, K., Cognitron: A Self-Organizing Multilayered Neural Network, Biological Cybernetics, Vol. 20, 1975.

13. Rumelhart, D. E.; Hinton, G. E.; and Williams, R. J., Learning Internal Representations by Error Propagation, Rumelhart and McClelland (Eds.), Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. I, Foundations, MIT Press, 1986.

14. Kirkpatrick, S.; Gelatt, C. D.; and Vecchi, M. P., Optimization by Simulated Annealing, Science, Vol. 220, 1983.

15. Werbos, P., Back Propagation Through Time: What It Does and How to Do It, Trans. IEEE, Vol. 78, No. 10, Oct 1990.

16. Minsky, M., Society of Mind, Simon and Schuster, NY, 1986.

17. Edelman, G., Neural Darwinism, New York Basic Books, 1987.

18. Reeke, G. N.; Sporns, O.; and Edelman, G., Synthetic Neural Modeling: The Darwin Series of Recognition Automata, Proc. IEEE, Sep 1990.

19. Novikoff, A., On Convergence Proofs for Perceptrons, Proceeding of the Symposium on Mathematical Theory of Automata, Apr 1962.

20. Tomplins, C. B., Methods of Steep Descent, Modern Mathematics for the Engineer, Ed. E.F. Beckenback, McGraw-Hill, 1956.

21. Carpenter, G. A. and Grossberg, S., The Art of Adaptive Pattern Recognition by a Self-Organizing Neural Network, IEEE Computer, Mar 1988.

22. Fukushima, K. and Miyake, S., Neocognitron: A New Algorithm for Pattern Recognition Tolerant of Defurmation and Shifts in Position, Pattern Recognition, Vol. 15, No. 6, 1982.

23. Fukushima, K., A Neural Network for Visual Pattern Recognition, IEEE Computer, Mar 1988.

24. Grogan, T. A. and Johnson, J. D., An Artificial Neural System for Object Classification and Orientation Estimation, STAS Report, Apr 1990.

25. Reeves, A. and Prokop, R. J., A Neural Network Object Recognition System, STAS Report, Feb 1990.

26. Rosenfield, I., The Invention of Memory, Basic Books, 1989.

27. Roth, M. W., Survey of Neural Network Technology for Automatic Target Recognition, IEEE Transactions on Neural Networks, Vol I, No. 1, Mar 1990.

28. Reggia, J. A. and Sutton, G. G., Self-Processing Networks and Their Biomedical Implications, Proc. IEEE, Vol. 76, No. 6, Jun 1988.

29. Von Neumann, J., The Computer and the Brain, Yale University Press, 1958.

30. Bilbru, G. L. and Hiriyannaiah, H., Fusion of Range and Reflectance Image Data Using Markov Random Fields, Proc. IEEE Int. Symp. Intelligent Control, Piscataway, NJ, 1988.

DISTRIBUTION LIST

Commander
Armament Research, Development and Engineering Center
U.S. Army Armament, Munitions and Chemical Command
ATTN: SMCAR-IMI-I (5)
SMCAR-FSF-RC (7)
Picatinny Arsenal, NJ 07806-5000

Commander
U.S. Army Armament, Munitions and Chemical Command
ATTN: AMSMC-GCL (D)
Picatinny Arsenal, NJ 07806-5000

Administrator
Defense Technical Information Center
ATTN: Accessions Division (12)
Cameron Station
Alexandria, VA 22304-6145

Director
U.S. Army Materiel Systems Analysis Activity
ATTN: AMXSY-MP
Aberdeen Proving Ground, MD 21005-5066

Commander
Chemical Research, Development and Engineering Center
U.S. Army Armament, Munitions and Chemical Command
ATTN: SMCCR-MSI
Aberdeen Proving Ground, MD 21010-5423

Commander
Chemical Research, Development and Engineering Center
U.S. Army Armament, Munitions and Chemical Command
ATTN: SMCCR-RSP-A
Aberdeen Proving Ground, MD 21010-5423

Director
Ballistic Research Laboratory
ATTN: AMXBR-OD-ST
Aberdeen Proving Ground, MD 21005-5066

Chief
Benet Weapons Laboratory, CCAC
Armament Research, Development and Engineering Center
U.S. Army Armament, Munitions and Chemical Command
ATTN: SMCAR-CCB-TL
Watervliet, NY 12189-5000

Commander
U.S. Army Armament, Munitions and Chemical Command
ATTN: AMSMC-IMF-L
Rock Island, IL 61299-6000

Director
U.S. Army TRADOC Systems Analysis Activity
ATTN: ATAA-SL
White Sands Missile Range, NM 88002