NEURAL NETWORKS

by Christos Stergiou and Dimitrios Siganos

Abstract

This report is an introduction to Artificial Neural Networks. The various types of neural networks are explained and demonstrated, applications of neural networks like ANNs in medicine are described, and a detailed historical background is provided. The connection between the artificial and the real thing is also investigated and explained. Finally, the mathematical models involved are presented and demonstrated.

Contents:

1. Introduction to neural networks

1.1 What is a Neural Network?

An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information. The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements (neurones) working in unison to solve specific problems. ANNs, like people, learn by example. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurones. This is true of ANNs as well.

1.2 Historical background

Neural network simulations appear to be a recent development. However, this field was established before the advent of computers, and has survived at least one major setback and several eras.

Many importand advances have been boosted by the use of inexpensive computer emulations. Following an initial period of enthusiasm, the field survived a period of frustration and disrepute. During this period when funding and professional support was minimal, important advances were made by relatively few reserchers. These pioneers were able to develop convincing technology which surpassed the limitations identified by Minsky and Papert. Minsky and Papert, published a book (in 1969) in which they summed up a general feeling of frustration (against neural networks) among researchers, and was thus accepted by most without further analysis.

Currently, the neural network field enjoys a resurgence of interest and a corresponding increase in funding.

For a more detailed description of the history click [here](#)

The first artificial neuron was produced in 1943 by the neurophysiologist Warren McCulloch and the logician Walter Pits. But the technology available at that time did not allow them to do too much.

1.3 Why use neural networks?

Neural networks, with their remarkable ability to derive meaning from complicated or imprecise data, can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques. A trained neural network can be thought of as an "expert" in the category of information it has been given to analyse. This expert can then be used to provide projections given new situations of interest and answer "what if" questions.
Other advantages include:

1. Adaptive learning: An ability to learn how to do tasks based on the data given for training or initial experience.
2. Self-Organisation: An ANN can create its own organisation or representation of the information it receives during learning time.
3. Real Time Operation: ANN computations may be carried out in parallel, and special hardware devices are being designed and manufactured which take advantage of this capability.
4. Fault Tolerance via Redundant Information Coding: Partial destruction of a network leads to the corresponding degradation of performance. However, some network capabilities may be retained even with major network damage.

1.4 Neural networks versus conventional computers

Neural networks take a different approach to problem solving than that of conventional computers. Conventional computers use an algorithmic approach i.e. the computer follows a set of instructions in order to solve a problem. Unless the specific steps that the computer needs to follow are known the computer cannot solve the problem. That restricts the problem solving capability of conventional computers to problems that we already understand and know how to solve. But computers would be so much more useful if they could do things that we don't exactly know how to do.

Neural networks process information in a similar way the human brain does. The network is composed of a large number of highly interconnected processing elements(neurones) working in parallel to solve a specific problem. Neural networks learn by example. They cannot be programmed to perform a specific task. The examples must be selected carefully otherwise useful time is wasted or even worse the network might be functioning incorrectly. The disadvantage is that because the network finds out how to solve the problem by itself, its operation can be unpredictable.

On the other hand, conventional computers use a cognitive approach to problem solving; the way the problem is to solved must be known and stated in small unambiguous instructions. These instructions are then converted to a high level language program and then into machine code that the computer can understand. These machines are totally predictable; if anything goes wrong is due to a software or hardware fault.

Neural networks and conventional algorithmic computers are not in competition but complement each other. There are tasks are more suited to an algorithmic approach like arithmetic operations and tasks that are more suited to neural networks. Even more, a large number of tasks, require systems that use a combination of the two approaches (normally a conventional computer is used to supervise the neural network) in order to perform at maximum efficiency.

Neural networks do not perform miracles. But if used sensibly they can produce some amazing results.

Back to Contents

2. Human and Artificial Neurones - investigating the similarities

2.1 How the Human Brain Learns?

Much is still unknown about how the brain trains itself to process information, so theories abound. In the human brain, a typical neuron collects signals from others through a host of fine structures called *dendrites*. The neuron sends out spikes of electrical activity through a long, thin stand known as an *axon*, which splits into thousands of branches. At the end of each branch, a structure called a *synapse* converts the activity from the axon into electrical effects that inhibit or excite activity from the axon into electrical effects that inhibit or excite activity in the connected neurones. When a neuron receives excitatory input that is sufficiently large compared with its inhibitory input, it sends a spike of electrical activity down its axon. Learning occurs by changing the effectiveness of the synapses so that the influence of one neuron on another changes.
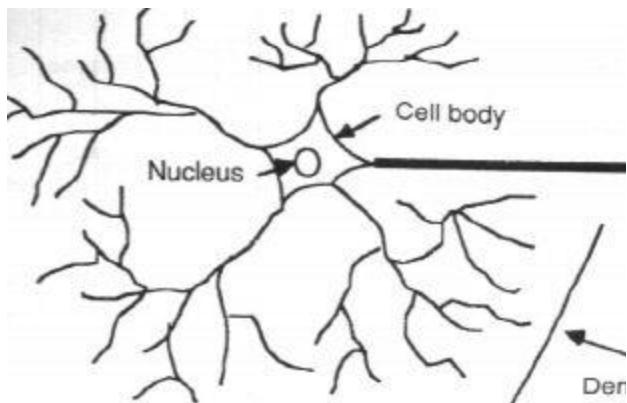
Components of a neuron
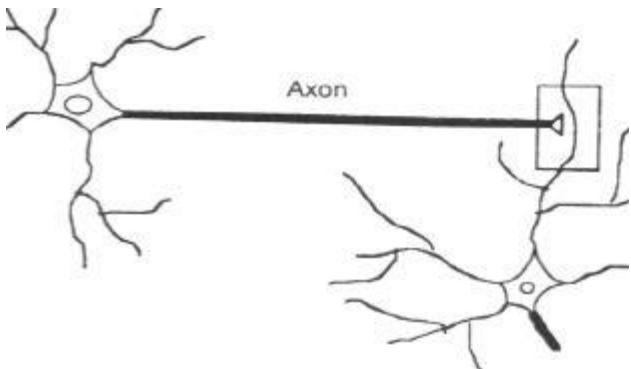
The synapse

2.2 From Human Neurones to Artificial Neurones

We conduct these neural networks by first trying to deduce the essential features of neurones and their interconnections. We then typically program a computer to simulate these features. However because our knowledge of neurones is incomplete and our computing power is limited, our models are necessarily gross idealisations of real networks of neurones.
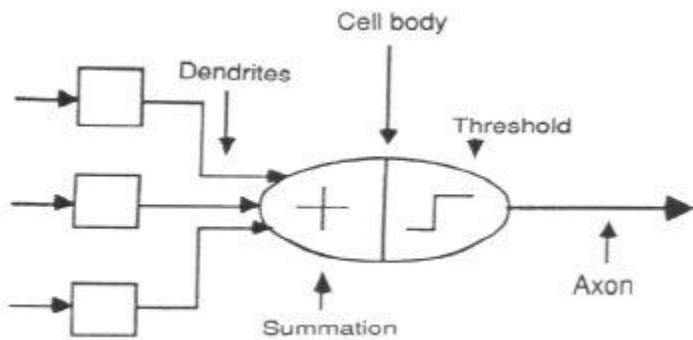
The neuron model

🔴**Back to Contents**

3. An engineering approach

3.1 A simple neuron

An artificial neuron is a device with many inputs and one output. The neuron has two modes of operation; the training mode and the using mode. In the training mode, the neuron can be trained to fire (or not), for particular input patterns. In the using mode, when a taught input pattern is detected at the input, its associated output becomes the current output. If the input pattern does not belong in the taught list of input patterns, the firing rule is used to determine whether to fire or not.
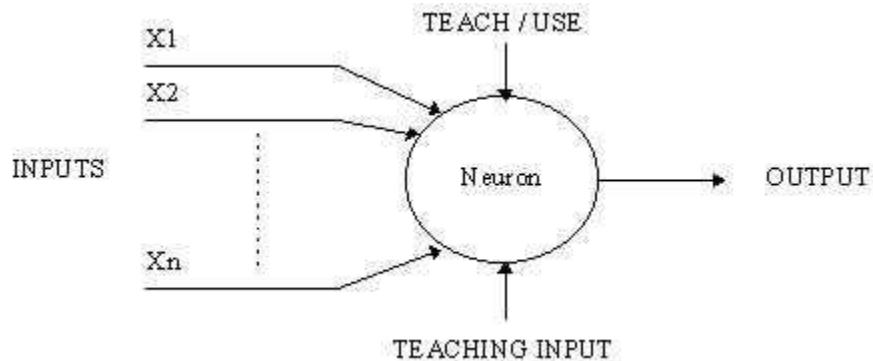
A simple neuron

3.2 Firing rules

The firing rule is an important concept in neural networks and accounts for their high flexibility. A firing rule determines how one calculates whether a neuron should fire for any input pattern. It relates to all the input patterns, not only the ones on which the node was trained.

A simple firing rule can be implemented by using Hamming distance technique. The rule goes as follows:

Take a collection of training patterns for a node, some of which cause it to fire (the 1-taught set of patterns) and others which prevent it from doing so (the 0-taught set). Then the patterns not in the collection cause the node to fire if, on comparison , they have more input elements in common with the 'nearest' pattern in the 1-taught set than with the 'nearest' pattern in the 0-taught set. If there is a tie, then the pattern remains in the undefined state.

For example, a 3-input neuron is taught to output 1 when the input (X1,X2 and X3) is 111 or 101 and to output 0 when the input is 000 or 001. Then, before applying the firing rule, the truth table is;

X1:		0	0	0	0	1	1	1	1
X2:		0	0	1	1	0	0	1	1
X3:		0	1	0	1	0	1	0	1
OUT:		0	0	0/1	0/1	0/1	1	0/1	1

As an example of the way the firing rule is applied, take the pattern 010. It differs from 000 in 1 element, from 001 in 2 elements, from 101 in 3 elements and from 111 in 2 elements. Therefore, the 'nearest' pattern is 000 which belongs in the 0-taught set. Thus the firing rule requires that the neuron should not fire when the input is 001. On the other hand, 011 is equally distant from two taught patterns that have different outputs and thus the output stays undefined (0/1).

By applying the firing in every column the following truth table is obtained;

X1:	0	0	0	0	1	1	1	1
X2:	0	0	1	1	0	0	1	1
X3:	0	1	0	1	0	1	0	1
OUT:	0	0	0	0/1	0/1	1	1	1

The difference between the two truth tables is called the *generalisation of the neuron.* Therefore the firing rule gives the neuron a sense of similarity and enables it to respond 'sensibly' to patterns not seen during training.

3.3 Pattern Recognition - an example

An important application of neural networks is pattern recognition. Pattern recognition can be implemented by using a feed-forward (figure 1) neural network that has been trained accordingly. During training, the network is trained to associate outputs with input patterns. When the network is used, it identifies the input pattern and tries to output the associated output pattern. The power of neural networks comes to life when a pattern that has no output associated with it, is given as an input. In this case, the network gives the output that corresponds to a taught input pattern that is least different from the given pattern.
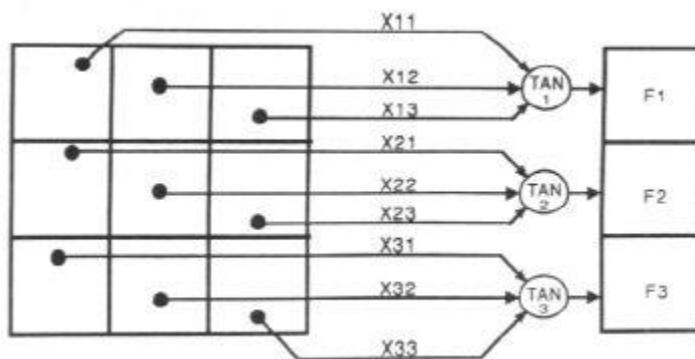
Figure 1.

For example:
The network of figure 1 is trained to recognise the patterns T and H. The associated patterns are all black and all white respectively as shown below.

INPUT   OUTPUT   INPUT   OUTPUT

If we represent black squares with 0 and white squares with 1 then the truth tables for the 3 neurones after generalisation are;

X11:	0	0	0	0	1	1	1	1
X12:	0	0	1	1	0	0	1	1
X13:	0	1	0	1	0	1	0	1
OUT:	0	0	1	1	0	0	1	1

Top neuron

X21:	0	0	0	0	1	1	1	1
X22:	0	0	1	1	0	0	1	1
X23:	0	1	0	1	0	1	0	1
OUT:	1	0/1	1	0/1	0/1	0	0/1	0

Middle neuron

X21:	0	0	0	0	1	1	1	1
X22:	0	0	1	1	0	0	1	1
X23:	0	1	0	1	0	1	0	1
OUT:	1	0	1	1	0	0	1	0

Bottom neuron

From the tables it can be seen the following associasions can be extracted:
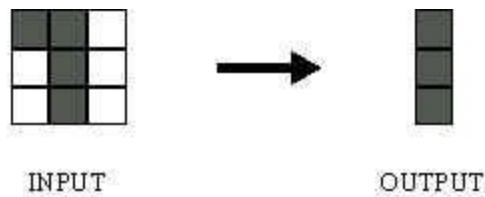
INPUT   OUTPUT

In this case, it is obvious that the output should be all blacks since the input pattern is almost the same as the 'T' pattern.
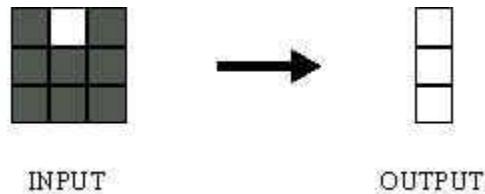
INPUT   OUTPUT

Here also, it is obvious that the output should be all whites since the input pattern is almost the same as the 'H' pattern.

INPUT   OUTPUT
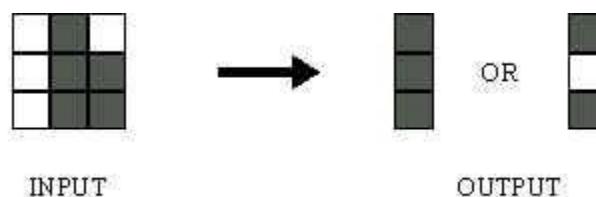
Here, the top row is 2 errors away from the a T and 3 from an H. So the top output is black. The middle row is 1 error away from both T and H so the output is random. The bottom row is 1 error away from T and 2 away from H. Therefore the output is black. The total output of the network is still in favour of the T shape.

3.4 A more complicated neuron

The previous neuron doesn't do anything that conventional conventional computers don't do already. A more sophisticated neuron (figure 2) is the McCulloch and Pitts model (MCP). The difference from the previous model is that the inputs are 'weighted', the effect that each input has at decision making is dependent on the weight of the particular input. The weight of an input is a number which when multiplied with the input gives the weighted input. These weighted inputs are then added together and if they exceed a pre-set threshold value, the neuron fires. In any other case the neuron does not fire.
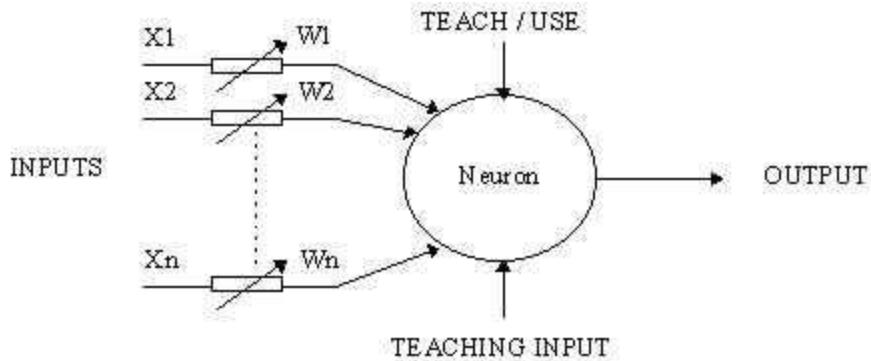
Figure 2. An MCP neuron

In mathematical terms, the neuron fires if and only if;

$$X_1W_1 + X_2W_2 + X_3W_3 + ... > T$$

The addition of input weights and of the threshold makes this neuron a very flexible and powerful one. The MCP neuron has the ability to adapt to a particular situation by changing its weights and/or threshold. Various algorithms exist that cause the neuron to 'adapt'; the most used ones are the Delta rule and the back error propagation. The former is used in feed-forward networks and the latter in feedback networks.

● **Back to Contents**

4 Architecture of neural networks

4.1 Feed-forward networks

Feed-forward ANNs (figure 1) allow signals to travel one way only; from input to output. There is no feedback (loops) i.e. the output of any layer does not affect that same layer. Feed-forward ANNs tend to be straight forward networks that associate inputs with outputs. They are extensively used in pattern recognition. This type of organisation is also referred to as bottom-up or top-down.

4.2 Feedback networks

Feedback networks (figure 1) can have signals travelling in both directions by introducing loops in the network. Feedback networks are very powerful and can get extremely complicated. Feedback networks are dynamic; their 'state' is changing continuously until they reach an equilibrium point. They remain at the equilibrium point until the input changes and a new equilibrium needs to be found. Feedback architectures are also referred to as interactive or recurrent, although the latter term is often used to denote feedback connections in single-layer organisations.
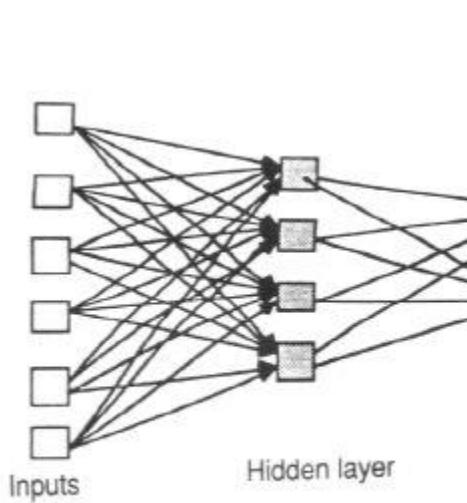
Figure 4.1 An example of a simple feedforward network
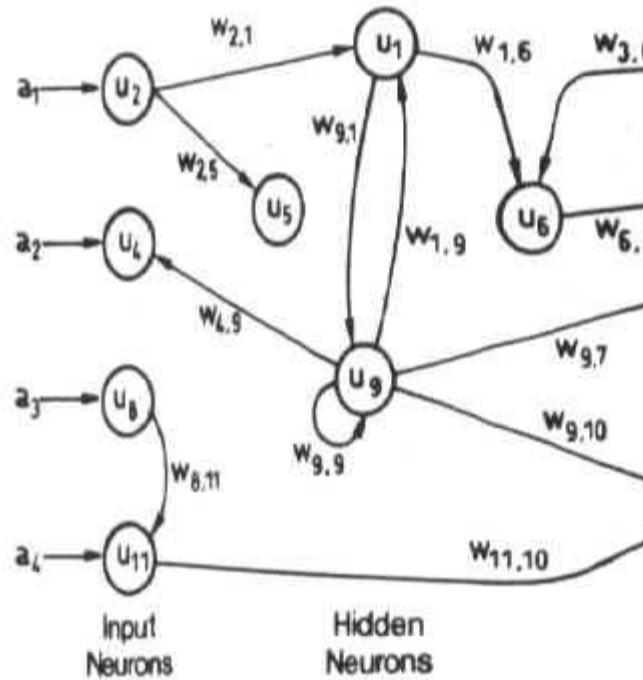
Figure 4.2 An example of a complicated network

4.3 Network layers

The commonest type of artificial neural network consists of three groups, or layers, of units: a layer of "**input**" units is connected to a layer of "**hidden**" units, which is connected to a layer of "**output**" units. (see Figure 4.1)

● The activity of the input units represents the raw information that is fed into the network.

● The activity of each hidden unit is determined by the activities of the input units and the weights on the connections between the input and the hidden units.

● The behaviour of the output units depends on the activity of the hidden units and the weights between the hidden and output units.

This simple type of network is interesting because the hidden units are free to construct their own representations of the input. The weights between the input and hidden units determine when each hidden unit is active, and so by modifying these weights, a hidden unit can choose what it represents.

We also distinguish single-layer and multi-layer architectures. The single-layer organisation, in which all units are connected to one another, constitutes the most general case and is of more potential computational power than hierarchically structured multi-layer organisations. In multi-layer networks, units are often numbered by layer, instead of following a global numbering.

4.4 Perceptrons

The most influential work on neural nets in the 60's went under the heading of 'perceptrons' a term coined by Frank Rosenblatt. The perceptron (figure 4.4) turns out to be an MCP model (neuron with weighted inputs) with some additional, fixed, pre--processing. Units labelled A1, A2, Aj , Ap are called association units and their task is to extract specific, localised featured from the input images. Perceptrons mimic the basic idea behind the mammalian visual system. They were mainly used in pattern recognition even though their capabilities extended a lot more.
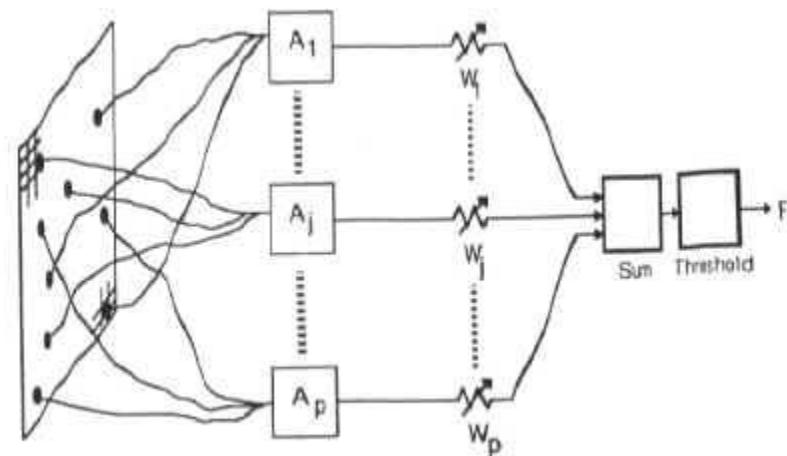
Figure 4.4

In 1969 Minsky and Papert wrote a book in which they described the limitations of single layer Perceptrons. The impact that the book had was tremendous and caused a lot of neural network researchers to loose their interest. The book was very well written and showed mathematically that *single layer* perceptrons could not do some basic pattern recognition operations like determining the parity of a shape or determining whether a shape is connected or not. What they did not realised, until the 80's, is that given the appropriate training, multilevel perceptrons can do these operations.

● **Back to Contents**

5. The Learning Process

The memorisation of patterns and the subsequent response of the network can be categorised into two general paradigms:

🔴**associative mapping** in which the network learns to produce a particular pattern on the set of input units whenever another particular pattern is applied on the set of input units. The associtive mapping can generally be broken down into two mechanisms:

🟢*auto-association*: an input pattern is associated with itself and the states of input and output units coincide. This is used to provide pattern completition, ie to produce a pattern whenever a portion of it or a distorted pattern is presented. In the second case, the network actually stores pairs of patterns building an association between two sets of patterns.

🟢*hetero-association*: is related to two recall mechanisms:

🔵*nearest-neighbour* recall, where the output pattern produced corresponds to the input pattern stored, which is closest to the pattern presented, and

🔵*interpolative* recall, where the output pattern is a similarity dependent interpolation of the patterns stored corresponding to the pattern presented. Yet another paradigm, which is a variant associative mapping is classification, ie when there is a fixed set of categories into which the input patterns are to be classified.

🔴**regularity detection** in which units learn to respond to particular properties of the input patterns. Whereas in asssociative mapping the network stores the relationships among patterns, in regularity detection the response of each unit has a particular 'meaning'. This type of learning mechanism is essential for feature discovery and knowledge representation.

Every neural network posseses knowledge which is contained in the values of the connections weights. Modifying the knowledge stored in the network as a function of experience implies a learning rule for changing the values of the weights.
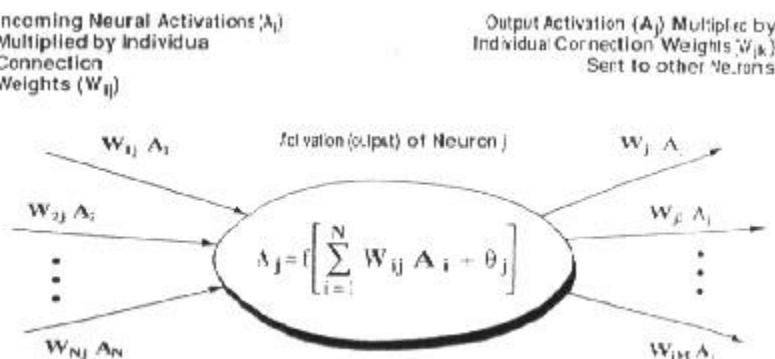
Information is stored in the weight matrix W of a neural network. Learning is the determination of the weights. Following the way learning is performed, we can distinguish two major categories of neural networks:

🐞 **fixed networks** in which the weights cannot be changed, ie dW/dt=0. In such networks, the weights are fixed a priori according to the problem to solve.

🐞 **adaptive networks** which are able to change their weights, ie dW/dt not= 0.

All learning methods used for adaptive neural networks can be classified into two major categories:

🐞 **Supervised learning** which incorporates an external teacher, so that each output unit is told what its desired response to input signals ought to be. During the learning process global information may be required. Paradigms of supervised learning include error-correction learning, reinforcement learning and stochastic learning.
An important issue conserning supervised learning is the problem of error convergence, ie the minimisation of error between the desired and computed unit values. The aim is to determine a set of weights which minimises the error. One well-known method, which is common to many learning paradigms is the least mean square (LMS) convergence.

🐞 **Unsupervised learning** uses no external teacher and is based upon only local information. It is also referred to as self-organisation, in the sense that it self-organises data presented to the network and detects their emergent collective properties. Paradigms of unsupervised learning are Hebbian lerning and competitive learning.
Ano2.2 From Human Neurones to Artificial Neuronesther aspect of learning concerns the distinction or not of a seperate phase, during which the network is trained, and a subsequent operation phase. We say that a neural network learns off-line if the learning phase and the operation phase are distinct. A neural network learns on-line if it learns and operates at the same time. Usually, supervised learning is performed off-line, whereas usupervised learning is performed on-line.

5.1 Transfer Function

The behaviour of an ANN (Artificial Neural Network) depends on both the weights and the input-output function (transfer function) that is specified for the units. This function typically falls into one of three categories:

🐞 linear (or ramp)

🐞 threshold

🐞 sigmoid

For **linear units**, the output activity is proportional to the total weighted output.

For **threshold units**, the output is set at one of two levels, depending on whether the total input is greater than or less than some threshold value.

For **sigmoid units**, the output varies continuously but not linearly as the input changes. Sigmoid units bear a greater resemblance to real neurones than do linear or threshold units, but all three must be considered rough approximations.

To make a neural network that performs some specific task, we must choose how the units are connected to one another (see figure 4.1), and we must set the weights on the connections appropriately. The connections determine whether it is possible for one unit to influence another. The weights specify the strength of the influence.

We can teach a three-layer network to perform a particular task by using the following procedure:

1. We present the network with training examples, which consist of a pattern of activities for the input units together with the desired pattern of activities for the output units.
2. We determine how closely the actual output of the network matches the desired output.
3. We change the weight of each connection so that the network produces a better approximation of the desired output.

5.2 An Example to illustrate the above teaching procedure:

Assume that we want a network to recognise hand-written digits. We might use an array of, say, 256 sensors, each recording the presence or absence of ink in a small area of a single digit. The network would therefore need 256 input units (one for each sensor), 10 output units (one for each kind of digit) and a number of hidden units.

For each kind of digit recorded by the sensors, the network should produce high activity in the appropriate output unit and low activity in the other output units.

To train the network, we present an image of a digit and compare the actual activity of the 10 output units with the desired activity. We then calculate the error, which is defined as the square of the difference between the actual and the desired activities. Next we change the weight of each connection so as to reduce the error.We repeat this training process for many different images of each different images of each kind of digit until the network classifies every image correctly.

To implement this procedure we need to calculate the error derivative for the weight (EW) in order to change the weight by an amount that is proportional to the rate at which the error changes as the weight is changed. One way to calculate the EW is to perturb a weight slightly and observe how the error changes. But that method is inefficient because it requires a separate perturbation for each of the many weights.

Another way to calculate the EW is to use the Back-propagation algorithm which is described below, and has become nowadays one of the most important tools for training neural networks. It was developed independently by two teams, one (Fogelman-Soulie, Gallinari and Le Cun) in France, the other (Rumelhart, Hinton and Williams) in U.S.

5.3 The Back-Propagation Algorithm

In order to train a neural network to perform some task, we must adjust the weights of each unit in such a way that the error between the desired output and the actual output is reduced. This process requires that the neural network compute the error derivative of the weights (**EW**). In other words, it must calculate how the error changes as each weight is increased or decreased slightly. The back propagation algorithm is the most widely used method for determining the **EW**.

The back-propagation algorithm is easiest to understand if all the units in the network are linear. The algorithm computes each **EW** by first computing the **EA**, the rate at which the error changes as the activity level of a unit is changed. For output units, the **EA** is simply the difference between the actual and the desired output. To compute the **EA** for a hidden unit in the layer just before the output layer, we first identify all the weights between that hidden unit and the output units to which it is connected. We then multiply those weights by the **EA**s of those output units and add the products. This sum equals the **EA** for the chosen hidden unit. After calculating all the **EA**s in the hidden layer just before the output layer, we can compute in like fashion the **EA**s for other layers, moving from layer to layer in a direction opposite to the way activities propagate through the network. This is what gives back propagation its name. Once the **EA** has been computed for a unit, it is straight forward to compute the **EW** for each incoming connection of the unit. The **EW** is the product of the EA and the activity through the incoming connection.

Note that for non-linear units, (see Appendix C) the back-propagation algorithm includes an extra step. Before back-propagating, the **EA** must be converted into the **EI**, the rate at which the error changes as the total input received by a unit is changed.

● **Back to Contents**

6. Applications of neural networks

6.1 Neural Networks in Practice

Given this description of neural networks and how they work, what real world applications are they suited for? Neural networks have broad applicability to real world business problems. In fact, they have already been successfully applied in many industries.

Since neural networks are best at identifying patterns or trends in data, they are well suited for prediction or forecasting needs including:

- sales forecasting

- industrial process control

- customer research

- data validation

- risk management

- target marketing

But to give you some more specific examples; ANN are also used in the following specific paradigms: recognition of speakers in communications; diagnosis of hepatitis; recovery of telecommunications from faulty software; interpretation of multimeaning Chinese words; undersea mine detection; texture analysis; three-dimensional object recognition; hand-written word recognition; and facial recognition.

6.2 Neural networks in medicine

Artificial Neural Networks (ANN) are currently a 'hot' research area in medicine and it is believed that they will receive extensive application to biomedical systems in the next few years. At the moment, the research is mostly on modelling parts of the human body and recognising diseases from various scans (e.g. cardiograms, CAT scans, ultrasonic scans, etc.).

Neural networks are ideal in recognising diseases using scans since there is no need to provide a specific algorithm on how to identify the disease. Neural networks learn by example so the details of how to recognise the disease are not needed. What is needed is a set of examples that are representative of all the variations of the disease. The quantity of examples is not as important as the 'quantity'. The examples need to be selected very carefully if the system is to perform reliably and efficiently.

6.2.1 Modelling and Diagnosing the Cardiovascular System

Neural Networks are used experimentally to model the human cardiovascular system. Diagnosis can be achieved by building a model of the cardiovascular system of an individual and comparing it with the real time physiological measurements taken from the patient. If this routine is carried out regularly, potential harmful medical conditions can be detected at an early stage and thus make the process of combating the disease much easier.

A model of an individual's cardiovascular system must mimic the relationship among physiological variables (i.e., heart rate, systolic and diastolic blood pressures, and breathing rate) at different physical activity levels. If a model is adapted to an individual, then it becomes a model of the physical condition of that individual. The simulator will have to be able to adapt to

the features of any individual without the supervision of an expert. This calls for a neural network.

Another reason that justifies the use of ANN technology, is the ability of ANNs to provide sensor fusion which is the combining of values from several different sensors. Sensor fusion enables the ANNs to learn complex relationships among the individual sensor values, which would otherwise be lost if the values were individually analysed. In medical modelling and diagnosis, this implies that even though each sensor in a set may be sensitive only to a specific physiological variable, ANNs are capable of detecting complex medical conditions by fusing the data from the individual biomedical sensors.

6.2.2 Electronic noses

ANNs are used experimentally to implement electronic noses. Electronic noses have several potential applications in telemedicine. Telemedicine is the practice of medicine over long distances via a communication link. The electronic nose would identify odours in the remote surgical environment. These identified odours would then be electronically transmitted to another site where an door generation system would recreate them. Because the sense of smell can be an important sense to the surgeon, telesmell would enhance telepresent surgery.
For more information on telemedicine and telepresent surgery click here.

6.2.3 Instant Physician

An application developed in the mid-1980s called the "instant physician" trained an autoassociative memory neural network to store a large number of medical records, each of which includes information on symptoms, diagnosis, and treatment for a particular case. After training, the net can be presented with input consisting of a set of symptoms; it will then find the full stored pattern that represents the "best" diagnosis and treatment.

6.3 Neural Networks in business

Business is a diverted field with several general areas of specialisation such as accounting or financial analysis. Almost any neural network application would fit into one business area or financial analysis.
There is some potential for using neural networks for business purposes, including resource allocation and scheduling. There is also a strong potential for using neural networks for database mining, that is, searching for patterns implicit within the explicitly stored information in databases. Most of the funded work in this area is classified as proprietary. Thus, it is not possible to report on the full extent of the work going on. Most work is applying neural networks, such as the Hopfield-Tank network for optimization and scheduling.

6.3.1 Marketing

There is a marketing application which has been integrated with a neural network system. The Airline Marketing Tactician (a trademark abbreviated as AMT) is a computer system made of various intelligent technologies including expert systems. A feedforward neural network is integrated with the AMT and was trained using back-propagation to assist the marketing control of airline seat allocations. The adaptive neural approach was amenable to rule expression. Additionaly, the application's environment changed rapidly and constantly, which required a continuously adaptive solution. The system is used to monitor and recommend booking advice for each departure. Such information has a direct impact on the profitability of an airline and can provide a technological advantage for users of the system. [Hutchison & Stephens, 1987]

While it is significant that neural networks have been applied to this problem, it is also important to see that this intelligent technology can be integrated with expert systems and other approaches to make a functional system. Neural networks were used to discover the influence of undefined interactions by the various variables. While these interactions were not defined, they were used by the neural system to develop useful conclusions. It is also noteworthy to see that neural networks can influence the bottom line.

6.3.2 Credit Evaluation

The HNC company, founded by Robert Hecht-Nielsen, has developed several neural network applications. One of them is the Credit Scoring system which increase the profitability of the existing model up to 27%. The HNC neural systems were also applied to mortgage screening. A neural network automated mortgage insurance underwritting system was developed by the Nestor Company. This system was trained with 5048 applications of which 2597 were certified. The data related to property and borrower qualifications. In a conservative mode the system agreed on the underwritters on 97% of the cases. In the liberal model the system agreed 84% of the cases. This is system run on an Apollo DN3000 and used 250K memory while processing a case file in approximately 1 sec.

Back to Contents

---

7. Conclusion

The computing world has a lot to gain fron neural networks. Their ability to learn by example makes them very flexible and powerful. Furthermore there is no need to devise an algorithm in order to perform a specific task; i.e. there is no need to understand the internal mechanisms of that task. They are also very well suited for real time systems because of their fast responseand computational times which are due to their parallel architecture.

Neural networks also contribute to other areas of research such as neurology and psychology. They are regularly used to model parts of living organisms and to investigate the internal mechanisms of the brain.

Perhaps the most exciting aspect of neural networks is the possibility that some day 'consious' networks might be produced. There is a number of scientists arguing that conciousness is a 'mechanical' property and that 'consious' neural networks are a realistic possibility.

Finally, I would like to state that even though neural networks have a huge potential we will only get the best of them when they are intergrated with computing, AI, fuzzy logic and related subjects.

● **Back to Contents**

References:

1. An introduction to neural computing. Aleksander, I. and Morton, H. 2nd edition
2. Neural Networks at Pacific Northwest National Laboratory
   http://www.emsl.pnl.gov:2080/docs/cie/neural/neural.homepage.html
3. Industrial Applications of Neural Networks (research reports Esprit, I.F.Croall, J.P.Mason)
4. A Novel Approach to Modelling and Diagnosing the Cardiovascular System
   http://www.emsl.pnl.gov:2080/docs/cie/neural/papers2/keller.wcnn95.abs.html
5. Artificial Neural Networks in Medicine
   http://www.emsl.pnl.gov:2080/docs/cie/techbrief/NN.techbrief.ht
6. Neural Networks by Eric Davalo and Patrick Naim
7. Learning internal representations by error propagation by Rumelhart, Hinton and Williams (1986).
8. Klimasauskas, CC. (1989). The 1989 Neuro Computing Bibliography. Hammerstrom, D. (1986). A Connectionist/Neural Network Bibliography.
9. DARPA Neural Network Study (October, 1987-February, 1989). MIT Lincoln Lab. Neural Networks, Eric Davalo and Patrick Naim
10. Assimov, I (1984, 1950), Robot, Ballatine, New York.
11. Electronic Noses for Telemedicine
    http://www.emsl.pnl.gov:2080/docs/cie/neural/papers2/keller.ccc95.abs.html
12. Pattern Recognition of Pathology Images
    http://kopernik-eth.npac.syr.edu:1200/Task4/pattern.html

● **Back to Contents**

Appendix A - Historical background in detail

The history of neural networks that was described above can be divided into several periods:

1. **First Attempts:** There were some initial simulations using formal logic. McCulloch and Pitts (1943) developed models of neural networks based on their understanding of neurology. These models made several assumptions about how neurons worked. Their networks were based on simple neurons which were considered to be binary devices with fixed thresholds. The results of their model were simple logic functions such as "a or b" and "a and b". Another attempt was by using computer simulations. Two groups (Farley and Clark, 1954; Rochester, Holland, Haibit and Duda, 1956). The first group (IBM reserchers) maintained closed contact with neuroscientists at McGill University. So whenever their models did not work, they consulted the neuroscientists. This interaction established a multidiscilinary trend which continues to the present day.

2. **Promising & Emerging Technology:** Not only was neroscience influential in the development of neural networks, but psychologists and engineers also contributed to the progress of neural network simulations. Rosenblatt (1958) stirred considerable interest and activity in the field when he designed and developed the Perceptron. The Perceptron had three layers with the middle layer known as the association layer.This system could learn to connect or associate a given input to a random output unit.
   Another system was the ADALINE (ADAptive LInear Element) which was developed in 1960 by Widrow and Hoff (of Stanford University). The ADALINE was an analogue electronic device made from simple components. The method used for learning was different to that of the Perceptron, it employed the Least-Mean-Squares (LMS) learning rule.

3. **Period of Frustration & Disrepute:** In 1969 Minsky and Papert wrote a book in which they generalised the limitations of single layer Perceptrons to multilayered systems. In the book they said: "...our intuitive judgment that the extension (to multilayer systems) is sterile". The significant result of their book was to eliminate funding for research with neural network simulations. The conclusions supported the disenhantment of reserchers in the field. As a result, considerable prejudice against this field was activated.

4. **Innovation:** Although public interest and available funding were minimal, several researchers continued working to develop neuromorphically based computaional methods for problems such as pattern recognition.
   During this period several paradigms were generated which modern work continues to enhance.Grossberg's (Steve Grossberg and Gail Carpenter in 1988) influence founded a school of thought which explores resonating algorithms. They developed the ART (Adaptive Resonance Theory) networks based on biologically plausible models. Anderson and Kohonen developed associative techniques independent of each other. Klopf (A. Henry Klopf) in 1972, developed a basis for learning in artificial neurons based on a biological principle for neuronal learning called heterostasis.
   Werbos (Paul Werbos 1974) developed and used the back-propagation learning method, however several years passed before this approach was popularized. Back-propagation nets are probably the most well known and widely applied of the neural networks today. In essence, the back-propagation net. is a Perceptron with multiple layers, a different threshold function in the artificial neuron, and a more robust and capable learning rule. Amari (A. Shun-Ichi 1967) was involved with theoretical developments: he published a paper which established a mathematical theory for a learning basis (error-correction

method) dealing with adaptive patern classification. While Fukushima (F. Kunihiko) developed a step wise trained multilayered neural network for interpretation of handwritten characters. The original network was published in 1975 and was called the Cognitron.

5. **Re-Emergence:** Progress during the late 1970s and early 1980s was important to the re-emergence on interest in the neural network field. Several factors influenced this movement. For example, comprehensive books and conferences provided a forum for people in diverse fields with specialized technical languages, and the response to conferences and publications was quite positive. The news media picked up on the increased activity and tutorials helped disseminate the technology. Academic programs appeared and courses were inroduced at most major Universities (in US and Europe). Attention is now focused on funding levels throughout Europe, Japan and the US and as this funding becomes available, several new commercial with applications in industry and finacial institutions are emerging.

6. **Today:** Significant progress has been made in the field of neural networks-enough to attract a great deal of attention and fund further research. Advancement beyond current commercial applications appears to be possible, and research is advancing the field on many fronts. Neurally based chips are emerging and applications to complex problems developing. Clearly, today is a period of transition for neural network technology.

Appendix B - The back-propagation Algorithm - a mathematical approach

Units are connected to one another. Connections correspond to the edges of the underlying directed graph. There is a real number associated with each connection, which is called the weight of the connection. We denote by W_{ij} the weight of the connection from unit u_i to unit u_j. It is then convenient to represent the pattern of connectivity in the network by a weight matrix W whose elements are the weights W_{ij}. Two types of connection are usually distinguished: excitatory and inhibitory. A positive weight represents an excitatory connection whereas a negative weight represents an inhibitory connection. The pattern of connectivity characterises the architecture of the network.

Input to neuron i | Output from neuron i | Input to other neurons
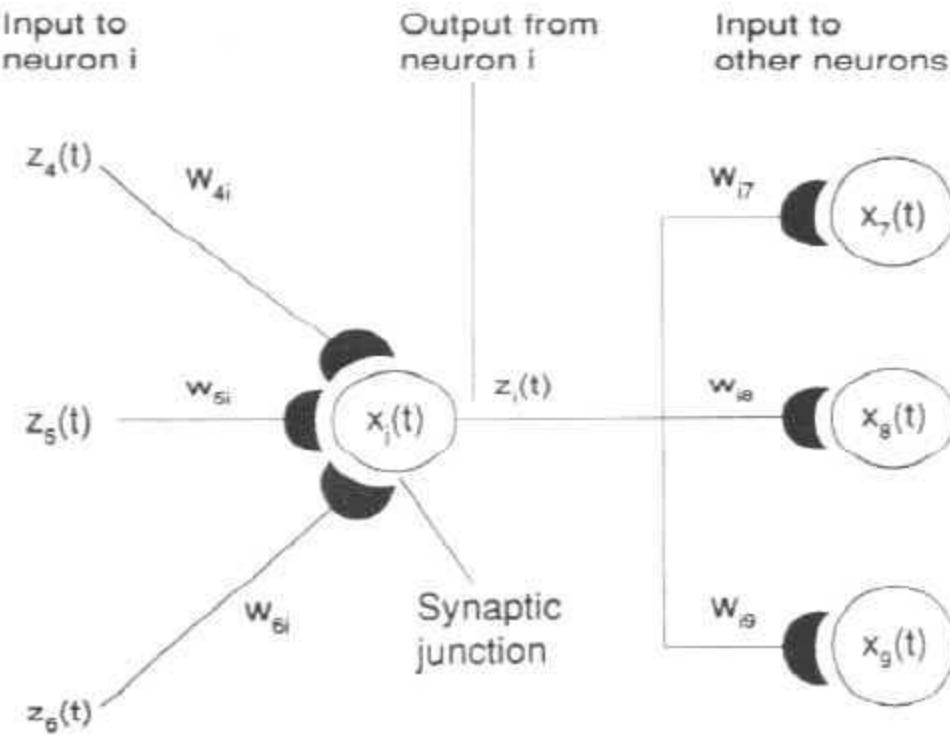
A unit in the output layer determines its activity by following a two step procedure.

● First, it computes the total weighted input x_j, using the formula:

$$X_j = \sum_i y_i W_{ij}$$

where y_i is the activity level of the jth unit in the previous layer and W_{ij} is the weight of the connection between the ith and the jth unit.

● Next, the unit calculates the activity y_j using some function of the total weighted input. Typically we use the sigmoid function:

$$y_j = \frac{1}{1+e^{-x_j}}$$

Once the activities of all output units have been determined, the network computes the error E, which is defined by the expression:

$$E = \frac{1}{2}\sum_i (y_i - d_i)^2$$

where y_j is the activity level of the jth unit in the top layer and d_j is the desired output of the jth unit.

The back-propagation algorithm consists of four steps:

1. Compute how fast the error changes as the activity of an output unit is changed. This error derivative (EA) is the difference between the actual and the desired activity.

$$EA_j = \frac{\partial E}{\partial y_j} = y_j - d_j$$

2. Compute how fast the error changes as the total input received by an output unit is changed. This quantity (EI) is the answer from step 1 multiplied by the rate at which the output of a unit changes as its total input is changed.

$$EI_j = \frac{\partial E}{\partial x_j} = \frac{\partial E}{\partial y_j} \times \frac{dy_j}{dx_j} = EA_j y_j \left(1 - y_j\right)$$

3. Compute how fast the error changes as a weight on the connection into an output unit is changed. This quantity (EW) is the answer from step 2 multiplied by the activity level of the unit from which the connection emanates.

$$EW_{ij} = \frac{\partial E}{\partial W_{ij}} = \frac{\partial E}{\partial x_j} \times \frac{\partial x_j}{\partial W_{ij}} = EI_j y_i$$

4. Compute how fast the error changes as the activity of a unit in the previous layer is changed. This crucial step allows back propagation to be applied to multilayer networks. When the activity of a unit in the previous layer changes, it affects the activites of all the output units to which it is connected. So to compute the overall effect on the error, we add together all these seperate effects on output units. But each effect is simple to calculate. It is the answer in step 2 multiplied by the weight on the connection to that output unit.

$$EA_i = \frac{\partial E}{\partial y_i} = \sum_j \frac{\partial E}{\partial x_j} \times \frac{\partial x_j}{\partial y_i} = \sum_j EI_j W_{ij}$$

By using steps 2 and 4, we can convert the EAs of one layer of units into EAs for the previous layer. This procedure can be repeated to get the EAs for as many previous layers as desired. Once we know the EA of a unit, we can use steps 2 and 3 to compute the EWs on its incoming connections.

🔴 **Back to Contents**

Appendix C - References used throughout the review

1. An introduction to neural computing. Aleksander, I. and Morton, H. 2nd edition
2. Neural Networks at Pacific Northwest National Laboratory
   http://www.emsl.pnl.gov:2080/docs/cie/neural/neural.homepage.html
3. Artificial Neural Networks in Medicine
   http://www.emsl.pnl.gov:2080/docs/cie/techbrief/NN.techbrief.ht
4. Industrial Applications of Neural Networks (research reports Esprit, I.F.Croall, J.P.Mason)
5. A Novel Approach to Modelling and Diagnosing the Cardiovascular System
   http://www.emsl.pnl.gov:2080/docs/cie/neural/papers2/keller.wcnn95.abs.html
6. Electronic Noses for Telemedicine
   http://www.emsl.pnl.gov:2080/docs/cie/neural/papers2/keller.ccc95.abs.html

7. An Introduction to Computing with Neural Nets (Richard P. Lipmann, IEEE ASSP Magazine, April 1987)
8. Pattern Recognition of Pathology Images
   http://kopernik-eth.npac.syr.edu:1200/Task4/pattern.html
9. Developments in autonomous vehicle navigation. Stefan Neuber, Jos Nijhuis, Lambert Spaanenburg. Institut fur Mikroelektronik Stuttgart, Allmandring 30A, 7000 Stuttgart-80
10. Klimasauskas, CC. (1989). The 1989 Neuro Computing Bibliography. Hammerstrom, D. (1986). A Connectionist/Neural Network Bibliography.
11. DARPA Neural Network Study (October, 1987-February, 1989). MIT Lincoln Lab.
12. Neural Networks, Eric Davalo and Patrick Naim.
13. Assimov, I (1984, 1950), Robot, Ballatine, New York.
14. Learning internal representations by error propagation by Rumelhart, Hinton and Williams (1986).
15. Alkon, D.L 1989, Memory Storage and Neural Systems, Scientific American, July, 42-50
16. Minsky and Papert (1969) Perceptrons, An introduction to computational geometry, MIT press, expanded edition.
17. Neural computers, NATO ASI series, Editors: Rolf Eckmiller Christoph v. d. Malsburg

Back to Contents