Instituto Tecnológico y de Estudios Superiores de Occidente
Departamento de Electrónica, Sistemas e Informática

Introductory Notes on Neural Networks

Dr. José Ernesto Rayas Sánchez

April 2002

Introductory Notes on Neural Networks

Dr. José Ernesto Rayas Sánchez

BIOLOGICAL NEURAL NETWORKS

The brain can be seen as a highly complex, nonlinear, and parallel computer. The neuron is the fundamental unit of the nervous system, particularly the brain. The *neuron* is a simple processing unit that receives and combines signals from many other neurons through filamentary input paths, the *dendrites*. A simplified representation of a neuron is shown in Fig. 1.

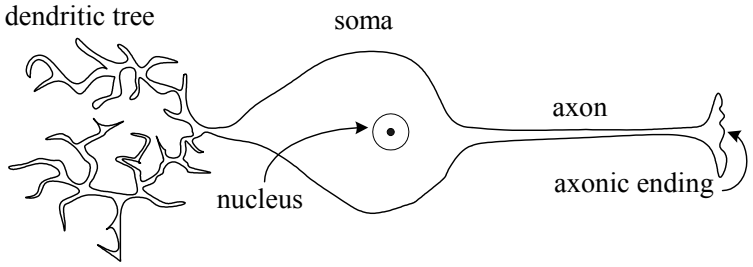


Fig. 1 Representation of a biological neuron.

Dendritic trees are connected with the main body of the nerve cell, the *soma*. The outer boundary of the cell is the *membrane*. When excited above a certain level, the *threshold*, the neuron fires; that is, it transmits an electrical signal, the *action potential*, along a single path called the *axon*. When the action potential reaches the axonic ending, chemical messengers, known as neurotransmitters, are released. The connection (or junction) between a neuron's axon and another neuron's dendrite is called a *synapse*.

A single neuron may have 1000 to 10,000 synapses and may be connected with some 1000 neurons. All signals coming from the dendritic tree are combined at the soma, and if the amplitude of the combined signals reaches the threshold of the neuron, a "firing" process is activated and an output signal is produced. The signal, either a single pulse or a sequence of pulses at a particular rate, is transmitted along the cell's axon to the axonic endings.

One neuron can make many contacts with other neurons, thus forming a complicated *neural network*. The axonic endings of a neuron may contact another neuron in different ways: a contact with a dendrite, a direct contact with the trunk of a dendritic tree, a contact with the soma of a neuron, o even a contact with the axon itself, as shown in Fig. 2. Some neurons feed back directly to themselves or indirectly via a third or fourth neuron.

When a signal appears at a synapse, a charge is generated. The magnitude of this charge depends on the strength of the incoming signal, which is weighted by a factor associated with this particular input. The *weighting factor* may not be constant over a long time.

In an embryonic neuron (i.e., a neuron under development) its axon is not fully developed and contacts with other neurons have not been fully formed. As the axon grows, however, the neuron recognizes the

correct pathway using a mechanism similar to an electrochemical seeking radar to contact a target neuron.

It is believed that a pattern is formed and stored when a group of events is associated in time. Patterns are stored and linked with already stored ones. This type of memory is called *associative*. Evidence suggests that each neuron can store thousands of patterns in the dendritric trees. During the learning phases (via input senses), patterns are formed, stored, and associated (and experience is gained). Memories are linked and associated. When a patterns is invoked, it triggers (recall) other patterns, which then triggers others, so that a cascade of memories (recalled patterns) occurs.

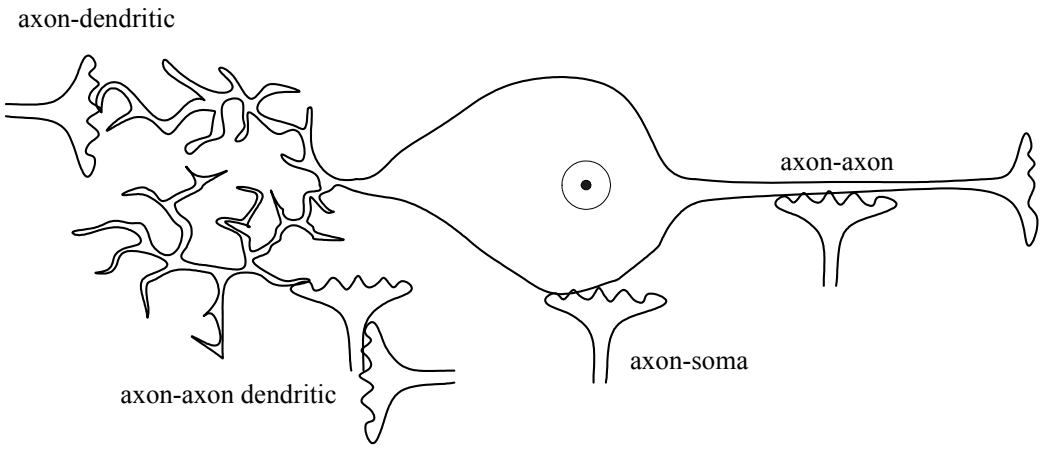


Fig. 2 Types of synapses.

All neurons are not the same, nor do they perform the same function. There are many categories of neurons with unique and highly *specialized functions*. Research has shown that some neurons have simple functions such as contracting and releasing a muscle, while other neurons control the excitatory or inhibitory behavior of other neurons; other types of neurons are specialized receptors (photonic, deflection, pressure, chemical, etc.), and so on.

Neural networks are quite diverse. Nevertheless, all neural network exhibit certain properties, namely:

a) Many parallel connections exist between many neurons.
b) Many of the parallel connections provide feedback mechanisms to other neurons and to themselves.
c) Some neurons may excite other neurons while inhibiting the operation of still others.
d) Some parts of the network may be pre-wired, whereas other parts may be evolving or under development.
e) Neural networks are asynchronous in operation.
f) Neural networks have a gross and slow synchronizing mechanism, as slow as the heartbeat, which supports their vital functions.
g) Neural networks execute a program that is fully distributed and not sequentially executed.
h) Neural networks do not have a central processor, but processing is distributed.

Biological neural networks are characterized by a hierarchical architecture. Lower neural networks preprocess raw information and pass their outcome to higher levels for higher-level processing.

Events in a silicon chip happen in the nanosecond range, whereas neural events happen in the

milliseconds range. Researchers estimate that there are 100 billion (10^{11}) neurons in the human cerebral cortex. Each neuron may have as many as 1,000 dendrites or more, so that the human cerebral cortex has at least 100,000 billion (10^{14}) synapses. Since each neuron can fire about 100 times per second, the brain can thus possibly execute 10,000 trillion (10^{16}) synaptic activities per second. This number is even more impressive when we realize that the human brain weights about 1.5 Kg. The energetic efficiency of the brain is about 10^{-16} joules per operation per second, whereas the corresponding value for the best computers in use today is about 10^{-6} J/s. The peak theoretical speed of the fastest computers now (around 35 gigaflops - 10^9 floating-point operations per second -) is not even close to our biological computer, the brain.

ARTIFICIAL NEURAL NETWORKS

An Artificial Neural Network (ANN) is a massively parallel distributed processor made up of simple processing units, that is able of acquiring knowledge from its environment though a learning process. ANNs are also information processing systems that emulate biological neural networks: they are inspired in the ability of human brain to learn from observation and generalize by abstraction.

An ANN may be thought of a sophisticated signal processor, in which the strength of each synapse (i.e., the synaptic weight), the bias and threshold values of each neuron at steady state constitute the network's program. Thus every neuron takes part in the massive parallel program execution. The *architecture* of an ANN defines the network structure, that is, the number of artificial neurons in the network and their interconnectivity. The *neurodynamics* of an ANN defines how it learns, recalls, associates and continuously compares new information with existing knowledge, as well as how it classifies new information and how it develops new classifications if necessary.

Basic models of a neuron

Several basic models to approximate the behavior of a biological neuron in a simplified manner are considered.

a) Linear Neuron

A general linear neuron can be modeled as shown in Fig. 3. Linear neurons are mostly used in the output layer of artificial neural networks.

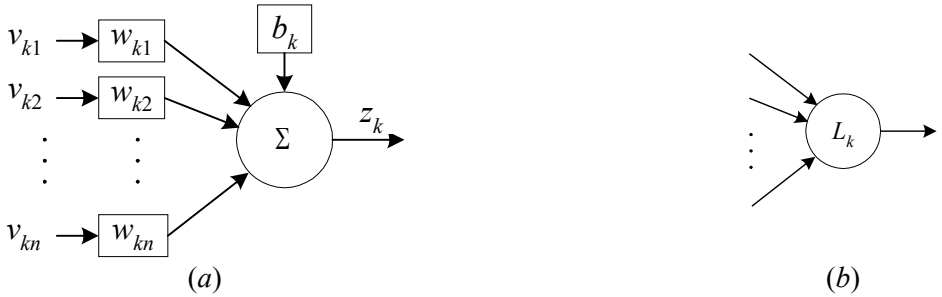


Fig. 3 Linear neuron: *a*) model, *b*) symbol.

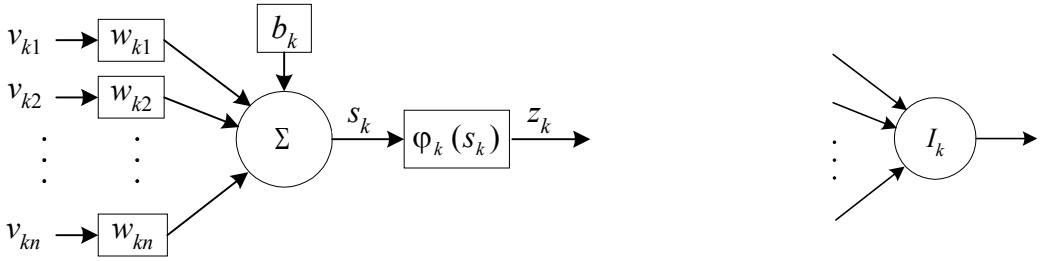
This *k*-th linear neuron has *n* inputs and one output, where $\boldsymbol{v}_k = [v_{k1} \ldots v_{kn}]^T$ is the vector of inputs, which

are signals coming from other neurons, $\boldsymbol{w}_k = [w_{k1} \ \ldots \ w_{kn}]^T$ is the vector of weights to represent the corresponding synapse strength and b_k is the bias or offset term. The output signal is the activation potential or induced local field given by

$$z_k = b_k + \boldsymbol{v}_k^T \boldsymbol{w}_k \tag{1}$$

b) Inner-Product Nonlinear Neuron

The most popular nonlinear neuron used mainly in the hidden layers of multiple-layer perceptrons is the inner-product nonlinear neuron, whose model is shown in Fig. 4.



(*a*) (*b*)

Fig. 4 Inner product nonlinear neuron: (*a*) model, (*b*) symbol.

This inner-product nonlinear *k*-th neuron has *n* inputs, and one output, where $\boldsymbol{v}_k = [v_{k1} \ \ldots \ v_{kn}]^T$ is the vector of inputs, which are signals coming from other neurons, $\boldsymbol{w}_k = [w_{k1} \ \ldots \ w_{kn}]^T$ is the vector of weights to represent the corresponding synapse strength, b_k is the bias or offset term, s_k is the activation potential or induced local field given by

$$s_k = b_k + \boldsymbol{v}_k^T \boldsymbol{w}_k \tag{2}$$

and z_k is the output signal in neuron *k* express as

$$z_k = \varphi_k(s_k) \tag{3}$$

The purpose of the nonlinear function $\varphi_k(s_k)$, called activation function or squashing function, is to ensure that the neuron's response is bounded, that is, the actual response of the neuron is conditioned or damped, as a result of large or small activating stimuli and thus is controllable. In the biological world, conditioning of stimuli is continuously done by all sensory inputs. Three popular nonlinearities for inner-product neurons are the sigmoid, the hard limiter and the hyperbolic tangent.

If a sigmoid or logistic function is used (see Fig. 5), then the response of neuron *k* can be given by

$$z_k = \varphi_k(s_k) = \frac{1}{1+e^{-s_k}} \tag{4}$$

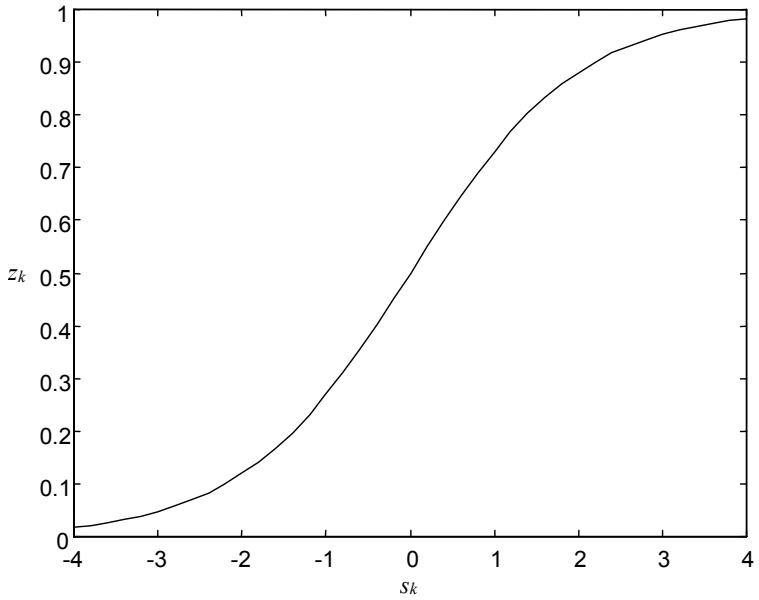


Fig. 5 Sigmoid activation function

The sigmoid function is the most popular nonlinear function for ANN because it is easy to implement and is continuously differentiable. It can be shown that a sigmoid function might be approximated by using

$$z_k = \varphi_k(s_k) = \frac{1}{1+e^{-s_k}} \approx \frac{1}{2}\left(\frac{1}{2}s_k + 1\right) \quad \text{for } |s_k| << 1 \tag{5}$$

The sigmoid function and its linear approximation are shown in Fig. 6*a*.

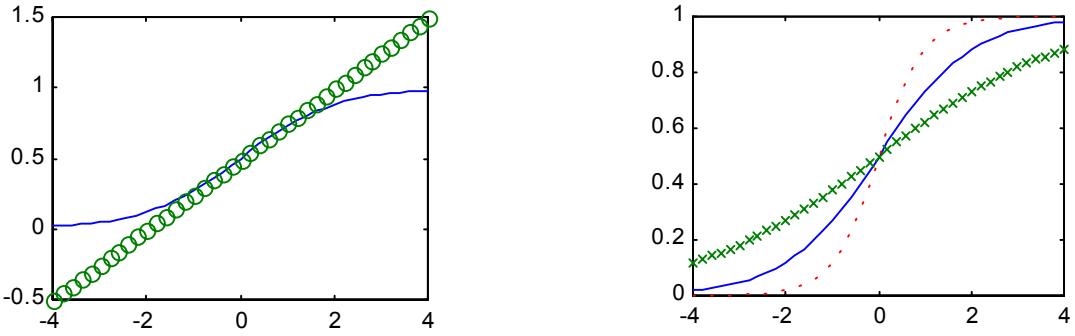


Fig. 6. Sigmoid activation function: (*a*) comparing it (—) with its linear approximation (o); (*b*) using different slope parameters: $a = 1$ (—), $a = 0.5$ (×) and $a = 2$ (--).

A slope parameter a for the sigmoid function can be introduced as follows

$$z_k = \varphi_k(s_k) = \frac{1}{1+e^{-a\,s_k}} \tag{6}$$

By varying a we can control the slope of the sigmoid, and therefore the amount of nonlinearity, as illustrated in Fig. 6*b*.

Taking the Taylor series expansion of an exponential function,

$$z_k = \varphi_k(s_k) = \frac{1}{1+e^{-s_k}} \approx \frac{1}{1+(1-s_k+\frac{s_k^{\,2}}{2!}-\frac{s_k^{\,3}}{3!}+\cdots)} \approx \frac{1}{2-s_k} \quad \text{for } |s_k| \text{ small} \tag{7}$$

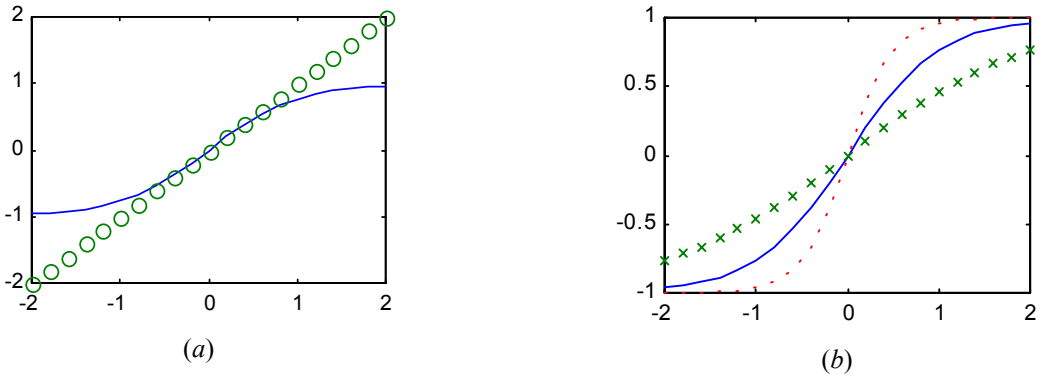
If a hyperbolic tangent is used, the response of the neuron is

$$z_k = \varphi_k(s_k) = \tanh(s_k) = \frac{e^{s_k}-e^{-s_k}}{e^{s_k}+e^{-s_k}} \tag{8}$$

The hyperbolic tangent activation function is illustrated in Fig. 7, considering its behavior for different slopes.

On the other hand, the hyperbolic tangent activation function exhibits a better behavior in the linear region, which is clear from its Taylor series expansion

$$z_k = \varphi_k(s_k) = \tanh(s_k) \approx s_k - \frac{s_k^{\,3}}{3} + \frac{2s_k^{\,5}}{15} - \frac{17s_k^{\,7}}{315} + \cdots \approx s_k \quad \text{for } |s_k| \text{ small} \tag{9}$$



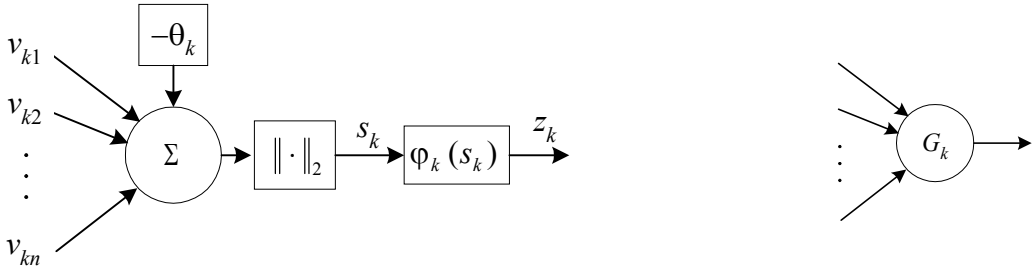
(*a*) (*b*)

Fig. 7. Hyperbolic tangent activation function: (*a*) comparing it (—) with its linear approximation (o); (*b*) using different slope parameters: $a = 1$ (—), $a = 0.5$ (×) and $a = 2$ (--).

Since each neuron in a biological neural network is different and is connected in a particular manner, the vector of inputs, the vector of weights, the bias and the nonlinear function can be different for each neuron in an ANN. In most practical cases, an ANN has the same nonlinear function at each layer of neurons.

c) Euclidean Distance Neuron

Euclidean distance neurons, shown in Fig. 8, are nonlinear neurons used to implement Radial Basis Functions ANNs.



(*a*) (*b*)

Fig. 8. Euclidean distance neuron: (*a*) model, (*b*) symbol.

This Euclidean distance *k*-th neuron has *n* inputs, and one output, where $\boldsymbol{v}_k = [v_{k1} \ \ldots \ v_{kn}]^T$ is the vector of inputs, which are signals coming from other neurons, $\theta_k = [\theta_{k1} \ \ldots \ \theta_{kn}]^T$ is the *k*-th center vector with respect to the Euclidean distance is measured, s_k is the activation potential or induced local field given by

$$s_k = \| \boldsymbol{v}_k - \boldsymbol{\theta}_k \|_2 \tag{10}$$

and z_k is the output signal in neuron *k* express as

$$z_k = \varphi_k(s_k) \tag{11}$$

The following nonlinear functions are of particular interest for RBF: multiquadratics, inverse multiquadratics and Gaussian functions. If a Gaussian function is used,

$$z_k = \varphi_k(s_k) = e^{-\frac{s_k^2}{2\sigma^2}} \quad \text{for some } \sigma > 0 \tag{12}$$

Gaussian activation functions for several values of σ are illustrated in Fig. 9.

General Characteristics of ANNs

Neural networks are characterized by a number of general features:

a) Collective and synergistic computation (or neurocomputing): program is executed collectively and synergistically; operations are decentralized.

b) Robustness: operation is insensitive to scattered failures and to partial inputs or inputs with inaccuracies.

c) Learning: the network makes associations automatically; it creates the program during leaning; the network adapts with or without a teacher (no programmer intervention).

d) Asynchronous operation: biological neural networks have no explicit clock to synchronize their

operation. Nevertheless, a number of ANNs require a clock.

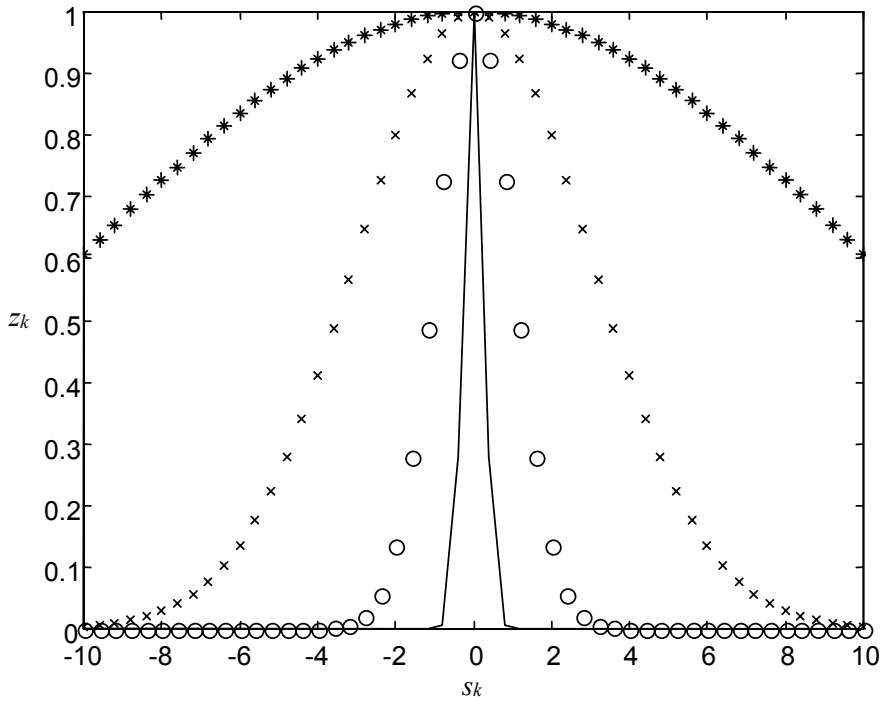


Fig. 9 Gaussian activation function using $\sigma = 0.25$ (—), $\sigma = 1$ (o), $\sigma = 3$ (×) and $\sigma = 10$ (∗).

Artificial Neural Network Topologies

An ANN configuration is also called a paradigm. It is characterized by a certain topology, a specific neuron model, and sometimes a particular learning strategy. Some of the most used topologies are illustrated in Fig. 10, where each circle represents a single neuron. The most widely used paradigm is the multilayer feedforward perceptron (MLP).

An MLP is formed by many basic neurons connected in a feedforward topology, with one or more hidden layers. The most common training algorithms for MLP are the Delta and the Back Propagation. The sigmoid is the most common nonlinearily for MLP, and usually is used only for the hidden neurons (the input and output layers are linear). The neurons in the input layer don't have any synaptic weight and bias element (they are just connecting nodes).

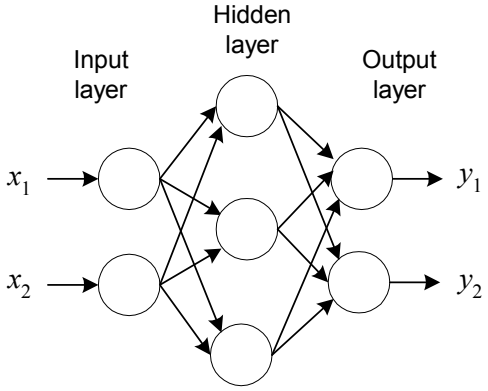
A two-layer perceptron (2LP) consists of only an input layer and an output layer. It can be shown that a 2LP actually implements a linear transformation on the input parameter space.

Another popular paradigm, not illustrated in Fig. 10, is the Recurrent Neural Network, which employes a feedback and delay units. It is widely used in time-domain applications.

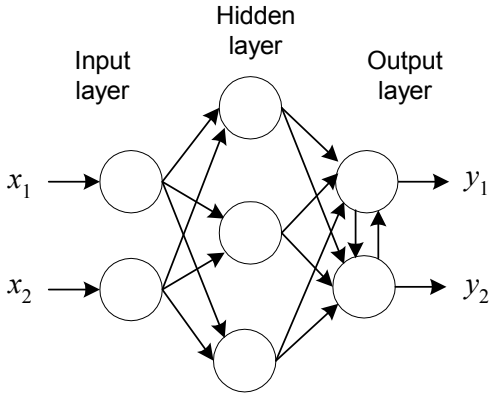
In most practical applications, input and output scaling is applied to the signals in the ANN. This is realized in order to control the relative importance of the different input parameters and to define a

suitable dynamical range for the region of interest. This is also very important to improve the numerical performance of the training algorithms.

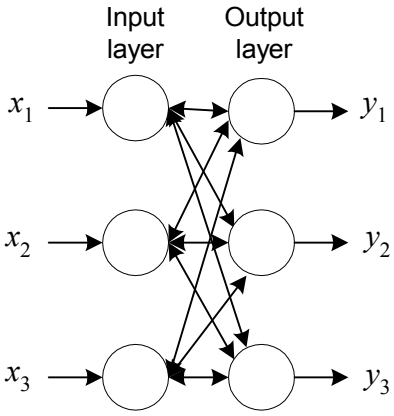
Multilayer perceptron (MLP):



Multilayer cooperative/competitive:



Bilayer: feedforward/backward:



Monolayer heterofeedback:

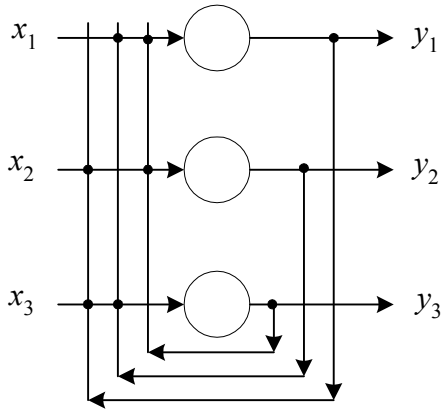


Fig. 10 Typical topologies for ANNs.

Learning in Artificial Neural Networks

Learning (or training) is the process by which an ANN adapts itself to a stimulus, and eventually (after making the proper parameter adjustment to its synaptic weights and bias) produces the desired response. When the actual output response is the same as the desired one, the network has completed the learning process; in other words, it has "acquired knowledge". Different learning techniques suit different

artificial neural networks:

a) Supervised Learning. During the training session of a neural network, an input stimulus is applied that results in an output response. This response is compared with an a priori desired output signal, the *target* response. If the actual response differs from the target response, an error signal is generated which is then used to calculate the adjustment that should be made to the network's synaptic weights and bias, so that the actual response matches the target output.

b) Unsupervised learning. During the training session, the NN receives many different excitations, or input patterns, and it organizes the patterns into categories. When a stimulus is later applied, the NN provides an output response indicating the class to which the stimulus belongs. If a class can not be found for the input stimulus, a new class is generated. Even though unsupervised learning does not require a target response, it requires guidelines or selecting criteria to determine how it will form groups. This kind of training of ANNs is extensively used in pattern recognition.

c) Competitive learning. In this scheme, several neurons are at the output layer. When an input stimulus is applied, each output neuron competes with the others to produce the closest signal to the target. This output then becomes the dominant one, and the other output cease producing an output signal for that stimulus. For another stimulus, another output neuron becomes the dominant one. Competitive learning can be viewed as random specialization process, which may not always be desirable.

d) Hebbian learning. In this technique the synaptic weight between two neurons is modified according to the degree of correlated activity between input and output. It is inspired on observations of metabolic changes in biological neural networks.

It is possible for an ANN to modify its own topology during learning, which is motivated by the fact that neurons in the human brain can die and new synaptic connections can grow.

The problem of training an ANN can be formulated as an optimization problem, where the ANN parameters (weights, bias, threshold, etc.) are to be found so that the ANN response matches the desired response. Some of the most used optimization techniques for ANN training are the Delta Rule, Boltzmann's algorithm, the back-propagation learning algorithm, simulation annealing, and the Markovian technique.

SUGGESTED REFERENCES

S. Haykin, *Neural Networks: A Comprehensive Foundation*. New Jersey, MA: Prentice Hall, 1999.

S.V. Kartalopoulos, Understanding Neural Networks and Fuzzy Logic. New York, NY: IEEE Press, 1996.