NEAR EAST UNIVERSITY

Faculty of Engineering

Department of Computer Engineering

NEURAL NETWORKS IN INDUSTRIAL APPLICATIONS

Graduation Project
COM- 400

Student: Ahmad Ziad Al-Kurdy

Supervisor: Assoc.Prof.Dr. Adnan Khashman

Nicosia - 2004

NEAR EAST UNIVERSITY

Faculty of Engineering

Department of Computer Engineering

NEURAL NETWORKS IN INDUSTRIAL APPLICATIONS

Graduation Project
COM- 400

Student: Ahmad Ziad Al-Kurdy

Supervisor: Assoc.Prof.Dr. Adnan Khashman

Nicosia - 2004

AKNOWLEDGEMENT

First, I would like to thank my supervisor Assoc. Prof. Dr Adnan Khashman for his invaluable advice and for the generosity he exhibited with his time and effort over this project and through the courses I have taken with him.

Second, I thank my family especially parents, brothers for their continuous encouragement and support during preparation of this project.

Third, I would like to express my gratitude to Near East University for the scholarship that made the work possible.

Finally, I would like to thank all my friends, especially Alaa Al-Shanableh, Mohammed al-Arrouri, Ahmed Wattad, And Zouran for their support and their constant moral sustain.

ABSTRACT

Artificial neural networks (usually just called neural networks) are interconnected collections of simple, independent processors. While loosely modeled after the brain, the details of neural network design are not guided by biology. Instead, for over 20 years researchers have been experimenting with different types of nodes, different patterns of interconnection, and different algorithms for adjusting connections.

The difference between the behavior of programmed computation and neural networks, compare the operation of computers with humans. For example, a computer can perform mathematical operations more quickly and precisely than a human can. However, a human can recognize faces and complex images in a more precise, efficient, and faster manner than can the best computer. One of the reasons for this performance difference can be attributed to the distinct organization forms of computers and biological neural systems. A computer generally consists of a processor working alone, executing instructions delivered by a programmer one by one. Biological neural systems consist of billions of nervous cells (i.e., neurons) with a high degree of interconnection. Neurons can perform simple calculations without the need to be previously programmed.

This project presents an investigation into neural networks and their application in industry.

TABLE OF CONTENTS

INTRODUCTION

Neural networks are named after the cells in the human brain that perform intelligent operations. The brain is made up of billions of neuron cells. Each of these cells is like a tiny computer with extremely limited capabilities; however, connected together, these cells form the most intelligent system known. Neural networks are formed from hundreds or thousands of simulated neurons connected together in much the same way as the brain's neurons.

Just like people, neural networks learn from experience, not from programming. Neural networks are good at pattern recognition, generalization, and trend prediction. They are fast, tolerant of imperfect data, and do not need formulas or rules. Neural networks are trained by repeatedly presenting examples to the network. Each example includes both inputs (information you would use to make a decision) and outputs (the resulting decision, prediction, or response).

Your network tries to learn each of your examples in turn, calculating its output based on the inputs you provided. If the network output doesn't match the target output, BrainMaker corrects the network by changing its internal connections. This trial-and-error process continues until the network reaches your specified level of accuracy. Once the network is trained and tested, you can give it new input information, and it will produce a prediction. Designing your neural network is largely a matter of identifying which data is input, and what you want to predict, assess, classify, or recognize.

Neural networks are called machine-learning algorithms because changing these connections (training) causes the network to learn the solution to a problem. This differs from other artificial intelligence technologies, such as expert systems, fuzzy logic or constraint-based reasoning which must be programmed to solve a problem.

The aim of this project is to show how neural networks are robust and useful in many industrial applications. Neural networks can be used in several ways to model a given process. The project consists of introduction, four chapters and conclusion.

Chapter One presents the Biological Foundation of Neurocomputing and the development of neurocomputing, how to build a neural network, and some benefits of a neural network.

Chapter Two describes the different neural network models have been explored. These models are described as either unsupervised or supervised. Unsupervised neural networks, such as self-organizing feature maps, find relationships between input examples by examining the similarities and differences between the examples. Supervised neural networks, such as back propagation, are used for pattern recognition or prediction. For supervised neural networks, the input examples must be an accompanied by the desired output.

Chapter Three presents some Industrial Application of neural network like neural computing in Oil and Gas Industry, Neural Network In Tetris Games, Neural network in the pulp and paper industry and the Modeling of plate Rolling Process using Neural network.

Chapter Four presents the application of neural network in the modeling of plate rolling process.The cases that neural network were applied to real life in steelwork that the still do not have hot rolling models developed. And in the Thermal profile of slaps a neural network model was developed to forecast the inner temperture of the slabs being reheated as a function of reheating time and superficial temperatures. One of the disadvantages of neural networks is the complexity of hardware and software necessary to enable industrial application. For example, if process conditions change from those to used when training the neural network, data must once again be collected, analyzed and used for retraning the system.

CHAPTER ONE

INTRODUCTION

TO NEURAL NETWORK

1.1 Overview

This chapter presents the Biological Foundation of Neurocomputing and the development of neurocomputing, how to build a neural network, and some benefits of a neural network.

1.2 What is a Neural Network?

In information technology, a neural network is a system of programs and data structure that approximate the operation of the human brain. A neural network usually involves a large number of processors operating in parallel, each with its own small sphere of knowledge and access to data in its local memory. Typically, a neural network is initially "trained" or fed large amounts of data relationships (for example, "a grandfather is older than a person's father"). A program can tell the network how to behave in response to an external stimulus (for example, to input from a computer user who is interacting with the network) or can initiate activity on its own (within the limits of its access to the external world).

In making determinations, neural networks use several principles, including gradient-based training, fuzzy logic, genetic algorithms, and Bayesian methods. Neural networks are sometimes described in items of knowledge layers, with, in general, more complex networks having deeper layers. In feed-forward systems, learned relationships about data can "feed forward" to higher layers of knowledge. Neural networks can also learn temporal concepts and have been widely used in signal processing and time series analysis.

Current applications of neural networks include: oil exploration data analysis, weather prediction, the interpretation of nucleotide sequences in biology labs, and the

exploration of models of thinking and consciousness. In his novel, Galatea [1], Richard Powers envisioned a neural network (named "Helen") that could be thought to pass a comprehensive exam in English literature.

1.3 The Sudden Rise of Neurocomputing

The majority of information processing today is carried out by digital computers. This has led to the widely held misperception that information processing is dependent on digital computers. However, if we look at cybernetics and the other disciplines that form the basis of information science, we see that information processing originates with living creatures in their struggle to survive in their environments, and that the information being processed by computers today accounts for only a small part - the automated portion - of this. Viewed in this light, we can begin to consider the possibility of information processing devices that differ from conventional computers. In fact, research aimed at realizing a variety of different types of information processing devices is already being carried out, albeit in the shadows of the major successes achieved in the realm of digital computers. One direction that this research is taking is toward the development of an information processing device that mimics the structures and operating principles found in the information processing systems possessed by humans and other living creatures.

Digital computers developed rapidly in and after the late 1940's, and after originally being applied to the field of mathematical computations, have found expanded applications in a variety of areas, to include text (word), symbol, image and voice processing, i.e. pattern information processing, robot control and artificial intelligence. However, the fundamental structure of digital computers is based on the principle of sequential (serial) processing, which has little if anything in common with the human nervous system.

The human nervous system, it is now known, consists of an extremely large number of nerve cells, or neurons, which operate in parallel to process various types of information. By taking a hint from the structure of the human nervous system, we should be able to build a new type of advanced parallel information processing device.

In addition to the increasingly large volumes of data that we must process as a result of recent developments in sensor technology and the progress of information technology, there is also a growing requirement to simultaneously gather and process huge amounts of data from multiple sensors and other sources. This situation is creating

a need in various fields to switch from conventional computers that process information sequentially, to parallel computers equipped with multiple processing elements aligned to operate in parallel to process information.

Besides the social requirements just cited, a number of other factors have been at work during the 1980 [3] pt research on new forms of information processing devices. For instance, recent neurophysiological experiments have shed considerable light on the structure of the brain, and even in fields such as cognitive science, which study human information processing processes at the macro level, we are beginning to see proposals for models that call for multiple processing elements aligned to operate in parallel. Research in the fields of mathematical science and physics is also concentrating more on the mathematical analysis of systems comprising multiple elements that interact in complex ways. These factors gave birth to a major research trend aimed at clarifying the structures and operating principles inherent in the information processing systems of human beings and other animals, and constructing an information processing device based on these structures and operating principles.

1.4 Building A Neural Network

Since 1958, when psychologist Frank Rosenblatt [2] proposed the "Perceptron," a pattern recognition device with learning capabilities, the hierarchical neural network has been the most widely studied form of network structure. A hierarchical neural network is one that links multiple neurons together hierarchically, as shown in Figure 1.3. The special characteristic of this type of network is its simple dynamics. That is, when a signal is input into the input layer, it is propagated to the next layer by the interconnections between the neurons. Simple processing is performed on this signal by the neurons of the receiving layer prior to its being propagated on to the next layer. This process is repeated until the signal reaches the output layer completing the processing process for that signal.

The manner in which the various neurons in the intermediary (hidden) layers process the input signal will determine the kind of output signal it becomes (how it is transformed). As wee can see, then, hierarchical network dynamics are determined by the weight and threshold parameters of each of their units. If input signals can be transformed to the proper output signals by adjusting these values (parameters), then hierarchical networks can be used effectively to perform information processing.

Since it is difficult to accurately determine multiple parameter values, a learning method is employed. This involves creating a network that randomly determines parameter values. This network is then used to carry out input-to-output transformations for actual problems. The correct final parameters are obtained by properly modifying the parameters in accordance with the errors that the network makes in the process. Quite a few such learning methods have been proposed. Probably the most representative of these is the error back-propagation learning method proposed by D. E. Rumelhart et al. in 1986 [3]. This learning method has played a major role in the recent neurocomputing boom.

The back-propagation paradigm has been tested in numerous applications including bond rating, mortgage application evaluation, protein structure determination, backgammon playing, and handwritten digit recognition. Choosing the right methodology, or backpropagation algorithm, is another important consideration. In working with the financial applications, many have found that the back-propagation algorithm can be very slow. Without using advanced learning techniques to speed the process up, it is hard to effectively apply backpropagation to real-world problems. Overfitting of a neural network model is another area which can cause beginners difficulty. Overfitting happens when an ANN model is trained on one set of data, and it learns that data too well. This may cause the model to have poor generalization abilities - the model may instead give quite poor results for other sets of data.

1.5 The Analogy To The Brain

The most basic components of neural networks are modeled after the structure of the brain. Some neural network structures are not closely to the brain and some does not have a biological counterpart in the brain. However, neural networks have a strong similarity to the biological brain and therefore a great deal of the terminology is borrowed from neuroscience.

1.5.1 The Biological Foundation of NeuroComputing

Neurocomputing involves processing information by means of changing the states of networks formed by interconnecting extremely large numbers of simple processing elements, which interact with one another by exchanging signals. Networks such as the one just described are called artificial neural networks (ANNs), in the sense that they represent simplified models of natural nerve or neural networks(figure 1.1.).

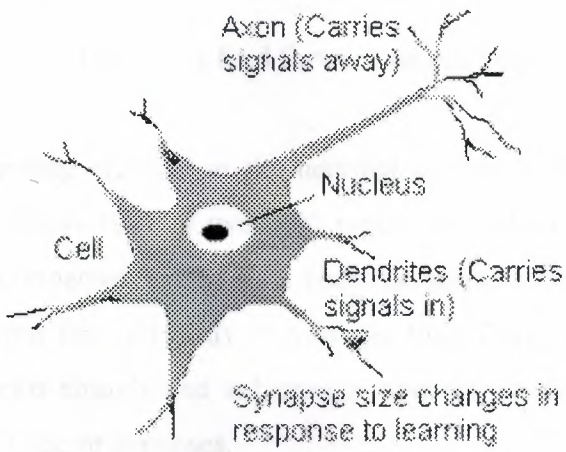

Fig. 1.1. A simple neuron cell

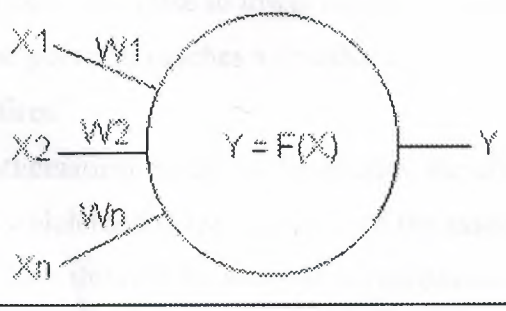

Fig. 1.2. A schematic diagram of a neuron

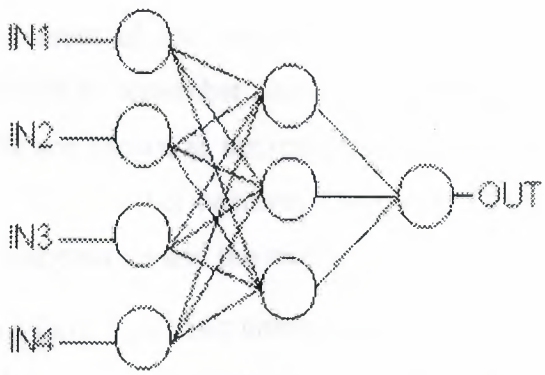

Fig.1.3. A feed forward neural network

The basic processing element in the nervous system is the neuron. The human brain is composed of about 1011 of over 100 types. Tree-like networks of nerve fiber called dendrites are connected to the cell body or soma, where the cell nucleus is located. Extending from the cell body is a single long fiber called the axon, which eventually branches into strands and substrands, and are connected to other neurons through synaptic junctions, or synapses.

The transmission of signals from one neuron to another at a synapses is a complex chemical process in which specific transmitter substances are released from the sending end of the junction. The effect is to raise to lower the electrical potential inside the body of the receiving cell. If the potential reaches a threshold, a pulse is sent down the axon - we then say the cell has "fired".

In a simplified mathematical model of the neuron, the effects of the synapses are represented by "weights" which modulates the effect of the associated input signals, and the nonlinear characteristics exhibited by neurons is represented by a transfer function which is usually the sigmoid function. The neuron impulse is then computed as the weighted sum of the input signals, transformed by the transfer function. The learning capability of an artificial neuron is achieved by adjusting the weights in accordance to the chosen learning algorithm, usually by a small amount *Wj = **Xj where * is called the learning rate and * the momentum rate.

1.6 The Biological Neuron

The most basic element of the human brain is a specific type of cell, which provides us with the abilities to remember, think, and apply previous experiences to our every action. These cells are known as neurons(see figure 1.4.), each of these neurons can connect with up to 200000 other neurons. The power of the brain comes from the numbers of these basic components and the multiple connections between them.

All natural neurons have four basic components, which are dendrites, soma, axon, and synapses. Basically, a biological neuron receives inputs from other sources, combines them in some way, performs a generally nonlinear operation on the result, and then output the final result. The figure below shows a simplified biological neuron and the relationship of its four components.

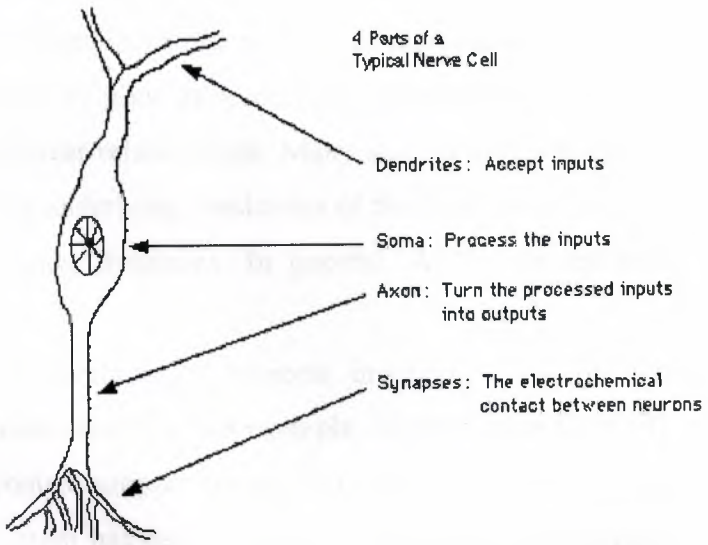

Figure 1.4. The biological neuron

1.7 Why we use neural network?

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 analyze. This expert can then be used to provide projections given new situations of interest and answer "what if" questions.Other advantages include:

Adaptive learning: An ability to learn how to do tasks based on the data given for training or initial experience.

Self-Organisation: An ANN can create its own organisation or representation of the information it receives during learning time.

Real Time Operation: ANN computations may be carried out in parallel, and special hardware devices are being designed and manifactured which take advantage of this capability.

Fault Tolerance via Redundant Information Coding: Partial destruction of a network leads to the corresponding degradation of performance. However, some network capabilites may be retained even with major network damage.

1.8 Some Real Life Application

ANNs can be regarded, in one respect, as multivariate nonlinear analytical tools, and are known to be very good at recognizing patterns from noisy, complex data, and estimating their nonlinear relationships. Many studies have shown that ANNs have the capability to learn the underlying mechanics of the time series, or, in the case of trading applications, the market dynamics. In general, ANNs are known to possess these capabilities:

A number of development projects involving ANN technology have been publicized in the media recently. For example, Nippon Steel Corp [4]. has built a blast furnace operation control support system that makes use of ANNs. The neural network employed in this system has been equipped with functions that enable it to learn the relationship between sensor data and the eight kinds of temperature distribution patterns known from experience to pertain to the overall operation of blast furnaces, and to instantaneously recognize and output that pattern which most closely approximates sensor data input into the system. The neural network learns very quickly, and achieves a better than 90% pattern recognition ratio following learning. Since this system has been performing extremely well during operational testing, Nippon Steel is planning to introduce it into other aspects of its operations in addition to blast furnace control, to include the diagnosis of malfunctions, and other control processes.

A second example is the experimental work started by Daiwa Securities Co. , Ltd. and NEC [5] Corporation on applying neural network technology to the learning and

recognition of stock price chart patterns for use in stock price forecasting. NEC had already developed neural network simulation software for use on its EWS 4800 series of workstations, and, by limiting stock price chart pattern learning to a few dozen major stocks, has improved the accuracy of this software's forecasting capabilities. Based on these results, the Daiwa Computer Services Co., Ltd., an information processing subsidiary of the Daiwa Securities Group, transferred the NEC system to its supercomputer and taught it to recognize the stock price chart patterns for 1,134 companies listed on the Tokyo Stock Exchange. DCS has since been putting this system to good use in the performance of stock price forecasting.

Mitsubishi Electric has combined neural network technology with optical technology to achieve the world's first basic optical neurocomputer system capable of recognizing the 26 letters of the alphabet. The system comprises a set of light-emitting diodes (LED) that output letter patterns as optical signals, optical fibers, liquid crystal displays (LCD) that display letter patterns and light receiving devices that read these letters. When letter data is input into this system, light emitted from the LEDs is input to the light receiving devices through the LCDs. At that time, the light receiving devices that receive the light, as well as the strength of the light they receive, is determined by the manner in which that light passes through the LCDs. The letter in question is delineated by the light receiving devices that receive the strongest light. This system is capable of 100% letter recognition even when slightly misshapen handwritten letters are input.

A fourth example is a development project for a facilities diagnosis system that employs a neural network system commenced by the Nippon Oil Co. [6], Ltd. in cooperation with CSK Research Institute. This project is attracting considerable attention as it is the first time research has been carried out on applying neural network systems to facilities diagnosis. Initially, the project will be aimed at developing a diagnosis system for pump facilities that employs vibration analysis. Nippon Oil operates a total of 1,500 pumps at its Negishi Oil Refinery in Yokohama, Kanagawa Prefecture alone, and must retain large numbers of experienced personnel to maintain these pumps. The company decided to apply neural network technology to pump facilities diagnosis operations with the ultimate goal of saving labor in mind.

1.9 The Future

Neural Networks will fascinate user-specific systems for education, information processing, and entertainment. "Alternative ralities", produced by comprehensive environments, are attractive in terms of their potential for systems control, education, and entertainment. This is not just a far-out research trend, but is something which is becoming an increasing part of our daily existence, as witnessed by the growing interest in comprehensive "entertainment centers" in each home. This "programming" would require feedback from the user in order to be effective but simple and "passive" sensors (e.g fingertip sensors, gloves, or wristbands to sense pulse, blood pressure, skin ionisation, and so on), could provide effective feedback into a neural control system. This could be achieved, for example, with sensors that would detect pulse, blood pressure, skin ionisation, and other variables which the system could learn to correlate with a person's response state.

Neural networks, integrated with other artificial intelligence technologies, methods for direct culture of nervous tissue, and other exotic technologies such as genetic engineering, will allow us to develop radical and exotic life-forms whether man, machine, or hybrid.

Neural networks will allow us to explore new realms of human capabillity realms previously available only with extensive training and personal discipline. So a specific state of consiously induced neurophysiologically observable awareness is necessary in order to facilitate a man machine system interface.

1.10 Some Advantages of Neural Network

Neural networks and artificial intelligence based models have potential advantages over other models due to following reasons

(1) Networks start processing the data without any preconceived hypothesis. They start with random weight assignment to various input variables. Adjustments are made based on the difference between predicted and actual output. This allows for unbiased and better understanding of data. It also some times help in unraveling subtle relationship between various input variables and out put being studied.

(2) Neural networks can be retrained using additional input variables and number of individuals. Once trained they can be called on to predict in a new patient.

(3) There are several neural network models available to chose from in a particular problem.

(4) Once trained, they are very fast.

(5) Due to increase accuracy, results in cost saving.

Disadvantages

(1) No set rule for network selection.

(2) Need experts in training the network and in-depth understanding of medical prpblem in hand.

(3) Training is quite time consuming and test the patience.

1.11 Summary

A neural network is a system of programs and data structure that approximate the operation of human brain . A neural network usually involves a large number of processors operating in parallel.

In this chapter we cover the development of neurocomputing, how to build a neural network and some benefits of a neural network.

In the next chapter we will cover the classifications of neural network, and the manner in which neural network are structed.

CHAPTER TWO
THE STRUCTURE OF NEURAL NETWORKS

2.1 Overview

This chapter presents two classifications of neural network according to first flow of information like recurrent network, feed forward network, and competitive network and second the way of learning like supervised learning and unsupervised learning.

2.2 Are There Any Limits To Neural Networks?

The major issues of concern today are the scalability problem, testing, verification, and integration of neural network systems into the modern environment. Neural network programs sometimes become unstable when applied to larger problems. The defense, nuclear and space industries are concerned about the issue of testing and verification. The mathematical theories used to guarantee the performance of an applied neural network are still under development. The solution for the time being may be to train and test these intelligent systems much as we do for humans. Also there are some more practical problems. The operational problem encountered when attempting to simulate the parallelism of neural networks. Since the majority of neural networks are simulated on sequential machines, giving rise to a very rapid increase in processing time requirements as size of the problem expands.

Solution: implement neural networks directly in hardware, but these need a lot of development still. Instability to explain any results that they obtain. Networks function as "black boxes" whose rules of operation are completely unknown.

2.3 The Artificial Neuron

The basic unit of neural networks, the artificial neurons, simulates the four basic functions of natural neurons. Artificial neurons are much simpler than the biological neuron; the figure below shows the basics of an artificial neuron.

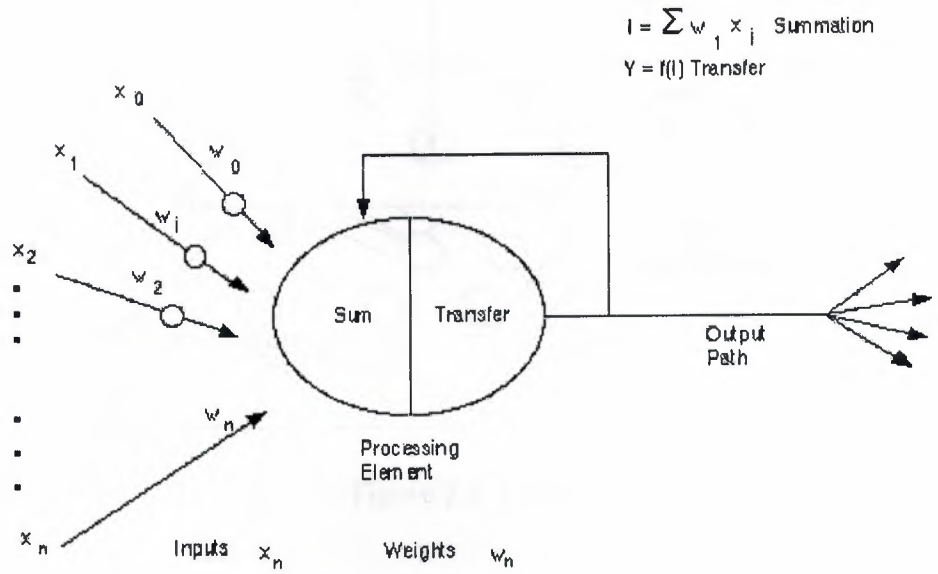

Figure 2.1. The Artificial Neuron

Note that various inputs to the network are represented by the mathematical symbol, x(n). Each of these inputs are multiplied by a connection weight, these weights are represented by w(n). In the simplest case, these products are simply summed, fed through a transfer function to generate a result, and then output.

Even though all artificial neural networks are constructed from this basic building block the fundamentals may vary in these building blocks and there are differences.

2.4 Layers

Biologically, neural networks are constructed in a three dimensional way from microscopic components. These neurons seem capable of nearly unrestricted interconnections. This is not true in any man-made network. Artificial neural networks are the simple clustering of the primitive artificial neurons. This clustering occurs by creating layers, which are then connected to one another. How these layers connect may

also vary. Basically, all artificial neural networks have a similar structure of topology. Some of the neurons interface the real world to receive its inputs and other neurons provide the real world with the network's outputs. All the rest of the neurons are hidden form view.

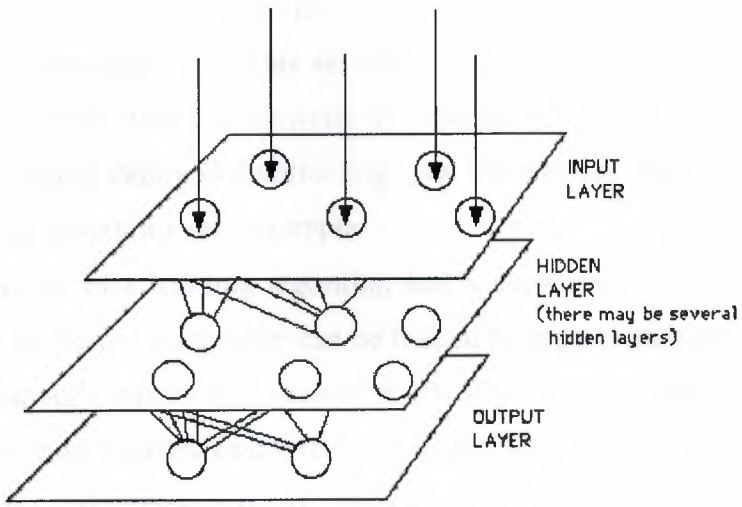

Figure 2.2. Layers

As the figure above shows, the neurons are grouped into layers The input layer consist of neurons that receive input form the external environment. The output layer consists of neurons that communicate the output of the system to the user or external environment. There are usually a number of hidden layers between these two layers; the figure above shows a simple structure with only one hidden layer.

When the input layer receives the input its neurons produce output, which becomes input to the other layers of the system. The process continues until a certain condition is satisfied or until the output layer is invoked and fires their output to the external environment.

To determine the number of hidden neurons the network should have to perform its best, one are often left out to the method trial and error. If you increase the hidden number of neurons too much you will get an over fit, that is the net will have problem to generalize. The training set of data will be memorized, making the network useless on new data sets.

2.5 Network Training

To train a neural network to approximate a desired function, a learning algorithm is used. In what is called unsupervised leaning, a learning algorithm automatically adjusts a neural network's weights in order improve its ability to give a desired output from a given input. Learning requires having a pre-defined set of inputs and desired outputs available to the algorithm. This set of training examples is called the training set. A learning algorithm trains a network by repeatedly looking at how a network responds to this training data and determining how the weights should be adjusted in order to improve the output for each example.

Through the use of a learning algorithm and a non-contradictory training set, a neural network of sufficient complexity can be trained to approximate any function. For example, a training set's input could consist of 100,000 hand-written letters, while the desired outputs for each training example could be the actual letter which was written. Each time the training algorithm is run on the training set, the neural network's weights are adjusted so that each training example gives an output which is closer to the desired output than it was before the training algorithm took place. Training algorithms are usually run over and over until the network produces outputs that are sufficiently close to the desired output for each training example.

2.6 Memorization and Generalization

To simulate intelligent behavior the abilities of memorization and generalization are essential. These are basic properties of artificial neural networks. The following definitions are according to the Collins English Dictionary: (table 2.1)

Table 2.1 defintion of memorizing and generalizing

To memorize:	To commit to memory; learn so as to remember.
To generalize:	To form general principles or conclusions from detailed facts, experience, etc.

Memorizing, given facts, is an obvious task in learning. This can be done by storing the input samples explicitly, or by identifying the concept behind the input data, and memorizing their general rules.

The ability to identify the rules, to *generalize*, allows the system to make predictions on unknown data.

Despite the strictly logical invalidity of this approach, the process of reasoning from specific samples to the general case can be observed in human learning.

Generalization also removes the need to store a large number of input samples. Features common to the whole class need not to be repeated for each sample - instead the system needs only to remember which features are part of the sample. This can dramatically reduce the amount of memory needed, and produce a very efficient method of memorization.

2.7 Classification of Neural Networks

Neural Network models can be classified in a number of ways. Using the network architecture as basis, there are three major types of neural networks:

2.7.1 Recurrent networks

The units are usually laid out in a two-dimensional array and are regularly connected. Typically, each unit sends its output to every other unit of the network and receives input from these same units. Recurrent networks are also called *feedback networks.* Such networks are "clamped" to some initial configuration by setting the activation values of each of the units. The network then goes through a stabilization process where the network units change their activation values and slowly evolve and converge toward a final configuration of "low energy". The final configuration of the network after stabilization constitutes the output or response of the network. This is the architecture of the *hopfield Model*

2.7.2 Feed forward networks

These networks distinguish between three types of units: input units, hidden units, and output units. The activity of this type of network propagates forward from one layer to the next, starting from the input layer up to the output layer. Sometimes called

multiplayer networks, feed forward networks are very popular because this is the inherent architecture of the Back propagation Model.

2.7.3 Competitive networks

These networks are characterized by lateral inhibitory connections between units within a layer such that the competition process between units causes the initially most active unit to be the only unit to remain active, while all the other units in the cluster will slowly be deactivated. This is referred to as a "winner-takes-all" mechanism. Self-Organizing Maps, Adaptive Resonance Theory, and Rumelhart & Zipser's [7] Competitive Learning Model are the best examples for these types of networks.

The network architecture can be further subdivided into whether the network structure is fixed or not. There are two broad categories:

Static architecture: Most of the seminal works on neural networks were based on static network structures, whose interconnectivity patterns are fixed *a priori*, although the connection weights themselves are still subject to training. Perceptrons, multi-layered perceptrons, self-organizing maps, and Hopfield networks all have static architecture.

Dynamic architecture: some neural networks do not constrain the network to a fixed structure but instead allow nodes and connections to be added and removed as needed during the learning process. Some examples are Grossberg's Adaptive Resonance Theory and Fritzke's "Neural Gas". Some adding-pruning approaches to Multi-Layered Perceptron networks have also been widely studied.

2.8 Learning

The brain basically learns from experience. Neural networks are sometimes called machine learning algorithms, because changing of its connection weights (training) causes the network to learn the solution to a problem. The strength of connection between the neurons is stored as a weight-value for the specific connection. The system learns new knowledge by adjusting these connection weights.

The learning ability of a neural network is determined by its architecture and by the algorithmic method chosen for training.

2.8.1 Off-line or On-line

One can categorize the learning methods into yet another group, off-line or on-line. When the system uses input data to change its weights to learn the domain knowledge, the system could be in training mode or learning mode. When the system is being used as a decision aid to make recommendations, it is in the operation mode, this is also sometimes called recall.

Off-line

In the off-line learning methods, once the systems enters into the operation mode, its weights are fixed and do not change any more. Most of the networks are of the off-line learning type.

On-line

In on-line or real time learning, when the system is in operating mode (recall), it continues to learn while being used as a decision tool. This type of learning has a more complex design structure.

2.8.2 Learning laws

There are a variety of learning laws which are in common use. These laws are mathematical algorithms used to update the connection weights. Most of these laws are some sort of variation of the best known and oldest learning law, Hebb's Rule[8]. Man's understanding of how neural processing actually works is very limited. Learning is certainly more complex than the simplification represented by the learning laws currently developed. Research into different learning functions continues as new ideas routinely show up in trade publications etc. A few of the major laws are given as an example below.

2.8.2.1 Hebb Law

The first and the best known learning rule was introduced by Donald Hebb. The description appeared in his book *The organization of Behavior* in 1949[9]. This basic rule is: If a neuron receives an input from another neuron, and if both are highly active (mathematically have the same sign), the weight between the neurons should be strengthened.

2.8.2.2 Hopefield Law

This law is similar to Hebb's Rule with the exception that it specifies the magnitude of the strengthening or weakening. It states, "if the desired output and the input are both active or both inactive, increment the connection weight by the learning rate, otherwise decrement the weight by the learning rate." (Most learning functions have some provision for a learning rate, or a learning constant. Usually this term is positive and between zero and one.)

2.8.2.3 The Delta Rule

The Delta Rule is a further variation of Hebb's Rule, and it is one of the most commonly used. This rule is based on the idea of continuously modifying the strengths of the input connections to reduce the difference (the delta) between the desired output value and the actual output of a neuron. This rule changes the connection weights in the way that minimizes the mean squared error of the network. The error is back propagated into previous layers one layer at a time. The process of back-propagating the network errors continues until the first layer is reached. The network type called Feed forward, Back-propagation derives its name from this method of computing the error term.This rule is also referred to as the Windrow-Hoff Learning Rule and the Least Mean Square Learning Rule.

2.8.2.4 Kohonen's Learning Law

This procedure, developed by Teuvo Kohonen, was inspired by learning in biological systems. In this procedure, the neurons compete for the opportunity to learn, or to update their weights. The processing neuron with the largest output is declared the winner and has the capability of inhibiting its competitors as well as exciting its neighbors. Only the winner is permitted output, and only the winner plus its neighbors are allowed to update their connectionweights. The Kohonen rule does not require desired output. Therefor it is implemented in the unsupervised methods of learning. Kohonen has used this rule to create the self-organizing neural network, which has an unsupervised learning method.

2.9 Supervised learning

These are generally the learn-by-example methods where user-supplied information is provided with each training pattern. These guide the neural network in adjusting its parameters. The perceptrons and back propagation networks are classic examples of supervised learning models.

2.10 Backpropagation

The term backpropagation has been used to refer to two distinct processes, one in the field of artificial neural networks (ANNs) and the other in neurobiology. Use of the same term is causing confusion: The existence of biological "backpropagation" hints that a process like ANN backpropagation may be present in biological neural networks. Here it is shown that the two are not related.

2.10.1 What is backpropagation?

There are at least three usages, the earliest from the field of artificial neural networks, and two that arose recently in neurobiology. A brief description of each, along with their distinctive features, will help exhibit the differences between the ANN and biological meanings.

2.10.2 Backpropagation in artificial neural networks

Is a mathematical technique for minimizing the discrepancy between a parametrized function and a set of pairs of inputs and "correct" outputs, where the overall function is partitioned into layers of vector functions. Each component of a layer is a weighted sum (or in some cases a product) of outputs from the previous layer, fed through some continuous scalar function. The "network" is regarded as the connections of outputs of one layer to inputs of the next. It is purely feed-forward -- there are no outputs from later layers that supply inputs of earlier layers. (Other forms of ANN contain explicit feedback connections of outputs to earlier inputs, but they aren't trained using backpropagation.)

The backpropagation algorithm adjusts the weights in the sums at each layer, in order to reduce the discrepancies between the network outputs before adjustment, and the desired values.

The technique originated with Werbos in 1974 [10], but did not come to the attention of the neural network community at that time. It was reinvented roughly simultaneously in 1985-6 by Le Cun , Parker , and Rumelhart[11], Hinton, and Williams [12], who named it backpropagation.

Some characteristics of ANN backpropagation, which will serve to distinguish it from biological "backpropagation" and to point out similarities, are:

ANN backpropagation provides direct feedback to the entire network. The process of updating the network weights takes information from the set of training values and from the current network output, and uses these to adjust all the network weights. Thus, in spite of the fact that the "network" is feed-forward, the training process adds unequivocal and global feedback. The feedback path looks like the figure below:

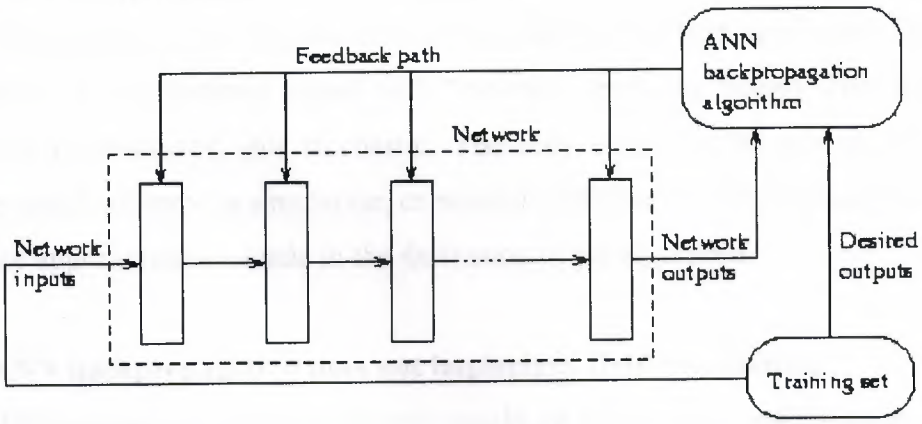

Figure 2.3. Backpropagation algorithm

There is no retrograde "flow" of material or information through the network layers. The updating of weights is effectively simultaneous across the network. The backpropagation algorithm uses all the network parameters, together with the actual and desired outputs, and computes new parameters, which it then installs in the network, while the network is not in use. The fact that the algorithm may, when run as a simulation, update the weights one at a time is an artifact of processing by a sequential computer. Even the flow of computation is primarily forward the algorithm iterates forward over the network layers twice, but only once backward. In fact, the final step, in which the weights are updated, is most efficiently performed while iterating forward over the network layers.

The above drawing might tempt one to say ANN feedback could be represented by a directed graph with cycles, rather than by an undirected (i.e. bidirectional) graph. But this is a bit misleading: In a directed graph, we can separately install edges in each direction, and control what is connected to what. But the ANN backpropagation algorithm uses all the weights, and updates all of them. ANN feedback can't be selectively blocked from parts of the network.

There is, however, a temporal restriction on the feedback path. ANN feedback is sequential, not combinational or recurrent (in the sense of being always operative). Production of the "next" set of weights does not interfere with production of the current network outputs.

2.10.3 ANN backpropagation is not a physical process

Although it may be implemented in hardware rather than simulated. There is no degradation of the feedback signal with "distance" from the output. The updating of weights is deterministic, not stochastic. The only errors in the process (other than possible round-off error in simulation, or noise if implemented electronically) arise from deliberate approximations made in the derivation of the algorithm.

2.10.4 ANN backpropagation does not implement Hebbian learning

Hebbian learning alters a network weight up or down depending on how closely the input and output connected by that weight do or do not match. Rather, the outcome of ANN backpropagation is function approximation. The network fits a continuous function to the data points supplied in the training set, that is, it matches the network's output to an externally-provided desired output.

There are a number of variants of ANN backpropagation, but they all intend to match network outputs to desired outputs. So one might say there is some "ideal" version of backpropagation, and others are approximations. In that sense, the operation of ANN backpropagation is fixed by its purpose we cannot alter it freely.

Because it attempts to match externally-supplied desired values, A backpropagation implements a type of supervised learning.

Note that traditional Hebbian learning is unsupervised. It uses only the inputs and output of the network. In so called "supervised Hebbian learning", a weight is adjusted depending on closeness of the input and desired output. This latter can implement an

autoassociative memory. But even it doesn't do function approximation, because it's not matching the network output to the desired output, nor is it truly Hebbian learning.

2.10.5 When to use (or not!) a BP Neural Network Solution

A back-propagation neural network is only practical in certain situations. Following are some guidelines on when you should use another approach:

- Can you write down a flow chart or a formula that accurately describes the problem? If so, then stick with a traditional programming method.
- Is there a simple piece of hardware or software that already does what you want? If so, then the development time for a NN might not be worth it.
- Do you want the functionality to "evolve" in a direction that is not pre-defined? If so, then consider using a Genetic Algorithm (that's another topic!).
- Do you have an easy way to generate a significant number of input/output examples of the desired behavior? If not, then you won't be able to train your NN to do anything.
- Is the problem is very "discrete"? Can the correct answer can be found in a look-up table of reasonable size? A look-up table is much simpler and more accurate.
- Are precise numeric output values required? NN's are not good at giving precise numeric answers.
- Conversely, here are some situations where a BP NN might be a good idea:
- A large amount of input/output data is available, but you're not sure how to relate it to the output.
- The problem appears to have overwhelming complexity, but there is clearly a solution.
- It is easy to create a number of examples of the correct behavior.
- The solution to the problem may change over time, within the bounds of the given input and output parameters (i.e., today 2+2=4, but in the future we may find that 2+2=3.8).
- Outputs can be "fuzzy", or non-numeric.

One of the most common applications of NNs is in image processing. Some examples would be: identifying hand-written characters; matching a photograph of a person's face with a different photo in a database; performing data compression on an image with minimal loss of content. Other applications could be: voice recognition; radar signature analysis; stock market prediction. All of these problems involve large amounts of data, and complex relationships between the different parameters.

It is important to remember that with a NN solution, you do not have to understand the solution at all! This is a major advantage of NN approaches. With more traditional techniques, you must understand the inputs, and the algorithms, and the outputs in great detail, to have any hope of implementing something that works. With a NN, you simply show it: "this is the correct output, given this input". With an adequate amount of training, the network will mimic the function that you are demonstrating. Further, with a NN, it is OK to apply some inputs that turn out to be irrelevant to the solution - during the training process, the network will learn to ignore any inputs that don't contribute to the output. Conversely, if you leave out some critical inputs, then you will find out because the network will fail to converge on a solution.

If your goal is stock market prediction, you don't need to know anything about economics, you only need to acquire the input and output data.

2.11 Unsupervised learning

Some neural network models do not need category information to accompany each training pattern, although such information would still be required in the interpretation and labeling of the resultant networks. Classical examples of these are Kohonen's self-organizing maps and Grossberg's Adaptive Resonance Theory.

It also makes sense to classify neural network models on the basis of their over-all task:

Pattern association: the neural network serves as an associative memory by retrieving an associated output pattern given some input pattern. The association can be *auto-associative* or *hetero-associative*, depending on whether or not the input and output patterns belong to the same set of patterns.

Classification: the network seeks to divide the set of training patterns into a pre-specified number of categories. Binary-valued output values are generally used for

classification, although continuous-valued outputs (coupled with a labeling procedure) can do classification just as well. For binary output representation, each category is generally represented by a vector (sequence) of 0's; with a single 1 whose position in the vector denotes the category.

Function approximation: the network is supposed to compute some mathematical function. The network's output represents the approximated value of the function given the input pattern as parameters. In certain areas, *regression* may be the more natural term.

There are other bases for classifying neural network models, but these are less fundamental than those mentioned earlier. Some of these include the type of input patterns that can be admitted (binary, discrete valued, real values), or the type of output values that are produced (binary, discrete-valued, real values).

2.12 Kohonen's Self Organizing Map (SOF)

Kohonon's SOMs are a type of unsupervised learning. The goal is to discover some underlying structure of the data. However, the kind of structure we are looking for is very different than, say, PCA or vector quantization.

Kohonen's SOM is called a topology-preserving map because there is a topological structure imposed on the nodes in the network. A topological map is simply a mapping that preserves neighborhood relations.

In the nets we have studied so far, we have ignored the geometrical arrangements of output nodes. Each node in a given layer has been identical in that each is connected with all of the nodes in the upper and/or lower layer. We are now going to take into consideration that physical arrangement of these nodes. Nodes that are "close" together are going to interact differently than nodes that are "far" apart.

What do we mean by "close" and "far"? We can think of organizing the output nodes in a line or in a planar configuration as shown in the figure below.

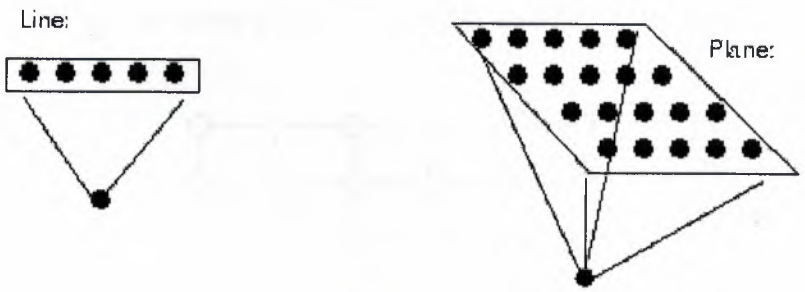

Figure 2.4. Kohonen's network

The goal is to train the net so that nearby outputs correspond to nearby inputs.E.g. if x1 and x2 are two input vectors and t1 and t2 are the locations of the corresponding winning output nodes, then t1 and t2 should be close if x1 and x2 are similar. A network that performs this kind of mapping is called a feature map.

In the brain, neurons tend to cluster in groups. The connections within the group are much greater than the connections with the neurons outside of the group. Kohonen's network tries to mimick this in a simple way.

2.12.1 Architecture of SOM

The architecture of the SOM can be a regular rectangular 2D structure, a hexagonal or honeycomb structure, a 3D rectangular structure, etc. 2-D structures are the most commonly used because maps are then more readily visualizable. Some other promising structures, like spherical maps where nodes are laid out regularly on the surface of the sphere, are also highly visualizable but are so far ignored in the literature.

The architecture affects the ordering of the data since neighborhood functions are used for the Kohonen training algorithm. The amount of change in the connection weights depends on its spatial distance in the map from the so called *winning unit*.

The original SOM, described by Teuvo Kohonen in 1980[13], has fixed map structures. The dimensions of the map structure has to be wisely chosen *a priori*. Some more recent variants of the SOM allow for dynamic allocation (both growth and pruning) of network units and the structure adapts to the inherent structure of the input environment.

Below are some examples of SOM fixed architectures.

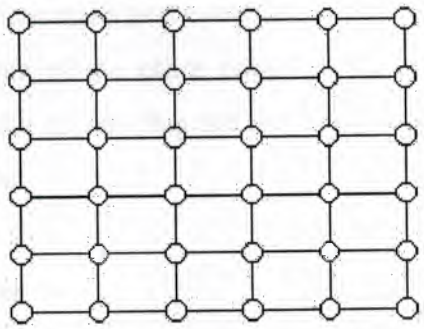

Figure 2.5. 2D rectangular architecture.

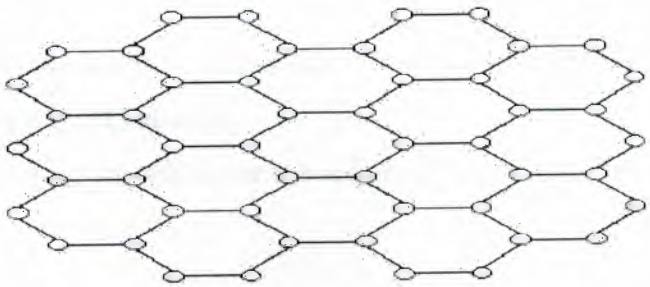

Figure 2.6. 2D honeycomb architecture

2.13 Characteristics of ANN

As discussed above, artificial neural networks are of different architectures, different learning schemes, varying weight update modalities, and may be called to perform different tasks. Yet, some basic characteristics can be said of most neural network models:

2.13.1 Inherent parallelism

Practically all-neural network models have some element of parallelism in the execution of its numerous components, although some degree of sequentiality of operation is observed in at least a component of these systems;

Regularity of components to a very a large extent, the basic components of a neural network, referred to in the literature as units, nodes, or neurons, all look alike and behave similarly.

2.13.2 Access to local information

Neural networks are composed of nodes that are interconnected with each other. Any given node's level of activation and eventual output will depend exclusively on its current state and the outputs of the other nodes to which it is connected. Whatever happens to other nodes in the system that are not connected to a given neuron will not directly affect its actions; and

2.13.3 Incremental learning

Neural networks do not learn any given concept in one go. Instead, the network parameters undergo several small changes, which, over time, would come to settle on their final values.

2.14 Summary

In this chapter we cover two classification of neural network according to Flow of information and the way of learning.

In the next chapter we will cover the industrial application of neural networks.

CHAPTER THREE
INDUSTRIAL APPLICATIONS
OF NEURAL NETORKS

3.1 Overview

This chapter presents some Industrial Application of neural network like neural computing in Oil and Gas Industry, Neural Network In Tetris Games, Neural network in the pulp and paper industry and the Modeling of plate Rolling Process using Neural network.

3.2 Neural Network in Papermaking Plant

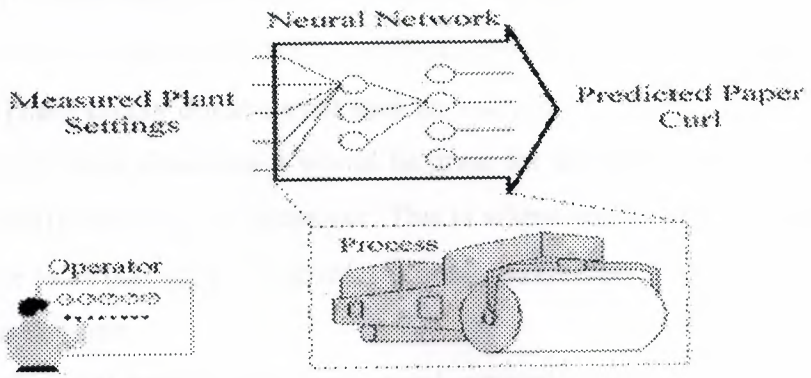

Figure 3.1. Neural Network in Papermaking Industry

The Neural Network Group (part of the Integrated Systems Group) is working together with two large paper-making firms to produce ANN (figure3.1.) based software to optimise the quality of paper from a paper-making plant.

In conventional operation a human operator, drawing on years of experience, continually modifies the settings on a huge machine which is rolling paper and coating it with a material that gives it a glossy surface. Among the variables to be controlled are the pressure on a lot of rollers, the temperature and the speed of the paper through the process. All are inter-related, usually in a highly non-linear fashion, and finding and maintaining the correct settings is a very inexact process more akin to alchemy than science.

Success is measured by, amongst other things, looking at the "curl" of the parer. "Curl" is to do with how a sheet of paper, initially flat, curls if it is held over a sharp edge, like the edge of a desk.

An ANN is currently being trained to mimic the work done by the human operator with the aim of first assisting him and finally relieving him from the tedium of constantly having to monitor the process.

3.3 Neural networks in the pulp and paper industry?

There are many processes in a pulp and paper mill where an on-line parameter analyzer cannot be used due to several reasons:

- The analyzer is very expensive to buy
- It cannot survive in the environment we want to use it
- It is not operational due to hardware problems, maintenance etc.
- There simply doesn't exist such an analyzer.

In all these situations it would be great for the mill to have an alternative way to continuously measure the parameter. This is where neural networks come to place. They can serve as virtual sensors that infer process parameters from other variables, which are measured on-line.

Inferential sensors based on neural network methodologies can be used for real time prediction of:

- Paper properties like tensile, stretch, brightness, opacity, softness etc.
- Digester kappa numbers
- Sodium chlorate concentrations and suspended solid levels in ClO_2 generators.
- Boiler stack emissions

3.4 What are the benefits an inferential sensor can offer?

3.4.1 Parameter monitoring in real time

The usual procedure followed in any of the above processes when an on-line sensor is not available, is to collect samples very infrequently, send them to the laboratory and wait for the results. This procedure may take several hours and if the lab

analysis shows that the product is out of specification, a substantial production time has been lost.

Suppose an inferential sensor is available to accurately predict process parameters in real time. This sensor can continuously monitor the parameters and is also able to provide us with a prediction of what is going to happen in the future. In case some parameters tend to be driven out of specifications, we have the time to take appropriate actions. Operational time will be saved and bad quality production will be minimized.

3.4.2 Process optimization

The inferential sensor based on neural networks is not only a tool that is used for parameter prediction. It also gives us the following important information about the process:

- The process input variables, which have the greatest impact on the product quality.
- The variables, which are not significant because they have little effect on the output of the process.
- The dead time for each process input variable, i.e. the time required for a change in the input variable to start affecting the output variable.

The model can also be used in off-line mode to run different scenarios. For example, we can simulate the response of the product quality parameters following a change in any input variable, like flow rate, temperature or pressure. In this manner we can improve the operation and optimize the quality properties of the product, by selecting the best possible set points of the input variables.

3.4.3 Process control

Model Predictive Control (MPC) is a relatively new control methodology, which became popular because it is robust, multivariable and takes care of input and output constraints. However, it requires an on-line sensor of the controlled variables. This is an excellent application of inferential analyzers based on neural network. They can serve as real time sensors in an MPC scheme, resulting in:

- Continuous control of the process.
- Process stabilization.

- Variability reduction.
- Disturbance rejection.

3.4.4 Neural Networks can save you money

Think of any process in your plant where you wish you could have some parameter prediction. A neural network can develop sample model for your process. Find out how powerful this technology is for process optimization and control is for process optimization and control

3.5 Neural Computing in the Oil and Gas Industry

Neural computing is now being applied successfully in the oil and gas industry. Neural computing applications include well log analysis, quality control, demand forecasting and machine health monitoring. So what is neural computing and what advantages can it offer to your industry?

Neural computers can succeed in many areas where conventional computers are unable to operate or can operate with only limited success. Conventional computers require someone to work out a step-by-step solution to the problem. Neural computers, however, are analogous to the human brain and learn from previous examples.

A neural computer can be trained to solve a particular problem by presenting it with a series of examples of problems and the desired solution in each case. Given enough training material, the neural computer is able to learn the underlying principles involved in the solution which it can then use to tackle similar problems. It has the ability to cope well with incomplete data and can deal with previously unspecified or unencountered situations. This contrasts with conventional systems, which, without a full set of data, are often unable to complete their tasks. Application areas for Neural Computing in the Oil and Gas industry.

3.6 Oil Exploration

A major oil exploration company to analyze seismic data and identify first break signals uses neural computing. The identification of first break signals is a laborious, time consuming manual task.

The vast quantities of seismic data involved are cluttered with noise and are highly dependent on the location being investigated. Classical statistical analysis techniques lose their effectiveness when the data is noisy and comes from an environment not previously encountered. Even a small improvement in correctly identifying first break signals could result in a considerable return on investment.

The neural network achieves better than 95% accuracy, easily outperforming existing manual and computer-based methods. As well as being more accurate, the system also achieves an 88% improvement in the time taken to identify first break signals. Considerable cost savings have been made as a result.

3.7 Well Log Analysis

Neural computing can be used to improve the accuracy of strata interpolation of core sample analysis.

Information on the underlying structure of the ground is typically obtained by drilling a series of bore holes, examining each core sample, comparing each core to its neighbours and inferring what structure might lie between the adjacent bores. Recent developments in neural computing, including dynamic warping analysis and section

3.8 Neural Networks in Logging Interpretation

Artificial neural network (ANN) theory has been widely applied in several aspects of logging interpretation in the petroleum industry, such as lithofacies identification, logging parameter prediction as well as other model recognition problems. Among several neural network models and algorithms applied in petroleum industry, the feed-forward neural network model and back-propagation algorithm (BPA) are most widely used for their easy implementation and high robustness. Although the BPA shows strong capability on solving model recognition, it has several problems, such as local optimum and low rate of convergence, when dealing with parameter prediction.

After comparing several kinds of neural network models and algorithms, the self-organization neural network model and Kohonen optimal algorithm are chosen to solve the lightfaces identification problem in the CaiNan oilfield in China [14]. The feed-forward neural network model and Levenberg-Marquardt [15] optimal algorithm are selected to predict logging parameters, such as porosity, permeability and water

saturation, in the same oilfield. Based on Mat lab tools, an artificial neural network logging interpretation program is compiled and applied to the logging evaluation of this oilfield. Simulation results show that the relative prediction error of porosity, permeability and water saturation can respectively reach 6%, 30% and 30%, which is higher than normal methods for parameter prediction. The results prove that the self-organization neural network model and Kohonen optimal algorithm are suitable for lithofacies identification and that the feed-forward neural network model and the optimal algorithm are suitable for logging parameter prediction in the oilfield

3.9 Ford Neural Chip

A new computer chip that mimics how the human mind works is making its way from the space program to American industry and may end up in millions of American cars in years to come.

Computer scientists at NASA's Jet Propulsion Laboratory [16] have made advanced neural network technology breakthroughs that can solve diagnostic problems in industries from automobiles and aerospace to manufacturing and electricity production.

JPL and the Ford Motor Company have signed a licensing agreement for use of an advanced neural network technology to diagnose misfiring under the hoods of Ford automobiles, among its many potential applications. With the advent of this new chip, vehicles should show a reduction in emission levels.

The smart fit between JPL's neural net hardware and Ford's automotive engineering algorithm expertise will enhance the industrial giant's ability to meet ever-stricter Clean Air Act requirements as they apply to continuous onboard diagnostics and control, officials said.

In addition, the chip is designed to improve fuel economy, resulting in financial savings for car owners. Ford engineers do not predict a price increase for installation of the chip because JPL designed a computationally powerful neuroprocessor that could be mass-produced in a highly cost-effective way. The technology also improves customer satisfaction by virtually eliminating distracting false alarms about misfiring that vehicle dashboards can signal with current under-the-hood diagnostic technology.

JPL and Ford scientists say the chip represents the first significant change in the way computing is done on vehicles since computers were first introduced into automobiles in the 1970s[17].

"Neural networks are a new discipline, and diagnostics, prognostics and control is a huge field. Ford's application is but the tip of the iceberg of this chip's potential use in American industry as a whole," said Tom Hamilton, program manager at JPL's Dual-Use Technology Office, one of JPL's many technology transfer arms. "JPL is proud to be able to make this revolutionary technology available for U.S. business."

The new license provides Ford with rights to intellectual property of the chip for auto industry applications, while JPL, which has applied for patents to the technology, retains general rights. JPL is managed by the California Institute of Technology, which serves as the party of record for this license.

Neural systems were inspired by the architecture of nervous systems of animals, which use neurons, a form of parallel processing elements, to process large volumes of information simultaneously. In vehicle applications, artificial neural networks will "learn" both how to diagnose problems like engine misfires and control the engine to optimize fuel economy and emissions.

What JPL has brought to the table is expertise in designing and building what are known as neural network application specific integrated circuits, said Dr. Raoul Tawel [18], who led the development at JPL for the chip. "With Ford, we are implementing highly complex neural network software code in dedicated hardware logic. This brings about a tremendous boost in computational ability compared to traditional software-based approaches, enabling real-time onboard diagnostics for the first time."

For misfire diagnostics, it is necessary to observe and diagnose every engine firing event, estimated at over one billion in the life of each car.

In addition, the diagnostic error rate has to be extremely small, less than one in a million, in order to avoid sending false alarm signals to the driver. The new chip will accomplish that task by "learning" diagnostic tasks during the vehicle development process, bypassing the need to develop conventional software that, in any event, can neither perform these tasks as well nor be implemented in large production volumes with standard microprocessors. The neural network chip, designed to carry out parallel neuron computations efficiently, overcomes the computational barriers that prevent this technology from being exploited today.

3.10 A Neural Network approach to the Tetris game

3.10.1 A brief Description of the Tetris Game

Tetris is a videogame of logic and skill. The game aim is to place in the right way the geometrical pieces falling from the top of the game field, composed of all the possible combinations of 4 basic blocks (see figure 3.2.).

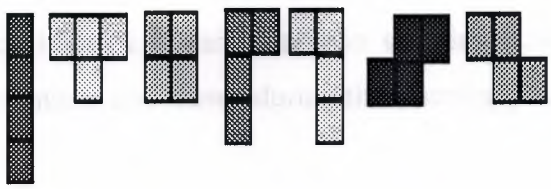

Figure 3.2. A Tetris game basic block

A whole line is erased every time the disposition of the pieces completes a horizontal line. This is the only way to erase pieces from the game pad, so it is worth to complete a line every time is possible. The game is over when the pieces fill completely all the game area.

The pieces can only be manipulated while falling, shifting and turning them, but when apiece has landed on other piece or on the bottom of the screen, it cannot be moved further.

3.10.2 Problem definition

The actions the Tetris player performs can be summarized in these 4 points

i. Identify which of the 7 pieces is falling down;
ii. Analyze the disposition of the previously fallen pieces;
iii. Choose the best (in some cases the least bad) place where to place the falling piece
iv. Shift and rotate the piece in order to perform the desired piece positioning

In some cases the best solution cannot exist (it is not physically possible to perform a perfect joint) and in other cases there can be multiple optimal solutions.

Therefore Tetris is a game where the player not only has to look for the perfect geometrical joint, but he/she has also to limit the damages.

In short, Tetris is a typical problem of real time 'decision making', a problem where you must react fast to a situation that can be very complex, or trivial, but in general consists of a very wide variety of possible scenarios.

A further problem in the Tetris game is that the decision previously taken affect the complexity of future decisions, thus it is very important to take always the best decision, or at least the 'minimum damage' one.

In our opinion the Tetris game can be very effective in showing Neural Network capabilities in taking decisions and particularly in showing how they can handle with good reliability even situations not shown during the learning phase.

3.10.3 Neural Networks and the Tetris game

We used Multi Layer Perceptron as the neural network model for the development of 'NeuroTetris' application.

The neural networks are only a component of a more general application that, using API (Application Program Interface) primitives for Windows, identifies the falling piece and the disposition of the game pad.

It is important to specify that we did not manipulate in any way the original Microsoft Tetris. A single neural network has been trained for each Tetris piece, except for the pieces that are mirror result of another one (yellow-violet green-blue couples, see figure 3.3.); therefore 5 networks are used in NeuroTetris.

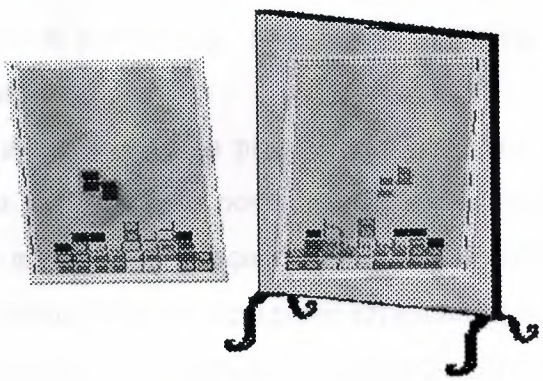

Figure 3.3. Mirror Result

After the identification of the falling piece, NeuroTetris activates the neural network associate with that particular piece: input neurons are fed with the pieces disposition on the game pad and one of the possible rotations; as output the network gives an evaluation for every possible combination of positions and rotations.

The combination with the best score is chosen and the piece is shifted and rotated accordingly. About one half of all the possible cases have been used during the phase of neural networks training.

The five networks are different in respect of both the training sets and the layers dimension.

In fact, input layer dimension depends on the number of possible rotations for the piece (cyan piece has no rotation; red, blue and green ones have 2 rotations; yellow, white and violet ones have 4 rotations) and on the dimensions of the piece.

Still the hidden layers differ from one network to another according to the hardness of the function implicitly hidden in the examples. Obviously the simpler the function the lower the number of neurons in the hidden layer.

3.10.4 INPUT

Window of the dimension of the piece (2, 3 or 4 neurons) where the pieces disposition is shown, obviously the window floates on the whole game pad. All possible rotations (0,2 or 4 neurons).

3.10.5 OUTPUT

Evaluation of the piece positioning. It is worth spending a word about the response speed of neural networks.

A window as wide as the maximum piece extent is shifted along the 10 Tetris game pad columns, and for every window position, all possible rotations are evaluated. We obtain thus that from a minimum of 9 (square piece) to a maximum of 32 ('T' and 'L' pieces) evaluations are performed at every new piece appearance (See figure 3.4.).

In spite of this high number of evaluations, while looking NeuroTetris playing, the bystander practically does not see any slackening in the game.

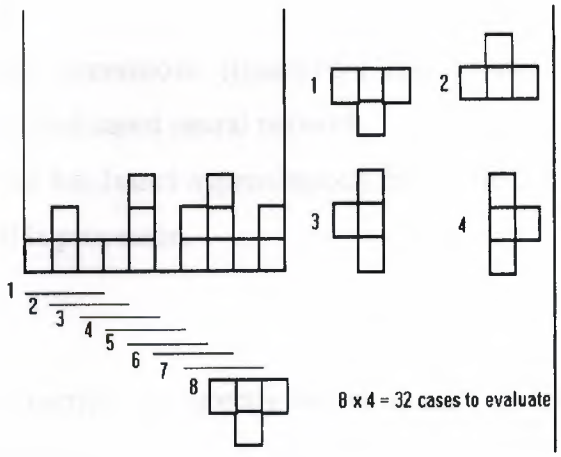

Fig. 3.4. Evaluation to be performed on the "T shaped" piece

We can observe that in this case the neural network has 3 input neurons used for the definition of the game pad piece disposition, 4 binary input neurons indicating the piece rotation and 1 output neuron for the evaluation of the piece positioning. (Remember that for every one of the 8 possible windows shifting, all the 4 possible rotations are evaluated).

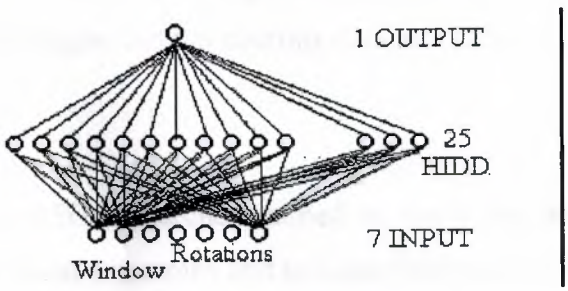

Fig.3.5. - The neural network for the "T" shaped piece (white)

3.10.6 Training phase

The hidden layer dimensions (figure3.5.) and epochs of training have been different on every piece-dedicated neural network.

The training phase has lasted approximately from 1 to 2 hours of computation on an Intel Pentium 60 MHz processor.

3.10.7 Test phase

Tests have been carried out directly on the quality of the games played by the networks on the Tetris game.

After a couple of weeks of neural network tuning, we reached the very encouraging record of 597 lines (remember that a very expert and skilled human player can reach up to 100 - 110 lines). The average game (on a test set of 200 plays) has been of 91 lines per game.

Unfortunately it is not possible to keep statistics directly on the score because Tetris' designers assumed that the limit of 32767 (2 bytes, the integer) could not be topped. Neurotetris often overshoots this score counter limit and makes it assume negative values.

A very interesting detail lays in the fact that neural networks with few hidden neurons (such as 4, 5) don't achieve good learning (10-15% of the examples are not perfectly learned) but perform a good game playing showing a very good skill. Best results are reached with higher hidden neurons number (such as 8, 25).

3.10.8 Conclusions

With this study ASG researchers aimed to show the feasibility of the use of artificial neural networks in a complex and delicate field such as decision-making.

Results have been encouraging and further developments in this area are planned for the immediate future.

The more immediate application area in the real world could be either automatic process control or the automatic management of financial goods [19].

NEAR EAST UNIVERSITY

Faculty of Engineering

Department of Computer Engineering

NEURAL NETWORKS IN INDUSTRIAL APPLICATIONS

Graduation Project
COM- 400

Student: Ahmad Ziad Al-Kurdy

Supervisor: Assoc.Prof.Dr. Adnan Khashman

Nicosia - 2004

AKNOWLEDGEMENT

First, I would like to thank my supervisor Assoc. Prof. Dr Adnan Khashman for his invaluable advice and for the generosity he exhibited with his time and effort over this project and through the courses I have taken with him.

Second, I thank my family especially parents, brothers for their continuous encouragement and support during preparation of this project.

Third, I would like to express my gratitude to Near East University for the scholarship that made the work possible.

Finally, I would like to thank all my friends, especially Alaa Al-Shanableh, Mohammed al-Arrouri, Ahmed Wattad, And Zouran for their support and their constant moral sustain.

ABSTRACT

Artificial neural networks (usually just called neural networks) are interconnected collections of simple, independent processors. While loosely modeled after the brain, the details of neural network design are not guided by biology. Instead, for over 20 years researchers have been experimenting with different types of nodes, different patterns of interconnection, and different algorithms for adjusting connections.

The difference between the behavior of programmed computation and neural networks, compare the operation of computers with humans. For example, a computer can perform mathematical operations more quickly and precisely than a human can. However, a human can recognize faces and complex images in a more precise, efficient, and faster manner than can the best computer. One of the reasons for this performance difference can be attributed to the distinct organization forms of computers and biological neural systems. A computer generally consists of a processor working alone, executing instructions delivered by a programmer one by one. Biological neural systems consist of billions of nervous cells (i.e., neurons) with a high degree of interconnection. Neurons can perform simple calculations without the need to be previously programmed.

This project presents an investigation into neural networks and their application in industry.

TABLE OF CONTENTS

INTRODUCTION

Neural networks are named after the cells in the human brain that perform intelligent operations. The brain is made up of billions of neuron cells. Each of these cells is like a tiny computer with extremely limited capabilities; however, connected together, these cells form the most intelligent system known. Neural networks are formed from hundreds or thousands of simulated neurons connected together in much the same way as the brain's neurons.

Just like people, neural networks learn from experience, not from programming. Neural networks are good at pattern recognition, generalization, and trend prediction. They are fast, tolerant of imperfect data, and do not need formulas or rules. Neural networks are trained by repeatedly presenting examples to the network. Each example includes both inputs (information you would use to make a decision) and outputs (the resulting decision, prediction, or response).

Your network tries to learn each of your examples in turn, calculating its output based on the inputs you provided. If the network output doesn't match the target output, BrainMaker corrects the network by changing its internal connections. This trial-and-error process continues until the network reaches your specified level of accuracy. Once the network is trained and tested, you can give it new input information, and it will produce a prediction. Designing your neural network is largely a matter of identifying which data is input, and what you want to predict, assess, classify, or recognize.

Neural networks are called machine-learning algorithms because changing these connections (training) causes the network to learn the solution to a problem. This differs from other artificial intelligence technologies, such as expert systems, fuzzy logic or constraint-based reasoning which must be programmed to solve a problem.

The aim of this project is to show how neural networks are robust and useful in many industrial applications. Neural networks can be used in several ways to model a given process. The project consists of introduction, four chapters and conclusion.

Chapter One presents the Biological Foundation of Neurocomputing and the development of neurocomputing, how to build a neural network, and some benefits of a neural network.

Chapter Two describes the different neural network models have been explored. These models are described as either unsupervised or supervised. Unsupervised neural networks, such as self-organizing feature maps, find relationships between input examples by examining the similarities and differences between the examples. Supervised neural networks, such as back propagation, are used for pattern recognition or prediction. For supervised neural networks, the input examples must be an accompanied by the desired output.

Chapter Three presents some Industrial Application of neural network like neural computing in Oil and Gas Industry, Neural Network In Tetris Games, Neural network in the pulp and paper industry and the Modeling of plate Rolling Process using Neural network.

Chapter Four presents the application of neural network in the modeling of plate rolling process.The cases that neural network were applied to real life in steelwork that the still do not have hot rolling models developed. And in the Thermal profile of slaps a neural network model was developed to forecast the inner temperture of the slabs being reheated as a function of reheating time and superficial temperatures. One of the disadvantages of neural networks is the complexity of hardware and software necessary to enable industrial application. For example, if process conditions change from those to used when training the neural network, data must once again be collected, analyzed and used for retraning the system.

CHAPTER ONE

INTRODUCTION

TO NEURAL NETWORK

1.1 Overview

This chapter presents the Biological Foundation of Neurocomputing and the development of neurocomputing, how to build a neural network, and some benefits of a neural network.

1.2 What is a Neural Network?

In information technology, a neural network is a system of programs and data structure that approximate the operation of the human brain. A neural network usually involves a large number of processors operating in parallel, each with its own small sphere of knowledge and access to data in its local memory. Typically, a neural network is initially "trained" or fed large amounts of data relationships (for example, "a grandfather is older than a person's father"). A program can tell the network how to behave in response to an external stimulus (for example, to input from a computer user who is interacting with the network) or can initiate activity on its own (within the limits of its access to the external world).

In making determinations, neural networks use several principles, including gradient-based training, fuzzy logic, genetic algorithms, and Bayesian methods. Neural networks are sometimes described in items of knowledge layers, with, in general, more complex networks having deeper layers. In feed-forward systems, learned relationships about data can "feed forward" to higher layers of knowledge. Neural networks can also learn temporal concepts and have been widely used in signal processing and time series analysis.

Current applications of neural networks include: oil exploration data analysis, weather prediction, the interpretation of nucleotide sequences in biology labs, and the

exploration of models of thinking and consciousness. In his novel, Galatea [1], Richard Powers envisioned a neural network (named "Helen") that could be thought to pass a comprehensive exam in English literature.

1.3 The Sudden Rise of Neurocomputing

The majority of information processing today is carried out by digital computers. This has led to the widely held misperception that information processing is dependent on digital computers. However, if we look at cybernetics and the other disciplines that form the basis of information science, we see that information processing originates with living creatures in their struggle to survive in their environments, and that the information being processed by computers today accounts for only a small part - the automated portion - of this. Viewed in this light, we can begin to consider the possibility of information processing devices that differ from conventional computers. In fact, research aimed at realizing a variety of different types of information processing devices is already being carried out, albeit in the shadows of the major successes achieved in the realm of digital computers. One direction that this research is taking is toward the development of an information processing device that mimics the structures and operating principles found in the information processing systems possessed by humans and other living creatures.

Digital computers developed rapidly in and after the late 1940's, and after originally being applied to the field of mathematical computations, have found expanded applications in a variety of areas, to include text (word), symbol, image and voice processing, i.e. pattern information processing, robot control and artificial intelligence. However, the fundamental structure of digital computers is based on the principle of sequential (serial) processing, which has little if anything in common with the human nervous system.

The human nervous system, it is now known, consists of an extremely large number of nerve cells, or neurons, which operate in parallel to process various types of information. By taking a hint from the structure of the human nervous system, we should be able to build a new type of advanced parallel information processing device.

In addition to the increasingly large volumes of data that we must process as a result of recent developments in sensor technology and the progress of information technology, there is also a growing requirement to simultaneously gather and process huge amounts of data from multiple sensors and other sources. This situation is creating

a need in various fields to switch from conventional computers that process information sequentially, to parallel computers equipped with multiple processing elements aligned to operate in parallel to process information.

Besides the social requirements just cited, a number of other factors have been at work during the 1980 [3] pt research on new forms of information processing devices. For instance, recent neurophysiological experiments have shed considerable light on the structure of the brain, and even in fields such as cognitive science, which study human information processing processes at the macro level, we are beginning to see proposals for models that call for multiple processing elements aligned to operate in parallel. Research in the fields of mathematical science and physics is also concentrating more on the mathematical analysis of systems comprising multiple elements that interact in complex ways. These factors gave birth to a major research trend aimed at clarifying the structures and operating principles inherent in the information processing systems of human beings and other animals, and constructing an information processing device based on these structures and operating principles.

1.4 Building A Neural Network

Since 1958, when psychologist Frank Rosenblatt [2] proposed the "Perceptron," a pattern recognition device with learning capabilities, the hierarchical neural network has been the most widely studied form of network structure. A hierarchical neural network is one that links multiple neurons together hierarchically, as shown in Figure 1.3. The special characteristic of this type of network is its simple dynamics. That is, when a signal is input into the input layer, it is propagated to the next layer by the interconnections between the neurons. Simple processing is performed on this signal by the neurons of the receiving layer prior to its being propagated on to the next layer. This process is repeated until the signal reaches the output layer completing the processing process for that signal.

The manner in which the various neurons in the intermediary (hidden) layers process the input signal will determine the kind of output signal it becomes (how it is transformed). As wee can see, then, hierarchical network dynamics are determined by the weight and threshold parameters of each of their units. If input signals can be transformed to the proper output signals by adjusting these values (parameters), then hierarchical networks can be used effectively to perform information processing.

Since it is difficult to accurately determine multiple parameter values, a learning method is employed. This involves creating a network that randomly determines parameter values. This network is then used to carry out input-to-output transformations for actual problems. The correct final parameters are obtained by properly modifying the parameters in accordance with the errors that the network makes in the process. Quite a few such learning methods have been proposed. Probably the most representative of these is the error back-propagation learning method proposed by D. E. Rumelhart et al. in 1986 [3]. This learning method has played a major role in the recent neurocomputing boom.

The back-propagation paradigm has been tested in numerous applications including bond rating, mortgage application evaluation, protein structure determination, backgammon playing, and handwritten digit recognition. Choosing the right methodology, or backpropagation algorithm, is another important consideration. In working with the financial applications, many have found that the back-propagation algorithm can be very slow. Without using advanced learning techniques to speed the process up, it is hard to effectively apply backpropagation to real-world problems. Overfitting of a neural network model is another area which can cause beginners difficulty. Overfitting happens when an ANN model is trained on one set of data, and it learns that data too well. This may cause the model to have poor generalization abilities - the model may instead give quite poor results for other sets of data.

1.5 The Analogy To The Brain

The most basic components of neural networks are modeled after the structure of the brain. Some neural network structures are not closely to the brain and some does not have a biological counterpart in the brain. However, neural networks have a strong similarity to the biological brain and therefore a great deal of the terminology is borrowed from neuroscience.

1.5.1 The Biological Foundation of NeuroComputing

Neurocomputing involves processing information by means of changing the states of networks formed by interconnecting extremely large numbers of simple processing elements, which interact with one another by exchanging signals. Networks such as the one just described are called artificial neural networks (ANNs), in the sense that they represent simplified models of natural nerve or neural networks(figure 1.1.).

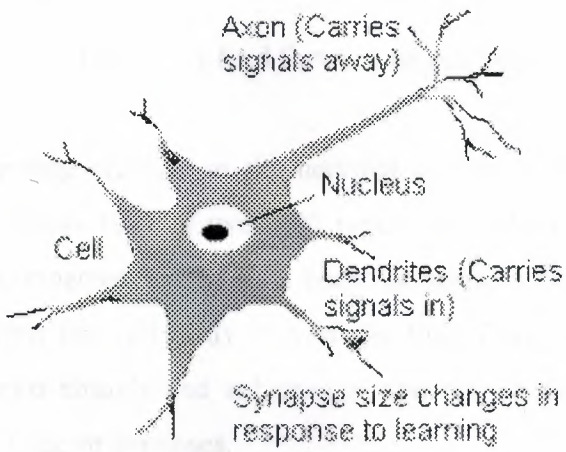

Fig. 1.1. A simple neuron cell

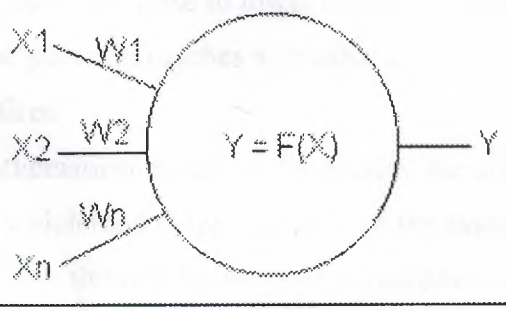

Fig. 1.2. A schematic diagram of a neuron

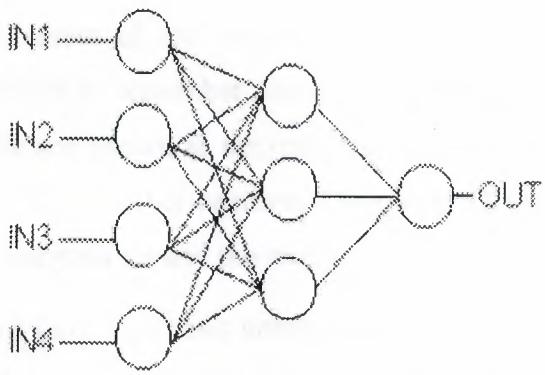

Fig.1.3. A feed forward neural network

The basic processing element in the nervous system is the neuron. The human brain is composed of about 1011 of over 100 types. Tree-like networks of nerve fiber called dendrites are connected to the cell body or soma, where the cell nucleus is located. Extending from the cell body is a single long fiber called the axon, which eventually branches into strands and substrands, and are connected to other neurons through synaptic junctions, or synapses.

The transmission of signals from one neuron to another at a synapses is a complex chemical process in which specific transmitter substances are released from the sending end of the junction. The effect is to raise to lower the electrical potential inside the body of the receiving cell. If the potential reaches a threshold, a pulse is sent down the axon - we then say the cell has "fired".

In a simplified mathematical model of the neuron, the effects of the synapses are represented by "weights" which modulates the effect of the associated input signals, and the nonlinear characteristics exhibited by neurons is represented by a transfer function which is usually the sigmoid function. The neuron impulse is then computed as the weighted sum of the input signals, transformed by the transfer function. The learning capability of an artificial neuron is achieved by adjusting the weights in accordance to the chosen learning algorithm, usually by a small amount *Wj = **Xj where * is called the learning rate and * the momentum rate.

1.6 The Biological Neuron

The most basic element of the human brain is a specific type of cell, which provides us with the abilities to remember, think, and apply previous experiences to our every action. These cells are known as neurons(see figure 1.4.), each of these neurons can connect with up to 200000 other neurons. The power of the brain comes from the numbers of these basic components and the multiple connections between them.

All natural neurons have four basic components, which are dendrites, soma, axon, and synapses. Basically, a biological neuron receives inputs from other sources, combines them in some way, performs a generally nonlinear operation on the result, and then output the final result. The figure below shows a simplified biological neuron and the relationship of its four components.

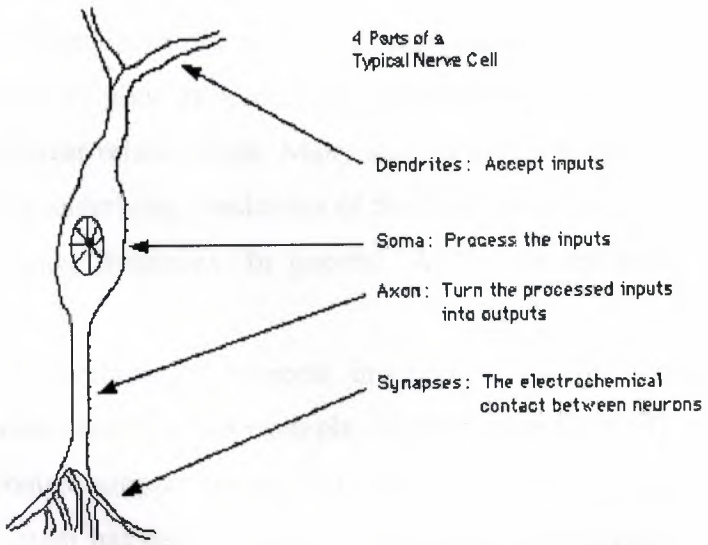

Figure 1.4. The biological neuron

1.7 Why we use neural network?

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 analyze. This expert can then be used to provide projections given new situations of interest and answer "what if" questions.Other advantages include:

Adaptive learning: An ability to learn how to do tasks based on the data given for training or initial experience.

Self-Organisation: An ANN can create its own organisation or representation of the information it receives during learning time.

Real Time Operation: ANN computations may be carried out in parallel, and special hardware devices are being designed and manifactured which take advantage of this capability.

Fault Tolerance via Redundant Information Coding: Partial destruction of a network leads to the corresponding degradation of performance. However, some network capabilites may be retained even with major network damage.

1.8 Some Real Life Application

ANNs can be regarded, in one respect, as multivariate nonlinear analytical tools, and are known to be very good at recognizing patterns from noisy, complex data, and estimating their nonlinear relationships. Many studies have shown that ANNs have the capability to learn the underlying mechanics of the time series, or, in the case of trading applications, the market dynamics. In general, ANNs are known to possess these capabilities:

A number of development projects involving ANN technology have been publicized in the media recently. For example, Nippon Steel Corp [4]. has built a blast furnace operation control support system that makes use of ANNs. The neural network employed in this system has been equipped with functions that enable it to learn the relationship between sensor data and the eight kinds of temperature distribution patterns known from experience to pertain to the overall operation of blast furnaces, and to instantaneously recognize and output that pattern which most closely approximates sensor data input into the system. The neural network learns very quickly, and achieves a better than 90% pattern recognition ratio following learning. Since this system has been performing extremely well during operational testing, Nippon Steel is planning to introduce it into other aspects of its operations in addition to blast furnace control, to include the diagnosis of malfunctions, and other control processes.

A second example is the experimental work started by Daiwa Securities Co. , Ltd. and NEC [5] Corporation on applying neural network technology to the learning and

recognition of stock price chart patterns for use in stock price forecasting. NEC had already developed neural network simulation software for use on its EWS 4800 series of workstations, and, by limiting stock price chart pattern learning to a few dozen major stocks, has improved the accuracy of this software's forecasting capabilities. Based on these results, the Daiwa Computer Services Co., Ltd., an information processing subsidiary of the Daiwa Securities Group, transferred the NEC system to its supercomputer and taught it to recognize the stock price chart patterns for 1,134 companies listed on the Tokyo Stock Exchange. DCS has since been putting this system to good use in the performance of stock price forecasting.

Mitsubishi Electric has combined neural network technology with optical technology to achieve the world's first basic optical neurocomputer system capable of recognizing the 26 letters of the alphabet. The system comprises a set of light-emitting diodes (LED) that output letter patterns as optical signals, optical fibers, liquid crystal displays (LCD) that display letter patterns and light receiving devices that read these letters. When letter data is input into this system, light emitted from the LEDs is input to the light receiving devices through the LCDs. At that time, the light receiving devices that receive the light, as well as the strength of the light they receive, is determined by the manner in which that light passes through the LCDs. The letter in question is delineated by the light receiving devices that receive the strongest light. This system is capable of 100% letter recognition even when slightly misshapen handwritten letters are input.

A fourth example is a development project for a facilities diagnosis system that employs a neural network system commenced by the Nippon Oil Co. [6], Ltd. in cooperation with CSK Research Institute. This project is attracting considerable attention as it is the first time research has been carried out on applying neural network systems to facilities diagnosis. Initially, the project will be aimed at developing a diagnosis system for pump facilities that employs vibration analysis. Nippon Oil operates a total of 1,500 pumps at its Negishi Oil Refinery in Yokohama, Kanagawa Prefecture alone, and must retain large numbers of experienced personnel to maintain these pumps. The company decided to apply neural network technology to pump facilities diagnosis operations with the ultimate goal of saving labor in mind.

1.9 The Future

Neural Networks will fascinate user-specific systems for education, information processing, and entertainment. "Alternative ralities", produced by comprehensive environments, are attractive in terms of their potential for systems control, education, and entertainment. This is not just a far-out research trend, but is something which is becoming an increasing part of our daily existence, as witnessed by the growing interest in comprehensive "entertainment centers" in each home. This "programming" would require feedback from the user in order to be effective but simple and "passive" sensors (e.g fingertip sensors, gloves, or wristbands to sense pulse, blood pressure, skin ionisation, and so on), could provide effective feedback into a neural control system. This could be achieved, for example, with sensors that would detect pulse, blood pressure, skin ionisation, and other variables which the system could learn to correlate with a person's response state.

Neural networks, integrated with other artificial intelligence technologies, methods for direct culture of nervous tissue, and other exotic technologies such as genetic engineering, will allow us to develop radical and exotic life-forms whether man, machine, or hybrid.

Neural networks will allow us to explore new realms of human capabillity realms previously available only with extensive training and personal discipline. So a specific state of consiously induced neurophysiologically observable awareness is necessary in order to facilitate a man machine system interface.

1.10 Some Advantages of Neural Network

Neural networks and artificial intelligence based models have potential advantages over other models due to following reasons

(1) Networks start processing the data without any preconceived hypothesis. They start with random weight assignment to various input variables. Adjustments are made based on the difference between predicted and actual output. This allows for unbiased and better understanding of data. It also some times help in unraveling subtle relationship between various input variables and out put being studied.

(2) Neural networks can be retrained using additional input variables and number of individuals. Once trained they can be called on to predict in a new patient.

(3) There are several neural network models available to chose from in a particular problem.

(4) Once trained, they are very fast.

(5) Due to increase accuracy, results in cost saving.

Disadvantages

(1) No set rule for network selection.

(2) Need experts in training the network and in-depth understanding of medical prpblem in hand.

(3) Training is quite time consuming and test the patience.

1.11 Summary

A neural network is a system of programs and data structure that approximate the operation of human brain . A neural network usually involves a large number of processors operating in parallel.

In this chapter we cover the development of neurocomputing, how to build a neural network and some benefits of a neural network.

In the next chapter we will cover the classifications of neural network, and the manner in which neural network are structed.

CHAPTER TWO
THE STRUCTURE OF NEURAL NETWORKS

2.1 Overview

This chapter presents two classifications of neural network according to first flow of information like recurrent network, feed forward network, and competitive network and second the way of learning like supervised learning and unsupervised learning.

2.2 Are There Any Limits To Neural Networks?

The major issues of concern today are the scalability problem, testing, verification, and integration of neural network systems into the modern environment. Neural network programs sometimes become unstable when applied to larger problems. The defense, nuclear and space industries are concerned about the issue of testing and verification. The mathematical theories used to guarantee the performance of an applied neural network are still under development. The solution for the time being may be to train and test these intelligent systems much as we do for humans. Also there are some more practical problems. The operational problem encountered when attempting to simulate the parallelism of neural networks. Since the majority of neural networks are simulated on sequential machines, giving rise to a very rapid increase in processing time requirements as size of the problem expands.

Solution: implement neural networks directly in hardware, but these need a lot of development still. Instability to explain any results that they obtain. Networks function as "black boxes" whose rules of operation are completely unknown.

2.3 The Artificial Neuron

The basic unit of neural networks, the artificial neurons, simulates the four basic functions of natural neurons. Artificial neurons are much simpler than the biological neuron; the figure below shows the basics of an artificial neuron.

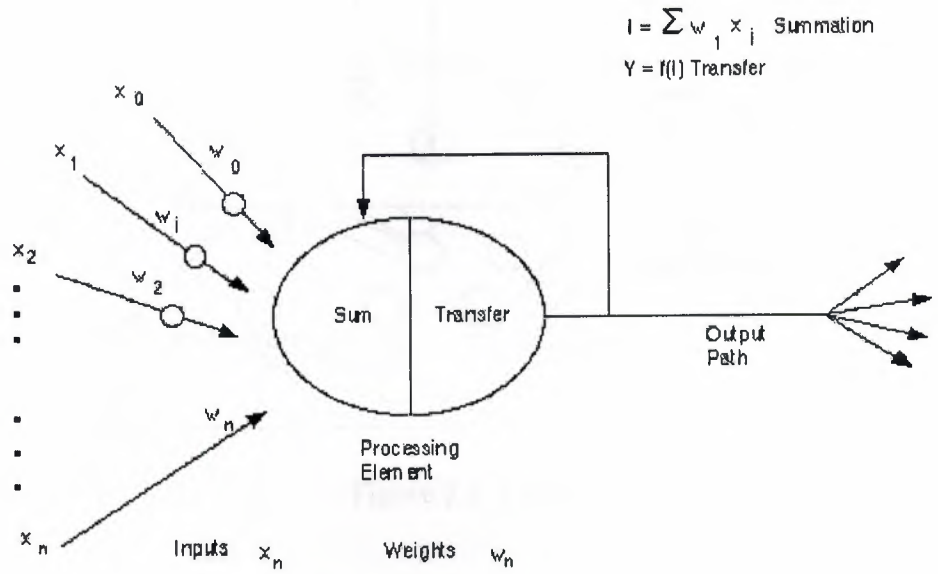

Figure 2.1. The Artificial Neuron

Note that various inputs to the network are represented by the mathematical symbol, x(n). Each of these inputs are multiplied by a connection weight, these weights are represented by w(n). In the simplest case, these products are simply summed, fed through a transfer function to generate a result, and then output.

Even though all artificial neural networks are constructed from this basic building block the fundamentals may vary in these building blocks and there are differences.

2.4 Layers

Biologically, neural networks are constructed in a three dimensional way from microscopic components. These neurons seem capable of nearly unrestricted interconnections. This is not true in any man-made network. Artificial neural networks are the simple clustering of the primitive artificial neurons. This clustering occurs by creating layers, which are then connected to one another. How these layers connect may

also vary. Basically, all artificial neural networks have a similar structure of topology. Some of the neurons interface the real world to receive its inputs and other neurons provide the real world with the network's outputs. All the rest of the neurons are hidden form view.

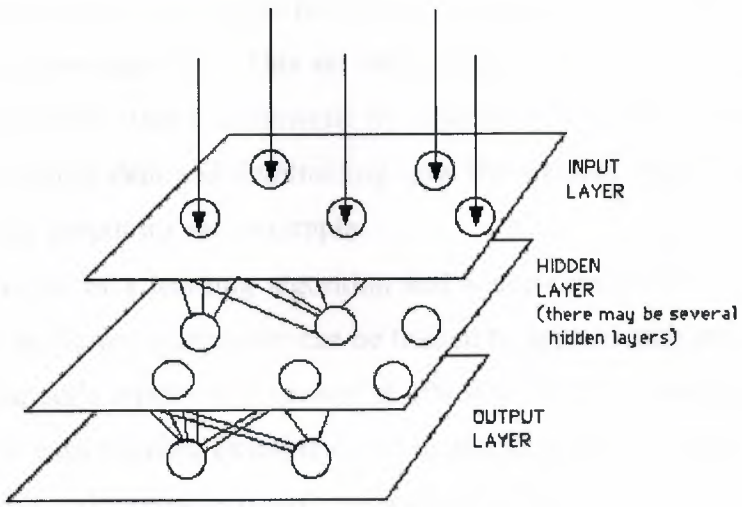

Figure 2.2. Layers

As the figure above shows, the neurons are grouped into layers The input layer consist of neurons that receive input form the external environment. The output layer consists of neurons that communicate the output of the system to the user or external environment. There are usually a number of hidden layers between these two layers; the figure above shows a simple structure with only one hidden layer.

When the input layer receives the input its neurons produce output, which becomes input to the other layers of the system. The process continues until a certain condition is satisfied or until the output layer is invoked and fires their output to the external environment.

To determine the number of hidden neurons the network should have to perform its best, one are often left out to the method trial and error. If you increase the hidden number of neurons too much you will get an over fit, that is the net will have problem to generalize. The training set of data will be memorized, making the network useless on new data sets.

2.5 Network Training

To train a neural network to approximate a desired function, a learning algorithm is used. In what is called unsupervised leaning, a learning algorithm automatically adjusts a neural network's weights in order improve its ability to give a desired output from a given input. Learning requires having a pre-defined set of inputs and desired outputs available to the algorithm. This set of training examples is called the training set. A learning algorithm trains a network by repeatedly looking at how a network responds to this training data and determining how the weights should be adjusted in order to improve the output for each example.

Through the use of a learning algorithm and a non-contradictory training set, a neural network of sufficient complexity can be trained to approximate any function. For example, a training set's input could consist of 100,000 hand-written letters, while the desired outputs for each training example could be the actual letter which was written. Each time the training algorithm is run on the training set, the neural network's weights are adjusted so that each training example gives an output which is closer to the desired output than it was before the training algorithm took place. Training algorithms are usually run over and over until the network produces outputs that are sufficiently close to the desired output for each training example.

2.6 Memorization and Generalization

To simulate intelligent behavior the abilities of memorization and generalization are essential. These are basic properties of artificial neural networks. The following definitions are according to the Collins English Dictionary: (table 2.1)

Table 2.1 defintion of memorizing and generalizing

To memorize:	To commit to memory; learn so as to remember.
To generalize:	To form general principles or conclusions from detailed facts, experience, etc.

Memorizing, given facts, is an obvious task in learning. This can be done by storing the input samples explicitly, or by identifying the concept behind the input data, and memorizing their general rules.

The ability to identify the rules, to *generalize*, allows the system to make predictions on unknown data.

Despite the strictly logical invalidity of this approach, the process of reasoning from specific samples to the general case can be observed in human learning.

Generalization also removes the need to store a large number of input samples. Features common to the whole class need not to be repeated for each sample - instead the system needs only to remember which features are part of the sample. This can dramatically reduce the amount of memory needed, and produce a very efficient method of memorization.

2.7 Classification of Neural Networks

Neural Network models can be classified in a number of ways. Using the network architecture as basis, there are three major types of neural networks:

2.7.1 Recurrent networks

The units are usually laid out in a two-dimensional array and are regularly connected. Typically, each unit sends its output to every other unit of the network and receives input from these same units. Recurrent networks are also called *feedback networks.* Such networks are "clamped" to some initial configuration by setting the activation values of each of the units. The network then goes through a stabilization process where the network units change their activation values and slowly evolve and converge toward a final configuration of "low energy". The final configuration of the network after stabilization constitutes the output or response of the network. This is the architecture of the *hopfield Model*

2.7.2 Feed forward networks

These networks distinguish between three types of units: input units, hidden units, and output units. The activity of this type of network propagates forward from one layer to the next, starting from the input layer up to the output layer. Sometimes called

multiplayer networks, feed forward networks are very popular because this is the inherent architecture of the Back propagation Model.

2.7.3 Competitive networks

These networks are characterized by lateral inhibitory connections between units within a layer such that the competition process between units causes the initially most active unit to be the only unit to remain active, while all the other units in the cluster will slowly be deactivated. This is referred to as a "winner-takes-all" mechanism. Self-Organizing Maps, Adaptive Resonance Theory, and Rumelhart & Zipser's [7] Competitive Learning Model are the best examples for these types of networks.

The network architecture can be further subdivided into whether the network structure is fixed or not. There are two broad categories:

Static architecture: Most of the seminal works on neural networks were based on static network structures, whose interconnectivity patterns are fixed *a priori*, although the connection weights themselves are still subject to training. Perceptrons, multi-layered perceptrons, self-organizing maps, and Hopfield networks all have static architecture.

Dynamic architecture: some neural networks do not constrain the network to a fixed structure but instead allow nodes and connections to be added and removed as needed during the learning process. Some examples are Grossberg's Adaptive Resonance Theory and Fritzke's "Neural Gas". Some adding-pruning approaches to Multi-Layered Perceptron networks have also been widely studied.

2.8 Learning

The brain basically learns from experience. Neural networks are sometimes called machine learning algorithms, because changing of its connection weights (training) causes the network to learn the solution to a problem. The strength of connection between the neurons is stored as a weight-value for the specific connection. The system learns new knowledge by adjusting these connection weights.

The learning ability of a neural network is determined by its architecture and by the algorithmic method chosen for training.

2.8.1 Off-line or On-line

One can categorize the learning methods into yet another group, off-line or on-line. When the system uses input data to change its weights to learn the domain knowledge, the system could be in training mode or learning mode. When the system is being used as a decision aid to make recommendations, it is in the operation mode, this is also sometimes called recall.

Off-line

In the off-line learning methods, once the systems enters into the operation mode, its weights are fixed and do not change any more. Most of the networks are of the off-line learning type.

On-line

In on-line or real time learning, when the system is in operating mode (recall), it continues to learn while being used as a decision tool. This type of learning has a more complex design structure.

2.8.2 Learning laws

There are a variety of learning laws which are in common use. These laws are mathematical algorithms used to update the connection weights. Most of these laws are some sort of variation of the best known and oldest learning law, Hebb's Rule[8]. Man's understanding of how neural processing actually works is very limited. Learning is certainly more complex than the simplification represented by the learning laws currently developed. Research into different learning functions continues as new ideas routinely show up in trade publications etc. A few of the major laws are given as an example below.

2.8.2.1 Hebb Law

The first and the best known learning rule was introduced by Donald Hebb. The description appeared in his book *The organization of Behavior* in 1949[9]. This basic rule is: If a neuron receives an input from another neuron, and if both are highly active (mathematically have the same sign), the weight between the neurons should be strengthened.

2.8.2.2 Hopefield Law

This law is similar to Hebb's Rule with the exception that it specifies the magnitude of the strengthening or weakening. It states, "if the desired output and the input are both active or both inactive, increment the connection weight by the learning rate, otherwise decrement the weight by the learning rate." (Most learning functions have some provision for a learning rate, or a learning constant. Usually this term is positive and between zero and one.)

2.8.2.3 The Delta Rule

The Delta Rule is a further variation of Hebb's Rule, and it is one of the most commonly used. This rule is based on the idea of continuously modifying the strengths of the input connections to reduce the difference (the delta) between the desired output value and the actual output of a neuron. This rule changes the connection weights in the way that minimizes the mean squared error of the network. The error is back propagated into previous layers one layer at a time. The process of back-propagating the network errors continues until the first layer is reached. The network type called Feed forward, Back-propagation derives its name from this method of computing the error term.This rule is also referred to as the Windrow-Hoff Learning Rule and the Least Mean Square Learning Rule.

2.8.2.4 Kohonen's Learning Law

This procedure, developed by Teuvo Kohonen, was inspired by learning in biological systems. In this procedure, the neurons compete for the opportunity to learn, or to update their weights. The processing neuron with the largest output is declared the winner and has the capability of inhibiting its competitors as well as exciting its neighbors. Only the winner is permitted output, and only the winner plus its neighbors are allowed to update their connectionweights. The Kohonen rule does not require desired output. Therefor it is implemented in the unsupervised methods of learning. Kohonen has used this rule to create the self-organizing neural network, which has an unsupervised learning method.

2.9 Supervised learning

These are generally the learn-by-example methods where user-supplied information is provided with each training pattern. These guide the neural network in adjusting its parameters. The perceptrons and back propagation networks are classic examples of supervised learning models.

2.10 Backpropagation

The term backpropagation has been used to refer to two distinct processes, one in the field of artificial neural networks (ANNs) and the other in neurobiology. Use of the same term is causing confusion: The existence of biological "backpropagation" hints that a process like ANN backpropagation may be present in biological neural networks. Here it is shown that the two are not related.

2.10.1 What is backpropagation?

There are at least three usages, the earliest from the field of artificial neural networks, and two that arose recently in neurobiology. A brief description of each, along with their distinctive features, will help exhibit the differences between the ANN and biological meanings.

2.10.2 Backpropagation in artificial neural networks

Is a mathematical technique for minimizing the discrepancy between a parametrized function and a set of pairs of inputs and "correct" outputs, where the overall function is partitioned into layers of vector functions. Each component of a layer is a weighted sum (or in some cases a product) of outputs from the previous layer, fed through some continuous scalar function. The "network" is regarded as the connections of outputs of one layer to inputs of the next. It is purely feed-forward -- there are no outputs from later layers that supply inputs of earlier layers. (Other forms of ANN contain explicit feedback connections of outputs to earlier inputs, but they aren't trained using backpropagation.)

The backpropagation algorithm adjusts the weights in the sums at each layer, in order to reduce the discrepancies between the network outputs before adjustment, and the desired values.

The technique originated with Werbos in 1974 [10], but did not come to the attention of the neural network community at that time. It was reinvented roughly simultaneously in 1985-6 by Le Cun , Parker , and Rumelhart[11], Hinton, and Williams [12], who named it backpropagation.

Some characteristics of ANN backpropagation, which will serve to distinguish it from biological "backpropagation" and to point out similarities, are:

ANN backpropagation provides direct feedback to the entire network. The process of updating the network weights takes information from the set of training values and from the current network output, and uses these to adjust all the network weights. Thus, in spite of the fact that the "network" is feed-forward, the training process adds unequivocal and global feedback. The feedback path looks like the figure below:

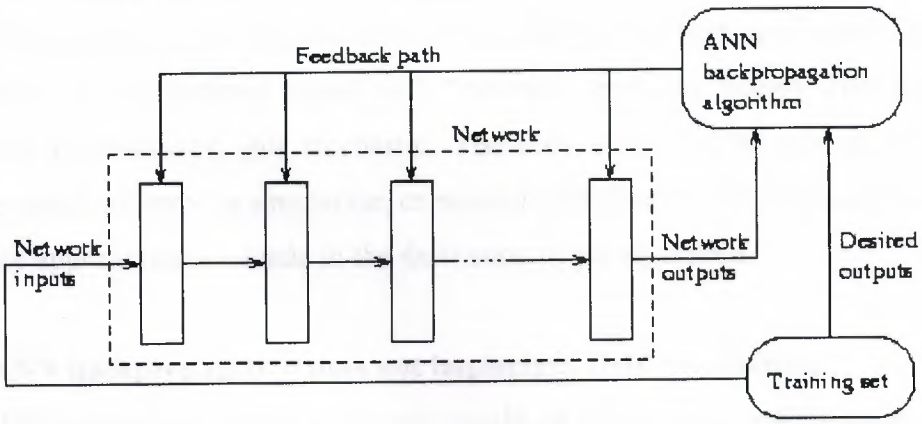

Figure 2.3. Backpropagation algorithm

There is no retrograde "flow" of material or information through the network layers. The updating of weights is effectively simultaneous across the network. The backpropagation algorithm uses all the network parameters, together with the actual and desired outputs, and computes new parameters, which it then installs in the network, while the network is not in use. The fact that the algorithm may, when run as a simulation, update the weights one at a time is an artifact of processing by a sequential computer. Even the flow of computation is primarily forward the algorithm iterates forward over the network layers twice, but only once backward. In fact, the final step, in which the weights are updated, is most efficiently performed while iterating forward over the network layers.

The above drawing might tempt one to say ANN feedback could be represented by a directed graph with cycles, rather than by an undirected (i.e. bidirectional) graph. But this is a bit misleading: In a directed graph, we can separately install edges in each direction, and control what is connected to what. But the ANN backpropagation algorithm uses all the weights, and updates all of them. ANN feedback can't be selectively blocked from parts of the network.

There is, however, a temporal restriction on the feedback path. ANN feedback is sequential, not combinational or recurrent (in the sense of being always operative). Production of the "next" set of weights does not interfere with production of the current network outputs.

2.10.3 ANN backpropagation is not a physical process

Although it may be implemented in hardware rather than simulated. There is no degradation of the feedback signal with "distance" from the output. The updating of weights is deterministic, not stochastic. The only errors in the process (other than possible round-off error in simulation, or noise if implemented electronically) arise from deliberate approximations made in the derivation of the algorithm.

2.10.4 ANN backpropagation does not implement Hebbian learning

Hebbian learning alters a network weight up or down depending on how closely the input and output connected by that weight do or do not match. Rather, the outcome of ANN backpropagation is function approximation. The network fits a continuous function to the data points supplied in the training set, that is, it matches the network's output to an externally-provided desired output.

There are a number of variants of ANN backpropagation, but they all intend to match network outputs to desired outputs. So one might say there is some "ideal" version of backpropagation, and others are approximations. In that sense, the operation of ANN backpropagation is fixed by its purpose we cannot alter it freely.

Because it attempts to match externally-supplied desired values, A backpropagation implements a type of supervised learning.

Note that traditional Hebbian learning is unsupervised. It uses only the inputs and output of the network. In so called "supervised Hebbian learning", a weight is adjusted depending on closeness of the input and desired output. This latter can implement an

autoassociative memory. But even it doesn't do function approximation, because it's not matching the network output to the desired output, nor is it truly Hebbian learning.

2.10.5 When to use (or not!) a BP Neural Network Solution

A back-propagation neural network is only practical in certain situations. Following are some guidelines on when you should use another approach:

- Can you write down a flow chart or a formula that accurately describes the problem? If so, then stick with a traditional programming method.
- Is there a simple piece of hardware or software that already does what you want? If so, then the development time for a NN might not be worth it.
- Do you want the functionality to "evolve" in a direction that is not pre-defined? If so, then consider using a Genetic Algorithm (that's another topic!).
- Do you have an easy way to generate a significant number of input/output examples of the desired behavior? If not, then you won't be able to train your NN to do anything.
- Is the problem is very "discrete"? Can the correct answer can be found in a look-up table of reasonable size? A look-up table is much simpler and more accurate.
- Are precise numeric output values required? NN's are not good at giving precise numeric answers.
- Conversely, here are some situations where a BP NN might be a good idea:
- A large amount of input/output data is available, but you're not sure how to relate it to the output.
- The problem appears to have overwhelming complexity, but there is clearly a solution.
- It is easy to create a number of examples of the correct behavior.
- The solution to the problem may change over time, within the bounds of the given input and output parameters (i.e., today 2+2=4, but in the future we may find that 2+2=3.8).
- Outputs can be "fuzzy", or non-numeric.

One of the most common applications of NNs is in image processing. Some examples would be: identifying hand-written characters; matching a photograph of a person's face with a different photo in a database; performing data compression on an image with minimal loss of content. Other applications could be: voice recognition; radar signature analysis; stock market prediction. All of these problems involve large amounts of data, and complex relationships between the different parameters.

It is important to remember that with a NN solution, you do not have to understand the solution at all! This is a major advantage of NN approaches. With more traditional techniques, you must understand the inputs, and the algorithms, and the outputs in great detail, to have any hope of implementing something that works. With a NN, you simply show it: "this is the correct output, given this input". With an adequate amount of training, the network will mimic the function that you are demonstrating. Further, with a NN, it is OK to apply some inputs that turn out to be irrelevant to the solution - during the training process, the network will learn to ignore any inputs that don't contribute to the output. Conversely, if you leave out some critical inputs, then you will find out because the network will fail to converge on a solution.

If your goal is stock market prediction, you don't need to know anything about economics, you only need to acquire the input and output data.

2.11 Unsupervised learning

Some neural network models do not need category information to accompany each training pattern, although such information would still be required in the interpretation and labeling of the resultant networks. Classical examples of these are Kohonen's self-organizing maps and Grossberg's Adaptive Resonance Theory.

It also makes sense to classify neural network models on the basis of their over-all task:

Pattern association: the neural network serves as an associative memory by retrieving an associated output pattern given some input pattern. The association can be *auto-associative* or *hetero-associative*, depending on whether or not the input and output patterns belong to the same set of patterns.

Classification: the network seeks to divide the set of training patterns into a pre-specified number of categories. Binary-valued output values are generally used for

classification, although continuous-valued outputs (coupled with a labeling procedure) can do classification just as well. For binary output representation, each category is generally represented by a vector (sequence) of 0's; with a single 1 whose position in the vector denotes the category.

Function approximation: the network is supposed to compute some mathematical function. The network's output represents the approximated value of the function given the input pattern as parameters. In certain areas, *regression* may be the more natural term.

There are other bases for classifying neural network models, but these are less fundamental than those mentioned earlier. Some of these include the type of input patterns that can be admitted (binary, discrete valued, real values), or the type of output values that are produced (binary, discrete-valued, real values).

2.12 Kohonen's Self Organizing Map (SOF)

Kohonon's SOMs are a type of unsupervised learning. The goal is to discover some underlying structure of the data. However, the kind of structure we are looking for is very different than, say, PCA or vector quantization.

Kohonen's SOM is called a topology-preserving map because there is a topological structure imposed on the nodes in the network. A topological map is simply a mapping that preserves neighborhood relations.

In the nets we have studied so far, we have ignored the geometrical arrangements of output nodes. Each node in a given layer has been identical in that each is connected with all of the nodes in the upper and/or lower layer. We are now going to take into consideration that physical arrangement of these nodes. Nodes that are "close" together are going to interact differently than nodes that are "far" apart.

What do we mean by "close" and "far"? We can think of organizing the output nodes in a line or in a planar configuration as shown in the figure below.

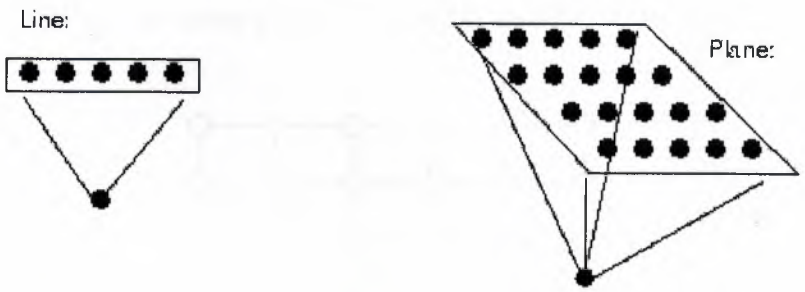

Figure 2.4. Kohonen's network

The goal is to train the net so that nearby outputs correspond to nearby inputs.E.g. if x1 and x2 are two input vectors and t1 and t2 are the locations of the corresponding winning output nodes, then t1 and t2 should be close if x1 and x2 are similar. A network that performs this kind of mapping is called a feature map.

In the brain, neurons tend to cluster in groups. The connections within the group are much greater than the connections with the neurons outside of the group. Kohonen's network tries to mimick this in a simple way.

2.12.1 Architecture of SOM

The architecture of the SOM can be a regular rectangular 2D structure, a hexagonal or honeycomb structure, a 3D rectangular structure, etc. 2-D structures are the most commonly used because maps are then more readily visualizable. Some other promising structures, like spherical maps where nodes are laid out regularly on the surface of the sphere, are also highly visualizable but are so far ignored in the literature.

The architecture affects the ordering of the data since neighborhood functions are used for the Kohonen training algorithm. The amount of change in the connection weights depends on its spatial distance in the map from the so called *winning unit*.

The original SOM, described by Teuvo Kohonen in 1980[13], has fixed map structures. The dimensions of the map structure has to be wisely chosen *a priori*. Some more recent variants of the SOM allow for dynamic allocation (both growth and pruning) of network units and the structure adapts to the inherent structure of the input environment.

Below are some examples of SOM fixed architectures.

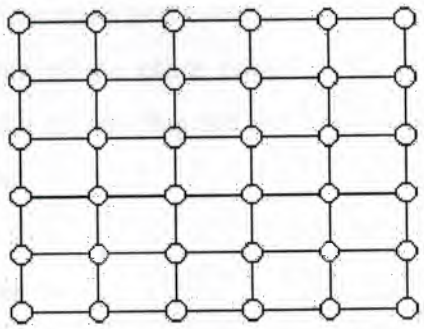

Figure 2.5. 2D rectangular architecture.

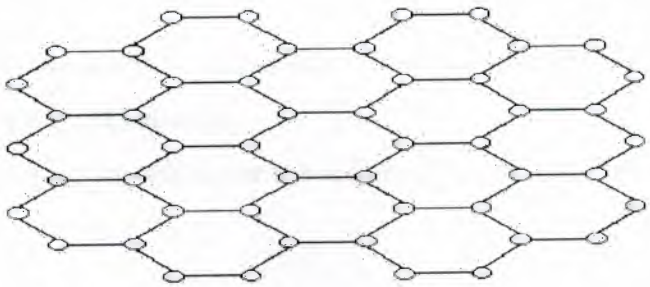

Figure 2.6. 2D honeycomb architecture

2.13 Characteristics of ANN

As discussed above, artificial neural networks are of different architectures, different learning schemes, varying weight update modalities, and may be called to perform different tasks. Yet, some basic characteristics can be said of most neural network models:

2.13.1 Inherent parallelism

Practically all-neural network models have some element of parallelism in the execution of its numerous components, although some degree of sequentiality of operation is observed in at least a component of these systems;

Regularity of components to a very a large extent, the basic components of a neural network, referred to in the literature as units, nodes, or neurons, all look alike and behave similarly.

2.13.2 Access to local information

Neural networks are composed of nodes that are interconnected with each other. Any given node's level of activation and eventual output will depend exclusively on its current state and the outputs of the other nodes to which it is connected. Whatever happens to other nodes in the system that are not connected to a given neuron will not directly affect its actions; and

2.13.3 Incremental learning

Neural networks do not learn any given concept in one go. Instead, the network parameters undergo several small changes, which, over time, would come to settle on their final values.

2.14 Summary

In this chapter we cover two classification of neural network according to Flow of information and the way of learning.

In the next chapter we will cover the industrial application of neural networks.

CHAPTER THREE
INDUSTRIAL APPLICATIONS
OF NEURAL NETORKS

3.1 Overview

This chapter presents some Industrial Application of neural network like neural computing in Oil and Gas Industry, Neural Network In Tetris Games, Neural network in the pulp and paper industry and the Modeling of plate Rolling Process using Neural network.

3.2 Neural Network in Papermaking Plant

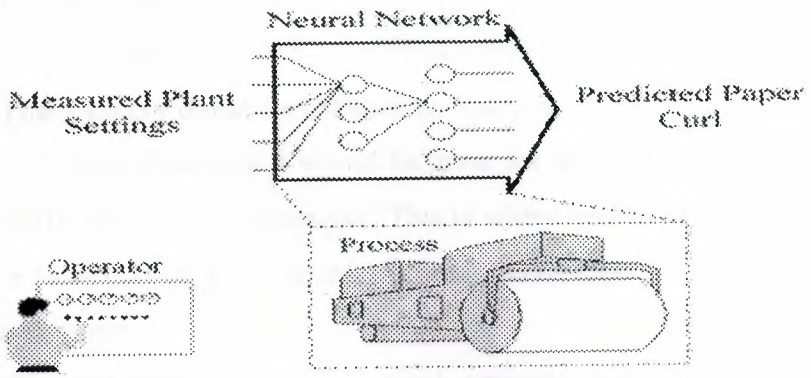

Figure 3.1. Neural Network in Papermaking Industry

The Neural Network Group (part of the Integrated Systems Group) is working together with two large paper-making firms to produce ANN (figure3.1.) based software to optimise the quality of paper from a paper-making plant.

In conventional operation a human operator, drawing on years of experience, continually modifies the settings on a huge machine which is rolling paper and coating it with a material that gives it a glossy surface. Among the variables to be controlled are the pressure on a lot of rollers, the temperature and the speed of the paper through the process. All are inter-related, usually in a highly non-linear fashion, and finding and maintaining the correct settings is a very inexact process more akin to alchemy than science.

Success is measured by, amongst other things, looking at the "curl" of the parer. "Curl" is to do with how a sheet of paper, initially flat, curls if it is held over a sharp edge, like the edge of a desk.

An ANN is currently being trained to mimic the work done by the human operator with the aim of first assisting him and finally relieving him from the tedium of constantly having to monitor the process.

3.3 Neural networks in the pulp and paper industry?

There are many processes in a pulp and paper mill where an on-line parameter analyzer cannot be used due to several reasons:

- The analyzer is very expensive to buy
- It cannot survive in the environment we want to use it
- It is not operational due to hardware problems, maintenance etc.
- There simply doesn't exist such an analyzer.

In all these situations it would be great for the mill to have an alternative way to continuously measure the parameter. This is where neural networks come to place. They can serve as virtual sensors that infer process parameters from other variables, which are measured on-line.

Inferential sensors based on neural network methodologies can be used for real time prediction of:

- Paper properties like tensile, stretch, brightness, opacity, softness etc.
- Digester kappa numbers
- Sodium chlorate concentrations and suspended solid levels in ClO_2 generators.
- Boiler stack emissions

3.4 What are the benefits an inferential sensor can offer?

3.4.1 Parameter monitoring in real time

The usual procedure followed in any of the above processes when an on-line sensor is not available, is to collect samples very infrequently, send them to the laboratory and wait for the results. This procedure may take several hours and if the lab

analysis shows that the product is out of specification, a substantial production time has been lost.

Suppose an inferential sensor is available to accurately predict process parameters in real time. This sensor can continuously monitor the parameters and is also able to provide us with a prediction of what is going to happen in the future. In case some parameters tend to be driven out of specifications, we have the time to take appropriate actions. Operational time will be saved and bad quality production will be minimized.

3.4.2 Process optimization

The inferential sensor based on neural networks is not only a tool that is used for parameter prediction. It also gives us the following important information about the process:

- The process input variables, which have the greatest impact on the product quality.
- The variables, which are not significant because they have little effect on the output of the process.
- The dead time for each process input variable, i.e. the time required for a change in the input variable to start affecting the output variable.

The model can also be used in off-line mode to run different scenarios. For example, we can simulate the response of the product quality parameters following a change in any input variable, like flow rate, temperature or pressure. In this manner we can improve the operation and optimize the quality properties of the product, by selecting the best possible set points of the input variables.

3.4.3 Process control

Model Predictive Control (MPC) is a relatively new control methodology, which became popular because it is robust, multivariable and takes care of input and output constraints. However, it requires an on-line sensor of the controlled variables. This is an excellent application of inferential analyzers based on neural network. They can serve as real time sensors in an MPC scheme, resulting in:

- Continuous control of the process.
- Process stabilization.

- Variability reduction.
- Disturbance rejection.

3.4.4 Neural Networks can save you money

Think of any process in your plant where you wish you could have some parameter prediction. A neural network can develop sample model for your process. Find out how powerful this technology is for process optimization and control is for process optimization and control

3.5 Neural Computing in the Oil and Gas Industry

Neural computing is now being applied successfully in the oil and gas industry. Neural computing applications include well log analysis, quality control, demand forecasting and machine health monitoring. So what is neural computing and what advantages can it offer to your industry?

Neural computers can succeed in many areas where conventional computers are unable to operate or can operate with only limited success. Conventional computers require someone to work out a step-by-step solution to the problem. Neural computers, however, are analogous to the human brain and learn from previous examples.

A neural computer can be trained to solve a particular problem by presenting it with a series of examples of problems and the desired solution in each case. Given enough training material, the neural computer is able to learn the underlying principles involved in the solution which it can then use to tackle similar problems. It has the ability to cope well with incomplete data and can deal with previously unspecified or unencountered situations. This contrasts with conventional systems, which, without a full set of data, are often unable to complete their tasks. Application areas for Neural Computing in the Oil and Gas industry.

3.6 Oil Exploration

A major oil exploration company to analyze seismic data and identify first break signals uses neural computing. The identification of first break signals is a laborious, time consuming manual task.

The vast quantities of seismic data involved are cluttered with noise and are highly dependent on the location being investigated. Classical statistical analysis techniques lose their effectiveness when the data is noisy and comes from an environment not previously encountered. Even a small improvement in correctly identifying first break signals could result in a considerable return on investment.

The neural network achieves better than 95% accuracy, easily outperforming existing manual and computer-based methods. As well as being more accurate, the system also achieves an 88% improvement in the time taken to identify first break signals. Considerable cost savings have been made as a result.

3.7 Well Log Analysis

Neural computing can be used to improve the accuracy of strata interpolation of core sample analysis.

Information on the underlying structure of the ground is typically obtained by drilling a series of bore holes, examining each core sample, comparing each core to its neighbours and inferring what structure might lie between the adjacent bores. Recent developments in neural computing, including dynamic warping analysis and section

3.8 Neural Networks in Logging Interpretation

Artificial neural network (ANN) theory has been widely applied in several aspects of logging interpretation in the petroleum industry, such as lithofacies identification, logging parameter prediction as well as other model recognition problems. Among several neural network models and algorithms applied in petroleum industry, the feed-forward neural network model and back-propagation algorithm (BPA) are most widely used for their easy implementation and high robustness. Although the BPA shows strong capability on solving model recognition, it has several problems, such as local optimum and low rate of convergence, when dealing with parameter prediction.

After comparing several kinds of neural network models and algorithms, the self-organization neural network model and Kohonen optimal algorithm are chosen to solve the lightfaces identification problem in the CaiNan oilfield in China [14]. The feed-forward neural network model and Levenberg-Marquardt [15] optimal algorithm are selected to predict logging parameters, such as porosity, permeability and water

saturation, in the same oilfield. Based on Mat lab tools, an artificial neural network logging interpretation program is compiled and applied to the logging evaluation of this oilfield. Simulation results show that the relative prediction error of porosity, permeability and water saturation can respectively reach 6%, 30% and 30%, which is higher than normal methods for parameter prediction. The results prove that the self-organization neural network model and Kohonen optimal algorithm are suitable for lithofacies identification and that the feed-forward neural network model and the optimal algorithm are suitable for logging parameter prediction in the oilfield

3.9 Ford Neural Chip

A new computer chip that mimics how the human mind works is making its way from the space program to American industry and may end up in millions of American cars in years to come.

Computer scientists at NASA's Jet Propulsion Laboratory [16] have made advanced neural network technology breakthroughs that can solve diagnostic problems in industries from automobiles and aerospace to manufacturing and electricity production.

JPL and the Ford Motor Company have signed a licensing agreement for use of an advanced neural network technology to diagnose misfiring under the hoods of Ford automobiles, among its many potential applications. With the advent of this new chip, vehicles should show a reduction in emission levels.

The smart fit between JPL's neural net hardware and Ford's automotive engineering algorithm expertise will enhance the industrial giant's ability to meet ever-stricter Clean Air Act requirements as they apply to continuous onboard diagnostics and control, officials said.

In addition, the chip is designed to improve fuel economy, resulting in financial savings for car owners. Ford engineers do not predict a price increase for installation of the chip because JPL designed a computationally powerful neuroprocessor that could be mass-produced in a highly cost-effective way. The technology also improves customer satisfaction by virtually eliminating distracting false alarms about misfiring that vehicle dashboards can signal with current under-the-hood diagnostic technology.

JPL and Ford scientists say the chip represents the first significant change in the way computing is done on vehicles since computers were first introduced into automobiles in the 1970s[17].

"Neural networks are a new discipline, and diagnostics, prognostics and control is a huge field. Ford's application is but the tip of the iceberg of this chip's potential use in American industry as a whole," said Tom Hamilton, program manager at JPL's Dual-Use Technology Office, one of JPL's many technology transfer arms. "JPL is proud to be able to make this revolutionary technology available for U.S. business."

The new license provides Ford with rights to intellectual property of the chip for auto industry applications, while JPL, which has applied for patents to the technology, retains general rights. JPL is managed by the California Institute of Technology, which serves as the party of record for this license.

Neural systems were inspired by the architecture of nervous systems of animals, which use neurons, a form of parallel processing elements, to process large volumes of information simultaneously. In vehicle applications, artificial neural networks will "learn" both how to diagnose problems like engine misfires and control the engine to optimize fuel economy and emissions.

What JPL has brought to the table is expertise in designing and building what are known as neural network application specific integrated circuits, said Dr. Raoul Tawel [18], who led the development at JPL for the chip. "With Ford, we are implementing highly complex neural network software code in dedicated hardware logic. This brings about a tremendous boost in computational ability compared to traditional software-based approaches, enabling real-time onboard diagnostics for the first time."

For misfire diagnostics, it is necessary to observe and diagnose every engine firing event, estimated at over one billion in the life of each car.

In addition, the diagnostic error rate has to be extremely small, less than one in a million, in order to avoid sending false alarm signals to the driver. The new chip will accomplish that task by "learning" diagnostic tasks during the vehicle development process, bypassing the need to develop conventional software that, in any event, can neither perform these tasks as well nor be implemented in large production volumes with standard microprocessors. The neural network chip, designed to carry out parallel neuron computations efficiently, overcomes the computational barriers that prevent this technology from being exploited today.

3.10 A Neural Network approach to the Tetris game

3.10.1 A brief Description of the Tetris Game

Tetris is a videogame of logic and skill. The game aim is to place in the right way the geometrical pieces falling from the top of the game field, composed of all the possible combinations of 4 basic blocks (see figure 3.2.).

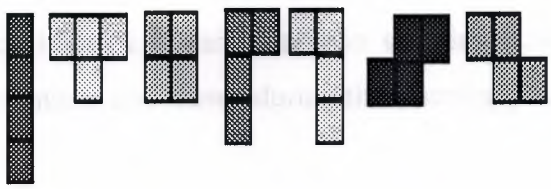

Figure 3.2. A Tetris game basic block

A whole line is erased every time the disposition of the pieces completes a horizontal line. This is the only way to erase pieces from the game pad, so it is worth to complete a line every time is possible. The game is over when the pieces fill completely all the game area.

The pieces can only be manipulated while falling, shifting and turning them, but when apiece has landed on other piece or on the bottom of the screen, it cannot be moved further.

3.10.2 Problem definition

The actions the Tetris player performs can be summarized in these 4 points

i. Identify which of the 7 pieces is falling down;
ii. Analyze the disposition of the previously fallen pieces;
iii. Choose the best (in some cases the least bad) place where to place the falling piece
iv. Shift and rotate the piece in order to perform the desired piece positioning

In some cases the best solution cannot exist (it is not physically possible to perform a perfect joint) and in other cases there can be multiple optimal solutions.

Therefore Tetris is a game where the player not only has to look for the perfect geometrical joint, but he/she has also to limit the damages.

In short, Tetris is a typical problem of real time 'decision making', a problem where you must react fast to a situation that can be very complex, or trivial, but in general consists of a very wide variety of possible scenarios.

A further problem in the Tetris game is that the decision previously taken affect the complexity of future decisions, thus it is very important to take always the best decision, or at least the 'minimum damage' one.

In our opinion the Tetris game can be very effective in showing Neural Network capabilities in taking decisions and particularly in showing how they can handle with good reliability even situations not shown during the learning phase.

3.10.3 Neural Networks and the Tetris game

We used Multi Layer Perceptron as the neural network model for the development of 'NeuroTetris' application.

The neural networks are only a component of a more general application that, using API (Application Program Interface) primitives for Windows, identifies the falling piece and the disposition of the game pad.

It is important to specify that we did not manipulate in any way the original Microsoft Tetris. A single neural network has been trained for each Tetris piece, except for the pieces that are mirror result of another one (yellow-violet green-blue couples, see figure 3.3.); therefore 5 networks are used in NeuroTetris.

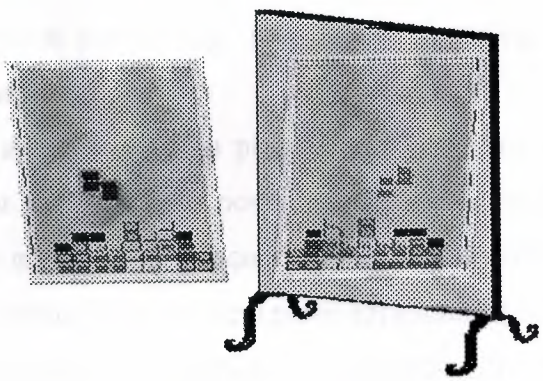

Figure 3.3. Mirror Result

After the identification of the falling piece, NeuroTetris activates the neural network associate with that particular piece: input neurons are fed with the pieces disposition on the game pad and one of the possible rotations; as output the network gives an evaluation for every possible combination of positions and rotations.

The combination with the best score is chosen and the piece is shifted and rotated accordingly. About one half of all the possible cases have been used during the phase of neural networks training.

The five networks are different in respect of both the training sets and the layers dimension.

In fact, input layer dimension depends on the number of possible rotations for the piece (cyan piece has no rotation; red, blue and green ones have 2 rotations; yellow, white and violet ones have 4 rotations) and on the dimensions of the piece.

Still the hidden layers differ from one network to another according to the hardness of the function implicitly hidden in the examples. Obviously the simpler the function the lower the number of neurons in the hidden layer.

3.10.4 INPUT

Window of the dimension of the piece (2, 3 or 4 neurons) where the pieces disposition is shown, obviously the window floates on the whole game pad. All possible rotations (0,2 or 4 neurons).

3.10.5 OUTPUT

Evaluation of the piece positioning. It is worth spending a word about the response speed of neural networks.

A window as wide as the maximum piece extent is shifted along the 10 Tetris game pad columns, and for every window position, all possible rotations are evaluated. We obtain thus that from a minimum of 9 (square piece) to a maximum of 32 ('T' and 'L' pieces) evaluations are performed at every new piece appearance (See figure 3.4.).

In spite of this high number of evaluations, while looking NeuroTetris playing, the bystander practically does not see any slackening in the game.

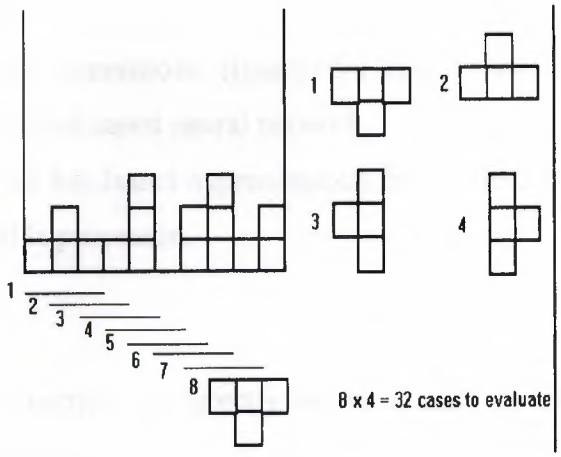

Fig. 3.4. Evaluation to be performed on the "T shaped" piece

We can observe that in this case the neural network has 3 input neurons used for the definition of the game pad piece disposition, 4 binary input neurons indicating the piece rotation and 1 output neuron for the evaluation of the piece positioning. (Remember that for every one of the 8 possible windows shifting, all the 4 possible rotations are evaluated).

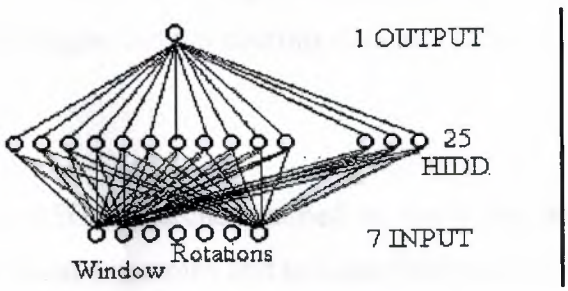

Fig.3.5. - The neural network for the "T" shaped piece (white)

3.10.6 Training phase

The hidden layer dimensions (figure3.5.) and epochs of training have been different on every piece-dedicated neural network.

The training phase has lasted approximately from 1 to 2 hours of computation on an Intel Pentium 60 MHz processor.

3.10.7 Test phase

Tests have been carried out directly on the quality of the games played by the networks on the Tetris game.

After a couple of weeks of neural network tuning, we reached the very encouraging record of 597 lines (remember that a very expert and skilled human player can reach up to 100 - 110 lines). The average game (on a test set of 200 plays) has been of 91 lines per game.

Unfortunately it is not possible to keep statistics directly on the score because Tetris' designers assumed that the limit of 32767 (2 bytes, the integer) could not be topped. Neurotetris often overshoots this score counter limit and makes it assume negative values.

A very interesting detail lays in the fact that neural networks with few hidden neurons (such as 4, 5) don't achieve good learning (10-15% of the examples are not perfectly learned) but perform a good game playing showing a very good skill. Best results are reached with higher hidden neurons number (such as 8, 25).

3.10.8 Conclusions

With this study ASG researchers aimed to show the feasibility of the use of artificial neural networks in a complex and delicate field such as decision-making.

Results have been encouraging and further developments in this area are planned for the immediate future.

The more immediate application area in the real world could be either automatic process control or the automatic management of financial goods [19].

3.11 Process Optimization in Cement Industry

Near Maastricht at the border between the Netherlands and Germany [20], ENCI produces more than three million tons of cement per year. In cooperation ENCI have inspected the process behavior of a ball mill, which is the main component of the manufacturing process. Therefore ball mills are used for grinding of clinker, gypsum and limestone. Ball mills consist of a horizontal cylindrical vessel in which the granular raw compounds are ground by rotation with mixed sized steel balls. Cement grinding is a process performed continuously. Material leaving the mill is carried to a separator there being partially fed back into the mill. Material that meets the required degree of fineness is transported to the storage silo. Air flows through a pulverizing chamber for the dissipation of heat imposed by raw materials and grinding friction. In addition, heat dissipation can be improved by water injection. The properties of the cement, such as its setting time and strength, are adjusted by the addition of gypsum and by grinding to specific degrees of fineness. Influence factors affecting the grinding process are: separator rotation, separator ventilation and water injection in the pulverizing chamber.

During grinding the quantities and composition of the material flows have to be controlled and regulated carefully. It is the goal of process optimization to increase the transparency of process states and thereby the process control by transforming the degree of cement fineness (Blaine-Fineness). This would be as a relevant value of quality of the process and regulation measurements.

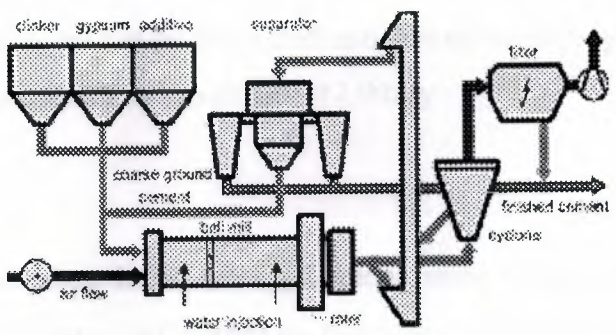

Figure 3.6. Process Optimization in Cement Making

3.11.1 Solution

In the first step, the process model is suitable for quality prediction depending on a given grinding process setting. This model is based on neural networks. In a second step important information can be derived about how manipulated variables have to be set in order to produce cement of a desired degree of fineness. Finally, in step three a genetic algorithm provides the optimal set point with respect to cost minimization for a given degree of fineness.

3.11.2 Benefits

The neural network supports the operator of the grinding plant in the following aspects:

- Decision support in process control for optimal cost operation of the cement mill and the separator under regular conditions.
- Prediction of processing results for verification of human operating decisions.
- The process knowledge is repeatably available at any time.
- Early process fault detection and safe process operation.
- Reduced variation in cement production output.

3.11.3 Software Tools

The models for process optimization were realized with the software tool Data Engine. This software for intelligent data analysis combines conventional statistical methods and approaches for signal processing with intelligent technologies, for example fuzzy logic and neural networks.

The developed model can be integrated into process control systems by the library Data Engine ADL - Application Development Library.

3.12 Summary

In this chapter some Industrial applications using neural networks have been described. Neural Computing in Oil and Gas Industry, Neural network in pulp and Paper industry, Neural network in Tetris Game and The Modeling of Plate Rolling Process using Neural Networks are examples of industrial applications.

CHAPTER FOUR
NEURAL NETWORKS IN THE MODELING OF PLATE ROLLING PROCESS

4.1 Overview

This chapter presents the application of neural network in the modeling of plate rolling process.The cases that neural network were applied to real life in steelwork that the still do not have hot rolling models developed. And in the Thermal profile of slaps a neural network model was developed to forecast the inner temperture of the slabs being reheated as a function of reheating time and superficial temperatures. One of the disadvantages of neural networks is the complexity of hardware and software necessary to enable industrial application. For example, if process conditions change from those to used when training the neural network, data must once again be collected, analyzed and used for retraning the system.

4.2 Plate Mill Applications

There are countless examples of neural network applications in the metallurgical field. Some cases regarding the hot rolling of steel are:

- The modeling of a steel's hot strength of steel based on temperature, strain, strain rate, and effect of chemical composition.
- The sizing of slabs for plate rolling.
- The determination of TTT diagrams from the chemical composition of a steel.
- Pass-schedule calculation for hot-strip mills.
- The feasibility of producing particular grades of at a particular steelworks.

In order evaluate the performance of neural networks under actual plant conditions at **Companhia Siderúrgica Paulista** (COSIPA) [21] industrial rolling mills, it was decided to utilize the technique for the off-line modeling of several plate mill processes. In some cases, these processes were already modeled using statistical techniques, permitting a comparison between both approaches.

All neural networks developed were of the Rummelhart type, with one hidden layer, and trained by the retropropagation method. The networks included a bias neuron in the input layer to improve the modeling capacity of the neuron network.

In each case, 80% of the global raw data was reserved for training the network; the remaining 20% was periodically used during the precision evolution check of the neural network. The training step of most of the neural networks studied in this work converged in 60,000 iterations. Evaluation of the neural network was performed by calculating Pearson's correlation coefficient (r) and the standard error of estimate (SEE); dispersion plots were also used. The software used for developing and training the neural networks was NeuralWorks.

4.3 Thermal Profile of Slabs in the Reheating Furnace

Thermal profiles of slabs being reheated are periodically collected at COSIPA's plate mill. These profiles are measured with an instrumented slab having drilled holes at several locations and depths. Chromel-alumel thermocouples are inserted into these holes and connected to a data logger, which measure temperature evolution during slab reheating. The data logger is sheltered in a nonoxidizing steel box coated with rock wool and filled with water and ice.

A neural network model was developed to forecast the inner temperature of the slabs being reheated as a function of reheating time and superficial temperatures. This is a case with a relatively easy mathematical solution, and, thus, allowed a ready comparison between the performance of the neural network and the conventional numerical models. After several configuration trials, a neural network with three layers was designed. The input layer comprised three neurons: reheating time (min.), slab upper surface temperature (°C), and slab lower surface temperature (°C). The hidden layer contained 13 neurons, and the output layer consisted of ten neurons, each of which representing a point in the instrumented slab where the temperature was measured.

Varying the number of neurons in the hidden layer, as well the use of more than one hidden layer, did not improve the performance of the neural network, which had its best performance in forecasting the temperature in the mid-thickness of the slab ($r = \sim 0.997$; SEE = 26.5°C). The worst performance occurred near the lower surface of the slab ($r = \sim 0.993$; SEE = 36.6°C).The figure below shows dispersion plots of the real and

calculated temperatures for both areas considered. The neural network's performance was considered adequate, as it was similar to previous developed mathematical models, which showed errors of ~30°C.

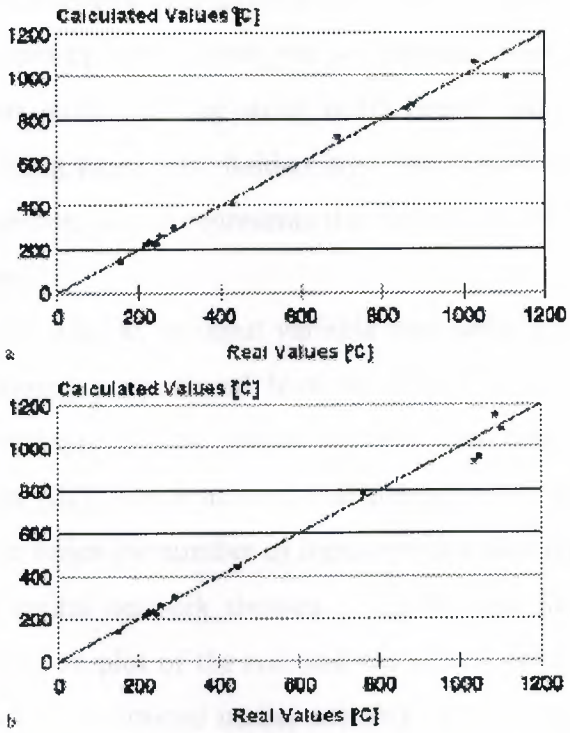

Figure 4.1. Precision achieved by the neural network in modeling thermal profiles of a steel slab during reheating: (a) mid-thickness temperature; (b) near- the-lower-surface temperature.

4.4 Detecting Turn-Up During Plate Rolling

The turn-up, or excessive bowing upwards, of rolled stock during plate rolling is a serious problem, as material being rolled can collide with the rolling stand or ancillary equipment, causing extensive damage. This problem was common at COSIPA's plate mill, especially during the processing of nickel steels.

Previous work showed that alterations in the pass schedule could minimize the occurrence of turn-up; this work led to the development of a statistical model for calculating an optimized pass schedule. There was a critical range of strain values to be avoided during plate rolling. However, this statistical model sometimes calculated unfeasible values of roll gaps. In some cases, the calculated values were excessively

low, jeopardizing productivity; on other occasions, they were excessively high well above the mill's capacity of load, torque, and power.

When neural networks became available, a natural application was to use them to address the problem of turn-up since the statistical model was unsatisfactory. After several trials, a neural network was developed consisting of an input layer with five neurons (i.e., desired turn-up index; work roll peripherical speed [rpm]; rolling load [t], calculated by the Sims model; rolling stock width [mm]; and roll-gap distance [mm] used in the former rolling pass). The hidden layer had eleven neurons, and the output layer contained one neuron, which represents the recommended roll gap distance (mm) for the next rolling pass.

The turn-up index used as an input variable was defined using an arbitrary scale from 0 to 5, representing porportional level of defect seriousness. The number of neurons in the hidden layer of this neural network was calculated after the Hecht-Kolmogorov's theorem [22], which affirms that the optimum number of neurons of a hidden layer is equal to twice the number of input neurons plus one.

The developed neural network showed r = 0.992 and SEE = ~3.0 mm. Figure below shows the dispersion plot of the real and calculated values. The most influential variables, as indicated by the trained neural network, are turn-up index and initial roll-gap distance, followed by (in a decreasing order of importance) rolling stock width, rolling load, and work-roll peripherical speed.

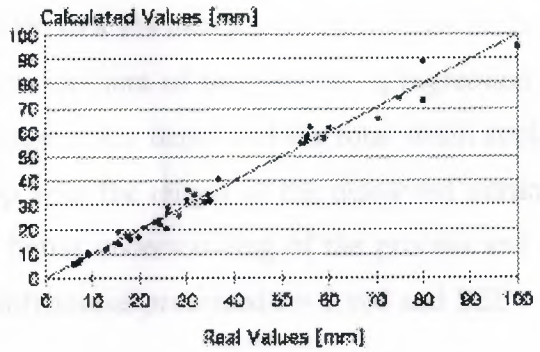

Figure 4.2 Dispersion plots of the neural-network-calculated and real values for controlling turn-up defects.

All calculated values are very near to the real values. That is, the neural network did not generate nonsense values as did the previously developed regression polynomial. Previous work about the turn-up occurrence at COSIPA's plate mill had revealed that this defect was more frequent in a specific range of roll-gap values: 60-80 mm. This is confirmed by the trained neural networks, as the initial roll-gap value is one of the most influencing variables of the model. In addition, the SEE is admissible, as it is equal to only 7.5% of the minimum roll-gap value. However, this is valid since the final thickness of plate is not between 40-80 mm.

4.5 Longitudinal Discard Calculation when Using Plane View Control

Plate rolled from continuously cast slabs generally presents longitudinal extremities with a tongue shape. This lowers the rectangular index of the rolled stock, which affects its metallic yield, as irregular portions in the extremities of the plate have to be cut. This characteristic can be attributed to the peculiar thickness profile of the continuously cast slabs. These slabs are slightly thicker in mid-width, resulting in a non-homogeneous mass distribution along the rolling stock, which affects the shape of its longitudinal extremities.

One potential solution to this problem is to apply a special thickness profile in the rolling stock during the application of the last pass of the broadsizing step. After this special pass, the width of the rolling stock presents a thickness profile that basically consists of a "V"-shape notch or the shape of a dog bone. The development of this process at COSIPA led to a 7% increase in the metallic yield of the plate mill.

During the development of this process, a regression polynomial was created to correlate the "V"-shape notch depth and the total strain applied to the rolled stock after the broadsizing step with the length of the discarded portion of the final rolled stock, thereby enabling a better understanding of the process and a check of its optimization possibilities. This polynomial presented r = 0.903 and SEE = 132 mm.

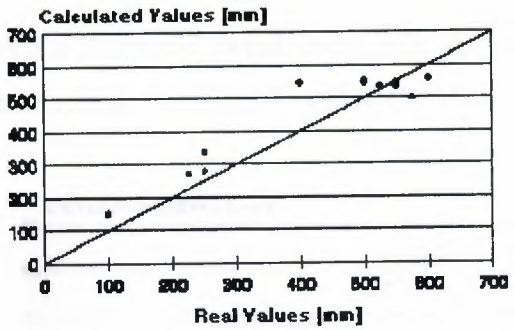

Figure 4.3 Dispersion plot of the neural-network-calculated and real values for estimating the length of the discarded portion in plates submitted to plane-view control during rolling.

Once more, this appeared to be a good application for a neural network, as it could be an opportunity to improve the forecasting of the length of the discarded portion. The best neural network designed to substitute the polynomial equation had an input layer with two neurons ("V"-shape notch depth [mm] and total strain applied to the rolled stock after broadsizing step [%]); a hidden layer with five neurons (according to the already mentioned Hecht-Kolmogorov's theorem); and an output layer with one neuron (representing the length of the discarded portion in the final rolled stock [mm]). The neural network showed r = 0.943 and SEE = 61 mm. The dispersion plot of real and calculated values for the length of the discarded portion in the final rolled stock can be seen in previous figure.

Although the neural network showed better performance than the regression polynomial (the SEE fell approximately 54%), errors observed (figure 4.3) are still significant. Perhaps the cause of these relatively high errors stems from the use of the discard length as an evaluation parameter of the metallic yield of the process. This parameter is less representative than the weight or area of the discarded portion but, in compensation, is an easier variable to measure under industrial conditions.

4.6 Pass Schedule calculation

One of the most stringent quality parameters of plate is its flatness index. The traditional approach to flatness control involves controling the rate crown variation: the thickness variation within a restricted range, especially during the last three passes of the rolling schedule. This fact has been confirmed at COSIPA's plate mill.

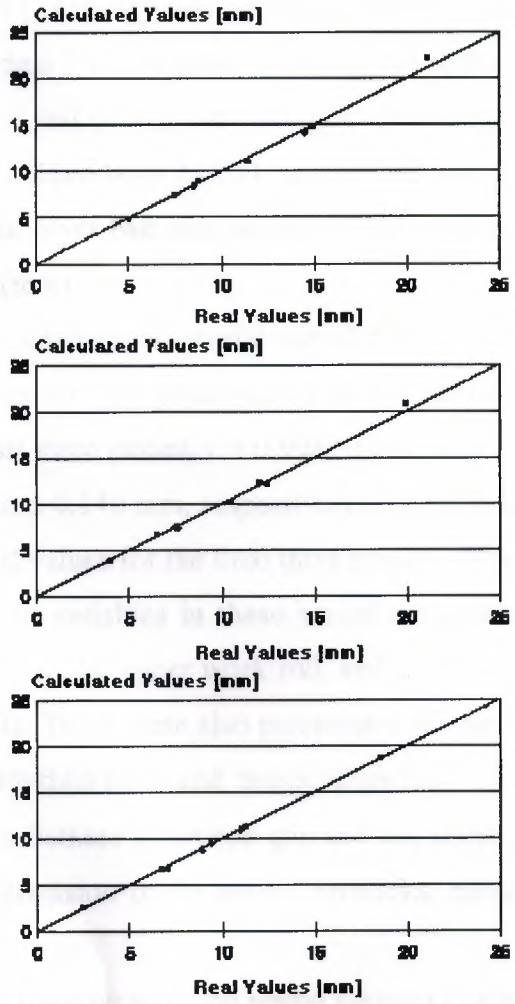

Figure 4.4 Dispersion plots for the pass schedule on optimizing plate flatness from the neural-network-calculated and real values of the roll gap for (a) the third to last, (b) of next to last, and (c) the last plate rolling passes.

Mathematical models for calculating pass schedules are relatively easy to develop, but the use of neural networks is simpler. The three last passes of the rolling schedule were modeled regarding optimization of plate flatness; one neural network was attributed for each pass. The three neural networks showed the same configuration. The input layer consisted of ten neurons (thickness of final plate [mm], width of final plate [mm], aimed flatness index in final plate, flatness index observed after prior pass, roll gap of prior pass [mm], rolling load measured during prior pass [t], temperature measured during prior pass [°C], original crown of the upper work roll [mm], original crown of the lower work roll [mm], and rolling stock tonnage since the last change of the work rolls [t]). The hidden layer had 21 neurons (according to Hecht-Kolmogorov's theorem), and the output layer had one neuron, which represents the roll-gap value of the corresponding pass (mm).

The flatness index used in this model varied from 0 to 5 (the higher the number, the worse the plate flatness). The performance of the neural network was very good. Corresponding to the last three passes, r = 0.998, 0.998, and 0.999, respectively; SEE = 0.430 mm, 0.394 mm, and 0.140 mm, respectively. Figure 6 shows the dispersion plots of the real and calculated values for the final three passes of the rolling schedule.

The most important variables in these neural networks were the aimed flatness index, the original crown in the upper work roll, and the rolling-stock tonnage after the last change of work rolls. There were also parameters of intermediate importance, such as the final plate width/thickness and temperature/load of the prior pass. Finally, variables like prior pass flatness index/roll gap did not show great influence, but were vital to improving the precision of the neural networks, making its use feasible under industrial conditions.

In fact, during the learning step, the neural networks identified the most important variables related to flatness that are traditionally defined by the rolling theory original crown of work rolls and rolling-stock tonnage since a change of work rolls. This last variable generally shows good correlation with thermal crown and wear of work rolls factors that affect the resultant roll crown and, consequently, plate flatness. Work roll deflection promoted by rolling load was also considered, as this was included in the input layer of the neural networks.

The last pass is the most important to define the final dimensions of plate, especially its thickness. The errors observed in the results calculated by the respective

neural networks varied from -0.26 mm to +0.17 mm practically within the commercial-plate-thickness tolerance range. The results can be further improved as more precise data acquisition systems become available. Such systems would avoid human error during data collection and improve the precision of the measured parameters.

4.7 Summary

This chapter described the application of neural network in the modeling of plate rolling process.The cases that neural network were applied to real life in steelwork that the still do not have hot rolling models developed. And in the Thermal profile of slaps a neural network model was developed to forecast the inner temperture of the slabs being reheated as a function of reheating time and superficial temperatures. One of the disadvantages of neural networks is the complexity of hardware and software necessary to enable industrial application. For example, if process conditions change from those to used when training the neural network, data must once again be collected, analyzed and used for retraning the system.

CONCLUSION

A neural network has a parallel-distributed architecture with a large number of nodes and connections. Each connection points from one node to another and associated with weights.

The neural network contains a large number of simple neuron like processing elements and a large number of weighted connections between the elements. The weights on the connections encode the knowledge of a network. Through biologically inspired, many of the neural network models developed do not duplicate the operation of the human brain. Some computational principles in these models are not even explicable from biological viewpoints.

Neural networks trying to mimic the structure and function of our nervous system. Many researches believe that al (Artificial Intelligence) and neural networks completely opposite in their approach. Conventional al is based on the symbol system hypothesis. Loosely speaking, a symbol system consists of indivisible entities called symbols, which can form many complex entities, by simple rule. The hypothesis then states such a system is capable of and is necessary for intelligence.

First chapter described the Biological Foundation of Neurocomputing and the development of neurocomputing, how to build a neural network, and some benefits of a neural network.

Second chapter described two classifications of neural network according to first flow of information like recurrent network, feed forward network, and competitive network and second the way of learning like supervised learning and unsupervised learning.

Third chapter described some Industrial Application of neural network like neural computing in Oil and Gas Industry, Neural Network In Tetris Games, Neural network in the pulp and paper industry and the Modeling of plate Rolling Process using Neural network.

Fourth chapter described the application of neural network in the modeling of plate rolling process.The cases that neural network were applied to real life in steelwork that the still do not have hot rolling models developed. And in the Thermal profile of slaps a neural network model was developed to forecast the inner temperture of the slabs being reheated as a function of reheating time and superficial temperatures. One of

the disadvantages of neural networks is the complexity of hardware and software necessary to enable industrial application. For example, if process conditions change from those to used when training the neural network, data must once again be collected, analyzed and used for retracing the system.

The objectives of this project were to show how neural networks are robust and useful in many industrial applications. Neural networks can be used in several ways to model a given process.

REFERENCES

[1] Galatea, "Exploration of model of thinking and consciousness Neural Network named (Helen)".

[2] Frank Rosenblatt (psychologist) "Purposed the Perceptron", 1958.

[3] Rumelhart D.E. "Purposed the error back-propagation learning method", 1986.

[4] Nippon Steel Crop "built a blast furnace operation control support system that makes use of ANN", 1992.

[5] Daiwa Securities Co. & NEC "Apply Neural Network to learn and recognize of stocks price".

[6] Nippon Oil Co. "The first applying of neural networks to facilities diagnosis".

[7] World Wide Web"http://www.fys.uio.no/bor/thesis/node26.htm2".

[8] Hebb's Rule "The oldest known learning low".

[9] Donald Hebb "The Organization of Behaviour", 1949.

[10] Werbos "Originated the backpropagation", 1974.

[11] Le Cun, Parker and Rumelhart "Simultaneously reinvented back propagation", 1985.

[12] Hinton, Williams "Named the backpropagation".

[13] Teuvo Kohonen "Described the original SOM ", 1980.

[14] CaiNan oilfield in China "Using Kohonen optimal algorithm to solve the lightfaces identification problem".

[15] Levenberg, and Marquardt "Predict logging parameter using optimal algorithm".

[16] NASA's Jet Propulsion Laboratory "Made advance NN that can solve diagnostic problem".

[17] World Wide Web "http://www.techtrans.jpl.nasa.gov/tu.htm1".

[18] Raoul Tawel "Who led the development at JPL for the chip".

[19] World Wide Web "http://www.yahoo.com".

[20] ENCI Co. in the border between Netherlands & Germany "Produce more than three million tons of cement per year".

[21] Companhia Siderurgica Paulista " Industrial rolling mills ", Brazil.

[22] EBERHART, R.C. & DOBBINS, R.W, Neural Network PC Tools - A Practical Guide. Academic Press, San Diego, 1990, 414 p.

[23] PORTMANN, N.F. et al. "Application of Neural Networks in Rolling Mill Automation". Iron and Steel Engineer, February 1995, 33-6.