NEAR EAST UNIVERSITY

Faculty of Engineering

Department of Computer Engineering

NEURAL NETWORKS IN INDUSTRIAL APPLICATIONS

Graduation Project
COM – 400

Student: Mazen Eleyan

Supervisor: Assoc. Prof. Dr Adnan Khashman

Nicosia - 2002

ACKNOWLEDGMENT

First, it is my honor to thank my supervisor the Assoc. Prof. Dr. Adnan Khashman for his co-operation and for his advices during my preparation to the graduation project.

As I would like to thank my family for giving me the opportunity to complete my academic study and specially I would like to thank my parents for supporting me and giving me the chance to achieve my goal in life.

Also I would like to thank all of the teachers with no exceptions for being so patients and for what they have taught us, specially Mr. Tayseer Al-Shanablah, ,Assoc. Prof. Dr. Rahib and Miss Besime, and I would like to thank the Assoc. Prof. Dr. Senol Bektas for standing beside us and helping us .

Finally I want to thank all my friends who helped and advised me during my preparation to the graduation project.

ABSTRACT

Neural Networks have been hailed as the greatest technological advance since the transistor. They are so named because their design is based on the neural structure of the brain on which scientists (Neurobiologists) have been doing intensive researches to understand its biological structure and behavior.

The principle behind the Artificial Neural Networks is to simulate and make decision just as the brain does, applying this concept by using both of hardware and software.

The aims of this project are to focus on the benefit of applying N.N. to various fields and also to concentrate on some problems that face or have faced technology of Neural Networks.

As Artificial Neural Networks learn by examples so, Artificial Neural Networks can be trained to solve the most difficult problems in many applications and especially in industry.

Artificial Neural Network has its own mark in many fields beside industry, such as it goes in business, image processing, medicine and many other fields. Thus, neural network have been applied to industry to help the manufacturers to develop their products.

TABLE OF CONTENTS

INTRODUCTION

Neural networks are computational constructs loosely modeled on the structure of the human and animal brain. They are comprised of neurons that are the information processors of a brain, and synapses, which are spaces between neurons that can be thought of as weighted buses that connect these processors.

Although, the neural network contains a large number of simple neuronlike 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.

Chapter one describes some definitions of Neural Networks and it includes a brief history of Neural Networks since the first days till the recent development of this technology.

Moreover, what Neural Networks use for, and where it is more applicable. Also this chapter includes the advantages and disadvantages of Neural Networks.

Chapter two describes the architecture of Neural Networks and how can it be trained, and also it describes the ways that Neural Networks can be trained with, such as Supervised and Unsupervised networks. Classification of Neural Networks will be also illustrated within this chapter.

Chapter three describes some applications of Neural Networks where can it be found in industry. Some of these applications that will be included within this chapter are: Oil and Gas Industry, Papermaking Plant and games.

Chapter four discusses a general application of Neural Networks in industry which is Modelling of Hot Rolling Processes.

CHAPTER ONE

Neural Networks Background

1.1 Overview

Neural networks are computational constructs loosely modeled on the structure of the human and animal brain. They are comprised of neurons that are the information processors of a brain, and synapses, which are spaces between neurons that can be thought of as weighted buses that connect these processors. Neurons in a network are arranged in layers and information flows through a network starting with external stimuli being presented to an input layer. The information continues flowing down the synapses through the neurons in zero or more hidden layers, and eventually ending up as a transformed activation pattern in the output layer. Thus neural networks essentially represents a function that can map a given input vector into a particular output vector based on the weights of the synapses in the network. The power of a neural network comes from the fact that the network through a process of training can learn this input/output mapping. A neural network consists of four main parts:

- Processing units, where each unit has certain activation level at any point in time.
- Weighted interconnections between the various processing units, which determine how the activation of one unit leads to input for another unit.
- An activation rule which acts on the set of input signals at a unit to produce a new output signal, or activation.
- Optionally, a learning rule that specifies how to adjust the weights for a given input/output pair.

1.2 What is the Neural Network?

A neural network is a system composed of many simple processing elements operating in parallel whose function is determined by network structure, connection strengths, and the processing performed at computing elements or nodes.

A neural network is a massively parallel-distributed processor that has a natural propensity for storing experiential knowledge and making it available for use.

A machine that is designed to model the way in which the brain performs a particular task or function of interest, the network is usually importuned using electronic components or simulated in software on digital computers.

1.3 Why Use a Neural Network?

Either humans or other computer techniques can use neural networks, with their remarkable ability to derive meaning from complicated or imprecise data, to extract patterns and detect trends that are too complex to be noticed. A trained neural network can be thought of as an "expert" in the category of information it has been given to analyze.

Other advantages include:

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

1.4 History of Neural Networks

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

Many important advances have been boosted by the use of inexpensive computer emulations. Following an initial period of enthusiasm, the field survived a period of frustration and disrepute. During this period when funding and professional support was minimal, relatively few researchers made important advances. These pioneers were able to develop convincing technology, which surpassed the limitations identified by Minsky and Papert. Minsky and Papert, published a book (in 1969) in which they summed up a general feeling of frustration (against neural networks) among researchers [1], And was thus accepted by most without further analysis. Currently, the neural network field enjoys a resurgence of interest and a corresponding increase in funding.

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

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

Promising & Emerging Technology: Not only was neuroscience influential in the development of neural networks, but psychologists and engineers also contributed to the progress of neural network simulations. Rosenblatt (1958) stirred considerable interest and activity in the field when he designed and developed the *Perceptron*. The Perceptron had three layers with the middle layer known as the association layer. This system could learn to connect or associate a given input to a random output unit. Another system was the ADALINE (*Adaptive Linear Element*), which was developed in 1960 by Widrow and Hoff (of Stanford University) [2].

2. The ADALINE was an analogue electronic device made from simple components. The method used for learning was different to that of the Perceptron, it employed the Least-Mean-Squares (LMS) learning rule.

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

Innovation: Although public interest and available funding were minimal, several researchers continued working to develop neuromorphically based computational methods for problems such as pattern recognition.

During this period several paradigms were generated which modern work continues to enhance. Grossberg's (Steve Grossberg and Gail Carpenter in 1988) influence founded a school of thought, which explores resonating algorithms. They developed the ART (Adaptive Resonance Theory) networks based on biologically plausible models.

4. Anderson and Kohonen developed associative techniques independent of each other. Klopf (A. Henry Klopf) in 1972 developed a basis for learning in artificial neurons based on a biological principle for neuronal learning called *heterostasis*. Werbos (Paul Werbos 1974) developed and used the *back-propagation* learning method, however several years passed before this approach was popularized. Back-propagation nets are probably the most well known and widely applied of the neural networks today [4]. In essence, the back-propagation net. Is a Perceptron with multiple layers, a different threshold function in the artificial neuron, and a more robust and capable learning rule?

5. Amari (A. Shun-Ichi 1967) was involved with theoretical developments: he published a paper, which established a mathematical theory for a learning basis (error-correction method) dealing with adaptive pattern classification. While Fukushima (F. Kunihiko) developed a stepwise trained multilayered neural network for interpretation of handwritten characters. The original network was published in 1975 and was called the *Cognition* [5].

Re-Emergence: Progress during the late 1970s and early 1980s was important to the re-emergence on interest in the neural network field. Several factors influenced this movement. For example, comprehensive books and conferences provided a forum for people in diverse fields with specialized technical languages, and the response to conferences and publications was quite positive. The news media picked up on the increased activity and tutorials helped disseminate the technology.

6. Academic programs appeared and courses were introduced at most major Universities (in US and Europe). Attention is now focused on funding levels throughout Europe, Japan and the US and as this funding becomes available, several new commercial with applications in industry and financial institutions are emerging.

7. Today: Significant progress has been made in the field of neural networks-enough to attract a great deal of attention and fund further research. Advancement

beyond current commercial applications appears to be possible, and research is advancing the field on many fronts. Neurally based chips are emerging and applications to complex problems developing. Clearly, today is a period of transition for neural network technology.

1.5 How Old are Neural Networks?

The idea of the neural networks has been around since the 1940s but only in the late 1980s were they advanced enough to prove useful in many areas such as computer vision, control and speech recognition. After that, interest exploded and neural networks were hailed as the miracle cure to all problems. Neural networks were quickly applied to financial forecasting with more or less success. Hard lessons were learned in those days, that is not enough to just 'throw some data at a neural network and, it will work.

1.6 Why Neural Network Now?

Current technology has run into a lot of bottlenecks-sequential processing, for one. When a computer can handle information only one small piece at a time, there are limits to how fast you can push a lot of information through. Even with many processors working in a parallel, much time is wasted waiting for sequential operation to complete. It's also difficult to write programs that can use parallel operation effectively.

1.7 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 like:

- 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.

1.8 Definition of a Neuron?

A neuron is a building block of a neural network. It is very loosely based on, The brains nerve cell. Neurons will receive inputs via weighted links from other neurons. This input will be processed according to the neuron activation function. Signals are based on to other neurons.

There are three types of neurons within neural networks. Input neurons receive encoded information from the external environment. Output neurons send signals out to external environment in the form of encoded answer to the problem presented in the input. Hidden neurons allow intermediate calculation between inputs and outputs.

1.9 Why is Neural Networks Useful?

Neural networks are unlike artificial intelligence software in that they are trained to learn relationships in the data they have been given. Just like a child learns the difference between a chair, and a table by being shown examples, a neural network learns by being given a training set. Due to its complex, non-linear structure, the neural network can find relationships in data that humans are unable to do.

1.10 What are Neural Networks used for?

Their applications are almost limitless but they fall into several main categories.

Classification

Business

- Credit rating and risk assessment
- Insurance risk evaluation
- Fraud detection
- Insider dealing detection
- Marketing analysis
- Mailshot profiling
- Signature verification

- Inventory control

Engineering

- Machinery defect diagnosis
- Signal processing
- Character recognition
- Process supervision
- Process fault analysis
- Speech recognition
- Machine vision and Image processing
- Speech recognition
- Radar signal classification

Security

- Face recognition
- Speaker verification
- Fingerprint analysis

Medicine

- General diagnosis
- Detection of heart defects

Science

- Recognizing genes
- Botanical classification
- Bacteria identification

1.11 Why Neural Network Do Not Work All the Times?

Neural networks can only learn if the training set consists of good examples. The old saying of 'garbage in garbage out' is doubly true fore neural networks. Great care should be taken to present decorrelated inputs, remove outliers in the data, and use as much prior knowledge to find relevant inputs as possible. Care must also be taken that the training set is representative, a neural network cannot learn from just a few examples.

1.12 Advantages of Neural Networks?

1. Neural networks can be retrained using additional input variables.
2. Once trained, they are very fast.

3. Due to increased accuracy, results in cost saving.
4. They deal with the non-linearity in the world in which we live.
5. They handle noisy or missing data.
6. They create their own relationships amongst information – no equation!
7. They provide general solutions with good predictive accuracy.

1.13 Disadvantages of Neural Networks?

1. No set rules for network selection.
2. Needs expertise in training the network.

1.14 Biological Neural Networks

Models of our own brains and nerve cell motivate neural Networks architectures. Although the knowledge of the brain is limited, we do have much detailed anatomical and physiological information. The basic anatomy of an individual nerve cell (also known as the neuron) is known, and the most important biochemical reactions that govern its activities have been identified.

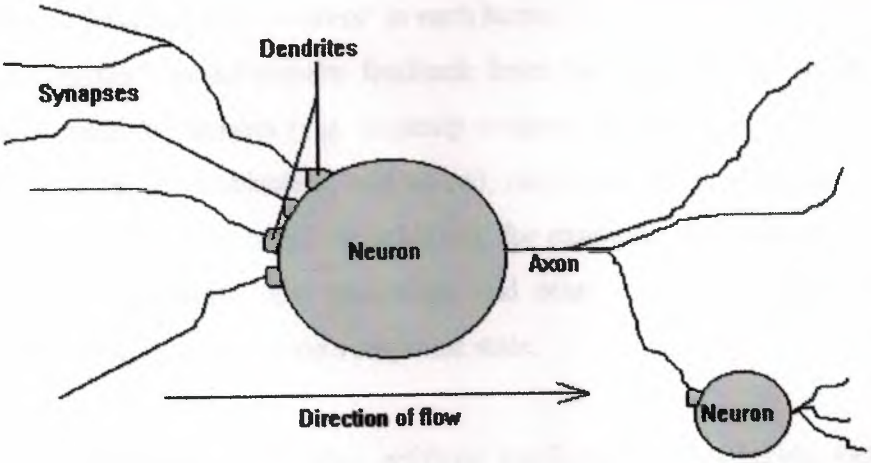


Figure 1.1. A simple neuron cell

The biological brain is an incredibly complex system of more than a 100 billion neurons of different types (not all) highly interconnected with each other via synapses of which there are more than a 150 billion. There is a set of synapses coming into each neuron which communicate with it through it's private dendrites, and each neuron also have an axon out of which it delivers it's messages to other neurons. It is also known that the human brain performs an average of 100 operations per second. Action potentials are fired from each neuron to others (depending on the task the brain is performing), which are electric pulses whose intensity level varies.

1.15 The Future

Because gazing into the future is somewhat like gazing into a crystal ball, so it is better to quote some "predictions". Each prediction rests on some sort of evidence or established trend, which, with extrapolation, clearly takes us into a new realm.

Prediction1:

Neural Networks will fascinate user-specific systems for education, information processing, and entertainment. "Alternative realities", 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 ionization, 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 ionization, and other variables, which the system could learn to correlate with a person's response state.

Prediction2:

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.

Prediction3:

Neural networks will allow us to explore new realms of human capability realms previously available only with extensive training and personal discipline.

So a specific state of consciously induced neurophysiologically observable awareness is necessary in order to facilitate a man machine system interface.

1.16 Summary

A neural network is a massively parallel-distributed processor that has a natural propensity for experiential knowledge and making it available for future use.

A neural network is applied in the situation where other methods cannot run because it uses the principle of the human brains.

Either humans or other computer techniques can use neural networks, with their remarkable ability to derive meaning from complicated or imprecise data, to extract patterns and detect trends that are too complex to be noticed.

In this chapter as mentioned before there are no limits to neural networks applications. They can be used for business, engineering, security, medicine and science.

In chapter two the structure of neural network will be applied.

CHAPTER TWO
STRUCTURE OF NEURAL NETWORKS

2.1 Overview

Neural Networks have been hailed as the greatest technological advance since the transistor. They are so named because their design is based on the neural structure of the brain on which scientists (Neurobiologists) have been doing intensive researches to understand its biological structure and behavior.

The neural network contains a large number of simple neuronlike 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.

2.2 Structure of An Artificial Neuron

The artificial neuron shown in Figure 2.1 is a very simple processing unit. The neuron has a fixed number of inputs n; each input is connected to the neuron by a weighted link w_i. The neuron sums up the net input according to the equation: net = $\sum_{i=1}^{n} x_i w_i$ or expressed as vectors net = x^T w. To calculate the output a activation function f is applied to the net input of the neuron. This function is either a simple threshold function or a continuous non linear function. Two often used activation functions are:

$$f_C(net) = \{11-e^{-net}\}$$

$$f_T(net) = \{\&lcub;\ \text{if } a > \theta \text{ then } 1 \text{ else } 0\ \}$$

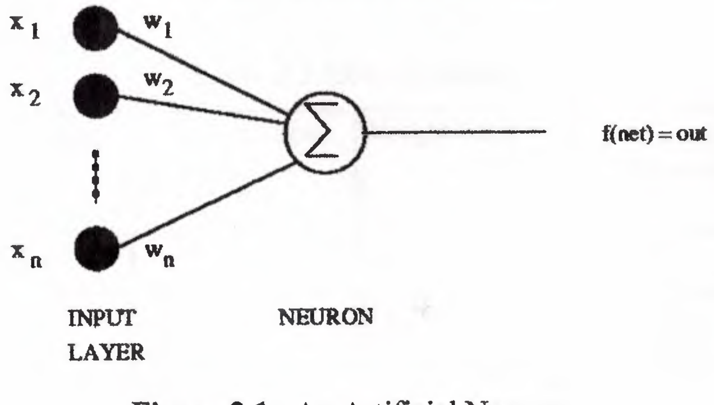


Figure 2.1 An Artificial Neuron.

The artificial neuron is an abstract model of the biological neuron. The strength of a connection is coded in the weight. The intensity of the input signal is modeled by using a real number instead of a temporal summation of spikes. The artificial neuron works in discrete time steps; the inputs are read and processed at one moment in time.

There are many different learning methods possible for a single neuron. Most of the supervised methods are based on the idea of changing the weight in a direction that the difference between the calculated output and the desired output is decreased. Examples of such rules are the Perceptron Learning Rule, the Widrow-Hoff Learning Rule, and the Gradient descent Learning Rule.

The Gradient descent Learning Rule operates on a differentiable activation function. The weight updates are a function of the input vector x, the calculated output f(net), the derivative of the calculated output f'(net), the desired output d, and the learning constant η.

$$net = x^T w$$

$$\Delta w = \eta f'(net)\ (d - f(net))\ x$$

The delta rule changes the weights to minimize the error. The error is defined by the difference between the calculated output and the desired output. The weights are adjusted for one pattern in one learning step. This process is repeated with the aim to find a weight vector that minimizes the error for the entire training set.

A set of weights can only be found if the training set is linearly separable. This limitation is independent of the learning algorithm used; it can be simply derived from the structure of the single neuron.

To illustrate this consider an artificial neuron with two inputs and a threshold activation function f_T; this neuron is intended to learn the XOR-problem (see table2.1). It can easily be shown that there are no real numbers w_1 and w_2 to solve the equations, and hence the neuron can not learn this problem.

Table 2.1 XOR-problem

Input Vector	Desired Output	Weight Equation
0 0	1	$0 w_1 + 0 w_2 > \theta$□$0 > \theta$
1 0	0	$1 w_1 + 0 w_2 < \theta$□$w_1 < \theta$
0 1	0	$0 w_1 + 1 w_2 < \theta$□$w_2 < \theta$
1 1	1	$1 w_1 + 1 w_2 > \theta$□$w_1 + w_2 > \theta$

2.3 How the Human Brain Learns?

Much is still unknown about how the brain trains itself to process information, so theories abound. In the human brain, a typical neuron collects signals from others through a host of fine structures called dendrites. The neuron sends out spikes of electrical activity through a long, thin stand known as an axon, which splits into thousands of branches. At the end of each branch, a structure called a synapse converts the activity from the axon into electrical effects that inhibit or excite activity from the axon into electrical effects that inhibit or excite activity in the connected neurons, as shown in figure 2.2a and figure 2.2b. When a neuron receives excitatory input that is sufficiently large compared with its inhibitory input, it sends a spike of electrical activity down its axon. Learning occurs by changing the effectiveness of the synapses so that the influence of one neuron on another changes.

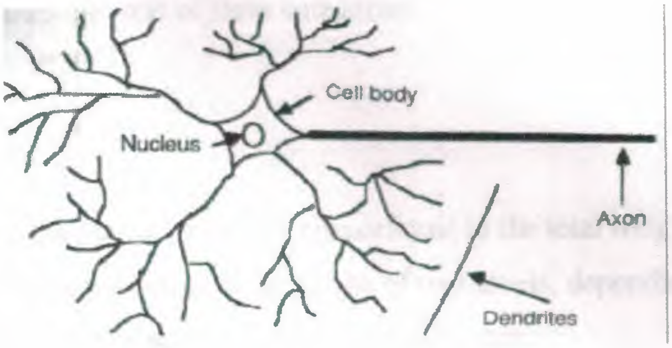


Figure 2.2a Components of a neuron

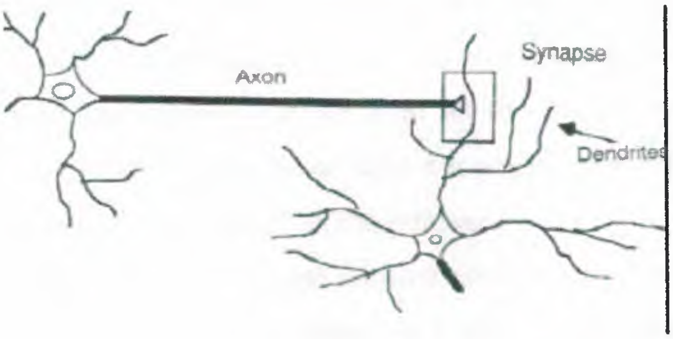


Figure 2.2b the synapse

2.4 How Neural Networks Learn?

Artificial neural networks are typically composed of interconnected "units", which serve as model neurones. The function of the *synapse* is modeled by a modifiable weight, which is associated with each connection. Each unit converts the pattern of incoming activities that it receives into a single outgoing activity that it broadcasts to other units. It performs this conversion in two stages:

1. It multiplies each incoming activity by the weight on the connection and adds together all these weighted inputs to get a quantity called the *total input*.
2. A unit uses an input-output function that transforms the total input into the outgoing activity.

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

- linear
- threshold
- sigmoid

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

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

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

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

The commonest type of artificial neural network consists of three groups, or layers, of units: a layer of "**input**" units is connected to a layer of "**hidden**" units, which is connected to a layer of "**output**" units as shown in figure 2.3.

- The activity of the input units represents the raw information that is fed into the network.
- The activity of each hidden unit is determined by the activities of the input units and the weights on the connections between the input and the hidden units.

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

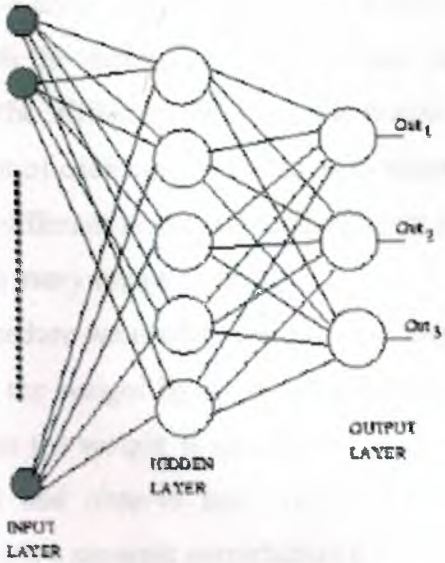


Figure 2.3 Structure of neural networks

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

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

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

2.5 An Example to illustrate the above teaching procedure:

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

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

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

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

Another way to calculate the EW is to use the Back-propagation algorithm which is described below, and has become nowadays one of the most important tools for training neural networks. It was developed independently by two teams, one (Fogelman-Soulie, Gallinari and Le Cun) in France, the other (Rumelhart, Hinton and Williams) in U.S [6].

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 2.2 Definition 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 The learning mechanism

Learning goes as follows : Each example is «shown» to the neural net (i.e. one puts the borrower's descriptive values into its inputs), then these values are «propagated» towards the output as described earlier. The prediction obtained at the network's output(s) is (most probably, especially at the beginning) erroneous. The «error value» is then computed (it is the difference between the expected, «right» value, and the actual output value). This error value is then «backpropagated» by going upwards in the network and modifying the weights proportionally to each one's contribution to the total error value. This mechanism is repeated for each example in the learning set and while performance on the test set improves. This is called «error backpropagation». Let us mention in passing that it is a general method («gradient method») applicable to other objects beside neural nets. For instance, a principal component analysis (PCA) matrix can thus be computed, by successive adjustments.

2.8 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.

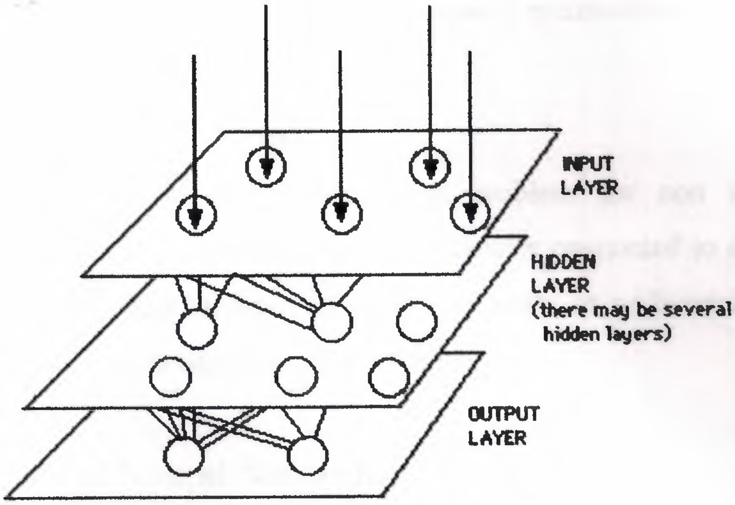


Figure 2.4 Layers of Nueral Networks

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.

2.8.1 A Single Layer Network

A single layer network is a simple structure consisting of m neurons each having n inputs. The system performs a mapping from the n-dimensional input space to the m-dimensional output space. To train the network the same learning algorithms as for a single neuron can be used.

This type of network is widely used for linear separable problems, but like a neuron, single layer network are not capable of classifying non linear separable data sets. One way to tackle this problem is to use a multilayer network architecture.

2.8.2 Multilayer Neural Network

Multilayer networks solve the classification problem for non linear sets by employing *hidden layers*, whose neurons are not directly connected to the output. The additional hidden layers can be interpreted geometrically as additional hyper-planes, which enhance the separation capacity of the network.

2.9 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:

- ***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 *Hop-field Model*

- ***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.***

- ***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 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 work 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.

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).

2.10 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.

2.11 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.

2.11.1 Supervised Learning

This is usually performed with feed forward nets where training patterns are composed of two parts, an input vector and an output vector, associated with the input and output nodes respectively. A training cycle consists of the following steps. An input vector is presented at the inputs together with a set of desired responses, one for each node, at the output layer. A forward pass is done and the errors or discrepancies, between the desired and actual response for each node in the output layer, are found. These are then used to determine weight changes in the net according to the prevailing rule. The term `supervised' originates from the fact that the desired signals on individual output nodes are provided by an external `teacher'. The best-known examples of this technique occur in the back propagation algorithm, the delta rule and Perceptron rule.

Examples of Supervised Learning processes:

- The Perceptron
- The back-propagation algorithm
- The hopfield network
- The Hamming network

Supervised learning divided into two parts:

1) ***Feedback nets:***

A) Back propagation through time

B) Real time recurrent learning

C) Recurrent extended kalman filter

2) Feed forward –only net: -

A) Perceptron

B) Adeline, Madeline

C) Time delay neural network

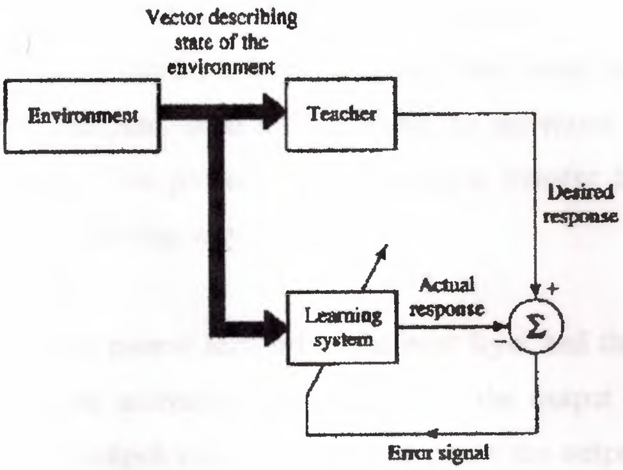


Figure 2.5 Supervised learning

How Supervised Learning Works (Back-Propagation)

This section contains a full mathematical description of how supervised neural networks learn (train). The most frequently used and effective supervised learning algorithm known in the world of neural networks is the "Back-Error Propagation Algorithm" or Back-Prop for short. The type of neural networks this learning algorithm requires is "the feed forward neural networks". It is for this reason they are also known as "back-propagation neural networks. Being a supervised learning algorithm, the back-error propagation relies on a teacher, which is a set of example pairs of patterns. The basic idea of the way this algorithm works is the following.

First a pair from the training data set is chosen randomly. The input pattern of the pair is given to the network at the input layer by assigning each signal of the pattern to one neuron on this layer. Then, the network passes these signals forward to the neurons on the next layer (hidden layer). But, how is this done?

For each neuron on the hidden layer, a Net Input value is computed. By doing the sum over the products of the output of each neuron on the input layer. Which is the original signal itself by the weight of the connection that connects it to the neuron on the hidden layer in question. I.e.,

NetL$_{pi}$ = å O (L-1)$_{pjWij}$

p : is the index of the pair of patterns chosen from the examples set.

NetL$_{pi}$: is the net input of neuron **i** on layer **L** corresponding to pattern p.

O (L-1)$_{pi}$: is the output of neuron **j** on the layer just below **L**. I.e., **(L-1)** corresponding to pattern **p**.

W_{ij}: is the weight of the connection from neuron **j** to neuron **i**.

When all the neurons on this layer have received a Net Input, the next step for each of these neurons is to compute, from it's Net Input, an activation value which is also considered as it's output. This process is done using a transfer function, usually the sigmoid function in the following way:

$$OL_{pi} = 1 / (1 + e^{-Net}L_{pi})$$

Then, these outputs are passed forward to the next layer and the same processes of computing net inputs and activation are done, until the output layer of the neural network is reached. The output values of the neurons on the output layer are taken as one pattern of signals, which is considered as the actual output pattern of the network.

The actual output pattern that the network produces for each input pattern is compared to the target output pattern it should have produced which is simply the second element of the example pair chosen randomly at the beginning of the whole process. An error value is computed using the actual and target patterns as follows:

$$E_p = \mathring{a}\ (O_{pi} - T_{pi})^{2}$$

Where:

E_p: is the error value that corresponds to example pair **p**.

O_{pi}: is the output value of neuron **i** on the output layer of the network.

T_{pi}: is the i'th signal value on the target output pattern of example pair **p**.

If the value of this error is zero, there will be no need to make any changes in the connectivity state. However, if the error value is not zero, some changes are to be made in the weights of the connections in the network reduce this error. The way this is done is as follows.

We should bear in mind that this process, as the title of the algorithm actually states, involves sweeping the error backwards through the network and at each layer (level) the relevant changes are made to the weights of the connections, which we will discuss in the following.

Each weight is either increased by some fraction or decreased. The mathematical formula used by this algorithm is known as the Delta Rule. Which is:

$$D\ _{p}W_{ij} = hd_{Lpi}O_{Lpi}$$

Where:

$D\ _{p}W_{ij:}$ is the amount by which the weight **W_{ij}** should change correspondingly to training pattern pair p

h : is the learning rate

$\mathbf{d_{Lpi}}$: is the error on the output of unit **i** on layer **L** for pattern pair **p**. The computation of it's value depends on the type of the neuron in question.

The way the error at the output of a neuron is computed depends on the type of the neuron. So if it's an output neuron then the error on it is:

$\mathbf{d_{Lpi} = (T_{pi} - O_{Lpi})OL_{pi}(1 - O_{Lpi})}$.

However if it's a hidden neuron then the error value on it is:

$\mathbf{d_{Lpi} = O_{Lpi}(1 - O_{Lpi})Sd(L+1)_{pk}W_{ki}}$.

Where:

$\mathbf{d(L+1)_{pk}}$: is the error value of neuron **k** on the layer just above layer **L**. That is layer **(L+1)**

$\mathbf{W_{ki}}$: is the weight of the connection going from the neuron in question **i** to neuron k on the layer just above.

The learning rate is a value that must be chosen between 0, and 0.9. It determines the size of the step by which the neural network system moves towards an optimal state. The actual idea behind the back-error propagation algorithm is to slide along the error surface performing a gradient descent in search of, ideally, what is known a global minima, i.e. a state of the network, where the error on it's output patterns in optimal (minium).

Figure 2.6 shows a typical example of an error surface of a neural network system on which the state of this system should slide in search of global minima.

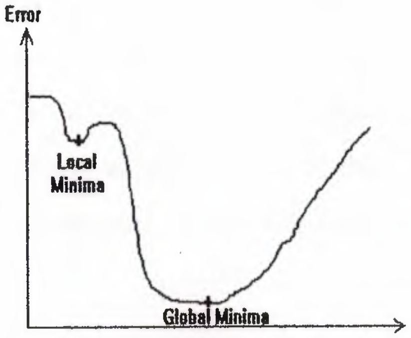


Figure 2.6 Example of Local Minima

The figure above also shows an example of what is known as local minima, which is simply an area on the error surface where the error of the system drops down, but it is not a good solution to the problem.

Choosing a value for the learning rate is very delicate, because, if it's assigned a large value then local minimas can easily be avoided by just jumping over them, but this *might end the system up in oscillation. I.e., jumping forward and backward over global* minima without ever getting there. However if the learning rate is given a small value, then may be global minimas cannot be missed, if there are any around, but the system is more likely to be trapped in a local minima. For this reason actually, a new variable has been introduced, known as the Momentum, whose value should be in the range 0 to 0.9 as well. The momentum times the old correction to the weights is added all the time a new correction is being proceeded. This way, the learning rate value can take a large value and the risk to end up in an oscillating state is minimised.

The final mathematical formula used by the back-error propagation algorithm to update the connection weights in a feed forward neural network is:

$\mathbf{NEWD_pW_{ij} = hd_{Lpi}O_{Lpi} + a\ OLDD_pW_{ij}}$

Where:

$\mathbf{NEWD_pW_{ij}}$: is the new weight correction value of W_{ij} concerning pattern p

$\mathbf{OLDD_pW_{ij}}$: is the old weight correction value of W_{ij} concerning pattern p

a : is the momentum.

This whole process is done for each and every example pair and for many epochs. Once a neural network has been trained to do a certain task, it should then be validated. The process of validation is in other words a process of checking its performance. This is done by providing a set of pairs of input/output patterns which is similar to the training set used to teach the network but different in contents. With this set of data, we give the input patterns to the network and observe the output produced then compare it to the target output. A judgment on the overall performance of the network, whether some more training is required or not, is taken there and then. Once the network is fully trained and validated, it can then be used as a black box system that one may query using it's input and output layers.

2.11.2 Unsupervised Learning

Unsupervised learning is a process when the network is able to discover statistical regularities in its input space, and automatically develops different modes of behavior to represent different classes of inputs (in practical applications some 'labeling' is required after training, since it is not known at the outset, which mode of behavior will be associated with a given input class). Kohonen's self-organizing (topographic) map neural networks use this type of learning.

Examples of Unsupervised Learning processes:

- Kohonen's Self-Organizing maps
- Competitive Learning
- Adaptive Resonance Theory (A.R.T)

Unsupervised divided into two parts:

1) Feedback nets:

A) Discrete hop filed

B) Analog adaptive resonance theory

C) Additive gross berg

2) Feed forward –only nets

A) Learning matrix

B) Linear associative memory

C) Counter propagation

Applications for Unsupervised Nets

Clustering data:

Exactly one of a small number of output units comes on in response to an input.

Reducing the dimensionality of data:

Data with high dimension (a large number of input units) is compressed into a lower dimension (small number of output units). Although learning in these nets can be slow, running the trained net is very fast - even on a computer simulation of a neural net.

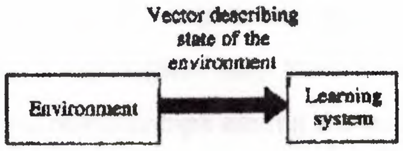


Figure 2.7 Unsupervised learning

Kohonen Self Organising Map (SOM) - Unsupervised learning

Very effective and frequently used un-supervised neural network architecture is the "Kohonen" neural network. These networks have only two layers, a standard input layer and an output layer known as the "Competitive (Kohonen)" layer (the reasons for which it is called so will be discussed later in a following paragraph).

Each input neuron is connected to each and every neuron on the competitive layer which are organized as a two dimensional grid. The picture bellow shows a typical example of. A Kohonen network with 2 inputs and 25 neurons on the competitive layer.

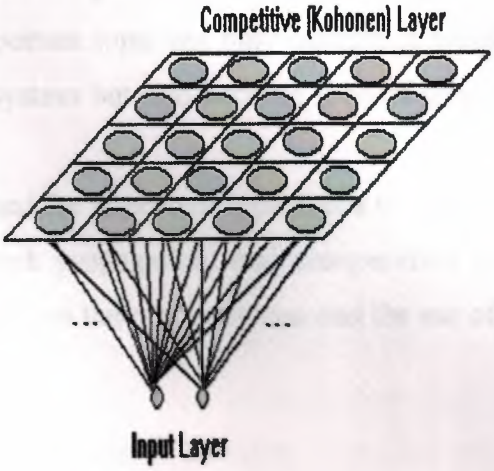


Figure 2.8 Kohonen learning

A Kohonen Self Organizing Grid - 2 Dimensional Output Layer

The input layer in a Kohonen network has the same function as the input layer described in the feed forward networks. However, the neurons on the output layer have a totally different property; they can actually find the organization of relationships among input patterns which are classified by the competitive neurons that they activate. Kohonen networks are known to be self-organizing feature maps (more details will be given). I.e., they can organize a topological map from a random starting point and the resulting map shows the natural relationships among the patterns that are given to them.

Topological mapping of sensory and motor phenomena exist on the surface of the brain. It is important to keep in mind, however, that the brain mechanisms are different from the paradigm described here. The detailed structure of the brain is different, and input patterns are represented differently in biological systems. Furthermore, biological neural systems have a much more complex interconnection topology. However, the basic idea of having a neural network organize a topological map is illustrated effectively with the Kohonen neural networks.

2.12 Summary

Neural Networks have been hailed as the greatest technological advance since the transistor.

The neural network contains a large number of simple neuronlike processing elements and a large number of weighted connections between the elements. The intelligence of a computer based system parallels the amount of knowledge it contains. Machine learning is an important topic not only because it would be an indispensable element in an intelligence system but also because it holds great promise in expediting scientific discovery.

In this chapter we obtained the important approaches to network learning, supervised learning with works by (back propagation) and unsupervised learning. . All learning seems to place more emphasis on the representation and the use of learning.

CHAPTER THREE
INDUSTRIAL APPLICATIONS
OF NEURAL NETORKS

3.1 Overview

Neural networks are a relatively new artificial intelligence technique that emulates the behavior of biological neural systems in digital software or hardware. These networks can "learn," automatically, complex relationships among data. This feature makes the technique very useful in modeling processes for which mathematical modeling is difficult or impossible. The work described here outlines some examples of the application of industrial neural networks.

Before any deeper analysis of the collected material, some experiences and best practices can already be named. Some of these subjects were included in questionnaires and others appear frequently in the reports and other papers of this field.

In about 20 percent of the industrial development projects using neural networks, a product actually has been developed. The corresponding figure including all product development projects is slightly lower. In addition results are utilised in embedded systems or they are parts of other systems. Also a few prototypes for further development were developed.

3.2 Application grounds

A few fundamental factors in the emergence and successful applications of neural networks and other intelligent methods can be named:

- Research history of the field
- Research history of related topics and thereby global contacts
- Formation of active research groups
- Suitable traditional industry sectors for methods
- New emerging industry sectors suitable for applications.

For example, in the case of Finland that has been used as the example for this draft report, favorable features can be found for each point. These together enabled another fundamental factor: setting up a national technology programme for the field.

3.2.1 Guidelines in applying neural networks

In the eighties, a lot of effort was placed in the research of neural networks. In the nineties this effort was utilised by a rapidly increasing number of applications in different industrial sectors. The first applications in Finland were in pulp and paper industry and in the commerce in the early 90's. Towards the mid 90's neural networks were used in a number of applications like controlling, monitoring, classification, recognition, profiling, optimization and forecasting.

After the launch of the technology programme in 1994 companies got more financial resources, contacts and knowledge. Soon the amount of new neural network applications reached 20 per year. This new stage in applying neural networks was also helped by the economic growth in the country.

The product development projects by the industry sector and by the application type have been analyzed. The target is to find out changes encountered during the last five years of the 90's. Both industry sectors and application types have been divided into three classes to simplify the examination. The industry sectors are: Base industry (pulp and paper, metal, forest, mining, chemical, power, food and building material industries), Information industry (electrical and electronics industry, communications), and Economy (retail, banking, insurance and medical applications, traffic).

The application types are: Control (modeling, control, monitoring, simulation, optimization, forecasting, quality and condition control, diagnostics), Signal processing (signal analysis, image analysis, classification, diagnostics), and Profiling and classification (profiling, classification, clustering, planning, configuration, analysis, forecasting, data mining, knowledge discovery).

The changes have been described with four pie charts: The shares of launched product development projects using neural networks in the years 1994-96 and 1997-1999 by the industry sector and by the application type, figures 3.1 and 3.2.

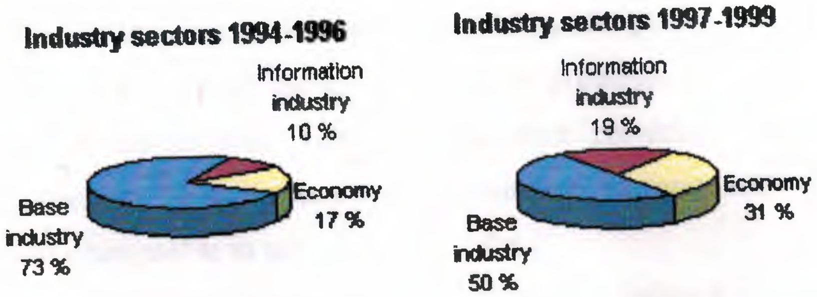


Figure 3.1 The shares of launched product development projects using neural networks in the years 1994-1996 and 1997-1999 by the industry sector.

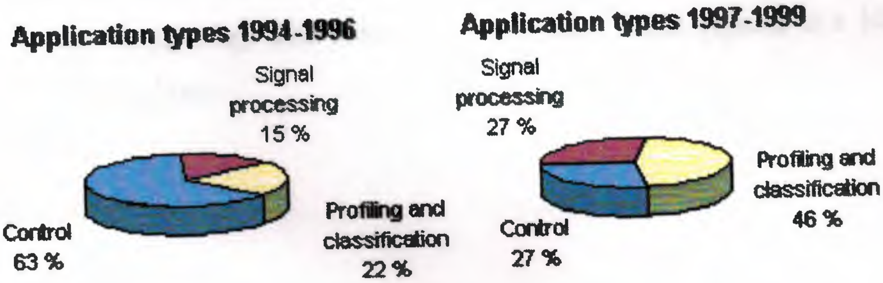


Figure 3.2 The shares of launched product development projects using neural networks in the years 1994-1996 and 1997-1999 by the application type.

By this kind of classification, the shares of information industry and economy seem to rise as well as signal processing together with profiling and classification.

The total number of projects was 67. The boom of applications hits the years 1995-1997 by nearly 20 new applications per year. The average length of a project was 15 months with average budget being EUR 200 000.

3.3 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:

3.3.1 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.

3.4 Process Control

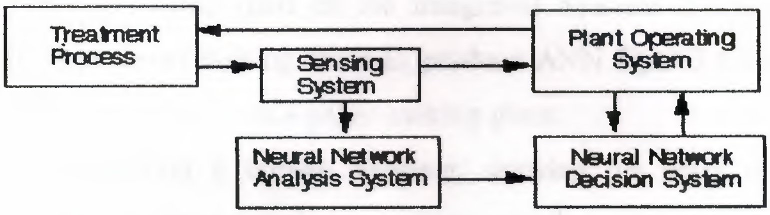

Treatment Process
Sensing System
Plant Operating System
Neural Network Analysis System
Neural Network Decision System

Figure 3.3 Neural Networks process control loop

ANNs have been successfully used in many chemical process control applications. The ANN can closely monitor and control complex chemical processes with a human operator functioning in a supervisory role. ANNs allow continuous, high-level monitoring of all process sensors and can be used as adaptive controllers as shown in figure 3.3. In many systems, performance degrades over time due to deterioration of the system components. To compensate, operational parameters are dynamically adjusted to optimize system performance. An ANN can be used to monitor the process, make decisions about system operation, and adjust the appropriate controls to keep the process operating with optimal efficiency. An advantage ANNs have over more traditional adaptive controllers is that the ANN can be continuously updated with new information by using a dynamic learning approach. The backpropagation algorithm is commonly used to train ANNs in process control with the training data composed of historical data about the process. Several applications are discussed in two recent special issues of the *IEEE Control Systems* magazine devoted to neural networks.

3.5 Neural Network in Papermaking Plant

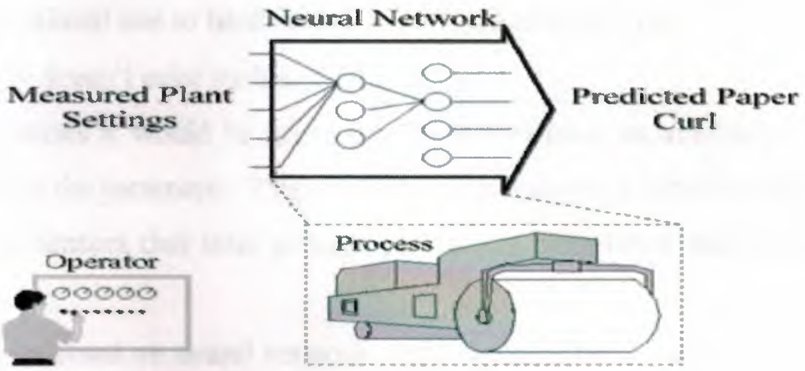


Figure 3.4 Neural Networks in Papermaking Industry

The Neural Network Group (part of the Integrated Systems Group) is working together with two large paper-making firms to produce ANN figure3.4 based software to optimize 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.*

3.5.1 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

3.6 Power Systems and High-Voltage Engineering

The Power Systems and High-Voltage Engineering Group is involved in: Voltage Collapse, Load Modeling, Bifurcations and Chaos in Power Systems, Adaptive Control of HVDC, Modeling and Control of Power Systems, Transient Stability and direct analysis, FACTS Devices Analysis and Control, Pattern Recognition with Neural Networks, Fuzzy-Logic/Neural-Network Control, Induction Motors Interfaced with Photovoltaic Arrays, Computer Simulation of Starting Transients and Performance of AC Drives, Distribution System Planning, Optimal Operation of Radial Distribution Systems, Power System State Estimation, Power System Optimization Techniques, Optimal Load Flow Methods, Reactive Power and Voltage Control, Fault Current Limiters, Digital Fault Location Algorithms; and Precipitators, Heat Transfer of Transformer Oil, Electrohydrodynamic Motion in Non-Polar Liquids, Electroconvection in Insulating Liquids, Vacuum Breakdown, Generation of High Voltages, Industrial Applications of High Voltage, Partial Discharges.

3.6.1 High Voltage and Insulation Engineering

The High Voltage and Insulation Laboratory of the Electrical and Computer Engineering Department at the University of Waterloo is a unique facility in Canadian universities, equipped with sophisticated diagnostic equipment for experimental research.

Research falls chiefly into two categories: (1) Application of high voltages to industrial processes: electric perforator; high voltage food sterilization; air and water pollution control; artificial neural networks for pattern recognition in insulation failure diagnosis; and (2) Study of Insulation Failure Mechanisms: pre-breakdown and breakdown phenomena in solid, liquid, gaseous and vacuum under controlled conditions.

3.6.2 Power Systems Analysis

Nonlinear systems theory, such as bifurcation analysis and Lyapunov functions, is used for the study of voltage and angle stability in combined FACTS-HVDC-AC power networks. Voltage collapse phenomena are analyzed using these techniques, so that efficient, commercial-grade software tools, as well as adequate models of the various network elements and their related controls, can be developed. Bifurcation and chaotic analysis is also used for the simulation and experimental study of stability and modeling issues in aggregated motor loads. All these studies are complemented with transient stability and small signal stability analysis, as well as electromagnetic transient (EMTP) simulations.

3.7 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 [7] 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.

"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 [8], 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.

3.8 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 3.5.); therefore 5 networks are used in NeuroTetris.

Figure 3.5 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.

3.8.1 INPUT

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

3.8.2 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 3.6).

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

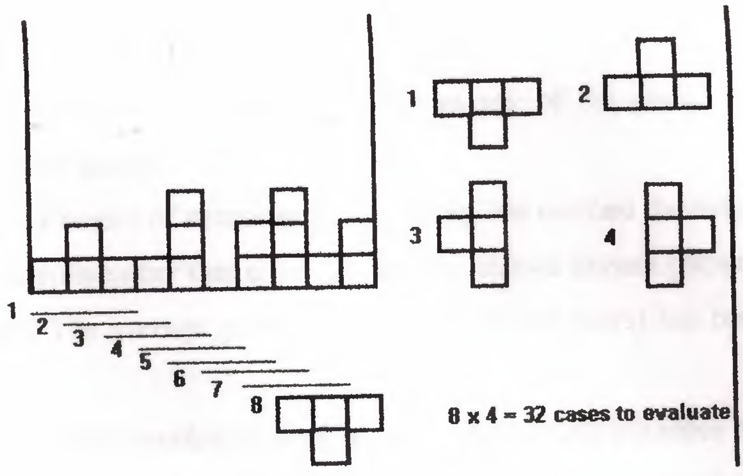


Figure 3.6. Evaluation to be performed on the "T shaped" piece

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).

3.8.3 Training phase

The hidden layer dimensions (figure3.7) 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.

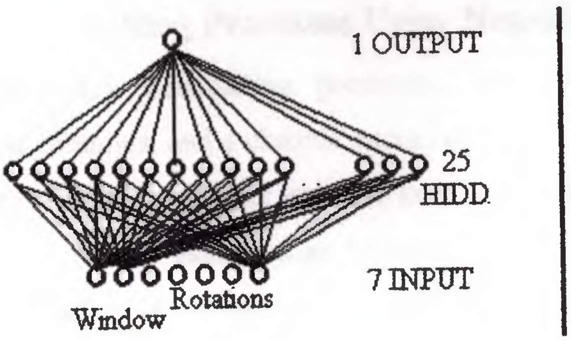


Figure 3.7 The Neural Network for the "T" shaped piece (white)

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).

3.8.5 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.

3.9 The Modeling of Hot Rolling Processes Using Neural Networks

The advent of revolutionary steelmaking processes, new materials like high performance polymers and ceramics, and a chronic excess of capacity production made steel market very competitive. If a steelmaker wants to keep or expand its market share, it must offer products with excellent quality at an affordable price. One of the keys to achieve this goal is the automation of the steelmaking process. In fact, this is one of the major stages of evolution in a steel plant, as it improves both process and product consistency, minimizes costs and make production control easy. All these factors promote a significant increase in the process cost/benefit ratio.

The automation of hot rolling processes requires the development of several mathematical models for the simulation and quantitative description of the industrial operations involved.

The main feature of the neural networks - the establishment of complex relationships between data through a learning process, with no need to previously propose any model to correlate the desired variables - makes this technique very attractive in the modeling of processes where traditional mathematical modeling is difficult or impossible. Besides that, they are almost immune to noise or spurious data. The development of neural network models is relatively quick and, in most cases, simple. Several researchers performed off-line tests on the modeling of hot rolling processes using this technique, frequently getting good results.

3.10 Summary

Neural networks are relatively new artificial intelligence techniques that emulate the behavior of biological neural systems in digital software or hardware.

As we noticed that Artificial Neural Networks has been successful in many fields in real life applications such as medicine, finance, business and industry.

In this chapter which was talking about Neural Networks in industry, several applications of industry have been discussed, like Neural Computing in the Oil and Gas Industry, Neural networks in the pulp and paper industry and The Modeling of Hot Rolling Processes Using Neural Networks which is going to be discussed in more details in the next chapter.

CHAPTER FOUR
THE MODELLING OF HOT ROLLING PROCESSES USING NEURAL NETWORKS

4.1 Overview

This work describes the application of neural networks in the modeling of hot rolling processes. This relatively new technique of Artificial Intelligence was conceived more than fifty years ago, but it only became really feasible with the arrival of low cost computer processing power. The first papers about its utilization in the hot rolling field were published in 1992 [10]. Although the first results were promising, there is still some lack of confidence about its real performance under industrial conditions, which is preventing the exclusive use of this new tool in the modeling of hot rolling processes. However, neural networks are already being used, as a standard feature, in hybrid automation models for hot strip mills, where they calculate adjusting coefficients for theoretical models. However, continuous use of these tools and its continuous development certainly will contribute to increase the general confidence in this revolutionary method and pave the way for a more intensive application in practical cases.

4.2 Sizing Slaps for Plate Rolling

A very first trial on the application of neural networks in the field of hot rolling was developed at Usiminas, a Brazilian steelworks. It was developed a neural network to replace a previous regression equation used for the calculation of the dimensions of the slab to be rolled at the plate mill, aiming minimal metal discard after hot rolling.

Some advantages are inherent to the use of neural networks instead of multiple regression equations [1]. There is no need to select the most important independent variables in the data set, as neural networks can automatically identify them. The synapses associated to irrelevant variables readily show negligible weight values; on its turn, relevant variables present significant synapse weight values. As said previously, there is also no need to propose a function as model, as required in multiple regression. The learning capability of neural networks allows them to "discover" more complex and subtle interactions between the independent variables, contributing to the development of a model with maximum precision. Besides that, neural networks are intrinsically robust, that is, they show more immunity to noise eventually present in real data; this is an important factor in the modeling of industrial processes.

Obviously, the forecasting performance of a slab sizing model will be consistent only if several operational parameters are kept under control: weight and dimensions of the slabs; precision of the pass schedules, including the broadsizing step; distribution of strain in the broadsizing step; plate crown and scale losses during the reheating of the slab. Other factors also must be considered, like the specific characteristics of the process of slab production (continuous cast or from ingots rolled at the slabbing mill), rolling type (normal or controlled) and broadsizing ratio.

This neural network was trained with data measured from 239 rolling stocks. Figure 4.1 shows the improvement in the metallic yield that was got with the use of this neural network. This amelioration can be summarized by the following parameters: increase of 0,5% in the plate-slab programmed yield; increase of 1,0% in the trimming yield and increase of 0,32% in the inspection yield.

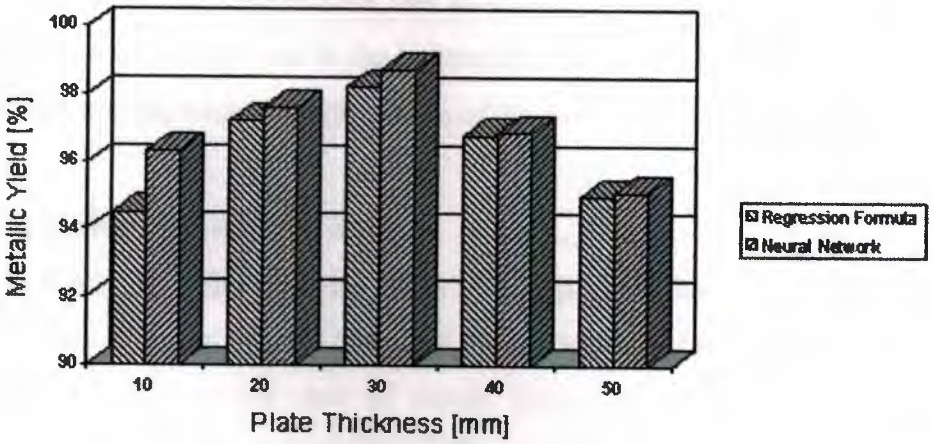


Figure 4.1 Comparison between metallic yield got with the application of the former regression formula and the newly developed neural network.

4.3 Modeling Thermal Profile of Slaps in The Reheating Furnace

At Cosipa, another Brazilian steelmaker, ofiles of slabs being reheated are periodically collected at the plate mill line. These profiles are measured with an instrumented slab, which has drilled holes at several locations and depths. Chromel-alumel thermocouples are inserted into these holes and connected to a data logger, which collects all the temperature evolution of these points during slab reheating. The data logger is sheltered in a stainless steel box coated with rock wool and filled with water and ice.

It was decided to develop a neural network model to forecast the inner temperature of the slabs being reheated as a function of their reheating time and their superficial temperature. This is a case with a relatively easy mathematical solution, and thus adequate to allow a comparison between the performance of the neural network and the conventional numerical models.

With this purpose in mind, a neural network with three layers was developed, after several configuration trials:

- Input layer with three neurons:
  - Reheating time [min]
  - Slab upper surface temperature [°C]
  - Slab lower surface temperature [°C];
- Hidden layer with thirteen neurons;
- Output layer with ten neurons, each of them representing a point in the instrumented slab where the temperature was measured.

The use of more or less neurons in the hidden layer, as well the use of more than one hidden layer, did not improve the performance of the neural network.

4.4 Modeling Hot Strength of Steel

Hot strength can be defined as the stress that begins and keeps the yielding of a material in a uniaxial stress state. It is one of the fundamental properties of a material under high temperature. In the case of metals being rolled, the knowledge of its magnitude is vital for the correct designing of the mechanical and electrical components of the rolling stands, as well in the development of mathematical models and automation algorithms for the hot rolling process.

Hot rolling of steels normally occurs at temperatures corresponding to its austenitic range. Figure 4.2 shows schematically a typical stress versus strain curve of steel at high temperature. As can be seen from this picture, three steps characterize this curve: *strain hardening*, *dynamic recrystallization* and *steady state*. During the initial step of strain hardening stress grows monotonically. As soon as stress reaches its maximum value, the advent of dynamical recrystallization causes a lowering on its magnitude, down to a steady-state value, which is approximately constant or eventually shows a cyclical behavior, which denotes an "equilibrium" between strain hardening and recovery processes.

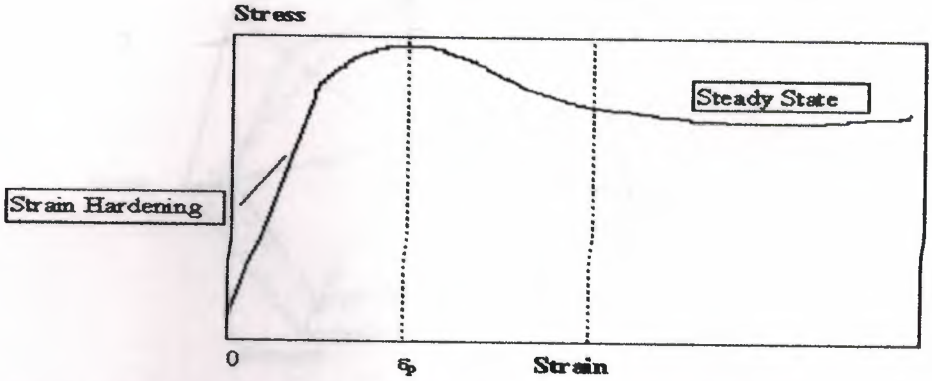


Figure 4.2 Schematic representation of the hot strength versus strain curve of steel in the austenite range.

The chemical composition of the steels can be seen in table 4.1. Data used in the non-linear regression program for the parameter fitting of the empirical equations and during the learning/testing steps of the neural network was got directly from the stress versus strain curves determined in the hot torsion machine, without any pre-processing.

Table 4.1 Chemical compositions of the HSLA steels tested.

Steel	C	Mn	Si	Al	Cr	Ni	Cu	Nb	V	Ti
Nb	0.14	1.02	0.40	0.040	0.55	-	0.24	0.022	-	-
NbV	0.11	1.24	0.28	0.036	-	-	-	0.036	0.042	-
NbTiV	0.10	1.58	0.27	0.025	0.22	0.26	-	0.042	0.054	0.018

The radar-type graphic in figure 4.3 shows that neural networks had a clear better forecasting power than the previous empirical equations: the best empirical model (Samanta) presented a mean standard error of estimate of approximately 1.5 kgf/mm^2, whereas the neural networks showed a value of 1.2 kgf/mm^2. This value was calculated during the testing step of the neural network, after its full training, using a data set not used during its training.

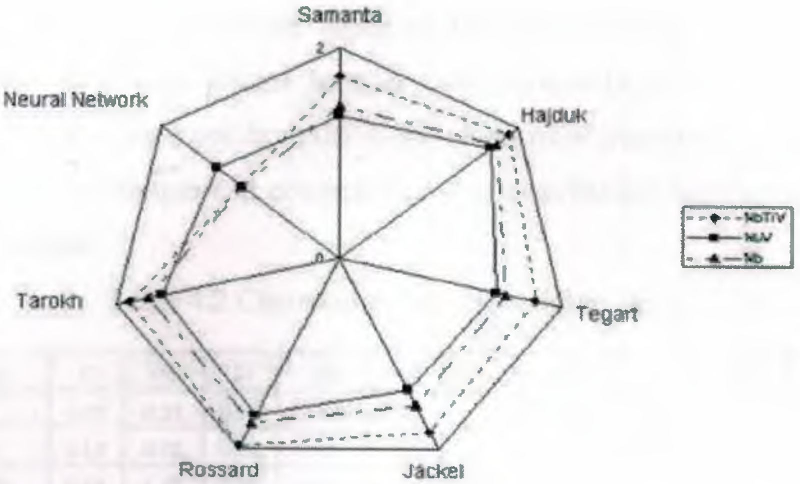


Figure 4.3 Radar-type graphic showing the forecasting power of the several models tested in terms of the standard error of estimate, in kgf/mm^2.

Besides that, it was possible to train these neural networks with data from the full strain range of the curve hot strength versus strain. That is, one single neural network can model the strain hardening, dynamic recrystallization and steady state steps of this curve. The neural networks used in this case have the same topology as the previously described. Its standard error of estimate of calculated during the testing after the training step was approximately 1.7 kgf/mm^2. This kind of modeling can not be performed with the conventional empirical equations cited here. Besides that, the use of neural networks does not require the division of the hot strength curve in distinct steps, a kind of task which is frequently hard. This made the modeling of the full hot strength versus strain curves very easy and quick. The best agreement given by the neural network for the dynamic recrystallization region was also verified in other works for an austenitic stainless steel.

This work was later expanded to a more wide selection of carbon and HSLA steels. The chemical composition of these steels can be seen in table 4.2. The specimens were heated to 1100°C for 15 minutes, and then cooled down to the aimed temperature. Tests were isothermically performed under temperatures of 1100, 1000, 900 and 800°C, and under strain rates of 0.5, 1 and 5 s^{-1}. The same six empirical equations cited before were used for the modeling of hot strength from temperature, strain and strain rate in the strain hardening step of the curve hot strength versus strain. The neural network used in this work was slightly different from the previous work: it has seven neurons distributed in two hidden layers (four in the second layer and three in the third layer). This arrangement showed to be better than the previously used. Data used in the non-linear regression program for the parameter fitting of the empirical equations and during the learning/testing steps of the neural network was previously processed. All the hot strength versus strain curves got from the torsion tests were smoothed using a modified Fourier's transform technique and compensated for the adiabatic heating effect arising from hot deformation.

Table 4.2 Chemical composition of the steels.

AÇO	C	Mn	Si	Al	Cr	Cu	Nb	V	Ti	N
C1	0.09	0.53	0.18	0.029	-	-	-	-	-	0.0047
C2	0.15	0.90	0.21	0.039	-	-	-	-	-	0.0053
CMn	0.16	1.48	0.36	0.039	-	-	-	-	-	0.0048
Nb	0.18	1.34	0.30	0.025	-	-	0.033	-	-	0.0074
NbTi1	0.14	1.11	0.30	0.044	-	-	0.020	-	0.015	0.0054
NbTi2	0.14	1.34	0.23	0.035	-	-	0.033	-	0.014	0.0048
NbTiV	0.12	1.50	0.31	0.038	-	-	0.047	0.051	0.020	0.0064
NbCrCu1	0.16	1.03	0.41	0.029	0.54	0.23	0.025	-	-	0.0107
NbCrCu2	0.13	0.99	0.38	0.042	0.50	0.22	0.014	-	-	0.0095

Surprisingly, in this case neural networks did not show the best forecast power, as can be seen in the radar graphic of figure 4.4. Considering the performance observed in all steels, the forecast power of the empirical models of Tegart, Jäckel, Samanta and Hajduk were greater than of the neural networks. The situation is even worse when one considers only carbon steels; in this case, also the Samanta model surpasses the forecast power of the neural networks. However, considering HSLA steels alone, only the equations of Tegart, Jäckel and Samanta are superior to the neural networks.

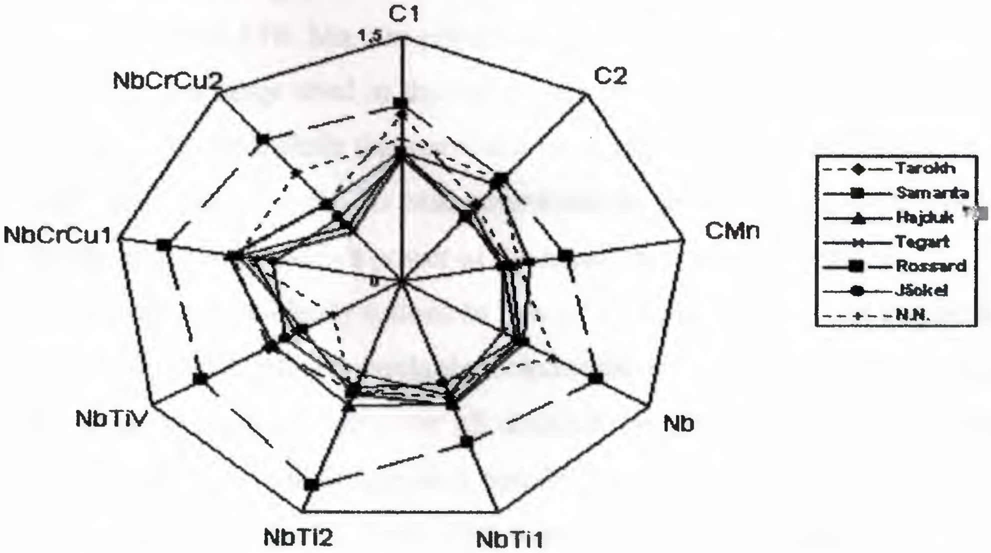


Figure 4.4 Radar-type graphic showing the forecasting power of the several models, in terms of the standard error of estimate, in kgf/mm^2.

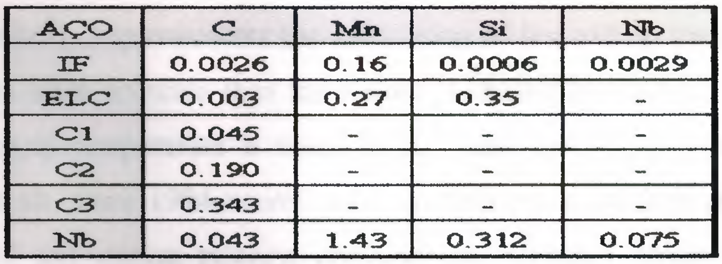
A probable reason for this unexpected conclusion can be attributed for the greater precision achieved in the models developed in comparison with the forecast power reported. In fact, the global standard error of estimate of the best model has a value of approximately 1.2 kgf/mm^2, while the same parameter was about the half of this value - 0.63 kgf/mm2. This better precision result from a better experimental practice and from data smoothing procedures carried out, which minimized the random errors present in the raw data. This enhanced the mathematical relationships between the variables, which simultaneously improved the forecast power of the empirical equations and minimized a great advantage of the neural networks, that is, its immunity to noise and spurious data. So, in the first work, which was based on raw data from torsion tests, the better forecast power of neural networks certainly arosed from its inherent noise filtering effect in raw data.

A neural network is as good as the data used during its training step. The alloys studied in this work were selected according to its participation in the productive mix of the plate mill. Unfortunately, this criterion was not adequate from a scientific standpoint. Another problem was the scarcity of data, as only nine steels are available to develop this model that can involve up to ten chemical elements. In the example of table 4.2, one can see that C content of the considered steels varies between 0.09 and 0.16%, while the range of Mn content is from 0.53 to 1.50%. However, C content is roughly proportional to Mn content. So, if a neural network is trained to consider the effect of chemical composition using these data, its results for a test steel containing, for instance, 0.16% C and 0.53% Mn, are potentially unreliable. These single values of C and Mn are within the range used in the training of the neural network, but data used during its training did not include this particular combination of these elements, that is, "high" C and "low" Mn. In fact, this case constitutes an extrapolation of the available data and, in this case, the predictive power of the neural networks is doubtful.

A neural network can be ideally trained to "learn" the effect of chemical composition in hot strength only if the data set available includes the results of tests performed with steels which chemical compositions cover all situations possible. This only can be made using a factorial experimental design, and certainly will involve the testing of some dozens or even hundreds of steel alloys. The work and cost involved in such research project would be very high if it would be developed from scratch. A possible solution to minimize this problem could be cooperative work between research institutions in this field, through the sharing of hot strength data.

Recently some results were published about the modeling of hot strength of steels from thermomechanical parameters and chemical composition. They tested six steels, which chemical composition can be seen in table 4.3: an I.F. steel, an extra-low C steel, three low C steels and one Nb steel. Hot strength was determined using isothermal compression tests with cylindrical samples, with exception of a low C steel, which was tested using isothermal ring compression tests. These steels were tested under temperatures varying from 860 to 1100°C and strain rates from 0.001 to 1 s-1. The input data of the neural networks included temperature, strain, strain rate, C, Mn, Si and Nb contents. Trials showed that the best results were got with one hidden layer with 25 neurons. The output layer, of course, was constituted of a single neuron, the value of hot strength. Input data included the full range of the hot strength versus strain curve, that is, the steps of strain hardening, dynamic recrystallization and steady state steps.

Table 4.3: Chemical composition of the steels.



AÇO	C	Mn	Si	Nb
IF	0.0026	0.16	0.0006	0.0029
ELC	0.003	0.27	0.35	-
C1	0.045	-	-	-
C2	0.190	-	-	-
C3	0.343	-	-	-
Nb	0.043	1.43	0.312	0.075

The predictive power of the neural network was very good. After a training step with 60,000 iterations, it showed an average error of approximately 0.10% during the test step with data not showed during the training step. The standard deviation of this error was of 3.77%, and the maximum error was about 18%. The predictive performance of this neural network was better than some empirical equations (Wang, Voce and Jonas) and even than a fuzzy inference model [11].

As tests were performed with data deriving from the same steels that generated data for the training step of the neural network, this work does not demonstrate the effective capability of the neural network to model the effect of the chemical composition on hot strength. This only could be checked if the neural network would be tested with data from a steel with different chemical composition from the steels that originated data for the training step. Once more, the problem of the scarcity of hot strength data appears.

As it was available data from only six steels, the use of data from one alloy only for testing purposes certainly would impair the predictive power of the neural network, as these data would not be available for its training step.

4.5 Calculation of Rolling Loads

The calculation of roll forces in hot rolling can be carried out through the use of neural networks, as described in [12]. Its entry layer had five neurons: reduction in thickness, initial thickness, peripheral speed of the work rolls, a deformation resistance factor and temperature. The hidden layer had only three layers. Naturally, the output layer is constituted of only one neuron, that is, the value of the rolling load.

A neural network was tested with real data from the first stand of a continuous hot rolling mill, showing a R.M.S. error less than 5% for the predicted values of load. However, at the moment of the publication of that paper, the model needed to be trained with supplementary data in order to widen its working range.

The use of this new approach of mathematical model, incorporating the use of neural networks, lead to an improvement in the mill performance, as can be seen in table 4.4. According to Siemens, improvements the calculation of hot rolling loads using neural networks are 20% more accurate than the values generated by classical methods. The improvement in strip temperature is about 35%. These first results were considered satisfactory; after all, since 1994 neural networks become a standard method used in every rolling mill automation systems delivered by Siemens. However, work still continues on improvements in new versions of these hybrid models.

Table 4.4: Test results for the neural networks used at the Westfalen hot strip mill.

Type of Neural Network	Month	Number of Alternating Class Strips	Improvement in Mean Absolute Error [%]	Improvement in Standard Deviation [%]
α Calculation	July 1993	4594	10.6%	12.8%
α Calculation	December 1993	3602	12.5%	13.0%
ΔT_1 Calculation	September 1993	3803	20.0%	17.1%
ΔT_2 Calculation	September 1993	3803	42.0%	39.2%

It is interesting to note that also Voest-Alpine developed its own hybrid automation system for hot strip mill automation using mathematical modeling and neural networks, despite the fact of its Linz Hot Strip Mill has a similar system delivered by Siemens.

However, there are no published data available about the performance of Voest-Alpine system under industrial conditions.

The approach of the Voest-Alpine for the calculation of hot strength of steel is somewhat different from Siemens. The practical use of the hybrid model showed that, for some kinds of steel, a specific value of a must be calculated for each rolling stand. So, it is being developed another neural network to identify such classes of steel that require calculation of several values. One of the functions of this parameter is to emulate the microstructural refining that the rolling stock undergoes when it is deformed in successive rolling stands.

4.5.1 Detection of "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's, causing extensive damage. This problem was frequent at COSIPA's plate mill, especially during the processing of Ni steels.

A previous work showed that alterations in the pass schedule could minimize the occurrence of turn-up, and led to the development of a statistical model for the calculation of an optimized pass schedule. It was showed then that there was a critical range of strain values to be avoided during plate rolling.

As soon the use of neural networks became available, it was a natural idea to use them in this application, as the statistical model was unsatisfactory. After several trials, it was developed the following neural network:

- Input layer with five neurons:
    - Desired turn-up index
    - Work roll peripheral speed [r.p.m.]
    - Rolling load [t], calculated by the Sims model
    - Rolling stock width [mm]
    - Roll gap distance [mm] used in the former rolling pass.
- Hidden layer with eleven neurons.
- Output layer with one neuron, which represents the recommended roll gap distance [mm] for the next rolling pass.

The turn-up index used here, one of the input variables, was defined using an arbitrary scale, from 0 to 5, that is proportional to defect seriousness.

The neurons number of the hidden layer of this neural network was calculated after the Hecht-Kolmogorov's theorem. In fact, it was verified that this was the best configuration for maximizing the precision of this neural network.

The developed neural network showed Pearson's correlation coefficient r of 0.992 and standard error of estimate of approximately 3.0 mm. Figure 4.5 shows the dispersion plot of the real and calculated values. The most influencing variables, as indicated by the trained neural network, are turn-up index and initial roll gap distance, followed by rolling stockwidth, rolling load and work roll peripheral speed, considering a decreasing rank of importance.

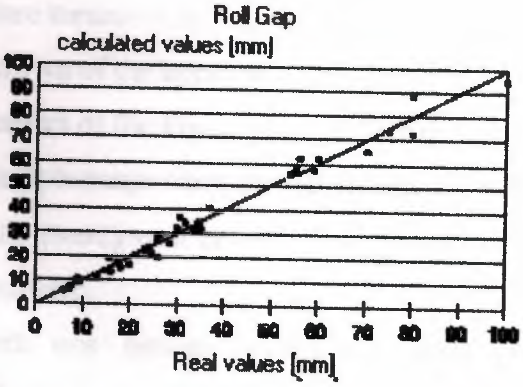


Figure 4.5 Dispersion plot of the calculated and real values from the model for roll gap calculation regarding control of the turn-up defect.

As can be seen in the graphic of figure 4.5, all calculated values are very near from the real values. That is, the neural network did not generate non-sense values as the previous developed regression polynomial equation.

4.5.2 Pass Schedule Calculation Aiming Plate Flatness Optimization

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

Mathematical models for the calculation of pass schedules are relatively easy to develop, but the use of neural networks is simpler. So, it was decided to use this new technique in this application too. The three last passes of the rolling schedule were modeled regarding optimization of plate flatness; one neural network was attributed for each pass. The respective three neural networks showed the same configuration, as follows:

- Input layer with 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]
  - Rolling stock tonnage since the last change of work rolls [t];
- Hidden layer with twenty-one neurons (according to the already mentioned Hecht-Kolmogorov's theorem);
- Output layer with one neuron, which represents roll gap value of the corresponding pass [mm].

The flatness index used in this model varied in the range from 0 to 5: this index was greater as plate flatness worsened.

The performance of the neural network was very good. The Pearson's correlation coefficient **r** corresponding to the last but two, penultimate and last pass were 0.998, 0.998 and 0.999, respectively; their standard error of estimate were 0.430, 0.394 and 0.140 mm, respectively. Figure 4.6 shows the dispersion plots of the real and calculated values for the three last passes of the rolling schedule.

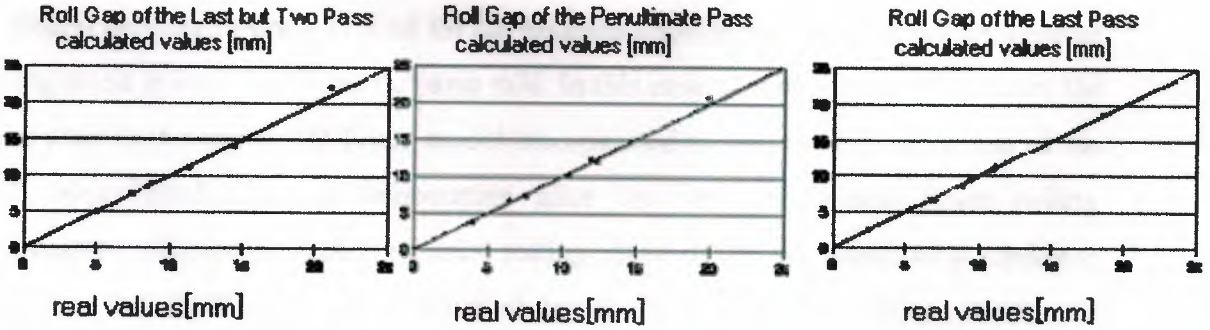


Figure 4.6 Dispersion plots of the calculated and real values of the last three passes of plate rolling, regarding the model for calculation of the pass schedule optimizing plate flatness.

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. Following to this group, there were parameters of intermediate importance like 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 improve the precision of the neural networks, making feasible its use under industrial conditions.

In fact, these neural networks identified during their "learning" step 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 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 last variable was included in the input layer of the neural networks.

The last pass is the most important to define the final dimensions of plate, specially its thickness. The errors observed in the results calculated by the respective neural network varied from -0.26 to +0.17 mm, practically comprised within commercial plate thickness tolerance range. This results can be even improved, as a more precise data acquisition system becomes available, thus avoiding human errors during data collection and improving the precision of the measured parameters. Those facts undoubtedly will contribute to a better accuracy of these neural networks.

4.6 Prediction of Process Temperatures in Hot Strip Mills

Neural networks were also used for the determination of the finishing temperature of rolling stock at the Hoogovens hot strip mill. In this case, the input parameters were the same used in the traditional linear model incorporated to the automation system of the mill: re-predicted finishing temperature after the last stand; base finish rolling temperature, adapted from strip to strip; rolling speed of the last stand; strip thickness after the last finishing stand; calculated temperature at first finishing stand; and calculated temperature at first finishing stand. A feed-forward network, with one hidden layer, was used. The input layer had eight neurons, corresponding to the parameters already mentioned. Best results were got using seven neurons in the hidden layer. The output layer had only one neuron, corresponding to the finishing temperature.

The performance of the neural network model was 25% better than the previous linear statistical model. The relative standard deviation of the former model was about 6.0°C, while the neural network showed a value of 4.4°C. As neural networks are capable of handling different, non-linear dependencies in different areas of the input space, its prediction power generally is improved in comparison with linear models. In fact, a detailed analysis revealed that linear approximations seemed acceptable for most input parameters of the statistical linear model. However, relationship between the finishing temperature and some input parameters are clearly non-linear. This explains the better performance of the neural network model.

Strip cooling at the runout table of hot strip mills can also be modeled using neural networks. The first part of the hot strip mill runout table at the Port Talbot works of British Steel was modeled using neural networks. Data used for training and testing of the neural network models consisted of 247 sets taken from three coils of C-Mn steel. They were all rolled in the same period, having identical target gauge, finishing temperature, interrupt temperature and coiling temperature.

Three feed-forward neural networks were tested. The input layer was the same for all these networks; it has eight neurons, corresponding to the finishing temperature, position of the coil segment in relation to the head end of the coil and water flow percentage of the first three top banks and the first three bottom banks. The output layer corresponded to the interrupt temperature, which is the strip temperature just after the first part of the runout table. Two of the proposed neural networks had only one hidden layer, with 20 and 4 neurons, respectively. The remaining network has two hidden layers, with 10 neurons each.

Surprisingly, the simpler neural network (one hidden layer with 4 neurons) showed best results, with an average error of 2.1°C and a maximum error of 14°C. This performance was far better than the standard model being used nowadays in the equipment, which showed an average error of 10.4°C and a maximum error of 25°C.

4.7 Feasibility of Production of A particular Steel Shape

Normally the decision about the feasibility of the production of a given steel grade or product is taken by an expert engineer, who intuitively evaluates the manufacturability difficulty grade of it. This ability is acquired through experience.

A model to systematize this technique in the specific case of shape steels, using a hybrid system, expert system and neural networks [13]. The function of the expert system is to select the neural network best fitted for the given case. The selected neural network simulates the judgement mechanism of the expert engineer, determining if the fabrication of a given product is feasible or not. Of course, this neural network must be previously trained with real examples to acquire the needed knowledge for this task.

An example of such neural network considers three variables: tensile strength lower limit, flange thickness and impact toughness guaranteed temperature. Their values are not directly input into the neural network. They are normalized according to a five level division. The values corresponding to each level are the data supplied to the neural network.

So, the input layer of the neural network has 15 neurons. Its hidden layer has 10 neurons. The output layer has five neurons, each one corresponding to a value of the manufacturing feasibility index. Only one neuron will react to a given set of data, signaling its corresponding manufacturing feasibility index. The first neuron of the output layer corresponds to the lowest value of the manufacturing feasibility index, that is, the worst condition of fabrication. For its turn, the last neuron corresponds to the maximum value of the manufacturing feasibility index, that is, the best condition of fabrication. The other neurons denote intermediate values of this manufacturing feasibility index.

4.8 Summary

This chapter described the application of neural networks in the modeling of hot rolling processes. However, almost ten years after the resurgence of neural networks, its application in the field of hot rolling still is shy, in spite of the good results that were got.

In the few cases where neural networks were applied to real life hot rolling automation systems, they played a coadjuvant role in the form of calculation of adaptive parameters to be used by the main conventional mathematical models.

Perhaps a more effective use of neural networks in the modeling of hot rolling processes is being considered more intensively in steelworks that still do not have hot rolling models developed.

However, it must be noted that the industrial scale application of neural networks in hot rolling mills has effectively begun. The build-up of expertise and experience in the use of this new technique that will be collected along time undoubtedly will encourage further practical applications.

CONCLUSION

Neural networks are computational constructs loosely modeled on the structure of the human and animal brain. They are comprised of neurons that are the information processors of a brain, and synapses, which are spaces between neurons that can be thought of as weighted buses that connect these processors.

Although, the neural network contains a large number of simple neuronlike 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.

Chapter one described some definitions of Neural Networks and it included a brief history of Neural Networks since the first days till the recent development of this technology.

Beside that, what Neural Networks used for also has been illustrated, and where it is more applicable. Also this chapter included some advantages and disadvantages of Neural Networks.

In Chapter two the architecture of Neural Networks and how can it be trained have been discussed, and also it described the ways that Neural Networks can be trained with, such as Supervised and Unsupervised networks. Classification of Neural Networks also has been illustrated within this chapter.

Chapter three described some applications of Neural Networks where can it be found in industry. Some of these applications have been included within this chapter are: Oil and Gas Industry, Papermaking Plant and games.

Chapter four discussed a general application of Neural Networks in industry which is Modelling of Hot Rolling Processes.

REFERENCES

[1]. Klimasauskas, CC. (1989). The 1989 Neuro Computing Bibliography

[2]. Hammerstrom, D. (1986). A Connectionist/Neural Network Bibliography. DARPA Neural Network Study (October, 1987-February, 1989).

[3]. Eric Davalo and Patrick Naim. Prof. Aleksander. articles and Books.

[4]. Berntson, M., and P. Stoll. 1997. Correcting for finite spatial scales of self-similarity.

[5]. Panico, J., and P. Sterling. 1995. Retinal Neurons and Vessels Are Not Fractal But Space-Filling. The Journal of Comparative Neurology 361:479-490

[6]. (http://www.dacs.dtic.mil/techs/neural/neural.title.html, printed November 1998)

[7]. ftp://media.mit.edu/pub/sturman/WholeHandInput/*.

[8]. ftp://whitechapel.media.mit.edu/pub/tech-reports/TR-316.ps.Z.

[9]. DONADILLE, C. et al. "Usage de Reseaux Neuronaux por Prevision des Courbes de Transformations de Acier". Revue de Metallurgie - CIT, October 1992, 892-94.

[10]. HECHT-NIELSEN, R. Neurocomputing. Addison-Wesley Publishing Company, Reading, 1990. 433 p.

[11]. GORNI, A.A. "Modeling the Effect of Alloy Elements in the Hot Strength of Steel". In: International Conference about Engineering Applications of Neural Networks (EANN '97). Proceedings. Stockholm, June 1997, 143-150.

[12]. LEVEN, J. et al. "Artificial Neural Networks for Rolling Applications". In: ATS Steelmaking Days, A.T.S., Paris, 1994.

[13]. BULSARI, A. & HOCKSELL, E. "Quality Control and Alloy Design System for Direct Hardened Steel Components Based on Neural Networks". Steel Technology International, July 1996, 133-8