Artificial Neural Networks (ANN)

- In data pattern recognition for monitoring purpose

LUNDS UNIVERSITET
Lunds Tekniska Högskola

LTH School of Engineering at Campus Helsingborg
Department of Computer Science, Lund University

Bachelor thesis:

Tom Stevens
Dalibor Lovric

LTH School of Engineering
Lund University
Box 882
SE-251 08 Helsingborg
Sweden

LTH Ingenjörshögskolan vid Campus Helsingborg
Lunds universitet
Box 882
251 08 Helsingborg

Printed in Sweden
Media-Tryck
Biblioteksdirektionen
Lunds universitet
Lund 2011

Abstract

This thesis is meant to be a simple introduction to the field of the Artificial Neural Networks and a survey of the possibility of using neural networks in fault diagnosis of sand moulding machines manufactured by DISA. The purpose of the survey is to give an indication of the direction the future work should take in order to obtain a sustainable solution for the use of artificial neural networks for monitoring molding machines in industrial plants.

A description of the sand moulding process and DISA's current fault detection and diagnosis procedure is given together with testing the potentials of feed-forward neural networks for recognizing patterns represented in control charts based on data of 16 sampled channels on the DISAMATIC moulding machine. The testing in Matlab and Encog environment proved that neural networks can learn to recognize periodic patterns in presented data but accepts too large deviations in patterns. The concluding part in this work reveals that an application based solely on neural networks, is not the sustainable solution and some prior signal processing of the sampled input is necessary.

Keywords: artificial neural networks, control chart, fault diagnosis, surveillance, pattern recognition

Sammanfattning

Detta examensarbete handlar om artificiella neurala nätverk och är en undersökning av om det är möjligt att använda artificiella neurala nätverk vid dignostisering och feldetektering på sandgjutningsmaskiner som företaget DISA tillverkar. Resultatet av undersökningen är en fingervisning i vilken riktning framtida arbete bör ta för att erhålla en hållbar lösning för användning av artificiella neurala nätverk för bevakning av gjutningsmaskiner på industrianläggningar och en översikt av den potential som neurala nätverk besitter.

En beskrivning av sandgjutningsprocessen och DISAs nuvarande feldiagnostiserings rutiner tas upp tillsammans med testing av möjligheter att använda feed-forward neurala nätverk för igenkänning av mönster representerade i styrdiagram (grafer) uppbyggda på samplad data av 16 kanaler på DISAMATIC gjutningmaskiner. Testningen i Matlab och Encog miljö har bevisat att neurala nätverk kan lära sig att känna igen återkommande mönster i den presenterade datan men att de accepterar alltför stora avvikelser i mönstren. Den avslutande delen i detta arbete visar att en applikation som enbart grundar sig på neurala nätverk inte är en hållbar lösning och att någon slags signalbehandling av den samplade inputen är nödvändig.

Nyckelord: artificiella neurala nätverk, styrdiagram, feldiagnostisering, övervakning, mönsterigenkänning

Foreword

This thesis is based on a collaboration between Lunds Tekniska Högskola and DISA in Denmark. During this thesis we had a close co-operation with Nils Assarsson and Daniel Bergquist at DISA and this thesis would not been possible if it were not for their dedication and support.

We would also like to thank our supervisor Mats Lilja for his excellent guidance and helping work through this thesis. Additionally we would like to thank Jacek Malek for his input and an interesting discussion about artificial neural networks.

Tom: I would like to thank my fiancée Michelle and my family for their support, and their belief in that I could succeed with this.

Tom Stevens & Dalibor Lovric

Helsingborg, June 2011

List of contents

1 Introduction

An Artificial Neural Network, hereby referred to as a neural network, is an abstract mathematical model developed in order to simulate the function and intelligence of the brain. A very attractive property of neural networks is the self-learning ability where a neural network can learn the behavior of a system from training data without requiring prior knowledge about the system. This technique has matured during the recent decade and is now used in many applications such as expert systems, speech recognition and many different kinds of classification problems. An example of successfully implemented application incorporated by the Swedish postal service is the application for pattern recognition used for reading the postcodes of letters on-the-fly. Furthermore, there are applications run on supercomputers for predicting the weather and development of stock prices in the financial sector. The objective in this thesis is to evaluate the use of neural network for monitoring an industrial process in which present methods, manual monitoring of all process variables and parameters is too tedious and too costly to apply.

The global economic development has led to that today's industry has high demands on availability, reliability and profitability. This raises the need for failure detection in the various industrial processes. The dream scenario is to apply monitoring to processes and to detect errors before they become severe, so adequate countermeasures can be initiated in time and minimize or avoid the risk of money loss due to a major breakdown. Such a monitoring process is requested by DISA in Copenhagen [8], a manufacturer and worldwide distributor of moulding equipment.

The production process on DISA's machinery is very complex and it is often composed of multiple synchronized steps that are dynamically controlled. It is common practice to monitor parts or even entire processes by checking the corresponding parameters, which have been recorded during the process. The monitoring on DISA's casting machines is performed on pressure in the hydraulic pistons, pumps and positions of the moving swing and pressure plates. In such a monitoring procedure, because of the dynamics of the processes, the large amounts of parameters are produced and need to be inspected. In addition to required knowledge and expertise in the relevant field, inspections also require a significant amount of time. It may be possible and economically acceptable to monitor a single machine, but when there is more than a thousand machines, it is very time consuming and thus uneconomic. Therefore, there is a need for some sort of automatic monitoring with alarm features at the occurrence of errors in processes. The dynamics of the processes reveal themselves by their parameters in different types of periodic patterns represented in the charts and hence the idea to try to apply neural networks for pattern recognition to discover deviations.

1.1 Goals

The primary aim of the case study described in this thesis, is to determine if neural networks are suitable for independent analysis, fault- and anomaly recognition of a series of parameters on DISA's moulding machine during the molding process.

1.2 Limitations

The work in this thesis is based on analyzing and testing the data that has been sampled on a moulding machine incorporated at DISA´s customer in Norway and downloaded via Remote Diagnostic Access. It is neither possible nor allowed to manipulate the machine for generating a specific error-rich data.

2 Methodology

The description of methods used to realize the goal of applying neural networks in fault detection and diagnosis is given in this chapter. Furthermore, the justification for the use of these methods is also brought up.

2.1 Literature

The first step is to gain knowledge of artificial neural networks. There is a relatively large supply of literature about ANN but sources chosen are books ([1], [2], [3], [5]) and trusted webpages ([4], [6]). Furthermore, an overall understanding of different architectures and training algorithms is needed to determine which one is suitable for testing of pattern recognition.

2.2 DISA and data analysis

The second step is to learn about DISA, moulding machine and its working principle. This information is provided to us by our contact personnel at DISA in form of sampled data and a film of DISAMATIC principle. Further information is gained by reading DISAs´ webpage. We have had a couple of meetings with technicians responsible for the machine monitoring that will illuminate us with the problems of current monitoring procedure and the representation of the sampled data.

2.3 Software search

A well proven and reliable testing environment is necessary to obtain reliable results. Matlab is an given candidate, but at least one open source is to be picked in order to compare and confirm the testing results. Time will be dedicated to evaluate the possibilities and limitations of the chosen software against the Matlab.

2.4 Testing

A quantitative testing will be performed to figure out and determine neural network architecture, software setup and the form of input data until the best results are obtained. The chosen testing technique is the black box model where our knowledge of software functionality is limited to its interfaces. The testing software is to run on computers with at least dual core processors at 2.0

GHz clock frequency, 2.0 GB memory (RAM) and operating both Windows 7 and Mac OSX 10.6.7.

2.5 Conclusions

The conclusion is to be presented based on presented results together with problems encountered.

3 Background

DISA is the world's leading manufacturer and global provider of innovative foundry technology. A complete range of metal casting production solutions for the ferrous and non-ferrous foundry industry is developed and manufactured by DISA. A large group of customers all over the world are using these machines in moulding production of all kinds of products, from car components such as brakes and engines to keys and fireplaces.

Figure 3.1 The DISAMATIC - DISA's sand moulding machine

Essentially a DISAMATIC machine [9] consists of two machine units: a molding machine (DMM) and Automatic Mold Conveyor (AMC). Additionally, a complete DISAMATIC line can be extended with additional units. Some of the most important are the Sand Supply Unit (SSU), Quick Pattern Change (QPC), Core Setting Device (CSE), Pneumatic Mold Conveyor (PMC), Synchronized Belt Conveyor (SBC), pouring device, casting cooling drums and sand preparation plants. The survey in this thesis is intended for monitoring of the molding machine. However the principle can be applied to monitor other equipment along the line as well.

3.1 Moulding process description

Sand moulding is quite a simple process and in this chapter we will give you a brief description on how it is performed on the DISAMATIC moulding machine. The DISAMATIC molding machine is designed and constructed for rapid and automatic creating a line of flaskless molds (without using multi-part molding boxes). The moulding machine is working on DISAMATIC principle of sand moulding where molds are created in a vertical molding environment and are used for vertical casting. The working cycle at DISA moulding machine can be divided into 6 operations. The complete animation can be found at [7].

Operation 1 - Sand shot.

The shot valve opens and compressed air from the air receiver blows sand from the feed hopper, placed on top, into the moulding chamber where two plates with pattern are placed on the two ends of the chamber. One of the plates is swingable and hence the name, the swing plate and the other one is pressure plate which is placed opposite to the swing plate and which presses the moulding sand. The molding sand is silica sand mixed with balanced doses of clay (bentonite) and some other additives.

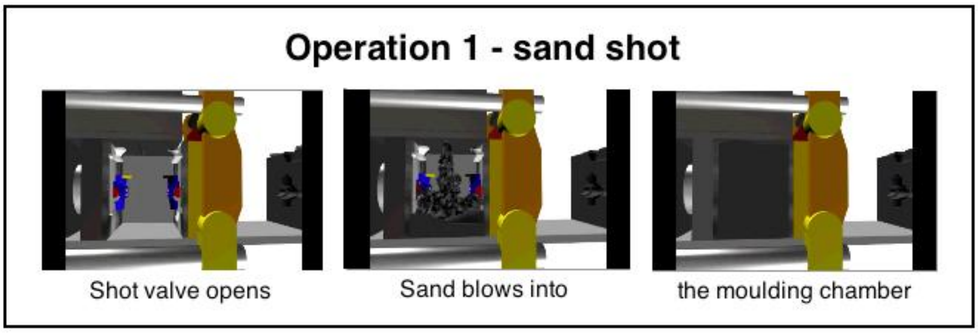

Figure 3.2 Operation 1 - sand shot

Operation 2 - Squeezing.

The mould is squeezed against the two pattern chamber plates until the pressure reaches the desired value. This pressure can be adjusted to give the moulds different hardness and characteristics.

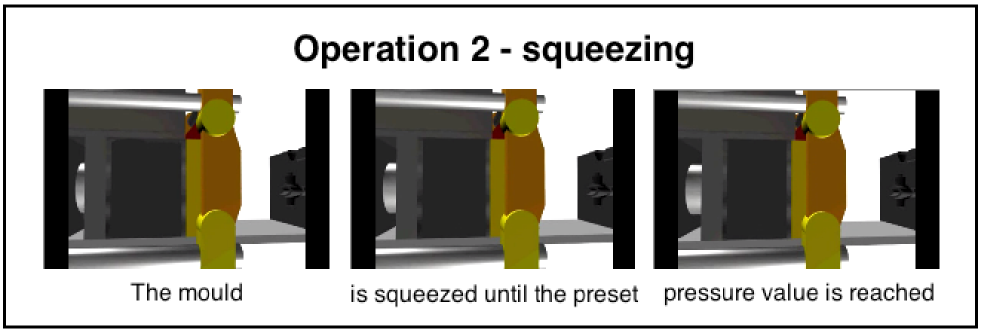

Figure 3.3 Operation 2 - squeezing

Operation 3 - Stripping SP.

The swing plate is slowly stripped from the mould and swings up to the horizontal position. The moulding chamber is now left open.

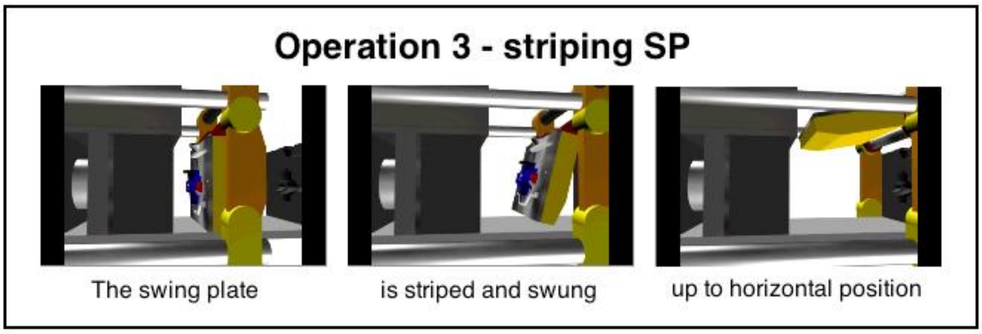

Figure 3.4 Operation 3 - stripping SP

Operation 4 - Mould transport.

The pressure plate pushes the mould out of the moulding chamber, into the back of the previous mould. Then the automatic mould conveyor conveys the mould string forward.

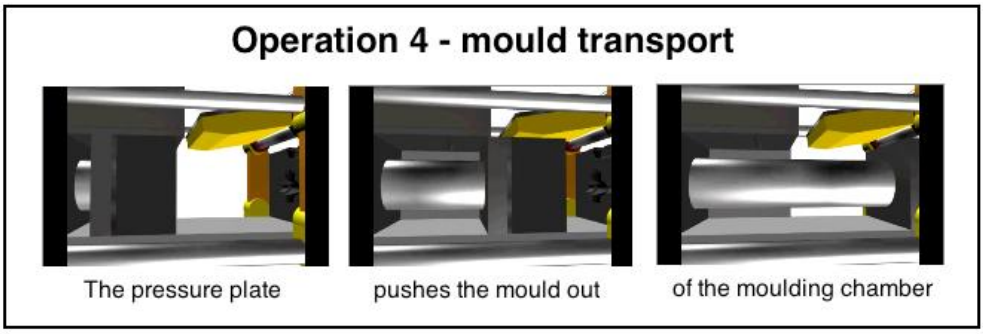

Figure 3.5 Operation 4 - mould transport

Operation 5 - Stripping PP.

The pressure plate is stripped from the mould and returns to starting position in the mould chamber.

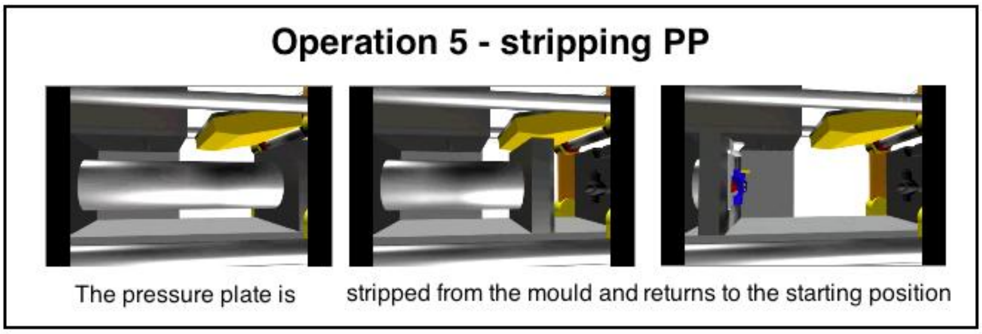

Figure 3.6 Operation 5 - stripping PP

Operation 6 - Closing chamber.
The swing plate swings down into vertical position and closes the moulding chamber ready for a new moulding cycle.

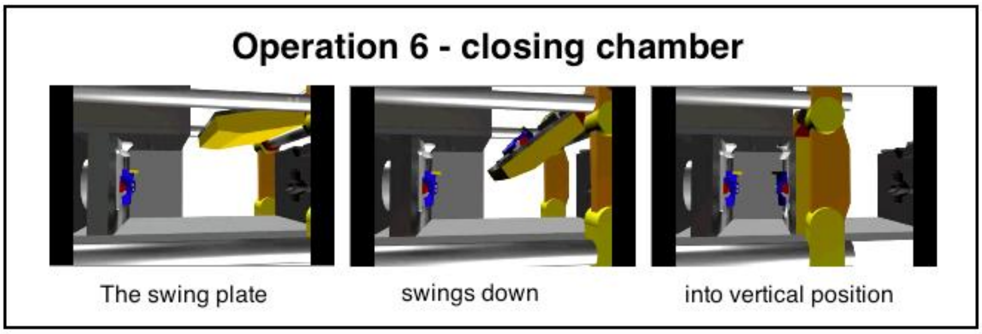

Figure 3.7 Operation 6 - closing chamber

If cores are needed, then they are automatically inserted by core setter into the mold cavity while the preparation of next form is performed. The cycle is repeated while a chain of finished molds are being linked up to each other on the rolling line. Once the molds by the conveyor have been transported out of the molding machine, they are filled with molten metal or some other desirable material and continue on a cooling conveyor to finally get to the end of the conveyor where the solidified castings get separated from its sand molds. The castings may need further processing and the form sand is reused in the next cycles of molding process.

3.2 Current fault detection and diagnosis

This chapter is intended to clarify the current methods of fault detection and diagnosis at DISA. The fault detection and diagnosis system currently used at DISA is called Remote Monitoring Service and is basically an extension to

DISAs Remote Diagnostic Access which is a dial-up-modem that allows technician to control the operating panel of the machine remotely. The Remote Monitoring Service is based on a model where the moulding machines system response signals, distributed over a number of channels are sampled with a frequency of 225 Hz and stored in a database. One single cycle of a specific parameter is presented with approximately 2400-2600 sample values on one channel and since the moulding machine can continuously produce approximately 240 - 300 moulds per hour, tremendous amounts of data over time is generated and stored.

While producing molds through each cycle the speed and position of every moving part in the machine is determined in advance, meaning movements of all interacting components are synchronized. This implies that all the pressure values, air and pneumatic, and the valves opening and closing timing correspond to predefined values. Each channel represents one of the significant machine components, and observing values on it can reveal its behavior. The representation of 16 channels on the moulding machine is listed in the table below.

Channel	Name	Unit	Description
1	Tilt_com_PP	m/s	The requested speed of the Pressure Plate
2	Tilt_fb_PP	m/s	The feedback speed of the Pressure Plate
3	Tilt_com_SP	m/s	The requested speed of the Swing Plate
4	Tilt_fb_SP	m/s	The feedback speed of the Swing Plate
5	PA_PP	bar	Pressure A on the Pressure Plate
6	PB_PP	bar	Pressure B on the Pressure Plate
7	PA_SP	bar	Pressure A on the Swing Plate
8	PB_SP	bar	Pressure B on the Swing Plate
9	Valve_setp	V	Control of valve for pressure
10	DP_setp	bar	The requested pressure in the piston
11	DP_FB	bar	The feedback pressure in the piston
12	P_servo	bar	The pressure in the main servo.
13	PP_speed	m/s	The actual speed of the Pressure Plate
14	SP_speed	m/s	The actual speed of the Swing Plate
15	P_shot	bar	The pressure inside the sandshooter
16	P_over_sand	bar	The pressure over the sandbox

Table 3.1 Input table

The sampled data during a time interval specified by the inspecting technician is later obtained via remote access to the database where the data is stored. Further control charts are generated out of the sampled data which represent time-series values. Then the control charts and are manually analyzed by technicians. The method is that control charts are used for visual pattern recognition in processes to identify problems in those. An example of control charts over 16 channels is depicted on the next page in Figure 3.8.

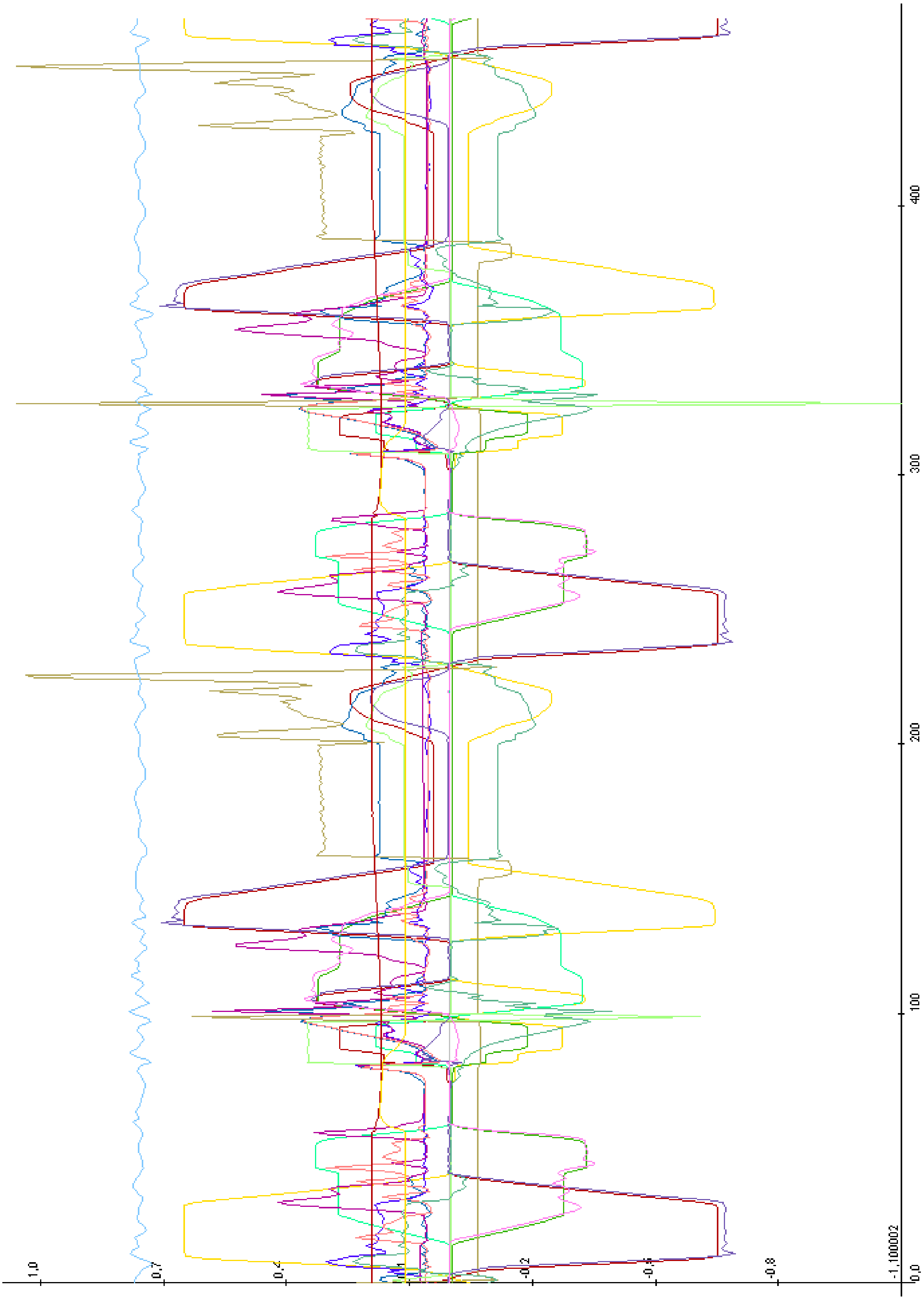

Figure 3.8 Control Chart over two moulding-cycles

Since the moulding process is periodic its periodicity is represented through the patterns repeating in the chart. Furthermore, the existence of correlations between the signals are clearly visible in the chart, this is a very useful property that makes monitoring based on neural networks theoretically feasible since neural networks are capable to learn correlations.

3.3 Error appearance and representation

Currently, the only method for detection of abnormalities in processes on the moulding machine is by manually inspecting the control charts. The chart for every channel presented in the table above is to be examined for deviations and unnatural patterns. Any deviation in charts can be associated with a specific problem on the performing component and may affect the moulding process. By examining these charts lots of faults of different kinds can be detected, everything from a malfunctioning valve to a worn or broken bearing on a component which malfunctions is affecting the moulding process. The detected error in the figure below is in fact a pneumatic valve leaking the pressure and the disturbance caused by its condition might increase and replacement is required.

Figure 3.9 Error detected - leaking pneumatic valve.

However, the example given in the Figure 3.9 indicates that detected malfunctions in the chart can be a barely noticeable, 2 mm bump, hence very difficult to detect in a control chart. Scanning charts for anomalies and errors requires great skill of the engineer performing the job. It is desirable that the engineer possesses many years of experience, knowledge and understanding of the forming process, machine structure and its behavior. Several issues can be addressed to this procedure. The detected deviations in charts may be difficult to determine and distinguish from signal distortions and normal cycle operation signals. This manual monitoring approach is tedious and time-consuming and is not applicable for a large scale monitoring project.

4 Artificial Neural Networks

The interest for artificial neural networks arises from the idea that the study of them may help us understand the brain, human cognition and perception and has lead to development of systems with improved functionality to solve complex problems. This means that the inspiration for development of neural networks comes from the fascination of human abilities for learning, understanding and performing complex tasks with simplicity and precise elegance. To mention something as simple as recognizing human faces, fetching a glass of water or kicking a football, things we just do without even knowing or realizing how these tasks are performed in our brain. A wish for developing a technology able to efficiently perform the same tasks as humans, but more accurately, faster and without ever getting tired or minding working in unsustainable conditions has been a driving force for researchers in academies and in the industry. It has been proven that neural networks can be used to efficiently solve many problems that are difficult using conventional programming techniques. Successful approaches have been achieved in approximation, categorization, prediction and pattern recognition fields.

4.1 Introduction to neural networks

4.1.1 The biological neural networks

It is known that there exist over hundred types of neurons; however most of their common characteristics have been found. The biological neural network consists of a tremendous number of nerve cells (neurons) which are connected with each other via synapses. The number of neurons is determined at the birth and is roughly estimated to 10 billions and is relatively constant throughout the life. The newborn human has a lot less synapses compared to a grownup, but is developing hundreds of thousands new ones per second due to the learning of the new world.

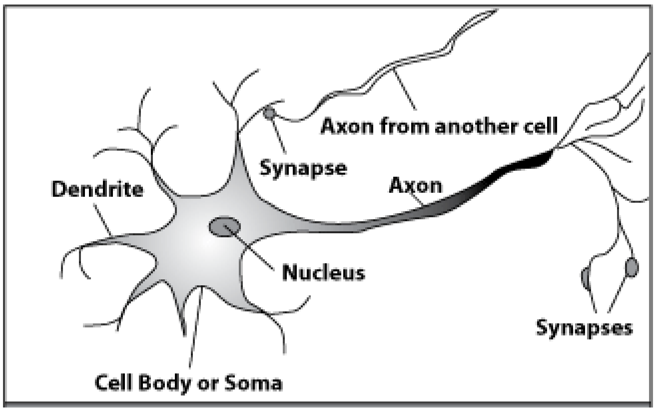

Figure 4.1 A biological neuron, from [8]

A neuron has a single output called axon and multiple inputs called dendrites. Axons can be branched to connect to multiple other neurons via their dendrites and the signal flow is unilateral. Signals, which are called action potentials, which are sent between neurons, are electrochemical. The signals are received via their dendrites and added together inside the cell through the processes of spatial and temporal summation. In case of a temporal summation a threshold has to be reached for the signal to be propagated down its axon to other neurons. When a human is learning something new the process of printing to the memory in the brain takes place. The synapses between the stimulated neurons are strengthened in the very moment when the learning occurs. Thus it is in both in the synapses and in the neurons with their threshold where all memory is stored. The human brain is capable to perform demanding perceptual acts and control activities and is effective in the use of parallelism.

4.1.2 The artificial neural networks

Artificial neural networks are a very simplified model of biological neural networks. There are many theories on how biological neural processing works but the main characteristics have been determined and used to implement the artificial ones. Input neurons are fed with the input signals and are further connected via weighted connections to next layer. Each neuron does a summation of signals received through its weighted connections, if this value reaches the determined threshold it is propagated to other neurons. Nodes with connections to itself are classified as feedback (recurrent) networks and those without are classified as feed-forward networks. Feed-forward networks consist of multiple layers of nodes, where each node is connected to all or some nodes in the next layer until the output layer is reached.

Neural networks as well as the biological ones ought to undergo some learning process before they become fully functional. This is accomplished through various learning algorithms, which in principle is to adjust the weights on the connections between the nodes and the threshold values of the neurons. The purpose of the neural networks use and desired properties, determine which algorithm shall be used. There are many different types of neural networks, from relatively simple to very complex ones with various learning algorithms.

4.1.3 The artificial neuron

A network consists of a number of elements or nodes, denoted as x_i. Each node receives signals from other nodes, processes and forwards them to other nodes. There is at every moment an activity in each node, here denoted as x_i. Nodes are connected by directed connections, denoted w, which have a weight or strength. Node i is connected with node j with connection w_{ij}.

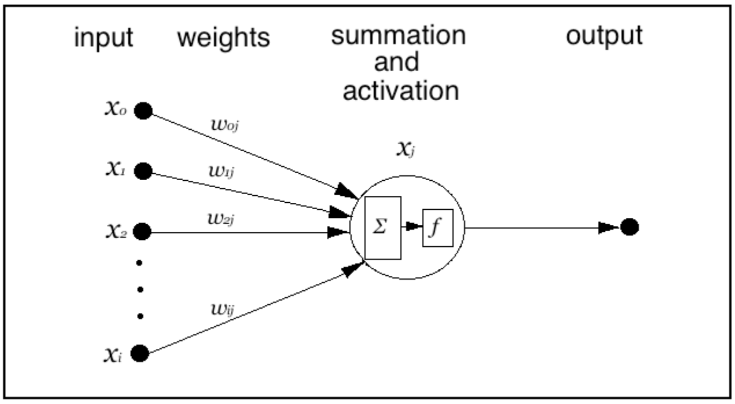

Figure 4.2 An artificial neuron

Signal dynamics of a network can be modeled either as continuous or discrete. The discrete is easier to explain. Input or other elements' activities are transformed into signals and proportionally strengthened by the weights. When inside the node, all signals are summarized. The inner sum corresponds to:

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

The activation function takes the summarized input as argument and the output value of this function is the nodes' resulting activity or output. The activating function is denoted with *f* and the resulting activity with x_j.

$$x_j = f(x\_in_j)$$

and an extended formula with summarized activity is

$$x_j = f\left(\sum_{i=1}^{n} x_i w_{ij}\right)$$

4.1.4 Activation functions

Activation functions are functions that are involved in deciding how the nodes and the network process signals. The function controls when the neuron should be active and that depending on if a given threshold is reached or not.
In the standard theory of neural network a few basic types of activation functions are elaborated. We mention here some of the simplest like linear, threshold functions, activating trough competition and sigmoid functions. When choosing the activation function some important factors need to be considered. Using linear functions in multilayer network is pointless because the biological correspondence is nonlinear. Another important factor in recurrent network is the stability aspect. Since the signals are strengthened and summarized, they can get arbitrarily high values and for that reason the activation function has to be of limiting nature. For the training procedure, especially in the error backpropagation network, it is important that the functions are of derivative nature. Some useful activation functions are signum, log-sigmoid transfer and tan-hyperbolic functions.

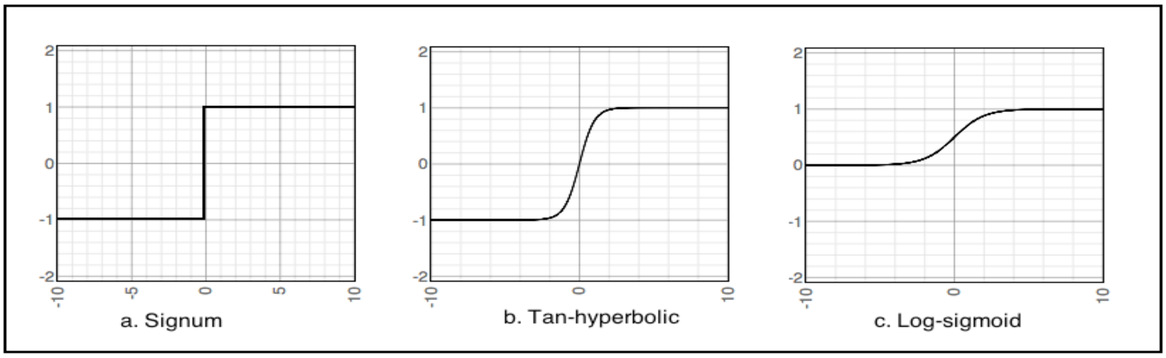

Figure 4.3 Activation functions

Graphical representations of useful activation functions are depicted in Figure 4.3. The log-sigmoid activation function (Figure 4.3 c) should only be used with positive values, since it blocks out negative ones and it is often used in feedforward and simple recurrent networks.

$$f(x) = \frac{1}{1+e^{-x}}$$

The tan-hyperbolic activation function (Figure 4.3 b) accepts input of both negative and positive values.

$$f(x) = \frac{e^{2x}-1}{e^{2x}+1}$$

4.2 Architecture

The arrangement of the individual elements in the structure of a neural network can be described as layered or non-layered. The categories are is defined in terms of signal flow direction and do not have anything to do with the spatial arrangement of the nodes in the neural network. According to this definition, the layered neural network is always of feedforward-type, in the sense that the signal flow through the network is always unidirectional. In contrast to feedforward networks, the feedback networks (also called recurrent networks); signals are fed back to some previous elements. The feedforward and recurrent network architecture contain all the essential characteristics for recognition of different patterns hence a description of these networks follows.

4.2.1 Multi-layered Feedforward Networks

Multi-layered feedforward network is a natural and important extension of single layered networks, where at least one layer has been placed between the input and output layers. The feed forward networks are designed such as that the signals are fed straight forward throughout the entire network. There is no back-feeding of signals into the previous layer, not even indirectly, see figure 3.4 below.

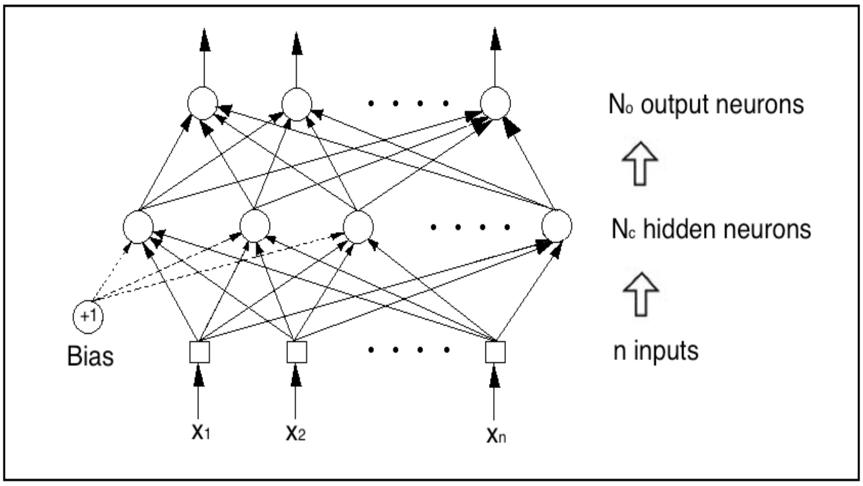

Figure 4.4 Multi-layered network

A network of this type consists of an input layer, an output layer and at least one intermediate layer. These intermediate layers are also called “hidden” layers because their activity is not accessible from outside the network as input and output layers are. The incoming information, from the external world or from another network, is represented as activities of input nodes, meaning that it is not performing any calculations and thus only acting containers for the input. Input nodes in this multilayer architecture represent the input layer. Signals are sent further to the hidden nodes where they are processed and while transferred over connections are also transformed by the weights. In turn, the output nodes are activated and deliver the entire network's output to the outside world, which can be the human user or the next neural network. Output nodes in this multilayer architecture represent the output layer.

Hidden nodes provide greatly increased opportunities for many forms of information processing that cannot be realized in single layer networks. It has been proved that more than two hidden layers do not improve computing power rather than getting the training procedure too complicated. Networks with two hidden layers are often enough to solve the most complex problems. To create really powerful multi-layer networks, it is essential that the weights on the connections to the hidden layers are modifiable through a training procedure. However, usually the number of neurons in the hidden layers can be difficult to determine as there are no specific rules or theorems to comply, but only some vague thumb rules. The number of neurons in the hidden layer can have a significant impact on network’s performance; with too few neurons the network might not be able to learn the problem and with too many it will result in training problems. In some cases, extra nodes, so-called bias nodes can be added for specific purpose networks. These bias nodes feed a constant signal into the node which can help the signal to gain enough to activate the node.

Because of their design feedforward networks are mostly suitable for pattern recognition. This due to the fact that they can handle static inputs very well, and learn the correlations between the different inputs.

4.2.2 Feedback or recurrent network

In the feedback network the signal flow is not unidirectional. Neural networks with one or more feedbacks are referred to as recurrent networks. There are two methods of realizing feedback structure in a neural network. One is local feedback at the level of a single neuron inside the network and the second is global feedback regarding the whole network, hence the classification of networks to local and global feedback network.

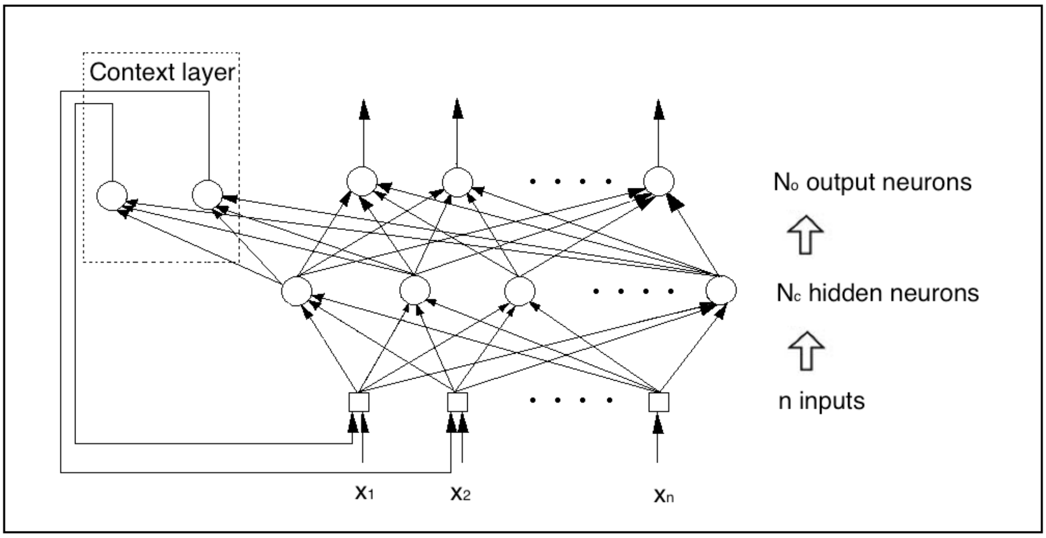

Figure 4.5 Recurrent network

The feedbacks have significant influence on the network's dynamics in the sense that individual neurons can be active a long time after the network received a single input signal from some external interface, sometimes even for arbitrarily long time. This approach of using time delays in the structure of a neural network provides the network with memory functions. Delayed inputs are used to remember the previous states of network allowing recurrent neural networks to predict outputs based on a number of previous inputs. The purpose of the various network types decides which nodes in a recurrent network to read the output values from. The activity pattern can either be read for each time step, i.e. as an ongoing process considered as a discrete time series, or be chosen to interpret the output as the final activity pattern for a certain time period. The latter is particularly meaningful for such networks that have a final pattern, that is, those for which the signal will eventually converge. Such networks are also called attractor networks and an example is the Hopfield network. [2]

4.3 Training

An important property of a neural network, in order to be useful is the ability to learn from its environment and through this learning improve its performing capabilities. The learning process is done by adjusting the weights and threshold values. Applying those adjustments changes the way the network deals with a given signal and therefore also the output signal. Hence it is important to find a sensible way to change the weights so that the network will perform the intended tasks with the desired output values as a result.
Although there is huge variety of diverse training algorithms, an intelligent selection of such may be the key for solving a difficult problem. During the learning process, the network is stimulated with a predefined set of data, called training set which is related to the environment in which the network operates. With respect to presence of supervision in training data set, learning algorithms can be classified as supervised or unsupervised learning.

Supervised learning means that the learning network obtains the knowledge about the unknown environment from a teacher providing a predefined set of input-output examples. Supervised learning is performed off-line meaning the learning phase and the operation phase is distinct. It is suitable when sufficient training examples are available and time delays in application due to training are acceptable. As the name implies, the unsupervised learning is done without a teacher providing the knowledge, where detailed input data is provided without any information on the desired output. Unsupervised training is performed on-line, meaning the network learns and operates at the same time. Unsupervised training is more suitable for applications where the training examples are insufficient (lacking the knowledge of the data representation), limited or the process is too dynamic.

4.3.1 Backpropagation

The backpropagation algorithm is one of the oldest learning algorithms for multi-layer feed forward artificial neural network. This algorithm is of supervised training type, it is relatively simple and it has some deficiencies. There are however, a large number of advanced and more useful variants of this algorithm, but since backpropagition is the most basic one, there is a good reason to understand the principle behind it.

The complete learning procedure involves calculating the total error in the output and then using a stepwise mathematical procedure to calculate how the output error depends on the weights of the network and correct them in reverse order. The total error in output is calculated of all inputs and for all output nodes. The total error in output represents the sum of the squared differences between desired and actual output.

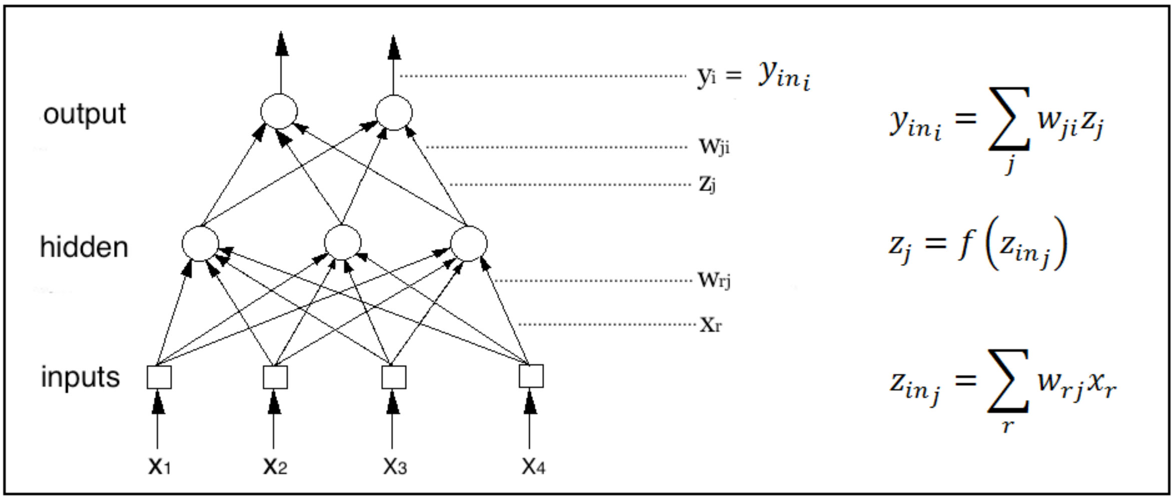

Figure 4.6 Neural network in BP - algorithm

The figure above presents a two layered network with four input units, three hidden sigmoid units and two fully linear output units. It is worth noting that the nodes in the network are numbered starting with the input layer, continuing through the hidden layers, layer by layer and finally ending with the output layer. Neurons in this network are numbered beginning with characters *x*, *z* and *y* denote the activities in the input, hidden and output layer. It is assumed that for a given input vector x to get the desired output $d_k = (d_{k1}, d_{k2})$. The actual output is $y = (y_{k1}, y_{k2})$. The appropriate measure of error for this task is now the sum

$$E_k = (d_{k1} - y_{k1})^2 + (d_{k2} - y_{k2})^2$$

and more generally would be if we had m outputs:

$$E_k = \sum_{i=1}^{m} (d_{ki} - y_{ki})^2$$

The squared differences between actual and desired outputs over all the *m* output nodes are summarized. So far we get the difference only for the considered input vector. So, to get the total difference for the entire data set means that we are summing the output over all input vectors where the number of inputs is denoted by *p*.

$$E = \sum_{k=1}^{p} \sum_{i=1}^{m} (d_{ki} - y_{ki})^2$$

We will proceed here considering the error as a function of the weights in the network and take a closer look at the partial derivatives of this function with respect to weights. It is clear that the output given an input is unambiguously determined by the weights and thus the error is too. So for some given number of weights *n*, the error is given by:

$$E = f(w) = f(w_1, w_2, \dots, w_n)$$

Since the error *E* is a function of all weights, we might consider, how changing a single weight *w* affects the error *E*. If *f* is a continuous and differentiable function, this is equivalent to observing the partial derivative of *E* with respect to w_i, $\partial/\partial \mathrm{w}_i$, in the point *w*. This partial derivative is a measure of how much the *E*-hyper surface is inclined in *w*-line in the considered point and is used to calculate the weight value that generates minimum error. By observing that, changing a connection weight affects the output error by changing the net-input of the unit the considered connection goes to. So, the changes in the net input are determined as derivative of quadratic function

$$\frac{\partial E_k}{\partial y_{\mathrm{ki}}} = 2(y_{ki} - d_{ki})$$

For the sake of simplicity of notation we skip the index *k*, and *p* is set to 1:

$$\frac{\partial E}{\partial y_{\mathrm{i}}} = 2(y_i - d_i)$$

Applying the chain rule for derivatives gives immediately all $\partial \mathrm{E}/\partial \mathrm{w}_{ji}$ for the weights in the last layer and noting that $y_i = y_{in_i}$

$$\frac{\partial E}{\partial \omega_{\mathrm{ji}}} = \frac{\partial E}{\partial y_{in_{\mathrm{i}}}} \cdot \frac{\partial y_{in_i}}{\partial w_{\mathrm{ji}}} = 2z_j(y_i - d_i)$$

Next step implies repeating the same procedure for the second last layer of nodes and also applying the chain rule:

$$y_{in_j} = \sum_j w_{ji} f\left(z_{in_j}\right)$$

$$\frac{\partial y_{in_i}}{\partial z_{in_{\mathrm{j}}}} = f'\left(z_{in_j}\right) w_{ji}$$

and summarize over all nodes:

$$\frac{\partial E}{\partial y_{in_{\mathrm{i}}}} = \sum_{i=1}^{m} \frac{\partial E}{\partial y_{in_{\mathrm{i}}}} \cdot \frac{\partial y_{in_i}}{\partial z_{in_j}} = f'\left(z_{in_j}\right) \sum_{i=1}^{m} w_{ji} \cdot \frac{\partial E}{\partial y_{in_{\mathrm{i}}}}$$

and output can be simply expressed by:

$$\frac{\partial E}{\partial y_{in_{\mathrm{i}}}} = f\left(z_{in_j}\right)\left(1 - f(z_j)\right) \sum_{i=1}^{m} w_{ji} \cdot 2(y_i - d_i)$$

If the Log-sigmoid is used we get:

$$f(z) = \frac{1}{1 + ke^{-z}} \Rightarrow f'(\mathrm{z}) = \mathrm{f}(z) \cdot \left(1 - f(z)\right)$$

The principle that has given the backpropagation algorithm its name, which makes it particularly calculation efficient, is that the contribution to the error

from the net-input to a node in some layer can be obtained from the corresponding contribution from net-input to the nodes in the next layer.
Since the contribution of the hidden nodes to the output error is calculated, it is easy to figure out that the weights to the hidden nodes affect the error in proportion to the activities of the nodes that are at the beginning of the connections. And with that said we get

$$\frac{\partial E}{\partial w_{\mathrm{rj}}} = x_r \cdot \frac{\partial E}{\partial z_{in_{\mathrm{j}}}}$$

For each additional layer of weights the last part of the procedure has to be repeated. With all the relevant partial derivatives that in this algorithm have been calculated, the gradient search is then utilized, i.e. letting the weight vector to move a little bit down in the direction in which weight-hyper-plane is mostly inclined. This direction can be found directly as direction with opposite sign, for the vector of all the partial derivatives. To go one step is therefore the same as to correct the weights in proportion to their respective contributions to the error. So this decreasing parameter which is used in the error gradient calculation is called learning rate. Optimizing weights and thresholds guaranties reaching a minimum point.

But there is one problem with this, the gradient search could converge to an error minimum which happens to be local, and thus there may exist a lower global minimum, with $E = 0$. This is prevented using a second parameter, called the momentum, together with the saved weight matrix from the previous iteration as an extra boost to jump out of local minima. There exist more efficient methods than simple gradient search, that use the second derivative of the error function where one can predict if one moves towards the local minimum.
Training data is processed for as many iterations as needed until the error rate reaches a predefined value which means that it is low enough. At the end of training all the data correlations in the network will be saved in the weight and threshold matrix.

4.3.2 Resilient backpropagation (RPROP)

Mathematicians are constantly working on developing new training algorithms in the belief that it is possible to the model a human brain. Although there already exists many training algorithms which are suitable for solving problems of different nature, one of these is the RPROP which can be described as an improvement over the backpropagation algorithm.

One of the problems with the backpropagation algorithm is that the error gradient often applies too large of a change to the weight matrix, and thus jumps over a possible low spot. RPROP on the other hand only sees to the sign of the gradient and discards big magnitudes. This means it is only important if the gradient is positive, negative or near zero. If the magnitude is small the RPROP won’t change any weight or thresholds matrix. If the magnitude is positive the weight or threshold will be increased by a constant, if it is negative it will be decreased by this constant. This constant has to be provided to the algorithm.

The RPROP uses special so called “deltas” that it calculates by itself, these “deltas” are not global, and there is one for each threshold and weight in the matrix. In the beginning each of these “deltas” is set to a very small number, and as the training succeeds they will increase or decrease with the help of the error gradients sign. By doing so each threshold and weight matrix will be individually adjusted and this is a huge advantage over the backpropagation.

5 Methods and Result

The description of applying neural networks in fault detection and diagnosis is given in this chapter. Our approach for analyzing neural network for monitoring purpose is quite straight forward. The idea has been to select neural networks with suitable architecture and training algorithms for solving our problems and to focus on applying a testing procedure to sampled data and the resulting outcome. Furthermore, a deeper analysis has been applied on the sampled residual signals of the moulding process.

5.1 Testing

Based on our knowledge of neural networks and the perception that control charts are represented with periodic patterns, we have decided to use the neural networks which are suitable for pattern recognition. Four tests have been performed with different network types and are presented here with the difficulties encountered during testing and the outcome of the testing.

5.1.1 Simulation environment

Due to the time constraints we have avoided developing software for testing neural networks. Hence, choosing adequate software with good testing performance has been important for getting sensible results. There were several workbenches for neural networks found, but most of them had not been updated for years and were too unstable and complicated to use. For this project two well proven alternatives have been chosen, Matlab and the open source Encog project.

Matlab is a very competent mathematical and simulation software tool. Some of the features are the simulation of neural networks using the Neural Network Toolbox [10]. The reason that Matlab is chosen it is because the Neural Network Toolbox has been used for educational purposes. One of the drawbacks of Matlab, besides that it is expensive, is that it mainly is a tool used for testing rather than as plug-in software to incorporate neural networks in software applications.

The Encog project [11] is a license free alternative and much like the Matlab Neural Network Toolbox contains a workbench for testing different types of designs on neural networks. The main difference is that the Encog software has plug-in classes written in both C# and Java. This provides possibilities to incorporate neural networks and train them inside one's own software applications. There are also support classes provided, that can normalize and process data for these neural networks.

Regardless of the test environment, there are no rules for determining the number of hidden layers in neural networks and the number of neurons in the hidden layers, but some vague thumb rules. For that reason a number of tests with varying numbers of neurons have to be carried out until the network performing with satisfying result was found.

5.1.2 Input analyzing

For the training procedure of our neural networks we have used a training and validation data set. The training set example data contains both correct, faulty values and desired output, and the validation set is consisting of data that is not a prior presented to the network and is not included in the training set and it is used to evaluate the capability of the trained network for generalizing. The data is fetched from a database and is represented with 16 bit signed integers in the range of -32768 to 32767. When fed into a control chart as in Figure 3.8 (page 11), repeating patterns can be visualized. Each line represents one of the significant machine components. When taking a closer look at the chart, it can be noted that some curves are following other curves and that some curves are their inversions. These correlations represented in the charts reveal that movements of all interacting machine components are highly synchronized. A classification of the line patterns into subgroups can be performed regarding to the line correlation to simplify the observing for the deviations and abnormalities. For examples on grouped signals by their correlation, see Figure 5.1 and Figure 5.2 on the next page.

A single machine cycle contains approximately 2400-2600 samples per channel and when plotted in Matlab lot of noisy parts on the lines can be discovered. This raises need to filter the input without cutting off or loosing important information in signals. The filtering method we have used here is a simple moving average of 10 values (i.e. $\sum_{i=1}^{10} x_i/10$). This procedure has cut the amount of samples down to a more reasonable level, from 2600 to 260, by a factor of 10.

As seen in Figure 3.8 (page 11) some of the lines follow each other, these have been identified as *Tilt_com_PP* and *PP_speed*, sorted in one group, and *Tilt_com_SP* and *SP_speed* in the other group. Additionally it can be noted that the *Tilt_fb_PP* and *Tilt_fb_SP* are feedbacks for their respective command (com) signals, as they appear as the inverse in the control charts.

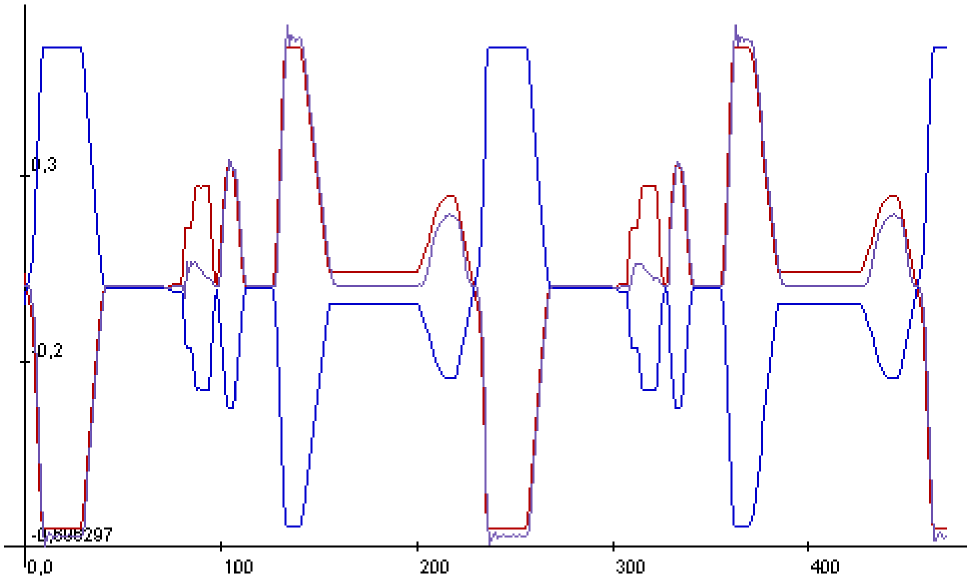

Figure 5.1 PP charts

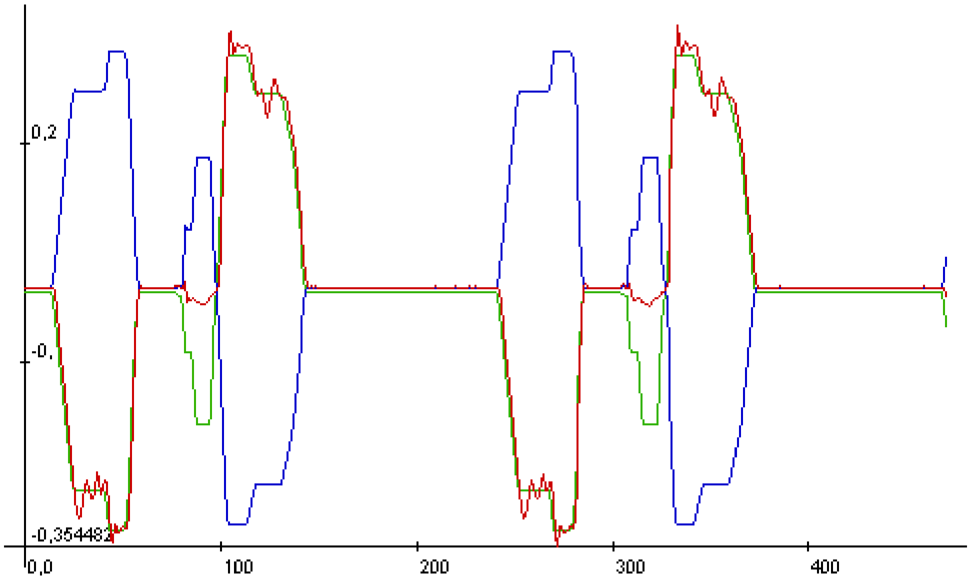

Figure 5.2 SP charts

The human brain can easily recognize the correlations between the 3 lines in each chart and the assumption is made that it is possible to teach the neural networks the correlations between the signals.

5.1.3 Test case 1 - Encog

The approach in this test is quite straight forward. The thought is to feed neural network with input consisting of all 16 channels at the same time. Based on this idea, we have built a feedforward network with 16 inputs and 1 output in Encog. The network consisted of 16 neurons in input layer, two hidden layers and 1 neuron in the output layer. Since there are no theoretical guidelines for determining the number of hidden neurons, we have experimented with 64, 80 and 128 neurons in the first hidden layer and 128, 160 and 256 neurons in the second hidden layer The training algorithm is resilient backpropagation and the activation function used for the network is the tan-hyperbolic function. The network is to output a 1 when there is an error and a 0 when there is no error. The training set in this test is built on data not containing faulty values or yet not identified faulty values. So there is a possibility that some errors in moulding processes will be accepted and learnt by the neural network as normal functioning parameters.

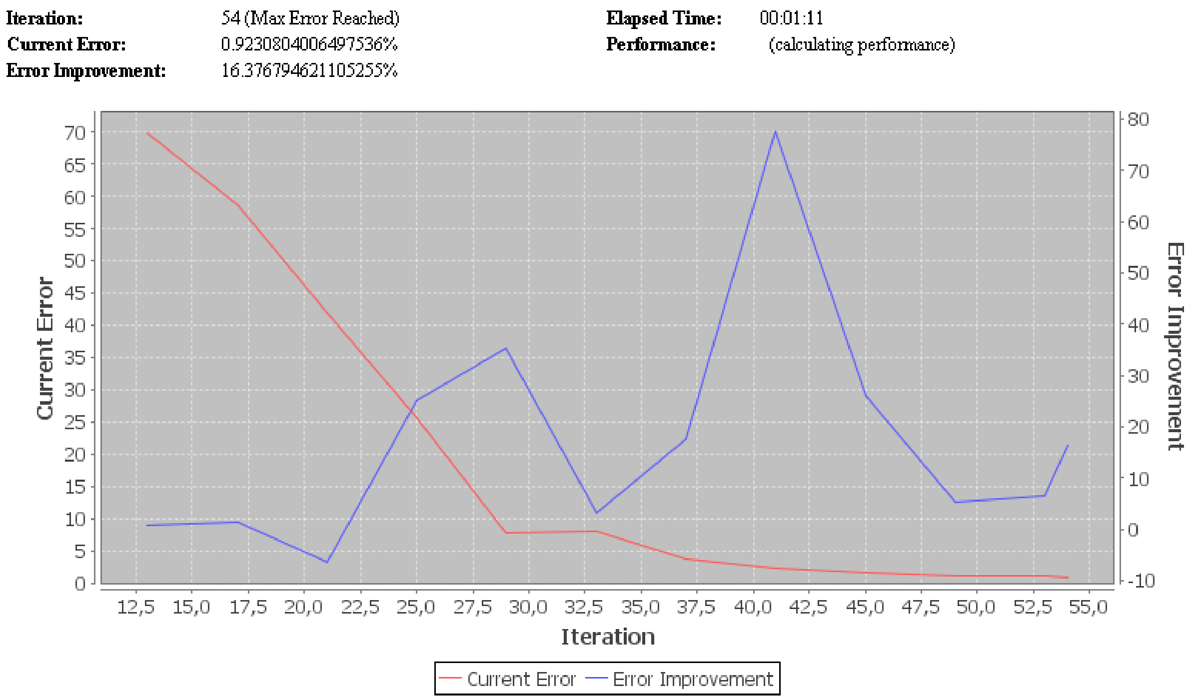

Figure 5.3 Training history

In Figure 5.3 we can see how the error (red line) decreases over time, we can also see how much the error decreases in percent over time (blue line). Although it looks like there is a lot of improvement and that the error is decreasing. The fact is that, the neural network only learns to adapt its weights and thresholds so that it always gives the desired output, regardless of the given input. This is done by adjusting most of the weights to values close to 0

(Figure 5.4). The reason that this happens is because the training set only consists of values which are supposed to be without faults. Due to this it couldn't be given a correct training which demands a big number of known faulty values and known correct values to teach the neural network how to recognize them.

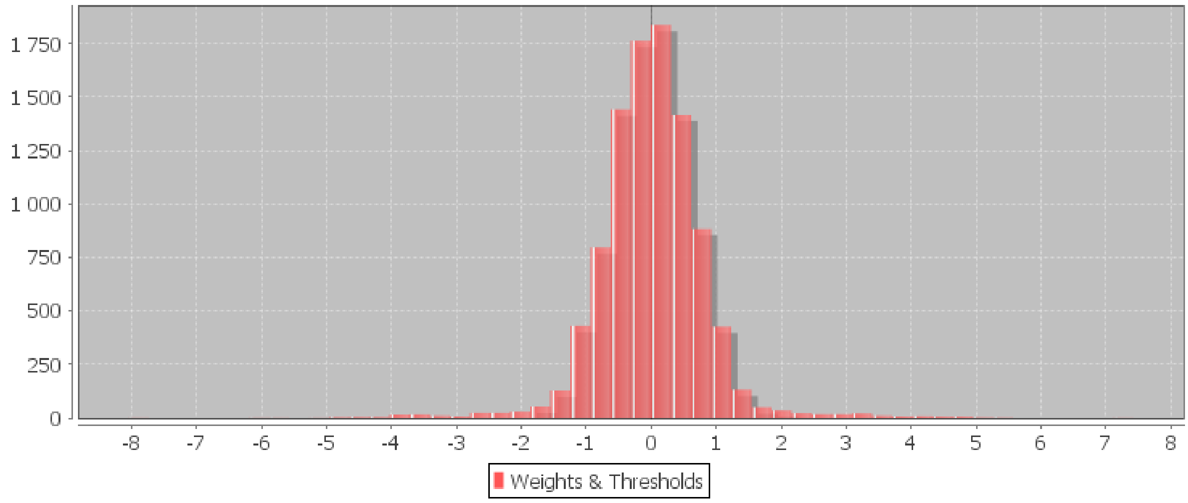

Figure 5.4 Weights diagram

5.1.4 Test case 2 – Encog 2

After the failure in the first test we assumed that the neural network could not handle all inputs at one time and therefore we might have a better chance by testing only one group of signals. So the second test case is similar to the test case 1 except that the input into the network consists of a training set built on only the three SP signals. However the end result from this test was exactly the same as in test case 1. This basically excludes supervised training from further testing since faulty inputs to train with cannot be provided.

5.1.5 Test case 3 - Matlab

In this test a Recurrent Neural Network called NARX (nonlinear autoregressive exogenous model) in Matlab is trained with all the 16 inputs. These kinds of neural networks are used in applications for prediction, like weather forecast and stock rate predictions in the financial sector, and our belief is that it can learn the normal behavior of the process through the fed input, meaning using unsupervised training procedure. If it can learn this and then compare its predicted output with the input and see if they match we will get a result. Multiple tests with different number of hidden neurons have been performed in order to find the optimal architecture. The best performing architecture was one with 512 neurons in the hidden layer, and although it took about 50 minutes running on full speed on a computer with 4 processor cores at a speed of 4GHz for Matlab to finish the test and that it occasionally used up to 9.6GB of RAM memory, the learning procedure failed. This gives the conclusion that a single big NARX network is not able to handle all 16 inputs simultaneously.

Since these two drawbacks with testing all 16 inputs, it was realized that some simplifications have to be done in terms of distinctly grouping or even pre-processing the inputs and in such a way lower the complexity grade in learning of the recognition of patterns in the inputs.

5.1.6 Test case 4 – Matlab 2

For this test a NARX network was built consisting of 300 neurons in the hidden layer with two delay layers. In the training of the neural network the input x(t) consisted of pressure plate speed signal, channel 13 in Table 3.1 (page 10) and the current output y(t) was compared with delayed signals x(t-2) and thereafter the error was calculated by taking the squared difference between them.

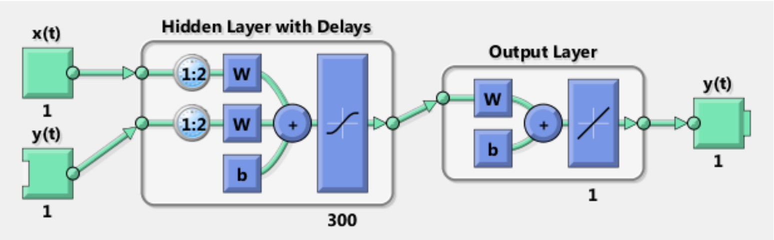

Figure 5.5 The NARX network tested.

This neural network took 10 seconds to train and it was done in 14 iterations and the error was calculated to 7.264e-5. As it can be seen in Figure 5.6 the network almost perfectly responds to the input of a correct PP signal. This approach is giving us optimism about recognizing patterns with neural networks.

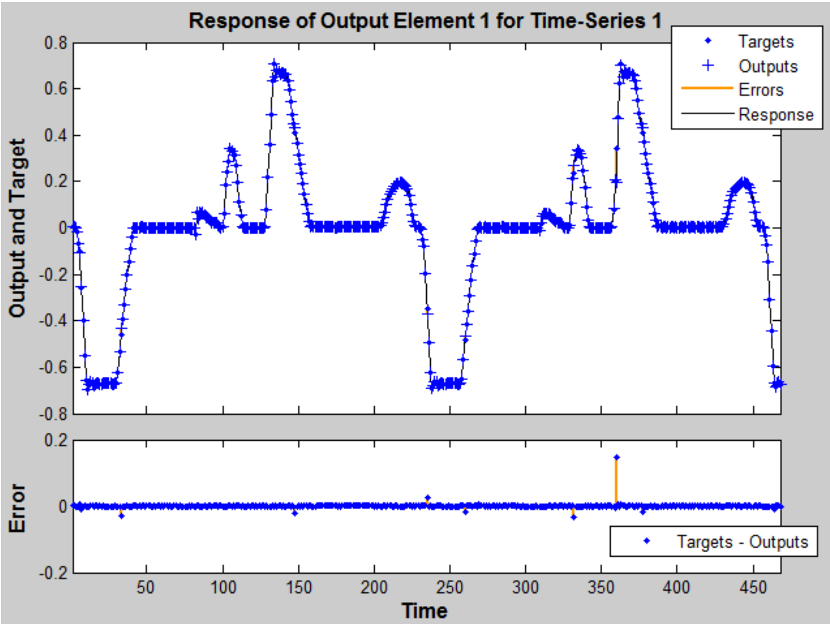

Figure 5.6 PP input into PP trained NARX

Since we lack the knowledge of presence of faults in the signals we decided to train the PP trained NARX network with a totally different signal in order to test the networks capability to detect errors in patterns or deviations in the signals. For that reason the neural network was fed with a swing plate speed signal (Channel 14) into the NARX network trained for PP speed signal, and it calculated the error to 1.32e-2. As seen in Figure 5.7 the network is capable to detect large signal deviations, but not capable to detect small signal deviations like the bump in Figure 3.9 (page 12). It can be noted that the NARX network does not seem to detect all the differences between the PP and the SP signals.

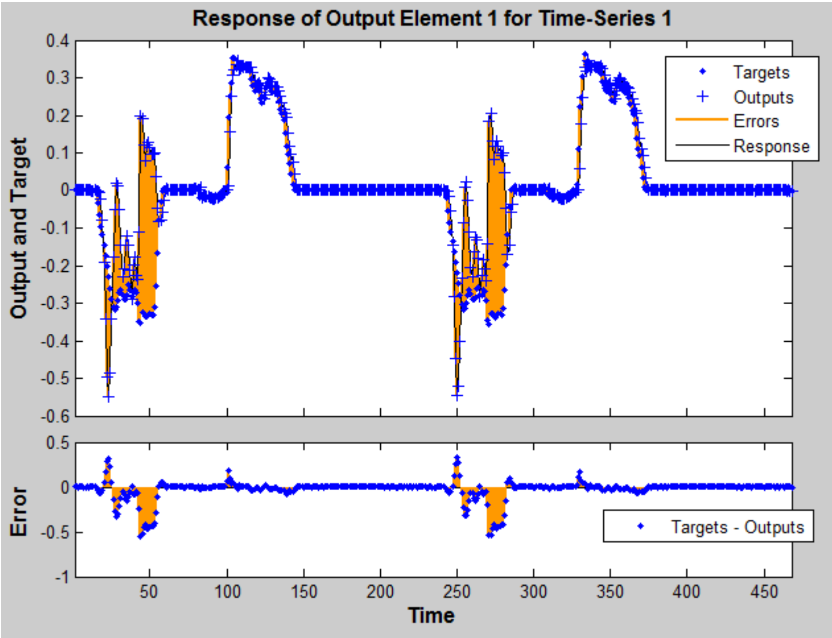

Figure 5.7 SP input into PP trained NARX

The same PP trained NARX network is fed with a modified input of PP with an offset of +0.2 over the entire period, and it calculated the error to 8.25e-2 which can be seen as a positive sign of that the NARX network can in fact detect deviations in the signals. This offset of +0.2 is considered as a big error, so this sensitivity is not satisfying referring to the described fault in chapter 3.3, Figure 3.9 (page 12). As it can be seen in Figure 5.8 the amount of detected faults is far from acceptable. This is a major drawback for the NARX network and leads us to the conclusion that it is not the right network for error recognition.

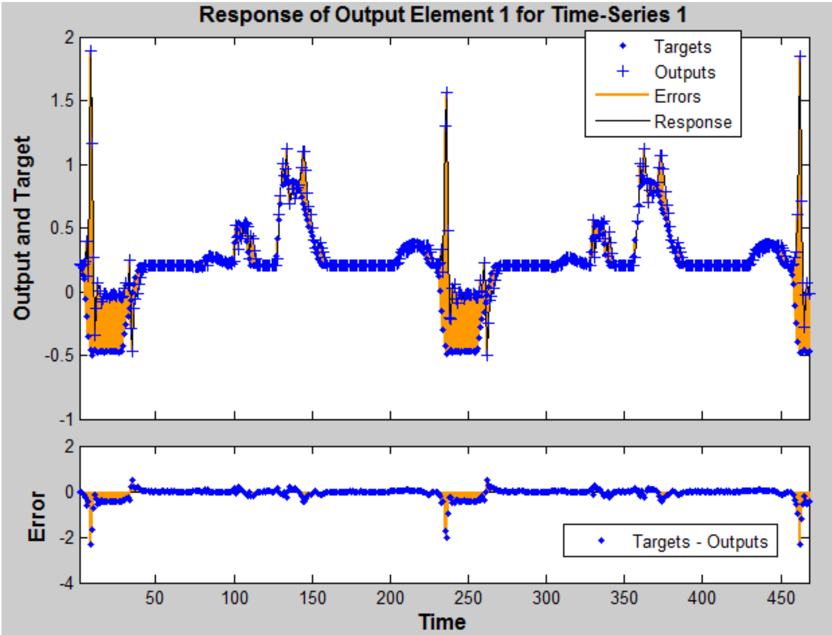

Figure 5.8 PP input with +0.2 offset error into PP trained NARX

Derived from this test, it can be said that the NARX network basically just learns to mimic the inputs to the output channel, small differences from the "right" input signal can be noted, but not satisfactorily enough considering that it should be used in error detection on expensive machines functioning.

5.2 Results

In theory neural networks are the adequate tool for control chart pattern recognition. However there are some characteristics of the control charts that limit the possibility to use neural networks for fault detection. In our work to investigate for possibilities of neural networks to detect deviations and anomalies in moulding processes on DISAMATIC machines, problems have emerged regarding signals and process dynamics.

5.2.1 Identified problems

The following problems have emerged during this case study:

1. Lack of proper training data

To properly train a neural network we need a training data set for supervised training that consists of a well-known errorless process cycle data, and also of cycle data for each known error and its behavior. We are lacking the knowledge of error existence in the sampled data. We are unable to collect such data since our work is based on sampled data of a machine already incorporated and optimized in an industrial environment and modifications to generate errors are not allowed. For that reason in the test cases with supervised training we have performed, there is a possibility that the neural networks are trained with errors in data and classified as correct machine behavior. So the errors are never discovered and hence the weariness of machine parts and that might jeopardize machine condition.

2. High sampling frequency

The high sampling rate produces approximately 2400 - 2600 sample values per channel for a single machine cycle period. This big data amount can be one of the reasons that neural networks fail to find and learn correlations represented in the data. The data amount guarantees the information richness and hence the presence of the noise.

3. Process periods are not consistent

The cycles period changes i.e. they are not of a fixed length and the reason why is unknown. The assumptions are made that the period length is changed every time some process parameter is changed due to some optimization of the machine settings carried out by technicians, control signals synchronizing the moving parts by adjusting the corresponding parameters, or some automatic calibration function is run on the machine or it can be varying sampling frequency. For this reason the neural network would have to be retrained. Because of the period varying between 2400 and 2600 samples the neural network has failed to learn and recognize the period pattern even though the data of a single channel is fed into the network.

In Figure 5.9 the varying period length can be seen. In this control chart the first 6 cycles had the same cycle length of 2494 samples, but on the 7th cycle the period length is shrunken with approximately 25 samples, then it continues to shrink and on the 10th cycle it shrinks with approximately 150 samples, which is more than a half second faster per cycle.

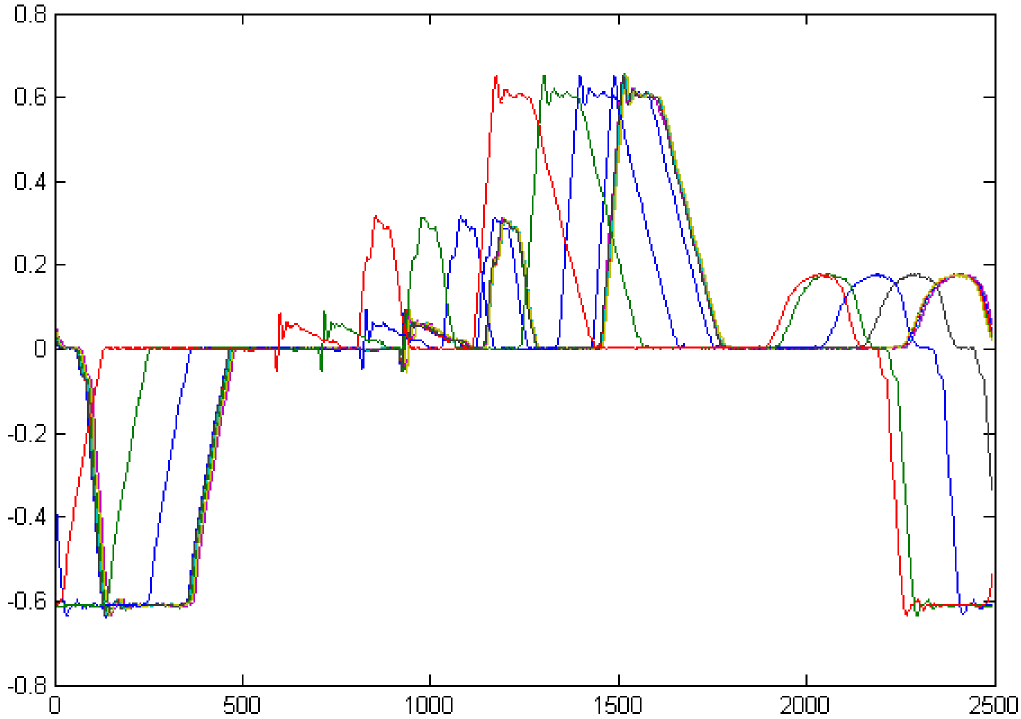

Figure 5.9 10 cycles plotted, the last 4 are shorter than the first.

4. Difficulty to make distinctions on noise from response signals.

The high sampling frequency guarantees the information richness in the sampled data and hence the presence of the noise which can be visualized and detected in charts. Even though chart lines seem to follow a regular pattern there is a lot of signal noise and distortions. Although, the human brain can distinguish these distortions and detect small faults in charts as the one in Figure 3.9 (page 12) a computer with a model of human brain is not capable of doing the same. The human brain is still tremendously superior to the computer in terms of detecting and classifying these details. By filtering the input data, the risk for cutting out the faults together with noise is taken and possibility for discovering errors and preventing the machine failure is lost.

5. Possibility to train in faults as correct behavior.

When some optimization of the machine settings is carried out by technicians, the neural network would have to be retrained. If optimizations are done frequently then using supervised training procedure for training the networks is less suitable. Using the online training procedure can solve this problem but the risk of training in the faults as correct behavior is imminent. If for example there is a model change and at the same time a valve breaks, then the neural networks trains as if the machine is running correctly even though it is not, and thus allowing a broken valve to exist.

6 Conclusion

To introduce automated monitoring and fault detection of a sand moulding process on the DISAMATIC molding machines, and to bring the neural networks' amazing abilities in detection and classification into the process has proved to be more difficult than expected. Problems in pattern recognition have been discovered and are presented in the results.

The conclusion of this case study is that software consisting solely of neural networks is not suitable for the monitoring purpose on a moulding machine. The main reason is the complex process dynamics which are represented in the inputs. Some pre-processing procedure on the inputs is required before they can be fed into the neural networks, such as signal processing using some transform technique.

The distinction on malfunctions and errors which can be represented in the charts in the form of small bumps has shown to be difficult and even more difficult to distinguish and to classify whether it is a correct signal, distortion signal or a signal of machine behavior.

The varying period length has had a huge impact on pattern recognition and for that reason the neural networks have not been able to learn a fixed number of patterns representing normal machine behavior, like kind of a fingerprint and to find the miss-matching ones.

The indeterminate but high sampling rate gives the information richness, includes the noise into the sample data and results in the periods of varying length. And this brings even more confusion in network learning of normal machine behavior.

6.1 Comments

We would like to emphasize, despite all the setbacks we have encountered that this work has been very interesting and knowledge enriching. To get a glimpse into the field of artificial neural networks, not least the potentials they possess have broadened our horizons.

We started this work with big enthusiasm and a belief in obtaining a functioning solution. The first problem we encountered was the lack of knowledge for signal processing, since it is required for data analyzing. We consider this as necessary for future work with development on the monitoring application.

But we still believed the neural networks could recognize and classify the error patterns so we proceeded with testing recognition on the existing data. And then we lacked a list over machine errors represented in either graphical or sampled form. DISA could not provide us with such a list because all error reports are kept in thesis form so retrieving those would be tedious and time consuming task. But it would be good idea and to some extent necessary to do so. Additionally, databases should be installed on as many moulding machines as possible in order to collect as many samples of malfunctioning and breakdowns as possible.

The obtained results in our work might not match our initial expectations but we are overall satisfied with them since they are only at pre-study level, and there is a lot of work left to reach a fully functional solution. We have discovered and specified problems that are the reason the neural networks failed to recognize the error patterns. Our opinion is that the discovered problems need to get solved first and due to time and knowledge constraints we have not been able to do that. We have got some ideas and proposals for solving those problems and we have presented them for the future work.

6.2 Tips and ideas for future work

In this chapter some ideas of how the future work on a sustainable solution for neural networks in monitoring applications could be directed, are presented.

An alternative approach suitable for future investigation in this area could be the data mining which is a popular technique for classification of large data quantities. With this technique further characteristic in sample data might get revealed and clarify the pattern recognition even more.

Due to the time and knowledge constraints we have not been able to implement the ideas for possible solutions and solve the problems that are presented in results. The main weight in future work could be assigned to solving those problems. Based on the experience obtained in this case study, possible solutions on these problems and approach ideas are proposed.

1. Lack of proper training data

In order to get proper training data set a good idea is to engage the engineer who is currently working on fault detection and diagnosis at DISA to set up the database over errors based on diagnosis history record. Then the representation of the errors can be identified in the sampled data and the training set containing those can be built.

2. High sampling rate

Since the high sampling rate is needed to get a good and precise resolution of the charts they also bring a lot of noise. One way to manage this noise is by using some filter which can guarantee that all the high resolution of the signals is left, but the noise disappears.

3. Process periods are not persistent

By getting a fix period length for all process cycles could make mapping the error patterns in the sample data using neural networks more plausible.

4. Difficulty to make distinctions on noise from response signals.

It seems to be an idea to process signal with Fourier and discrete wavelet transform. When the representation for all the errors have been mapped a comparison of error frequency components can be made against the noise frequency components. This procedure can greatly facilitate the pattern recognition.

5. The risk of training in faults as correct behavior.

The application of the neural networks with alarming function on error discovery reduces the risk for training in the errors or malfunctions on the machine as correct behavior. This requires that the engineer check the machine condition for errors before making any adjustments or replacements.

7 References

1. Bertil Thomas, Studentlitteratur 2003. Framtidens Intelligens
2. Helge Malmgren, Studentlitteratur 2007. Inlärning och minne i neurala nätverk
3. Christopher M. Bishop, Oxford 1995. Neural Networks for Pattern Recognition
4. http://www.heatonresearch.com/online/programming-neural-networks-encog-cs 2011-03-29
5. Simon Haykin, Prentice-Hall Inc 1999. Neural Networks - a comprehensive foundation
6. http://www.neuralpower.com/technology.htm 2011-05-06
7. http://youtu.be/VQRuBbfRe7o 2011-06-11
8. http://www.disagroup.com 2011-06-11
9. http://www.disagroup.com/sites/disa/content/equipment/molding_solutions/vertical_lines.aspx 2011-06-11
10. http://www.mathworks.com/products/neuralnet/ 2011-06-11
11. http://www.heatonresearch.com/encog 2011-06-12