VTT INFORMATION TECHNOLOGY

RESEARCH REPORT TTE1-2000-27

MODUS-Project
Case Study ODT2

Off-Line Analysis and Prototyping of Paper Machine Drive Monitoring System

Version 1.1-1

6.11.2000

Esa Rinta-Runsala

Version history

Version	Date	Author(s)	Reviewer	Description
0.1-1	15.9.2000	E. Rinta-Runsala	J. Kiviniemi	First draft
1.0-1	2.10.2000	E. Rinta-Runsala	J. Kiviniemi	First version to the steering group
1.1-1	6.11.2000	E. Rinta-Runsala	I. Karanta	Corrected version

Contact information

Esa Rinta-Runsala
VTT Information Technology
P.O. Box 1201, FIN-02044 VTT, Finland
Street Address: Tekniikantie 4 B, Espoo
Tel. +358 9 4561, fax +358 9 456 6027
Email: Esa.Rinta-Runsala@vtt.fi
Web: http://www.vtt.fi/tte

Last modified on 6 November, 2000

Abstract

The report concentrates on a specific case study of a single paper machine. Data from the paper machine consists of motor speed and torque measurements taken along the machine as well as supply power and mains voltage measurements. The data is analyzed off-line and some interesting abnormal phenomena in the signals are described.

A review of state identification methods is presented, including cluster analysis, decision trees, their fuzzy versions, multi-layer perceptron and radial basis function neural networks. Additionally, self-organizing map (SOM), and adaptive resonance theory (ART) neural networks are presented in more detail and they are prototyped on monitoring of the paper machine. The performance of the prototypes in detecting abnormal phenomena in the data as well as demands and properties of both methods are compared. Finally, a suggestion for implementation is made for using SOM in on-line monitoring. Along the suggestion there are some notions about the limitations of using SOM in monitoring.

Tiivistelmä

Raportissa esitetään yksittäisen paperikoneen tapaustutkimus liittyen koneen monitorointiin. Analysoitavana datana ovat paperikoneen moottoreiden nopeuden ja momenttien, sekä syöttöjännitteen ja -tehon mittaukset. Raportissa kuvataan analyysi ja datassa havaitut monitoroinnin kannalta mielenkiintoiset ilmiöt.

Monitoroinnissa käytettävistä tilojen tunnistusmenetelmistä esitetään lyhyt katsaus, jossa käydään läpi klusterianalyysi ja päätöspuut sekä niiden sumeutetut versiot, multi-layer perceptron- ja radial basis function neuroverkot. Lisäksi itseorganisoituva kartta ja adaptiivisen resonanssiteorian neuroverkot esitetään yksityiskohtaisemmin ja näitä menetelmiä prototypoidaan tapaustutkimuksen paperikoneen monitoroinnissa. Prototypoitujen monitorointijärjestelmien antamia tuloksia, vaatimuksia ja muita ominaisuuksia vertaillaan ja lopuksi esitetään implementaatioehdotus itseorganisoituvaan karttaan perustuvasta monitorointijärjestelmästä. Ehdotuksessa esitetään myös menetelmän monitoroinnille asettamia rajoituksia.

Table of Contents

Nomenclature

χ	: grade of membership
σ	: standard deviation
$\phi(.)$	: kernel function
$\Re$	: set of teal numbers
2D	: 2-Dimensional
ART	: Adaptive Resonance Theory
ART1	: Binary ART
ART2	: Analogue ART
BMU	: Best Matching Unit
DT	: Decision Tree
LTM	: Long Term Memory
MLP	: Multi-Layer Perceptron
NN	: Neural Network
PC	: Principal Component
PCA	: Principal Component Analysis
QE	: Quantization Error
RBF	: Radial Basis Functions
SOM	: Self-Organizing Map
STFT	: Short Time Fourier Transform
STM	: Short Term Memory

1 Introduction

This report is a continuation for a previous report on Modus project ODT2 case study [Rinta-Runsala 2000]. The previous report presented some requirements and suggestions for paper machine drive control monitoring as well as a few feature extraction methods.

This report continues the case study but it concentrates on a specific machine. The specific case of this report is presented in Chapter 2. In Chapter 3 the feature extraction methods from the previous report are used for data from the studied paper machine and the data is analyzed based on the features extracted. After that a few state identification methods are introduced in Chapter 4. The introduction to state identification methods includes a review of common methods and more specific presentations of self-organizing map and adaptive resonance theory neural networks. Self-organizing map and adaptive resonance theory networks are prototyped in Chapters 5 and 6, respectively. The prototyping is based on the data presented. Chapter 7 compares the prototyped methods in many respects and Chapter 8 presents a suggestion for implementation of monitoring system. Chapter 9 summarizes the report.

2 Presentation of the Specific Case Study

This report concentrates on a specific paper machine instead of a wide variety of paper machines as the previous report [Rinta-Runsala 2000]. The paper machine studied is monitored with 50 measurements of its drive controls. The measurements are power and mains voltage of the machine and speed (in rotations per minute) and torque (in percents of nominal torque) of 24 rolls in different sections along the paper web with sampling interval of 0.5 seconds. A list of the signals measured is in Appendix A.

The measurements were conducted during six days in three series. The series are referred in the text with Roman numbers I, II, and III. Series I consists of 118,397 observations, were observation is an event with measurements of all the 50 signals. The series I is about 16 hours. The series II is the longest, 300,000 observations, lasting about 42 hours. The series III is 147,145 observations long with about 21 hours span.

There were no known malfunctions during the measurement period but some unstable abnormalities can be seen in the measurements. These abnormalities are studied in more detail in Section 3.1.

The main use for the data collected is to construct a state identifier to detect abnormal situations. The definition of an abnormal situation is quite obvious; the situation differs from a normal situation. An abnormal situation may be undesirable or harmful but it may also be totally harmless. The task of the identifier is simply to detect abnormal situations, not to classify them into harmful and harmless ones. This kind of an identifier is used to keep track on events in the process, and the records can be used in the future for search of reasons for some unexpected events as stated in [Rinta-Runsala 2000].

3 Off-Line Analysis of Data

Off-line analysis of the data is a basis of the monitoring system's construction. The analysis is conducted using visual examination of the data as well as more sophisticated feature extraction methods like principal component analysis, Fourier, and wavelet analysis. First, the data is described in terms of phenomena found and the importance of the phenomena is considered. Second, the ability of the features to detect the important phenomena is considered and the suitable features are selected for further analysis.

3.1 Description of Data

The data studied is as a whole quite normal in its nature. There are usual rapid decreases, increases, and instabilities due to breakages and run-ups of the machine, slower changes in the values due to set point changes, some periodicities due to controllers, and noise. Moreover, there are also some unexplained periodicities and spikes.

The circumstances change a little bit during the measurements, since a part of the measurements are conducted when the center drive spool is not run. This is due to a certain paper type made on that period. Additionally, the machine calender was not used during the measurements and its measurements, 4 signals altogether, are excluded from further analysis. There are also periods of missing values in some signals.

Since the objective of the study is to detect unexpected phenomena, the study is concentrated on the abnormal periodicities and spikes. Additionally, there can be abnormal rapid decreases, increases, and instabilities, which do not result from breakages or run-ups. However, due to the fact that no information about the times of occurrence of the breakages or run-ups is available, all the short time changes are taken as results of breakages or run-ups. Examples of phenomena in signals are presented in Figure 1.

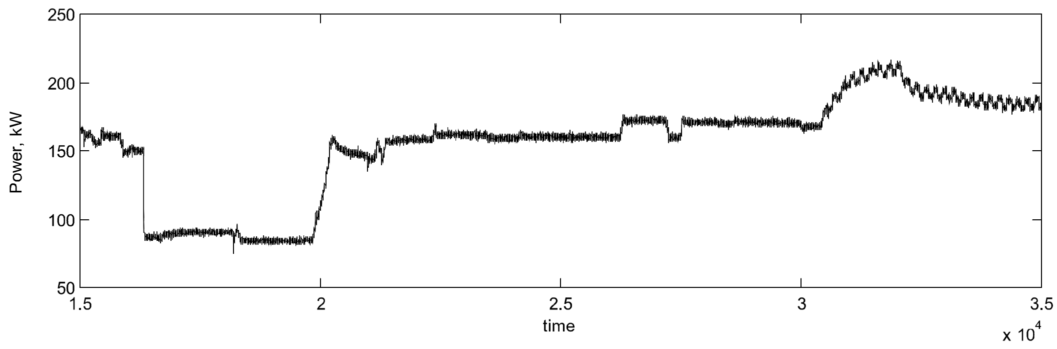

Figure 1. Some phenomena detected in measurements. The time scale is in observations with sampling time of 0.5 seconds. On the left there is a (normal) rapid decrease due to a web breakage and later on a short time increase caused by a run-up. On the right there is an unexplained abnormal periodicity with a period of about 100 seconds.

In preliminary analysis, the data was analyzed manually to find the possibly interesting abnormal phenomena. After preliminary analysis, the phenomena found were discussed with experts to determine their severity and importance for the automatic monitoring. The most important abnormal phenomena detected are listed in Table 1 and examples of them are presented in Figures 1-6. The goal of the monitoring system is to detect these kinds of phenomena. However, there are some limitations for the detection. For example, the phenomenon P4 is present almost all the time, although it is abnormal in the sense that it is an undesired state. Thus, phenomenon P4 cannot always be treated like an abnormal state.

Table 1. The most important phenomena detected in the studied time series.

Code	*Description of the phenomenon*	*Possible explanations for the phenomenon*	*Signals in which the phenomenon was detected*
P1	Periodicity with a period of about 100 seconds. Lasts for about 4 hours.	Resonance due to increase in speed or change in paper type.	1 (Figure 1), 40, 44, 46, 48, 50
P2	Several spikes following each other at a rapid pace. The magnitudes of the greatest spikes in signal 10 are about 20 % of the average value. The phenomenon lasts for 1 hour.	Series of web breakages[1] due to too high a tension on the reel drum.	9, 10 (Figure 2), 11, 12, 13, 15, 16, 17, 18, 35, 36, 37, 38, 39, 40, 41, 42, 45, 46, 47
P3	Slow periodicity (period ca. 8.3 minutes) in the yankee section applicator roll. Lasts for over 6 hours.	Deterioration of some mechanical part of the machine.	14 (Figure 3)
P4	The varying periodicities detected especially in the coating section but also through the whole machine.	Slip or other non-optimal behavior of the machine.	15 (Figure 4), almost in every signal
P5	Slow periodicity in one-nip calender with period of ca. 3 minutes. Lasts for ca. 1.4 hours.	Mechanical resistance of the roll.	20 (Figure 5)
P6	Spikes occurring in uniform time intervals of ca. 7 minutes. Lasts for ca. 4 hours.	Slip of paper.	46 (Figure 6)

[1] The reason for spikes of web breakages to be interesting enough to be observed is their unusually great magnitude.

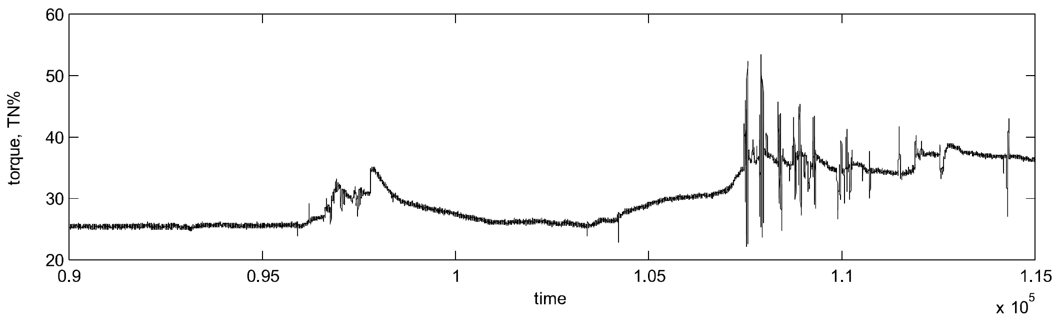

Figure 2. The series of spikes (P2, at the right side of the figure) on torque of a dryer cylinder (signal 10). The time scale is in observations with sampling time of 0.5 seconds.

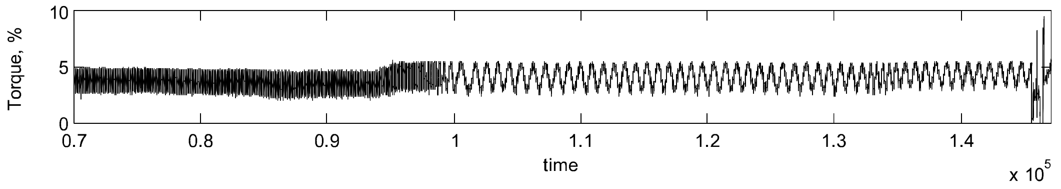

Figure 3. The slow periodicity (P3, starting at time 100,000) of torque of an applicator roll (signal 14). The time scale is in observations with sampling time of 0.5 seconds.

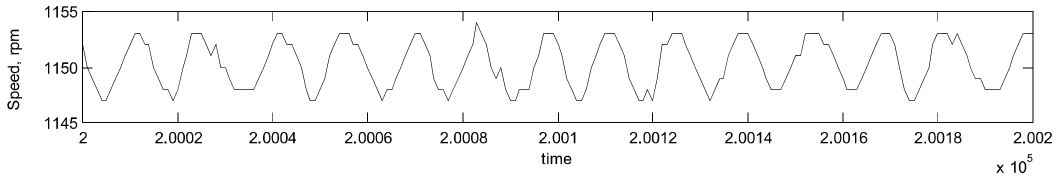

Figure 4. The periodicity of speed of a roll (P4, signal 15) in the coating section. The time scale is in observations with sampling time of 0.5 seconds.

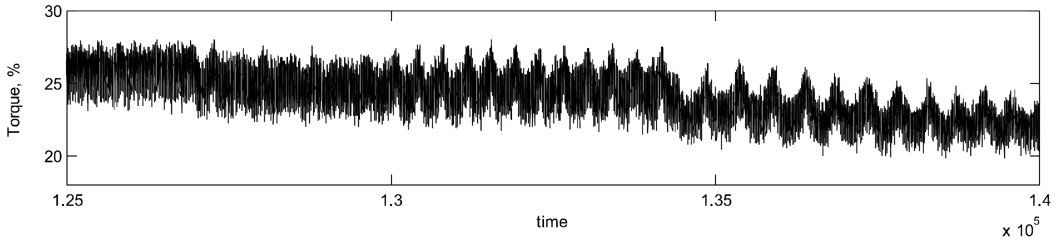

Figure 5. The slow periodicity in torque of a roll in a one nip calender (P5, signal 20). The time scale is in observations with sampling time of 0.5 seconds.

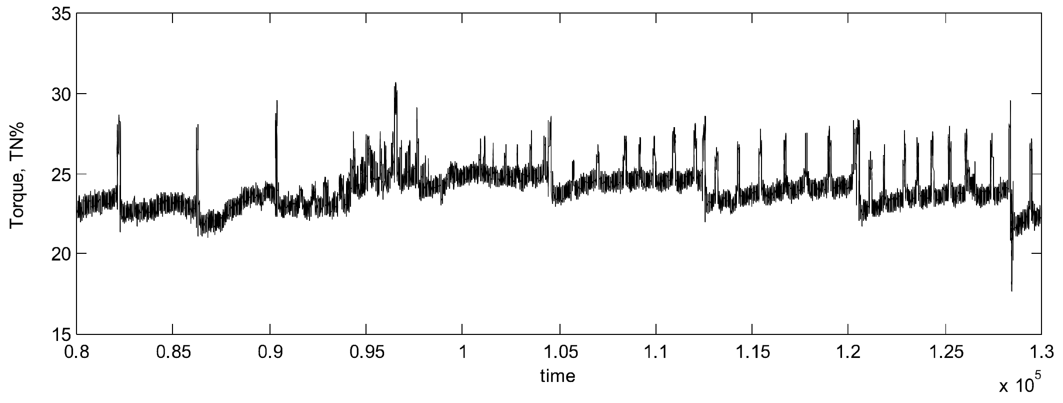

Figure 6. Spikes with uniform intervals in torque of a dryer section roll (P6, signal 46). The time scale is in observations with sampling time of 0.5 seconds.

3.2 Feature Selection

The idea behind the automatic monitoring of the states is to use the features presented in [Rinta-Runsala 2000] to discriminate the abnormal states P1-P6 (Table 1) from the normal states of the paper machine.

Since there are as many as 46 signals to monitor, the first thing to consider is the reduction of the dimensionality of the data. Principal component analysis (PCA) can be used for this purpose. However, comparing the correlation matrices of different days in Figure 7, one can see that the correlation matrix of the data is not invariant. This means also the principal components of the data change with time and the whole data set cannot be presented reliably with principal components (PC) estimated from one fixed period of time. Thus, there is no guarantee the future measurements would have the same PCs.

The variability of PCs is true for the whole process but there are some parts of the correlation matrices in Figure 7 that seem to be more invariant. One option is to represent the signals with invariant correlation with PCs and leave the variant signals untouched. Unfortunately, there are also serious disadvantages in this approach. In Figure 7, the signals of the yankee section seem to have relatively high correlation and thus one could assume PCA to be a proper technique to apply. However, in the state P3 only torque of the applicator roll deviates from the normal situation and PCA would distort the detection of this phenomenon.

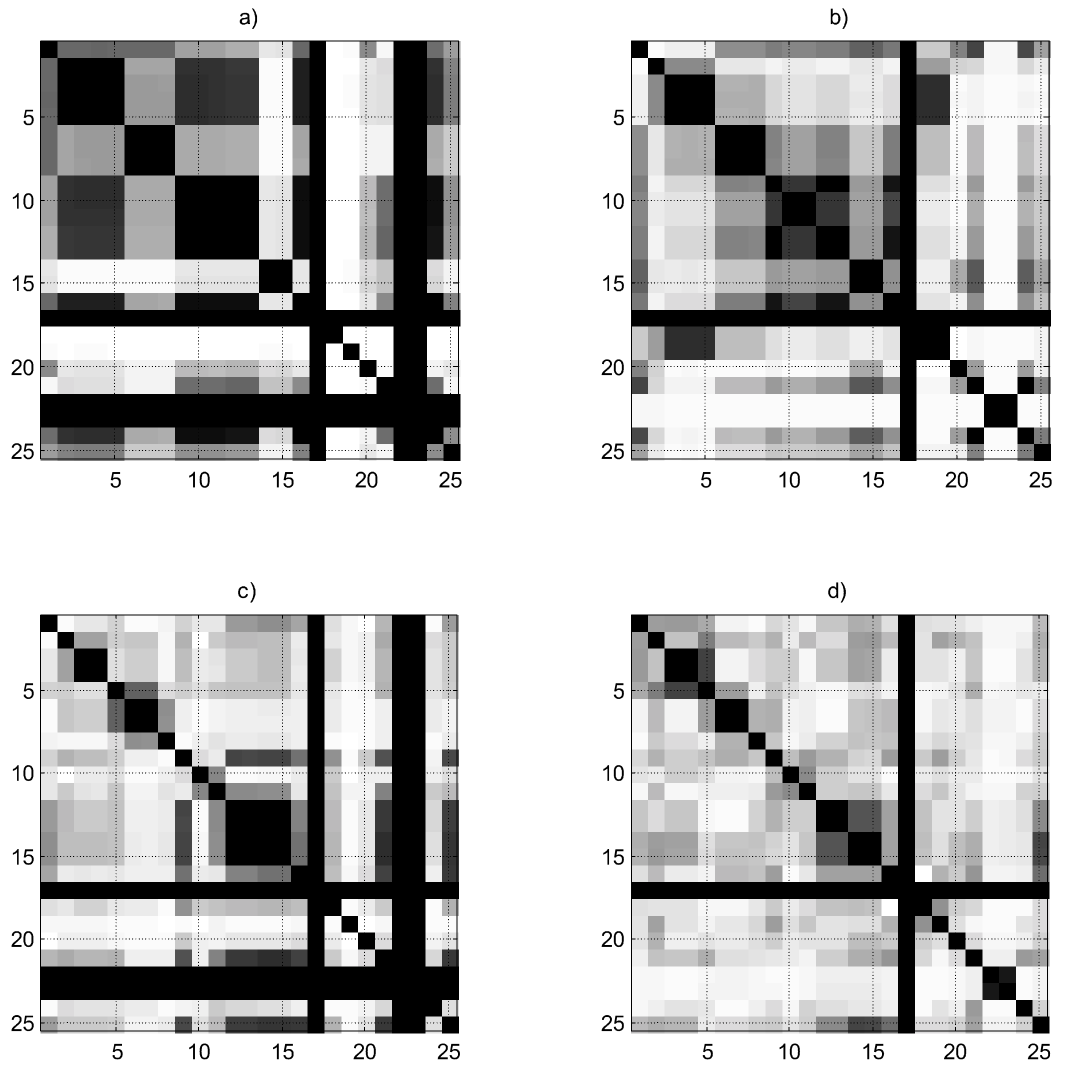

Figure 7. Graphical representation of the correlation matrices of a) the speed measurements in the series I, b) the speed measurements in the series III, c) the torque measurements in the series I, and d) the torque measurements in series III. The numbering of the measurement stations begins from the forward end of the paper machine. The darker the shade, the higher the correlation between the measurement points. If the correlation matrices were invariant with respect to time, the matrices a) and b) would be similar, as well as matrices c) and d). However, this is not the case and the correlation matrix cannot assumed to be invariant. The yankee section mentioned in the text consists of the measurements numbered 10 - 13 in the matrices.

There were only a few abnormal states during the period of study, though experience has showed there are many more to detect. This gives rise to the fact that the monitor cannot rely on redundancy of the measurements as the previous example of the state P3 showed. Although the signals act similarly in normal situations, they can differ from each other in faulty or abnormal situations. Thus, the measurements will be studied in further analysis without dimensionality reduction.

The next thing to consider is how to detect the abnormal states by monitoring the measurements, and which features would help in this task. The states P1, P3, P4, P5 are related to periodicities so Fourier, short time Fourier or wavelet analysis are natural choices for their detection. On the other hand, states P2 and P6 are essentially rapid changes in the signal and thus deviation from moving average, trends or wavelets could be used.

Simple Fourier analysis of the signals is discarded as a monitoring tool because of its lack of temporal resolution. The second option is short time Fourier transform (STFT). However, the assumption of occurrence of both very slow periodicities in hour scale and very rapid changes in scale of seconds limits the usefulness of STFT as stated in [Rinta-Runsala 2000]. Instead, wavelet analysis can extract the short time features with high temporal resolution and long time features with lower resolution. This fact favors the use of wavelets for monitoring of periodicities.

Since wavelets can also be used in monitoring rapid changes, they are an important aid in monitoring. Wavelets have also other advantages. To achieve some generality for the monitoring the features used cannot depend on the machine monitored. This means that in addition to machine-dependent phenomena presented the same features have to be able to be used to detect many other phenomena. Due to the multi-resolution properties of the wavelets, they are quite useful also in this way. By using all the wavelet coefficients of the signals one is able to detect a wide range of phenomena in the process.

These facts result in the selection of wavelet-coefficients as the features to be monitored. The mother wavelet function used is Daubechies-4 [Daubechies 1992]. Daubechies-4 was chosen, because it suits well approximating the behavior of signals with peaks and troughs.

Further, advantages of wavelets as monitoring features is the ease of automatization of model identification. One can assume that by using wavelets the monitoring system can detect the states desired, so the ease of automatization objects the use of other features. However, if an adequate monitoring needs additional features, the importance of an automatization has to be reconsidered.

Finally, attention must be paid to the translation invariance of the features. This means that a phenomenon can be detected independent of its time of occurrence within the monitoring window. To achieve this the features, i.e. all the wavelet coefficients, are calculated in 1-second interval for the teaching data. Thus, they cover the whole range of occurrence within the phenomena detected, and later on a similar phenomena can be identified independent of their time of occurrence within the monitoring window.

The wavelet-coefficients used in further analysis were calculated with Matlab and a freeware wavelet function library Wavelab for Matlab [Donoho et al. 1999] written at the Department of Statistics at Stanford University. Because of the limitation to monitoring window lengths of powers of two posed by the wavelet algorithm, the monitoring window length is set to 8,192 observations. This corresponds to 4,096 seconds, i.e. time period of approximately 68.3 minutes. The window length 8,192 = 2^{13} observations gives total of 13 wavelet-coefficients per signal, i.e. scales of 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048 and 4196 seconds. This gives a total number of 598 for the features to be monitored. As a result, the input pattern for the state identification methods consists only of wavelet-coefficients and not the signals themselves.

4 State Identification Methods

State identification is essentially assigning of a single state to a more general group of states. The state is represented by the features mentioned in the previous chapter and together all the features form a pattern, or a feature vector. The task in state identification is to classify a newly observed pattern to a class of patterns. However, before the classification can be done, the previously observed patterns have to be clustered to form the classes for the classification. It is important to notice this difference between clustering and classification. Clustering means determining of previously unknown clusters and classifying means assigning a new observation to a known class. In some cases, e.g. with neural networks, the clustering is also known as training, or teaching, of the network. In training, the network is presented with training data, from which the network learns the implicit connections needed for clustering of the data.

Since the states of the paper machine are not known *a priori* both clustering and classification techniques are studied in this chapter. Some clustering and classification techniques are presented briefly in Section 4.1 to give some background for further analysis. Self-organizing maps and adaptive resonance theory neural networks are later used in classification of the paper machine states, so they are described in more detail in Sections 4.2 and 4.3.

4.1 Review of Identification Methods

This section presents shortly some pattern recognition methods: cluster analysis, decision trees, multi-layer perceptron, and radial basis functions neural networks. For further reading there is quite an extensive presentation of clustering methods in [Jain et al. 1999] and a somewhat shorter presentation of pattern recognition methods in [Du et al. 1995].

Cluster analysis is essentially grouping of the observations into groups by some similarity measure. The similarity measure can be e.g. distance or variance between the observations or groups. The two main branches of classical cluster analysis are hierarchical and nonhierarchical clustering [Sharma 1996].

In hierarchical cluster analysis, the number of clusters is not known *a priori*. In the beginning of the analysis, every observation forms its own cluster. During the clustering, the clusters are combined two at a time until at last there is only one cluster consisting of all the observations. The proper number of clusters is based on statistics calculated from the groups, e.g. the ratio of sum of squares between the groups and the total sum of squares. Some other statistics to determine the number of clusters are described in [Sharma 1996].

Nonhierarchical clustering differs from hierarchical one in the sense that the number of clusters *k* is known beforehand. First *k* cluster centroids are selected and then all the observations are assigned to the closest cluster. After that the observations are reassigned over and over again according to a predetermined rule until a stopping criterion is satisfied [Sharma 1996]. The weakness of nonhierarchical clustering methods is the sensitivity to the selection of initial cluster centroids.

Decision trees (DT) are techniques used to extract simple rules for classification of data. Before teaching (growing) a DT, the teaching data has to be classified and the tree is grown according to the preclassified data.

In every node of the tree there is a simple condition concerning the features and the samples are classified to different branches of the node depending on if the condition is fulfilled or not. For example, in a binary tree a node can contain a condition "speed ≥ 80 m/s". If the condition is fulfilled, the sample belongs to the right branch of the node and otherwise to the left one. Together the condition and actions form a rule. This way the sample space is divided in ever-smaller subspaces until there is only samples of one class in every subspace. Figure 8 illustrates this idea. In Figure 8 is also seen a couple of important properties of DT. Firstly, the boundaries between the subspaces, or classes, are linear, and secondly, the boundaries are crisp, i.e. strict, in sense that an observation can belong to only one class.

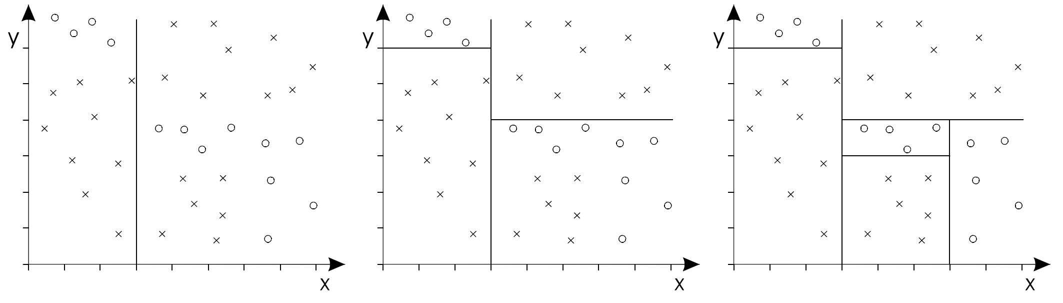

Figure 8. 2D example of a decision tree dividing the sample space to smaller subspaces with lines parallel to axes. The first division is done parallel to y-axis (left) and the seconds parallel to x-axis (center). The final division to subspaces with perfect discrimination is seen on the right.

There are a lot of methods for extracting the rules for the sample space divisions and the structure of the tree is dependent on the method used. Some methods, like ID3 [Quinlan 1986] need a predefined partitioning of the features for the tree nodes, others like CART [Breiman et al. 1984] does the partitioning while processing the data. Despite of the rule extraction method used, a DT performing a perfect classification may be very large. This may be due to noise in the data or the complexity of the process behind the data. To overcome the problem of large trees and too many conditions, some pruning heuristics are developed. In pruning, the tree is simplified with the trade-off of worse classification ability [Breiman et al. 1984].

Weakness of cluster analysis and decision trees is their usage of crisp or strict boundaries in partition. This weakness is tried to overcome in fuzzy approach [Zadeh 1988] of these techniques, namely in fuzzy cluster analysis and fuzzy decision trees. The idea behind the fuzzy techniques is that there is imprecision in the boundaries and an observation can belong to more than one group with a nonzero membership (Figure 9). Grade one membership means the observation belongs certainly to a group and grade zero that the observation does not belong to the group at all.

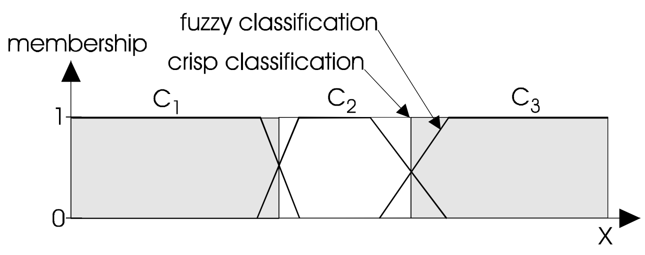

Figure 9. The difference between crisp (ordinary) and fuzzy classification to clusters C_1, C_2, and C_3.

By using this kind of definition of membership, clustering methods can be fuzzified. For example, c-fuzzy partition of the sample space $\{\mathbf{X}_1, \mathbf{X}_2, ..., \mathbf{X}_m\}$ is defined as [Friedman & Kandel 1999]

$$\begin{aligned} &\sum_{i=1}^{c} \chi_{ij} = 1, \quad 1 \le j \le m \\ &0 < \sum_{j=1}^{m} \chi_{ij} < m, \quad 1 \le i \le c \end{aligned} \tag{1}$$

where χ_{ij} denotes the grade of membership of $\mathbf{X}_j$ in cluster C_i. The membership grade is defined as

$$\chi_{ij} = \left[\sum_{k=1}^{c} \left(\frac{\left\| \mathbf{X}_j - \mathbf{Y}_i \right\|}{\left\| \mathbf{X}_j - \mathbf{Y}_k \right\|} \right)^{2/(\beta-1)} \right]^{-1}, \quad 1 \le i \le c, \quad 1 \le j \le m \tag{2}$$

where $\mathbf{Y}_1, \mathbf{Y}_2, ..., \mathbf{Y}_c$ are cluster centers and parameter $\beta > 1$ controls the degree of fuzziness. In fuzzy clustering process, the goal is to find the cluster centers $\mathbf{Y}_1, \mathbf{Y}_2, ..., \mathbf{Y}_c$ minimizing the fuzzy performance index [Friedman & Kandel 1999]

$$I_f = \sum_{j=1}^{m} \sum_{i=1}^{c} \chi_{ij} \left\| \mathbf{Y}_i - \mathbf{X}_j \right\|^2 ; \mathbf{Y}_i \in \Re^n, 1 \le i \le c \tag{3}$$

This is a complex nonlinear problem with respect to $\mathbf{Y}_i$ and there are many ways to find the solution of minimizing (3), e.g. c-fuzzy means iterative algorithm [Friedman & Kandel 1999].

Decision tree algorithms can be fuzzified in the same way as in cluster analysis [Chang & Pavlidis 1977]. This approach has two kinds of consequences. First, an observation can belong to more than one terminal node of the tree with nonzero membership. Second, if the training data is preclassified, a terminal node can have observations with nonzero membership from different classes. These consequences complicate inference based on fuzzy decision trees and the interpretation of the classification of a fuzzy tree is ambiguous. For example, [Janikow 1998] proposes inferences based on different center of gravity calculations of the fuzzy sets of the tree with good results. However, the use of fuzzy theory with DT do not change the fact that paper machine is too complex a system to monitor with decision trees.

Neural network (NN) based identification methods are very popular in a wide variety of application areas. In NN methods, the data is modeled using a group of neurons connected to each other in some specific way. The observations belonging to different classes are discriminated from each other based on the activations of the neurons.

The NN and the neurons themselves have a number of parameters, which affect the classification results of the network. Before the NN can be used in classification, it has to be taught, i.e. the parameters producing the desired classification have to be determined. The teaching is done using adequate amount of teaching data depending on the NN method. Additionally, some NN methods require the teaching data to be preclassified. These methods use the error between the desired and network outputs to improve the classification. The methods using output error are called supervised methods in contrast to unsupervised methods [Bishop 1995], which determine the suitable classification during the teaching. Next, multi-layer perceptron and radial basis functions neural networks are presented briefly and in Sections 4.2 and 4.3 self-organizing map and adaptive resonance theory are presented, respectively.

Multi-layer Perceptron (MLP) neural network is a generalization of single layer networks that arose from linear discriminant analysis [Bishop 1995]. In MLP network, there are three kinds of layers of neurons connected to each other (Figure 10). The three layers are input, hidden, and output layer. The layers are connected via adaptive weights w_{ij}, which store the "knowledge" learned by the network. The teaching of an MLP network is supervised, so it needs preclassified data for learning. One of the most popular methods to teach an MLP network is error backpropagation [Bishop 1995], which is based on the derivatives of error with respect to the weights.

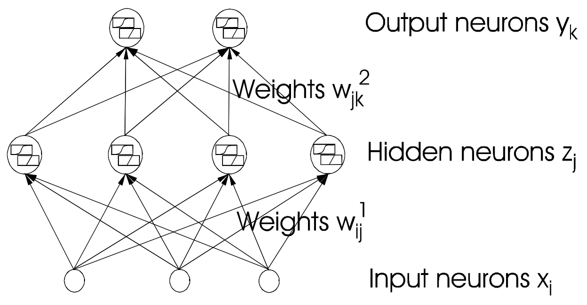

Figure 10. An example of MLP network with one hidden layer. The neurons in hidden and output layer can have either piecewise-linear or nonlinear (e.g. sigmoidal) response to their inputs.

The possibility to nonlinear mapping gives MLP networks better classification ability than classical discriminant analysis [Gallinari et al. 1988]. However, the requirement of supervised learning makes MLP an inappropriate tool for the case in hand.

Another popular neural network method is radial basis functions (RBF). In a RBF neural network, the neurons are centers of radial kernel functions. The activations of the neurons are of form $\phi(\|\boldsymbol{\mu} - \mathbf{X}\|)$ where $\boldsymbol{\mu}$ is the center of the kernel, $\mathbf{X}$ is the input vector (pattern), and $\phi(\cdot)$ is the kernel function. The most common form of $\phi(\cdot)$ is Gaussian

$$\phi(x) = \exp\left(-\frac{x^2}{2\sigma^2}\right) \tag{4}$$

where parameter σ controls the width of the kernel. During the teaching, the locations of the neurons $\boldsymbol{\mu}$ are organized to represent the data as well as possible. The goodness of the data representation of RBF NN depends on the kernel functions, width, and number of the kernels [Bishop 1995].

The use of RBF NNs for classification is based on the local kernels, which model the class distributions [Lowe 1995]. This approach is quite different from the MLP approach, which concentrates on the class boundaries instead of the sample densities. The difference between MLP and RBF neural networks in classification is illustrated in Figure 11.

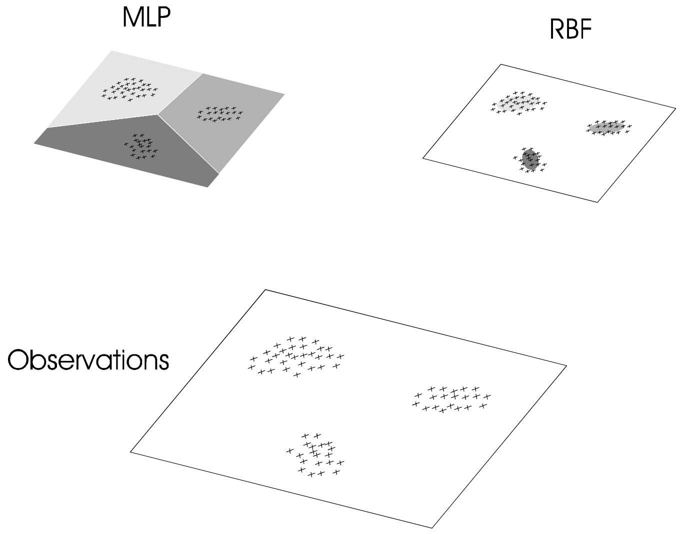

Figure 11. Classification methods of MLP and RBF networks.

4.2 Self-Organizing Map

Self-Organizing Map (SOM) is a neural network method for analyzing, modeling and visualizing multidimensional data [Kohonen 1995]. SOM is usually a 2D hexagonal or rectangular regular grid of neurons. During the training, the neurons fold onto the data 'cloud' thus mapping the multidimensional observations to a 2D-grid. The SOM-grid suits well for clustering and visualization purposes [Vesanto 1999].

The iterative algorithm for teaching SOM is simple. The input patterns $\mathbf{X} = (X_1, ..., X_n)$ are presented to the map one at a time. For each input **X,** the nearest neuron prototype vector $\mathbf{m_k} = (m_{k1}, ..., m_{kn})$ is found on the map. The nearest prototype vector $\mathbf{m_c}$ is called best-matching-unit (BMU). The prototype vector and its grid neighbors are moved towards the input pattern on iteration *t* according to the formula

$$\mathbf{m}_k := \mathbf{m}_k + \alpha(t)h_{ck}(t)(\mathbf{X} - \mathbf{m}_k), \quad 1 \le k \le l \tag{5}$$

where notation $x :=$ means assigning a new value to x, l is the number of prototype vectors, $\alpha(t)$ is a learning factor and $h_{ck}(t)$ is the neighborhood kernel centered on the BMU c. The principle of teaching SOM is illustrated in Figure 12. Both the learning factor and the radius of the neighborhood kernel are decreased monotonically with time resulting the map to become more and more stable during the training. The training is usually done in two phases. First large values of $a(t)$ are used ($a(t_0) = 0.3 \ldots 0.99$) to get a rough approximation of the data and then fine-tuning the map with smaller values of $a(t)$ ($a(t_0) = 0.01 \ldots 0.1$) [Alhoniemi et al. 1999b]. Since usually more iteration steps are needed in training of SOM to achieve a good statistical accuracy than there are data samples available, the training data is often used reiteratively [Kohonen 1995].

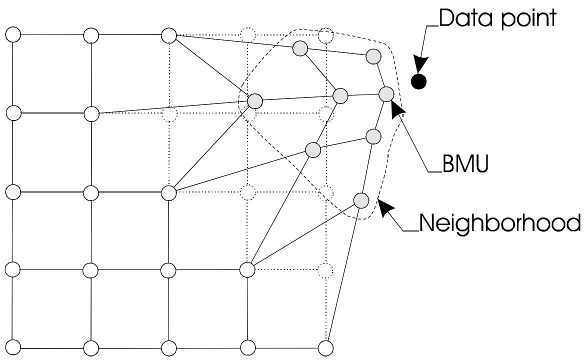

Figure 12. The principle of an iterative teaching of SOM. One data point is presented to the map at a time. BMU of the data point and the nodes in BMU's neighborhood move towards the data point thus gradually folding the map to the data 'cloud'.

The iterative, or sequential, algorithm for training of SOM is only one possibility. Another algorithm is batch training, which is computationally lighter than sequential training. In batch training the whole data set is presented to the map before any adjustments to the locations of the nodes is done. Essentially, the data set is partitioned according to the Voronoi regions of the nodes [Vesanto 2000]. Thus, each data point belongs to the partition of the closest map node. Now the new locations of the nodes are calculated from the formula

$$\mathbf{m}_k := \frac{\sum_{j=1}^{n} h_{ck}(t)\mathbf{X}_j}{\sum_{j=1}^{n} h_{ck}(t)} \tag{6}$$

where n is the number of observations and c is the index of BMU of observation $\mathbf{X}_j$. Therefore, the new location of a node is a weighted average of the observations. The weight of an observation is the neighborhood function $h_{ck}(t)$ at observation's BMU $\mathbf{m}_c$ [Vesanto 2000].

A common problem with real-world data is the issue of missing values. Some sensors can be broken, the measurements can be late or missing due to other reasons. However, with proper methods also observations with missing components can be used in teaching of the map like in [Kaski & Kohonen 1996].

Another problem is the use of categorical data. Categorical values are usually represented with integers, and used the same way in training as numerical variables. However, this is not entirely sufficient solution. There is no order for nominal values but labelling nominal values with integers creates an order for the values. Consequently, this distorts both the data and the training.

Due to the iterative nature of the learning algorithm, the presentation order of the data points affects the formation of the map during the learning. This way the presentation order affects also the final map achieved. Thus, the same data can result in different maps. As a matter of fact the maps are rarely exactly similar after two different learning periods and there is no "right" map to be found.

As one can see from the Formula (5) change of location of a neuron is proportional to the Euclidean distance between the neuron $\mathbf{m}_c$ and the input pattern $\mathbf{X}$. This gives rise to the question of scaling. If some components of the input patterns are much larger than others, the large components affect the map much more than the small ones. To avoid this problem, the components have to be scaled properly before the teaching. One of the most popular scaling methods is to scale the components to equal variance [Kohonen 1995]. This kind of scaling puts the same weight to every component independent of its original absolute values.

There are two important properties of the map that describe the quality of the map. One is average quantization error (QE_{avg}), which measures how well the map is fitted to the data [Kohonen 1995]. Average quantization error is the average distance between a data point and the respective BMU $\mathbf{m}_c$:

$$QE_{avg} = \frac{1}{n}\sum_{j=1}^{n}\left\|\mathbf{X}_j - \mathbf{m}_c\right\| \tag{7}$$

The greater QE_{avg} the farther the map nodes are from the data points and the poorer fit to the data. Another important aspect of map quality is its ability to preserve topology. The mapping is topology preserving, if data points near each other in sample space are also near each other on the map. In practice, this means the map does not fold onto itself (Figure 13). One way to measure this is to calculate the portion of the data samples, which do not have their best- and second-best matching units adjacent [Kiviluoto 1996]:

$$E_t = \frac{1}{n}\sum_{j=1}^{n}u(\mathbf{X}_j), \quad \text{where } u(\mathbf{X}_j) = \begin{cases} 1, \text{best - and second - best matching} \\ \quad \text{units not - adjacent} \\ 0, \text{otherwise} \end{cases} \tag{8}$$

$u(\mathbf{X}_j) = 1$ means a data sample has almost similar prototypes in different parts of the map. Thus, the mapping of $\mathbf{X}_j$ is not topology preserving. On the other hand, $u(\mathbf{X}_j) = 0$ indicates topology preservation, since almost similar prototypes are near each other. The measure E_t is called topographic error and its value is between 0 and 1, 0 describing the best possible topology. However, if the lattice of SOM is lower dimensional than the input space, the topology cannot be perfectly preserved [Kiviluoto 1996], i.e. the $E_t > 0$.

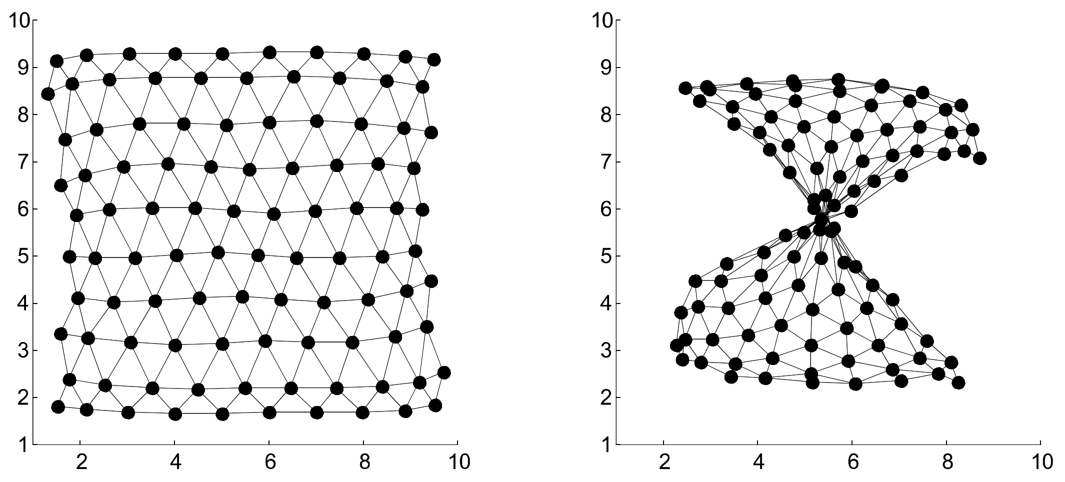

Figure 13. SOM with hexagonal grid and training set with 2D-data points on [1,10]×[1,10] area. On the left is a map with proper topology. On the right, the map has folded to itself resulting to an undesired topology. The topographic error of the map on the left is 0 (this is possible because the mapping is $\Re^2 \to \Re^2$) and of the map on right 0.07.

The conventional SOM is a single continuous grid of neurons. However, the data to be clustered may consist of several disjoint groups of observations. Thus, some of the neurons of the final SOM may be located between the data cluster, i.e. they are interpolating units. This means they are not necessarily BMUs of any of the input patterns. One solution to this problem to use a variable-size map or variable-structure as in [Fritzke 1991]. In this approach, a neuron can be removed, if it has not been selected as a BMU for a long time during the training. Thus, interpolating units are removed from the map.

There are a couple of ways SOM can be used in process monitoring, depending if there is a need for identification of all the states or just to detect the abnormal ones [Alhoniemi et al. 1999b]. In an identification case, the map is made to learn both the normal and faulty states and thus to cluster the observations to different states. After learning, the map can be used to classify new observations to one of the previously encountered states. SOM is used this way e.g. in [Simula & Alhoniemi 1999]. By drawing the path of the state to component planes the operator can also monitor the development of the process [Tryba & Goser 1991]. In this approach all the features are in the same map. Another kind of approach is presented in [Iivarinen & Visa 1998], where each of the different types of features has a map of its own. The identification is based on the responses of all the maps. This kind of approach could be used also in paper machine monitoring if many types of features were used.

The approach to simple detection of abnormal situations is a little bit different. This time only the normal observations are presented to the map so it learns only the normal states of the process. When new data is presented to the map, the quantization error of a normal observation is small (i.e. the data point is close to the BMU on the map) compared to that of an abnormal observation (i.e. the data point lies far from any of the units of the map). Thus, the detection can be based on the distance between the data point and its BMU (Figure 14). This kind of technique is used in process monitoring in [Alander et al. 1991] and [Harris 1993]. The actual indication can be strict, i.e. the decision of abnormality depends on if the data point is inside a predefined radius or not, or it can use some probabilistic measure [Alhoniemi et al. 1999a] to flexibly indicate the abnormality of the state.

Some other ways to exploit SOM in error detection or monitoring are presented in [Hamad & Delsert 1995] and [Zhang et al. 1996].

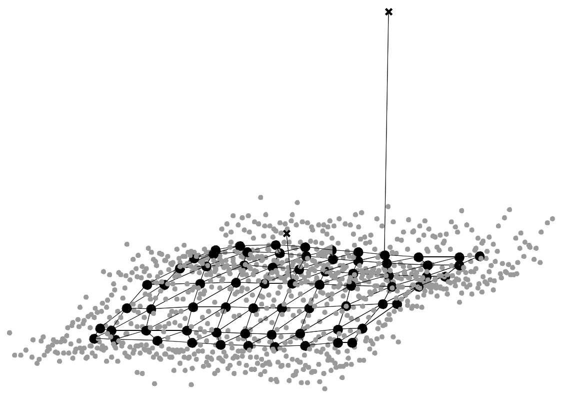

Figure 14. The idea of detecting abnormal states with SOM. The map (nodes shown as black dots connected to each other) is taught with data shown as gray dots. The new observations are marked with crosses connected with line to their BMUs. The new observation on the right is clearly farther from the map than data points in general. Thus, it is classified as an abnormal state.

4.3 Adaptive Resonance Theory

A common property of all artificial neural networks is their ability to learn connections from training data presented to them. However, depending on the algorithm and the structure of the network, the ability to adapt to new data after the teaching varies. NN should be flexible enough to adapt to new states - a property called plasticity, and on the other hand, stable enough not to react to irrelevant changes. The balancing between these two properties is known as stability-plasticity dilemma. Adaptive Resonance Theory (ART) neural network was originally designed to overcome this problem [Carpenter & Grossberg 1987a].

Another way to describe the problem is to ask "how to preserve the learned knowledge while continuing to learn new things?" [Carpenter & Grossberg 1988]. ART neural network solves the problem with two layers of neurons, which interact with each other as if a human would do in trying to compare her memories to a new situation.

The two-layer structure of ART network is illustrated in Figure 15. In ART vocabulary, the input layer (cf. Figure 10) is called input representation field. This may be due to the fact that in ART network the calculations are a little bit more complex than e.g. in MLP, which usually uses sigmoid activations without further calculations. In the same way the output layer is called category representation field, since the output of the net is classification, or categorization, of an input pattern. Together the two layers are called attentional subsystem. The outputs of the layers are called short-term memory (STM) traces. The STM traces represent the activations of the nodes of a conventional MLP NN. On the other hand, the weights between the layers in MLP NN are represented by long term memory (LTM) traces, which adapt to the inputs presented to the net.

An important property of ART network is the direct feedback from category representation field to input representation field. This allows the net to compare the activations caused by the input pattern and by the category the input is classified to. The comparison between the input and the category is done in a separate part of the network, called an orienting subsystem. A resetting unit in the orienting subsystem controls the classification by rejecting the mismatching categories from the classification.

The connections between an attentional and an orienting subsystem are shown in Figure 16. Figure 16 presents the first ART network called ART1, which is designed for binary input patterns [Carpenter & Grossberg 1987a].

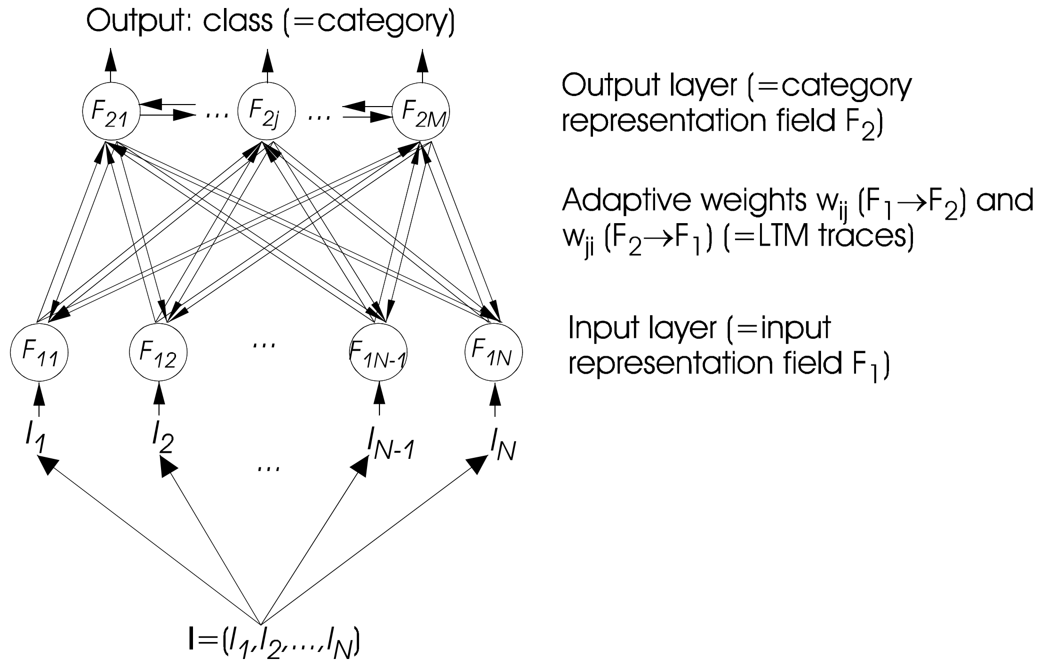
Output: class (=category)
F_{21}
F_{2j}
F_{2M}
Output layer (=category representation field F_2)
Adaptive weights w_{ij} ($F_1 \to F_2$) and w_{ji} ($F_2 \to F_1$) (=LTM traces)
F_{11}
F_{12}
F_{1N-1}
F_{1N}
Input layer (=input representation field F_1)
I_1
I_2
I_{N-1}
I_N
$\mathbf{I}=(I_1,I_2,\ldots,I_N)$

Figure 15. Two-layer structure of ART network. The number of nodes in the input layer is the number of components in the input pattern **I,** *and the number of nodes in the output layer is the maximum number of classes. The nodes F_{11}, ..., F_{1N} in the input layer consist of calculations presented in Equations (16)-(21). The output layer is a competitive net where only one node is activated. The only activated node in the output layer is the ART's guess for the class of the input pattern. The NN learns, when the feedback from output layer to input layer (i.e. top-down signal) is similar enough with the input.*

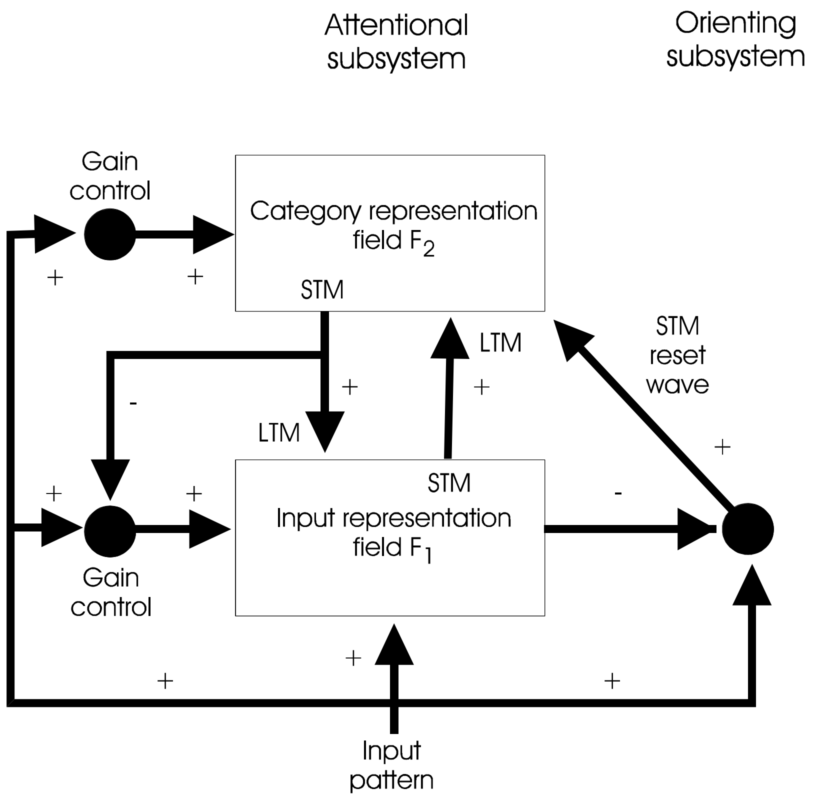

Figure 16. Structure of an ART1 network. The input representation field F_1 codes the input pattern to short term memory (STM) pattern. The STM pattern of F_1 is weighted with the bottom-up long-term memory (LTM) traces and sent to the category representation field F_2. Further, the STM pattern in field F_2 is the activation of the nodes caused by the signal from the field F_1. The category node J with the highest activation inhibits the other category nodes to send a feedback signal to field F_1. Thus, the signal from node J is the only one sent back to F_1. The signal from node J is weighted with the top-down LTM traces to form an expected input pattern. The actual and expected input patterns are compared, and if they match well enough, the network updates the active category J. In case the actual and expected inputs are too different, the orienting subsystem resets the category activation and discards category J from further search. After this, the search is continued until a category matching the input is found or all the categories are searched. If no matching category is found, a new one is created. The gain controls are required for the stability of the categorical learning [Carpenter & Grossberg 1987a].

The search for a right category is the following (Figure 17) [Carpenter & Grossberg 1987a]: The input pattern I is presented to the input representation field F_1 and the field responses with a specific STM pattern X (Figure 17a). The F_1 field also nonspecifically (i.e. on/off -type of signal without a pattern) inhibits the orienting subsystem A from sending a reset wave. The pattern X is weighted with LTM traces, or adaptive weights, to form a bottom-up output signal S from F_1. The outputs from the nodes of F_1 are summed for every node in F_2 thus forming an input signal T for the field F_2.

The signal T activates the STM pattern Y in the category representation field F_2. In the pattern Y, the only non-zero component is the one representing the category, or node, with the highest activation caused by the signal T. The pattern Y generates further top-down signal U, which is transformed with LTM traces to expected pattern V (Figure 17b). The common features in the expected pattern *V* and in the input pattern *I* activate a new STM pattern *X** in F_1. The less the patterns *V* and *I* have in common, the smaller the new STM activations *X**. Thus, the whole activation sequence described above is:

$$\text{Bottom} - \text{up}:$$

$$\underset{\text{input}}{I} \rightarrow \underset{\text{STM activation}}{X} \xrightarrow{\text{LTM weighting}} S \xrightarrow{\text{summing}} T \rightarrow \underset{\text{category found}}{Y}$$

$$\text{Top - down (feedback)}:$$

$$Y \xrightarrow{\text{LTM weighting}} U \xrightarrow{\text{summing}} \left.\begin{matrix} V \\ I \end{matrix}\right\} \xrightarrow{\text{combining of patterns}} \underset{\text{new STM activation}}{X^*}$$

If the total STM activation is reduced due to change $X \rightarrow X^*$, i.e. the patterns *V* and *I* mismatch, the inhibition from F_1 to *A* is decreased. The lower inhibition level in *A* releases a nonspecific reset wave from *A* to F_2 (Figure 17c). The sensitivity of the resetting is controlled by a vigilance parameter $0 \leq \rho \leq 1$. The lower the vigilance the more mismatch between *V* and *I* is allowed without resetting and the more heterogeneous the patterns within categories.

The wave resets STM activity pattern *Y* in F_2 and thus eliminates the top-down signal from F_2. Without the top-down signal *U* the STM pattern *X* can be reinstated in F_1. The reinstation of *X* generates anew the bottom-up signal *S* and activates the F_2 field. However, since the previously activated pattern *Y* remains inhibited, another STM pattern *Y** is activated. In case the expected pattern of *Y** also mismatches the input pattern *I*, the search continues further (Figure 17d).

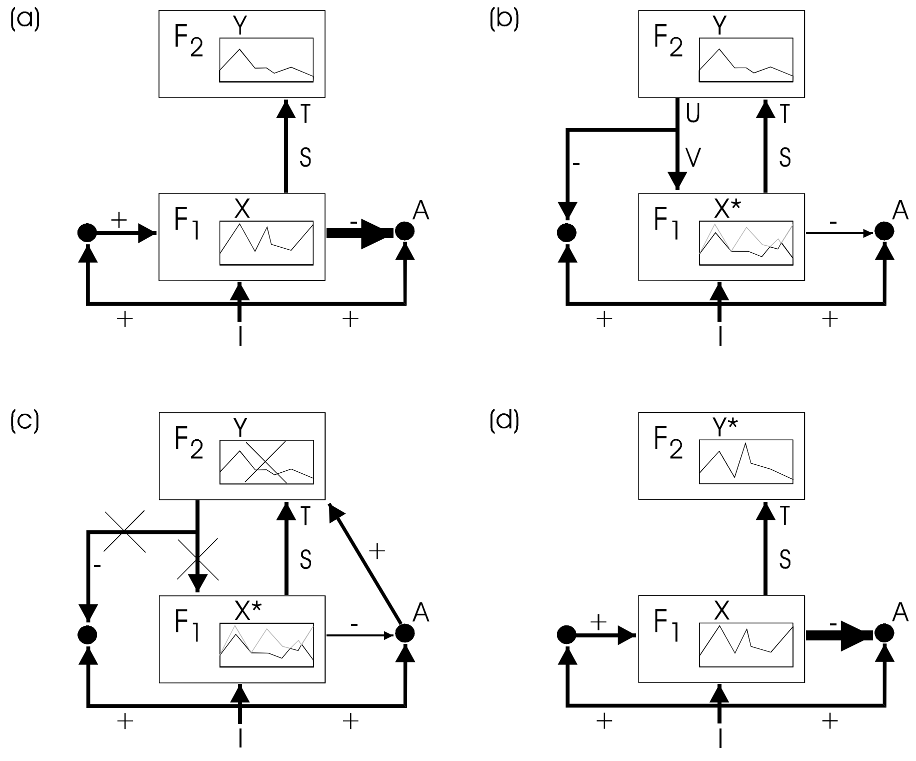

Figure 17. The search procedure for a right category in field F_2.

If an approximately correct category is found, the network remains in a so-called resonant state with both bottom-up and top-down signals active. Learning occurs only when the ART networks is in its resonant state. This is due to the fact that the STM computations in the search phase proceed too fast for the slowly varying LTM traces to react to them. In the learning phase, the LTM traces are refined according to the new information from the input pattern *V*. Thus, the changes in LTM traces, or in adaptive weights, occur only if the presented input pattern *I* is similar enough to an already known category [Carpenter & Grossberg 1988].

In a case no sufficiently similar category with the input pattern is found, and there is enough free space, a new category is created. The new category learns all the information from the input pattern.

The learning principle of ART2 network, which is capable of learning from analogue patterns, is quite similar to the one of ART1 [Carpenter & Grossberg 1987b]. One possible ART2 architecture is presented in Figure 18. This kind of architecture is prototyped and tested in Chapter 6. In addition to fields F_1 and F_2, there is also a pre-processing field F_0, which removes noise and normalizes the original input pattern $\mathbf{I}^0$. The exact equations defining the learning are presented next.

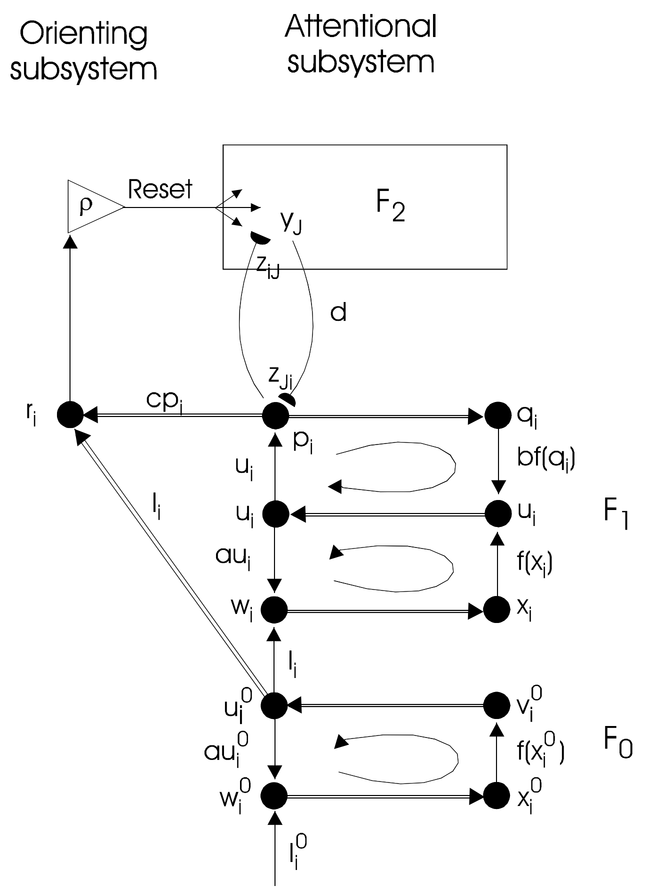

Figure 18. An example of ART2 architecture. The double lines represent Euclidean normalization.

First transformations of *M*-dimensional input pattern $\mathbf{I}^0$ are done in the pre-processing field F_0 according to the equations [Carpenter et al. 1991]

$$\mathbf{w}^0 = \mathbf{I}^0 + a\mathbf{u}^0 \qquad (9)$$

$$\mathbf{x}^0 = N\,\mathbf{w}^0 \qquad (10)$$

$$\mathbf{v}^0 = F_0\mathbf{x}^0 \qquad (11)$$

$$\mathbf{u}^0 = N\,\mathbf{v}^0 \qquad (12)$$

where $a > 0$ is a constant and operator N means Euclidean normalization:

$$N\,\mathbf{w}^0 = \frac{\mathbf{w}^0}{\left\|\mathbf{w}^0\right\|} \qquad (13)$$

The nonlinear signal function $F_0\mathbf{x}$ is

$$\left(\mathcal{F}_0 \mathbf{x}^0\right)_i \equiv f\left(x_i^0\right) = \begin{cases} x_i^0, & \text{if } x_i^0 > \theta \\ 0, & \text{otherwise} \end{cases} \tag{14}$$

where θ is a threshold value. The threshold θ should satisfy the constraints

$$0 < \theta \leq \frac{1}{\sqrt{M}} \tag{15}$$

This assures that the M-dimensional vector $\mathbf{v}^0$ is always non-zero if $\mathbf{I}^0$ is non-uniform. As a whole the pre-processing field carries out thresholding and normalization of the input pattern $\mathbf{I}^0$. The output vector $\mathbf{u}^0$ of the pre-processing field is also input vector $\mathbf{I}$ for the input representation field F_1. The calculations in the input representation field are carried out through quite similar feedback loops as in pre-processing field in equations (9)-(12):

$$\mathbf{w} = \mathbf{I} + a\mathbf{u} \tag{16}$$

$$\mathbf{x} = \mathcal{N}\,\mathbf{w} \tag{17}$$

$$\mathbf{v} = \mathcal{F}\;\mathbf{x} + b\mathcal{F}\;\mathbf{q} \tag{18}$$

$$\mathbf{u} = \mathcal{N}\,\mathbf{v} \tag{19}$$

$$\mathbf{q} = \mathcal{N}\,\mathbf{p} \tag{20}$$

$$p_i = u_i + \sum_j g(y_j) z_{ji} \tag{21}$$

where b is a constant, $g(y_j)$ is the activation of jth node in F_2 and z_{ji} are the LTM traces from node j in F_2 to node i in F_1. If the category representation field F_2 is designed to make a choice, only the node J with the largest input from F_1, i.e.

$$\sum_i p_i z_{iJ} = \max_j \left\{ \sum_i p_i z_{ij} \right\}, \tag{22}$$

becomes active. The coefficients z_{ij} are the LTM traces from F_1 to F_2. If the activation function $g(y_j)$ is a constant d, the Equation (21) reduces to the sum

$$p_i = u_i + dz_{Ji} \tag{23}$$

The category representation field F_2 works in ART2 in the same manner as in ART1. The choice of an active node is based on Equation (22) and the top-down output is d for the active node and zero for the others. If the network enters the resonant state, the LTM traces z_{ij} and z_{ji} are adapted to the new information. In case of a single active node in F_2 field, the adaptation is ruled by equations

$$\text{top}-\text{down}\left(F_2 \rightarrow F_1\right): \frac{d}{dt} z_{Ji} = d\left[p_i - z_{Ji}\right] \tag{24}$$

$$\text{bottom}-\text{up}\left(F_1 \rightarrow F_2\right): \frac{d}{dt} z_{iJ} = d\left[p_i - z_{iJ}\right] \tag{25}$$

with $0 < d < 1$.

The resetting and inhibition of tested nodes in F_2 is controlled by the orienting subsystem. The decision of sending or not sending the reset wave is based on a vector **r** measuring the similarity of the vectors **I** and **p**. The reset wave is sent iff

$$\|\mathbf{r}\| < \rho \tag{26}$$

where $0 < \rho < 1$ is a vigilance parameter. The vector **r** is defined by equation

$$\mathbf{r} = \frac{\mathbf{I} + c\mathbf{p}}{\|\mathbf{I}\| + \|c\mathbf{p}\|} \tag{27}$$

where $c > 0$ is a constant. If the expected pattern vector **p** is proportional to the input pattern vector **I**, the denominator in the right side of (27) equals $\|\mathbf{I} + c\mathbf{p}\|$ and $\|\mathbf{r}\| = 1$, so reset does not occur. In case of divergent vectors **I** and **p,** the resetting is ruled by the value of ρ.

A proper selection of both initial top-down and bottom-up LTM traces is important for the good classification performance of ART. The initial traces are the traces of the uncommitted nodes of the net, and their values have an effect on the learning of new categories. The initial bottom-up LTM traces should be large enough, so that also other than committed nodes can activate in F_2 during the search phase. The necessity of this property is evident in the case were only one node is committed. If the uncommitted nodes have no chance to activate, i.e. their bottom-up LTM traces are too small compared to the ones of the committed node, all the patterns would be classified to the category of the single committed node. Thus, no learning would occur.

On the other hand, the initial top-down LTM traces should be small. This is because if the top-down traces are large, also the difference between expected pattern and input pattern could be large when committing a previously uncommitted node. Large difference would lead to instant resetting of the newly committed node, which would prevent the learning process.

As with SOM training data, the issue of scaling of the patterns is important also with ART networks. An input pattern is normalized in the network as seen in Equations (10), (12), (17), (19) and (20), so only the relative magnitudes of the components are important. Hence, the purpose of scaling is not only to limit the magnitude of features, but also enhance the contrast of the feature vector to achieve a better separation of the observations. One way to enhance the contrast of patterns is thresholding and this is a part of ART2 network as stated in Equations (11) and (18). For thresholding to be effective, all the features have to have the same scale. In summary of all the above-mentioned scaling issues, one can conclude that the effect of poor scaling on classification performance is more fatal with ART than with SOM.

ART has both advantages and disadvantages compared to other NN structures. The definite advantages are its ability to refine the old categories and learn new ones as long as there is memory left for the new information [Carpenter & Grossberg 1987a]. The fact associated to the adaptivity is needlessness to know the number of classes beforehand. Some algorithms, however, need the maximum number of classes to be defined. In practice the larger the maximum number of classes, the slower the execution of the algorithm, since there are more classes to test.

The major drawbacks of ART are associated with its ability to handle noisy or missing data. The competitive learning occurring in layer F_2 is known to become unstable if the variety of input patterns or the number of categories is too high [Carpenter & Grossberg 1988]. One way to overcome this problem is a proper scaling of inputs as mentioned earlier. In addition to the scaling done by user, ART algorithm is self-scaling according to the complexity of a pattern [Carpenter & Grossberg 1987a]. If the patterns are simple, just few mismatching features result the patterns to be classified into different classes. With more complex patterns and the same number of mismatching features, the patterns would be classified into the same class. However, the self-scaling properties of ART make the analyzing of the functioning of the network somewhat harder.

There are two different ways to use ART in process monitoring as with self-organizing map. The first way is to use ART as a classifier to detect and identify abnormal states. As stated before, this approach needs data from both normal and abnormal states for the initial learning. However, because of the continuously adapting nature of ART, there is no difference between the learning phase and the classification phase. If a new, previously not occurred state is detected, a new category is created and the information about the new state is adapted to the system. Despite of this fact, the parameters of the system have an effect on the classification results of ART. Thus, some time has to be spent to consider if the classification of the training data is appropriate, before the system can be used for on-line monitoring. One version of ART, so called Multichannel-ART (MART) [Fernández-Delgado & Barro 1998], is used this way for classifying ECG patterns [Barro et al. 1998].

The other task in monitoring is the simple detection of abnormal states. This is done by training the ART network with normal data and informing the monitor if a new state does not match with any of the previously observed states. This also gives the monitor an opportunity to decide if the mismatching new state is a normal or abnormal one.

5 SOM Monitoring Prototype

The SOM monitoring is prototyped with Matlab and SOM toolbox [Vesanto el al. 2000] for Matlab written in Laboratory of Computer and Information Science of Helsinki University of Technology. The version used is SOM Toolbox 2.0beta and it can be found at http://www.cis.hut.fi/projects/somtoolbox/.

5.1 Teaching of the Prototype

Essentially, the goal of paper machine monitoring is to detect abnormal states as stated in [Rinta-Runsala 2000]. Since there are no comprehensive data about the abnormal states available and classification of abnormal states is not required, the construction of the SOM monitoring prototype is based on normal data. The principle for this kind of monitoring is presented in Section 4.2. Essentially, SOM is taught with a normal data and the abnormality of a new observation is determined by the QE of the observation.

There are a few points to consider while constructing the monitoring prototype. The most important ones are the selection of proper training data, the scaling of the values and the effect of missing values.

The trained SOM should be sufficiently sensitive for the abnormal states. Better sensitivity can be achieved by using as stable training data as possible. This results in a map with nodes representing only stable normal states of the paper machine. Every deviant state, like change in set point values, closing of a nip, change of a paper reel, or abnormal periodicities, has a large QE and is therefore classified as an abnormal state. Thus, the monitoring system declares many normal states as abnormal ones. However, the information about the instantaneous changes made by the operators can be inputed in the system and be used to eliminate the states with normal reasons for large QE. In this case study, the information about the changes made by operators was not available and the elimination of false detections was not possible. Instead, the reason for abnormality of a state could be determined by examining the underlying signals and the false detections could be eliminated this way manually.

The usual scaling of a training set to zero mean, unit variance by using sample mean and standard deviation is enough if the training set has sufficiently high variability. Since the training set consists of periods of stable states, the variability within the periods is small (Figure 19). However, the variability between the stable states can be large compared to the variability within a state as seen in Figure 19. Thus, a high enough variability has to be between the stable states in the training set to ensure proper scaling. If this condition is fulfilled, zero mean, unit variance scaling can be used.

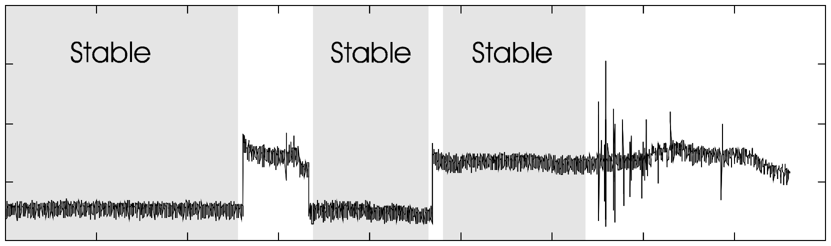

Figure 19. The variability within each of the stable periods is small, so large enough variability has to be achieved by taking samples of stable periods of different levels (e.g. of last and second last stable periods).

The effect of missing values in the monitoring performance of SOM in this case study is related to both the temporal nature of wavelets and the scaling of coefficients. To calculate the wavelet coefficients, there has to be a certain number of past measurements available. For example, if the length of the wavelet is 4,096 seconds, i.e. 8,192 observations with 0.5 s sampling interval, the calculation of the wavelet coefficient requires 8,192 past observations. If one of the observations is missing, the coefficient cannot be calculated. Thus, one missing observation in this particular case implies that the wavelet coefficient cannot be calculated for 4,096 seconds. The missing coefficients do not affect directly the training or monitoring algorithm. However, many missing values in the training set decrease the variability and may result in an inappropriate scaling before the training.

The requirements for the training data set described above are summarized in the following list:

1. Data from normal stable states
2. Enough variability between the states
3. As few missing values as possible and filling of the data if necessary

The data used for training of SOM consists of four periods with total number of 18,193 observations. One of the periods is from the time series I, two from the time series II and one from the time series III. Three of the periods are 4,096 seconds long and one is 5,905 seconds long. The lengths of the periods are short for two reasons. Firstly, the lengths of stable states of the paper machine (i.e. periods without changes in set points, web breakages, etc.) are quite limited. Secondly, and more importantly, the size of the training set is limited by the computational and memory capacity on hand. If there had been more data, the training data should have been improved by increasing the number of sample periods instead of the length of the periods thus leading to a greater variability of data.

Wavelet coefficients are the only features used in monitoring, as stated in Section 3.2. However, this does not make the proper scaling of the data irrelevant. The coefficients of the wavelets in different scales have different magnitudes. The coefficients of wavelets of short lengths are in general smaller than the coefficients of wavelets of longer length (Figure 20). The coefficients of the wavelet with the coarsest scale (i.e. the longest wavelet) are usually also the largest ones, since the coarsest scale essentially determines the level of the signal studied.

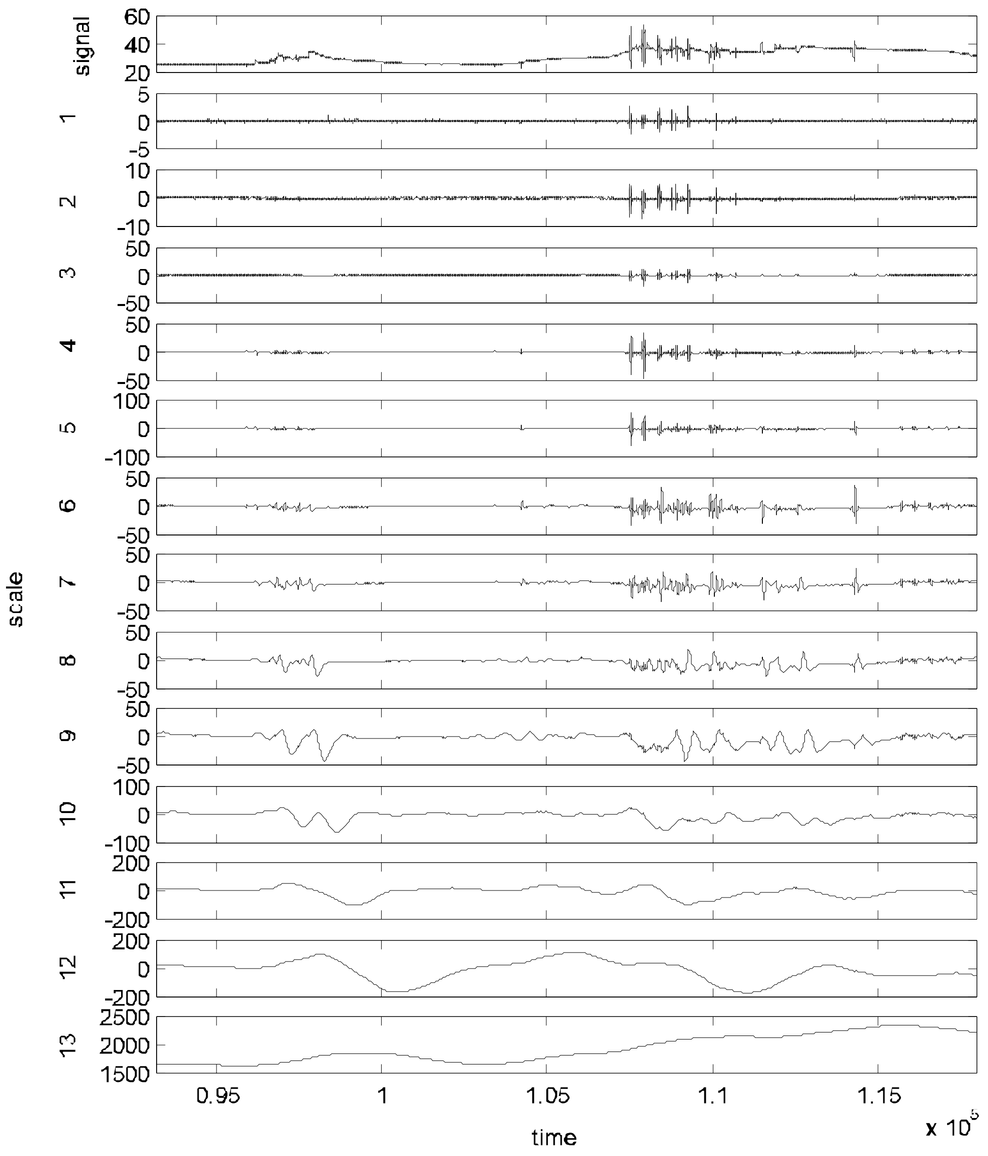

Figure 20. Example of Daubechies-4 wavelet coefficients. The coefficients are calculated from the signal in the upper most subfigure. The time axis is the same in all figures, with time units of 0.5 s (sampling interval).

The wavelet coefficients of different scales describe events in different time scales. The events in all the time scales are supposed to be equally important, so the weight of all the wavelet coefficients should be the same in SOM. This is done by scaling all the wavelet coefficients to zero mean and unit variance before training the map. An important aspect in scaling of the wavelet coefficients is not only to limit the magnitude of the coefficients used in training of SOM, but also of the coefficients of the observations not belonging to the training set. This is especially important since the monitoring is based on QE. Inappropriate scaling would easily result in huge QEs for the new observations and ruin the monitoring ability of SOM.

The variability of the signals between the periods is sufficient for zero mean, unit variance scaling in all but one signal, namely the speed of a cylinder in the second dryer section. The reason for the scaling problems of that signal is a period of missing values. The scaling problems were overcome by replacing the missing values with a constant value. This was justified by the fact that the speed in question gets only two values around the period of missing values. The constant replacing the missing values is the median of the values before the period of missing values.

The data set size of 18,193 observations can be considered medium size for the SOM Toolbox used [Vesanto 2000]. Data set of this size needs a lot of memory, so the batch training algorithm presented in Section 4.2 is used to minimize the memory requirements. Due to memory limitations, also the number of map units is set to 300. The map units are set in rectangular sheet with hexagonal lattice with side lengths of 15 and 20 units. The other training parameters are determined by routines in SOM Toolbox.

In training, a Gaussian neighborhood function

$$h_{ci}(t) = e^{-\frac{\delta_{ci}^2}{2r(t)^2}} \tag{28}$$

where δ_{ci} is the distance between units c and i on the map grid and $r(t)$ is the neighborhood radius at time t, is used. The training is done in two phases. First the coarse training is done with neighborhood radius decreasing from 3 to 1 during the training. After this the map is fine-tuned with constant neighborhood radius of 1. The trained map has an average quantization error 15.3 and a topographic error 0.089.

5.2 Monitoring with the Prototype

Monitoring of the process states is based on the QEs of new states as presented in Section 4.2. The exact monitoring procedure is illustrated in Figure 21. First the wavelet coefficients of the signals are calculated (feature extraction). After that the coefficients are scaled to the same scale as the components of the feature vectors in the map. Next, the best matching unit for the scaled feature vector is found on the map and the respective quantization error is calculated. Finally, the decision about the abnormality of the new state is made based on the calculated QE.

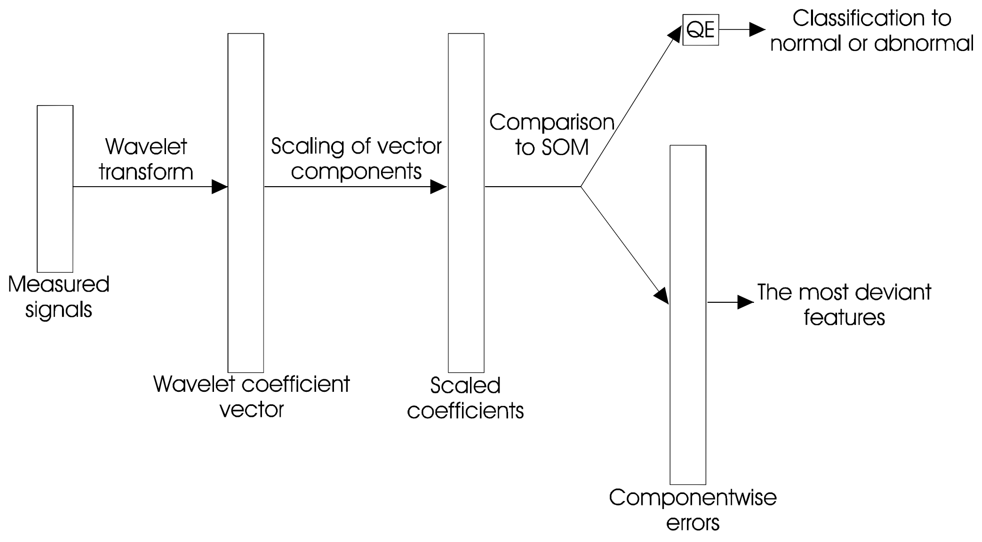
QE
Classification to normal or abnormal
Comparison to SOM
Wavelet transform
Scaling of vector components
Measured signals
The most deviant features
Wavelet coefficient vector
Scaled coefficients
Componentwise errors

Figure 21. Monitoring procedure using SOM for abnormal state detection. The results of the monitoring are the possible detection of an abnormal state and the most deviant features, i.e. wavelet coefficients. This means the abnormal signals and the scale of the abnormal events in them can be determined.

In preliminary examination of the calculated QE, the QEs can be roughly divided in three groups. The normal steady states have QE around 10-20, while the states with abnormal periodicies have QE between 20 and 100. The third group is the observations with QE more than 100, sometimes even 1000. This large a QE usually results from instantaneous or rapid changes in the process, typically opening or closing of a section of the machine, change in set points, etc. The effect of this kind of changes should be eliminated from the QE, but without data about their times of occurrences this is difficult and laborious.

The QE calculated from Formula 7 can be understood to be composed of componentwise differences ΔX_i between the observation and its BMU

$$QE = \sqrt{(\Delta X_1)^2 + (\Delta X_2)^2 + \ldots + (\Delta X_n)^2}\ , \tag{29}$$

The contributions of the pattern components to QE can be determined from the magnitudes of the differences ΔX_i. The greater the difference the more abnormal the component is. Let us consider a case where the largest componentwise differences are in scale 5 wavelet-coefficients of a speed signal of roll in first coater section. Now one can conclude, that the abnormal phenomenon in first coater section is in scale of $0.5s * 2^5 = 16$

seconds. This can further aid in corrective actions or starting of a more thorough analysis of the section.

5.3 Test results

The monitoring performance of SOM is tested against the phenomena P1-P6 described in Table 1. However, there is no way to eliminate the distortion in QE resulting from instantaneous changes with the data currently available. Thus, only the signals contained in each phenomenon are taken into account when calculating QE resulting in a partial QE. Figures 22-27 present the calculated partial QEs for the phenomena. As a summary, the phenomena P2 and P3 are clearly detected in QE as increases in values, phenomena P1 and P6 more weakly and the phenomena P4 and P5 cannot be distinguished from the pictures.

The obvious reason of inability to detect phenomenon P4 is the fact that periodicity like that is present almost all the time in the measurements. Thus, during training SOM learns P4 as a normal state and cannot distinguish it with larger QE later on. For phenomenon P5 the reasons for monitoring inability are related to the selection of features and their scaling.

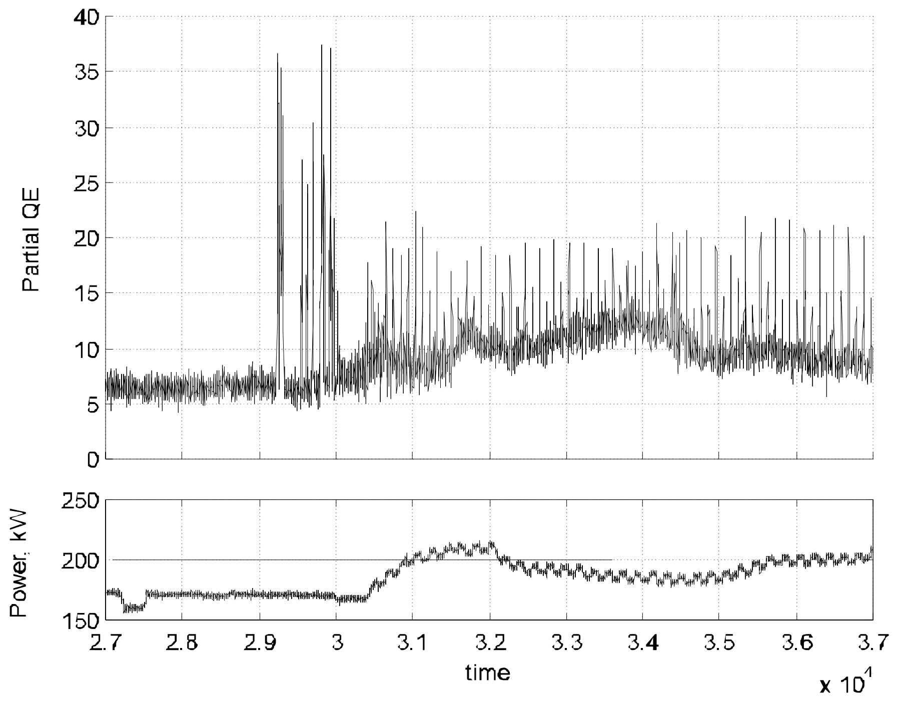

Figure 22. The effect of the phenomenon P1 (lower picture [one of the signals where P1 is shown], the periodicity between 30,000 and 58,000) to the QE of the respective signals (upper picture). The time axis is the same in both pictures with time units of 0.5 s.

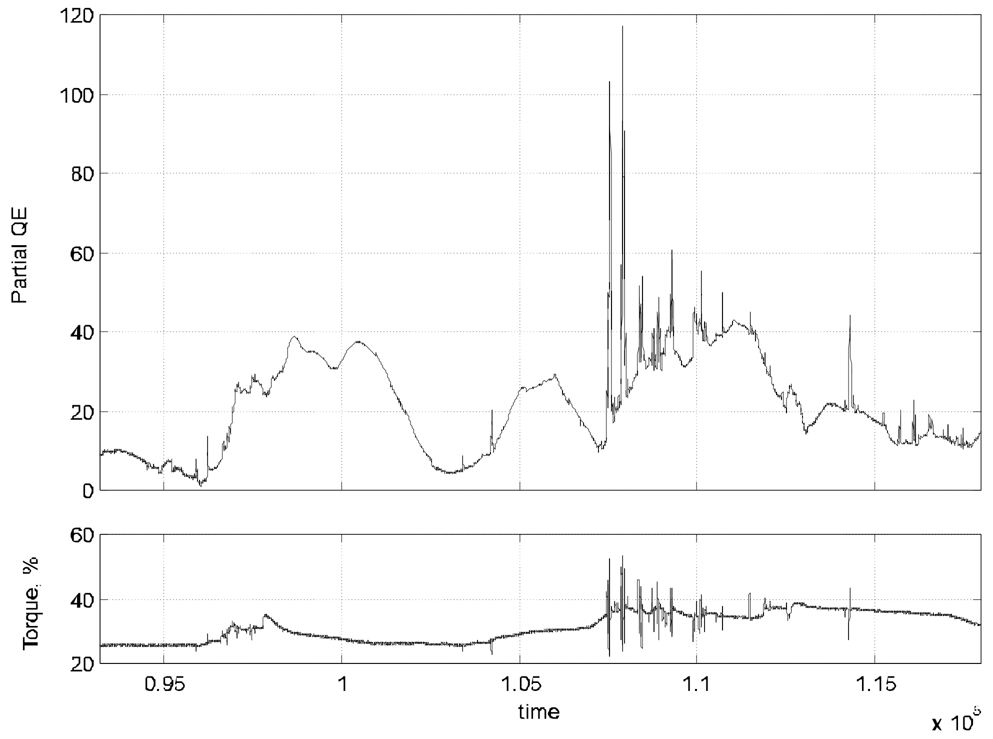

Figure 23. The effect of the phenomenon P2 (lower picture, the spikes between 107,000 and 110,000) to the QE of the respective signal (upper picture). The time axis is the same in both pictures with time units of 0.5 s. Phenomenon P2 can be clearly detected as peaks in the partial QE.

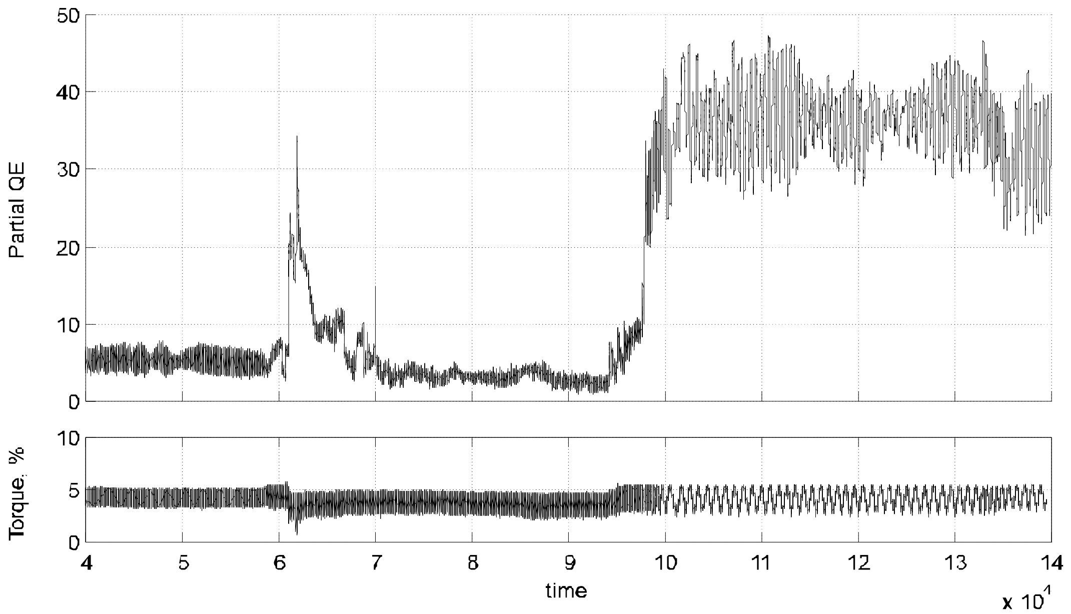

Figure 24. The effect of the phenomenon P3 (lower picture, the periodicity starting at 100,000) to the QE of the respective signal (upper picture). The time axis is the same in both pictures with time units of 0.5 s. P3 can be clearly detected from the partial QE.

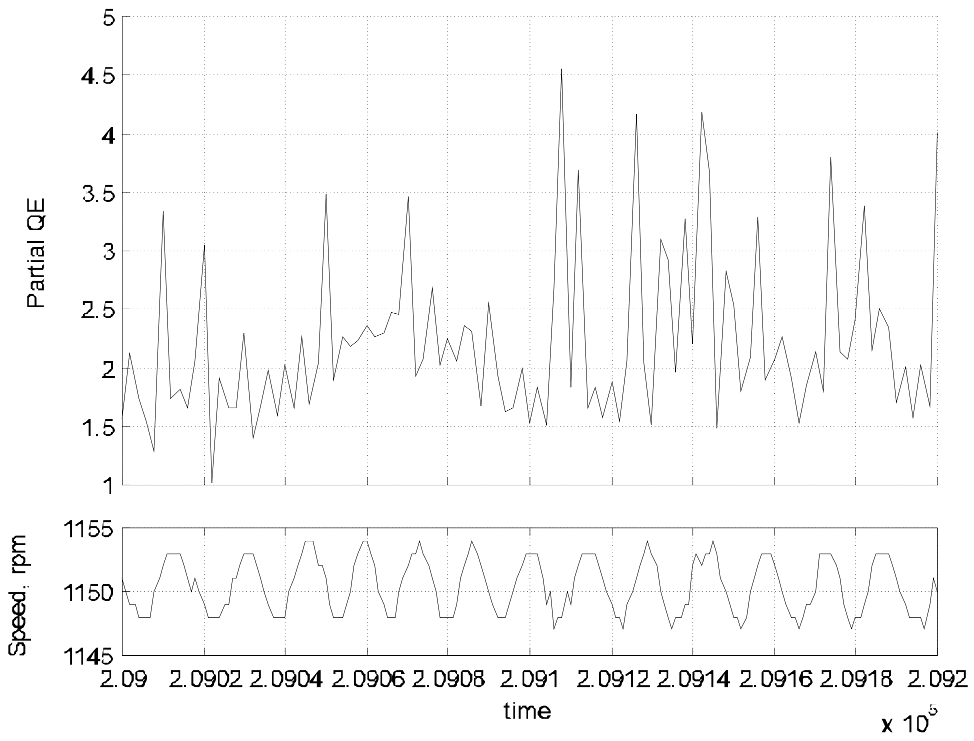

Figure 25.The phenomenon P4 (lower picture, the periodicity) and the QE of the respective signal (upper picture). The time axis is the same in both pictures with time units of 0.5 s. The phenomenon cannot be distinguished in the partial QE.

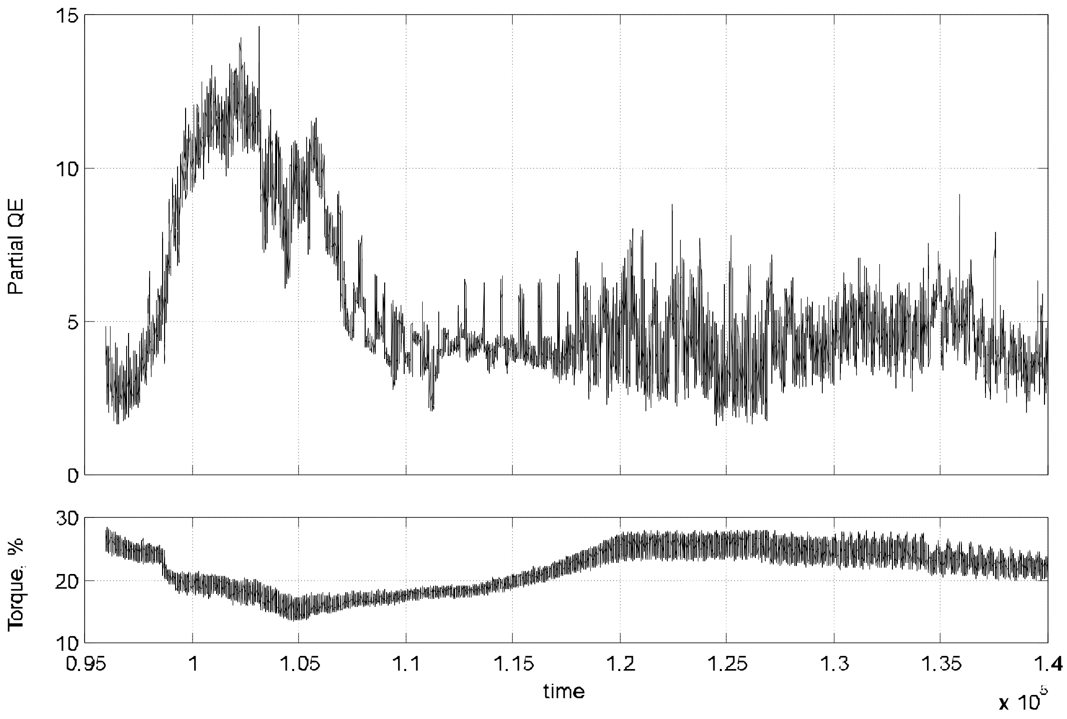

Figure 26. The effect of the phenomenon P5 (lower picture, the periodicity starting around 126,000) to the QE of the respective signal (upper picture). The time axis is the same in both pictures with time units of 0.5 s. P5 is not perceivable from QE. The effect of decrease between 98,000 and 105,000 to QE with respect to effect of P5 is notable.

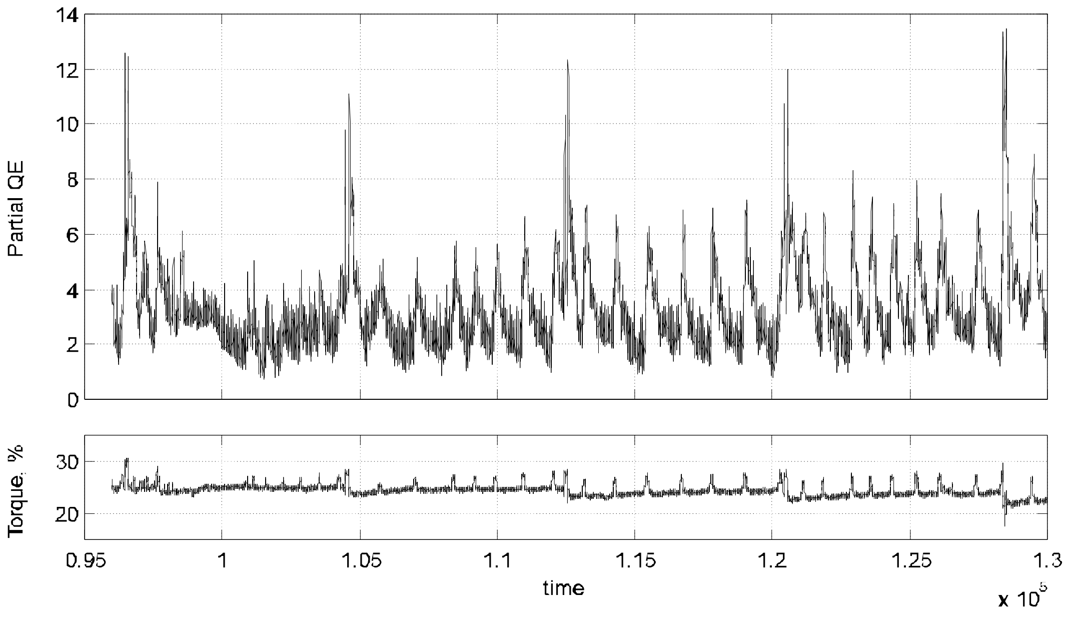

Figure 27. The effect of the phenomenon P6 (lower picture, the periodic spikes) to the QE of the respective signal (upper picture). The time axis is the same in both pictures with time units of 0.5 s. The effect is clear, but the magnitude of partial QE is quite small.

Since there are many other disturbances affecting the total QE, the actual monitoring performance is worse than the one described in Figures 22-27. How much worse depends on the ability to take into account the normal changes in the signals and their effect on the total QE.

In the future, when information about the discrete events, like operator interventions, is available, the false detections caused by normal changes can be eliminated to improve the monitoring ability of SOM. However, as mentioned before, this was not possible with the data on hand.

6 ART Monitoring Prototype

The ART monitoring prototype was coded in Matlab environment. The ART algorithm used is ART2 presented in Section 4.3.

6.1 Parameters of Prototype

Training of ART network does not differ in principle from real time monitoring with it. ART learns as well during the monitoring as on the training period. However, training before starting of the actual monitoring is usually useful.

As stated in Section 4.3 the scaling and the contrast within the patterns are important when using ART. The coarseness of classification can be adjusted by changing the vigilance parameter of the network. However, if the input patterns do not differ enough from each other, tuning of the network parameters cannot enhance the classification much. This may result either in too coarse or too fine a classification, both being poor for monitoring purposes.

Another thing related to scaling is the handling of a component with an extraordinary high absolute value. For example, if a component in the pattern vector has an extremely high value, the normalization and thresholding process in layer F_1 may result in classification of otherwise dissimilar patterns into the same category (Figure 28). Thus, in addition to the contrast enhancement, also the magnitude of the extreme values has to be limited in the scaling.

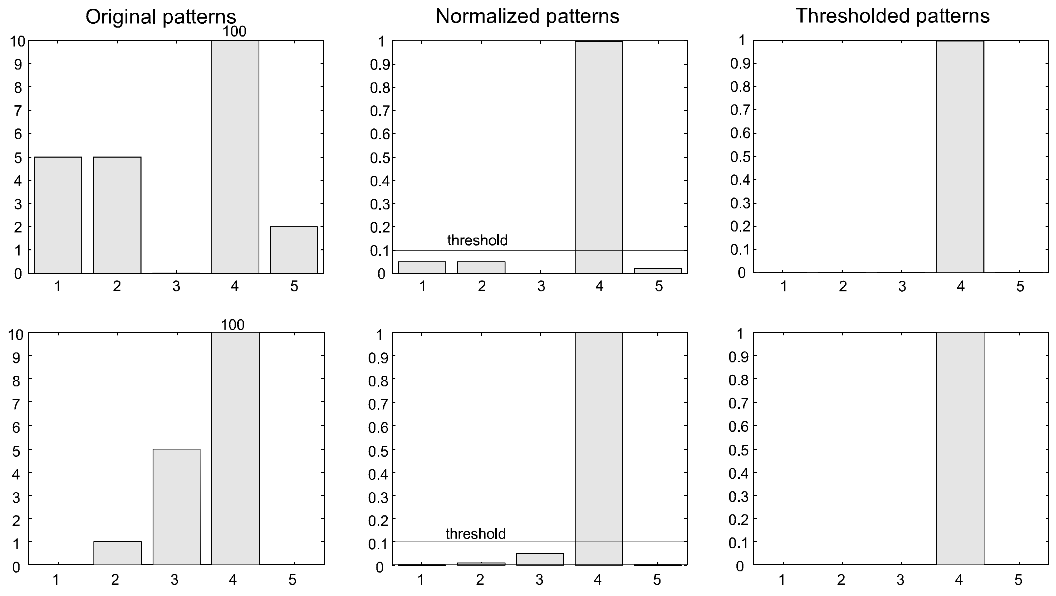

Figure 28. An example of how two dissimilar patterns may be classified into the same category because of normalization and thresholding. The 4th component of the patterns is 100 and the threshold is 0.1.

After the training set has been presented to ART, the next thing is to ensure the stability of the network. This is done by presenting the same set of observations again to the network. If ART has been trained well, learning should be stable and the observations would be classified to the same classes as in the previous presentation time. One must notice, however, that changing of the presentation order of patterns may also change the classification results due to the ever-adapting net. A more thorough discussion about the stability issues can be found in [Carpenter & Grossberg 1987a] and [Carpenter & Grossberg 1988].

When considering the stability and performance issues of ART, one also has to consider the expected number of classes of the process. In standard pattern recognition tasks such as in letter recognition, this is not a problem, since the number of classes is limited. However, this is not the case with an industrial process like paper machine. The number of classes may not be limited or at least the large number of features and the complexity of the process suggest a vast number of classes. When dealing with a large number of classes, one is tempted to limit the number of classes by decreasing vigilance of ART. However, this may complicate the interpretation of the classification results, since classes representing large number of states are more abstract compared to classification with more strict vigilance [Carpenter & Grossberg 1987a]. Thus, also this has to be taken into account when deciding the training parameters of ART.

The scaling used for ART patterns is following:

$$X_i' = \tanh\left(\left|\frac{X_i - \overline{X}_{i\_ref}}{a\, s_{i\_ref}}\right|\right) \tag{30}$$

where X_i is the original value of ith component of input pattern **X**, $\overline{X}_{i\_ref}$ and s_{i_ref} are the sample mean and the standard deviation of X_i in the reference data and a is a constant scaling factor. The reference data consists of both normal and abnormal observations and its purpose is to unify the scale of the pattern components. The scaling factor a was chosen to be 5. Absolute values are used because only the deviations are interesting and taking *tanh* limits the scaled components X_i' to interval [0,1].

The initial LTM traces have to fulfill the conditions set in Section 4.3. Thus, the initial values of bottom-up LTM traces are chosen randomly from uniform distribution U(0,0.02) and the initial top-down values are set to zero. The other ART parameters are: $a = 10$, $b = 10$, $c = 0.1$, $d = 0.9$ and threshold $\theta = 0.05$. The vigilance parameter ρ is set to 0.80 and the maximum number of categories is limited to 1000. The parameters for ART are partially the same as used in [Carpenter & Grossberg 1987b] and partially their values are tuned for the problem on hand.

Preliminary tests with ART trained with specifically selected training data were not considerably better compared to a situation with no specific training data. Thus, no specific training data was selected and the features of time series III were presented to the network in chronological order. This approach simulates a real situation, where ART has to classify many previously undetected states on-line. Hence, the results of this prototyping tell much about ART's possibilities in real-life monitoring.

6.2 Results of Prototyping

ART network with scaling of patterns and net parameters as presented in the previous section produces at the first run through the time series III a classification with 194 categories. The second run through the data uses only 32 categories but the rest of the categories are mostly categories with few observations and the classification can be considered stable.

Results of the second classification are in Figure 29. The number of states found is still quite large and this confirms the hypothesis of vast number of states in a complex process like paper machine. An important fact related to this many states is that the monitoring speed slows down gradually. This is true especially if one could expect many new states to appear also in the future. The reason for this fact is that the longer the network has run the more categories it has to go through before creating a new category for a new state. For complex processes this is quite a limiting notion.

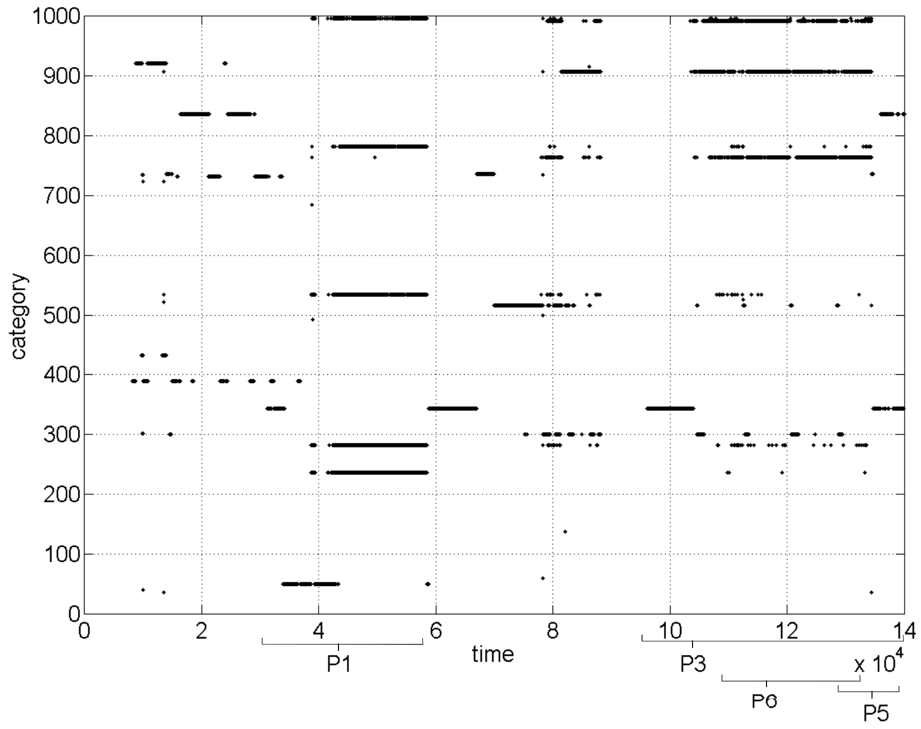

Figure 29. Categories of observations in time series III. The numbering of categories is arbitrary. Phenomena P1, P3, P5 and P6 are marked in the figure. E.g. phenomena P1 is most of its duration mapped to categories 237, 283, 534, 782, and 996. In an ideal situation P1 would be mapped to only one category.

When examining Figure 29, one notices that the phenomena P1, P3, P5, and P6 are not clearly classified into separate abnormal categories, or at least there are many categories for the phenomena. Because there is no other indication of abnormality, operator should study each new category separately to determine if it is abnormal or not and label it respectively. With as many categories as in the case study, this would be very laborious task, which discourages the use of ART.

The classification of phenomena P2 and P4 with the net trained with time series III is in Figures 30 and 31. As can be seen, the classification is not clear in these cases either.

Phenomenon P2 is classified into a single category, but the category stretches outside P2. Phenomenon P4, on the other hand, is classified 4-5 categories, some of which are abnormal and some of which are normal.

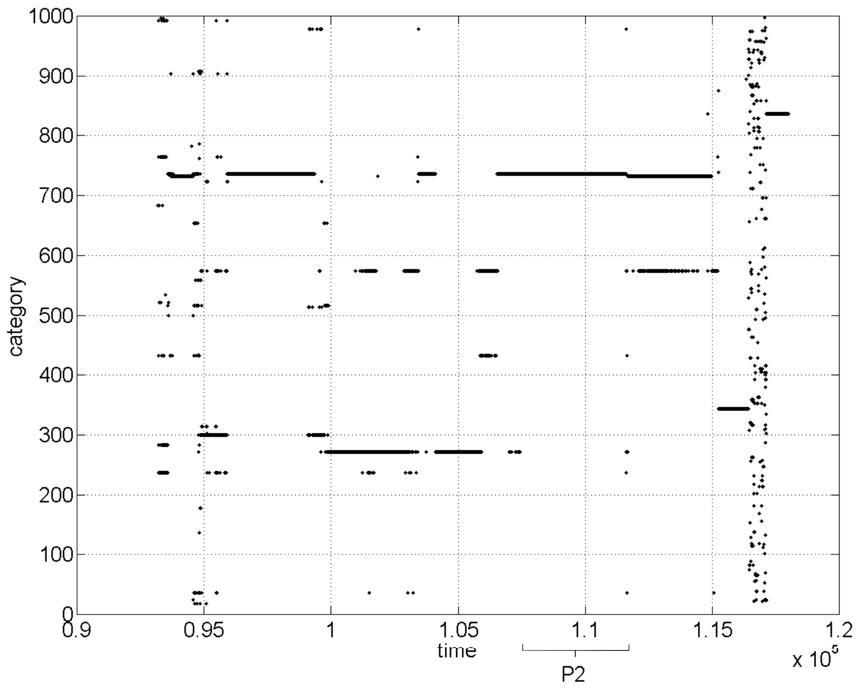

Figure 30. Classification of phenomenon P2 (in time series I) with ART trained with time series III. The occurrence of the phenomenon is drawn to the figure. The numbering of categories is arbitrary.

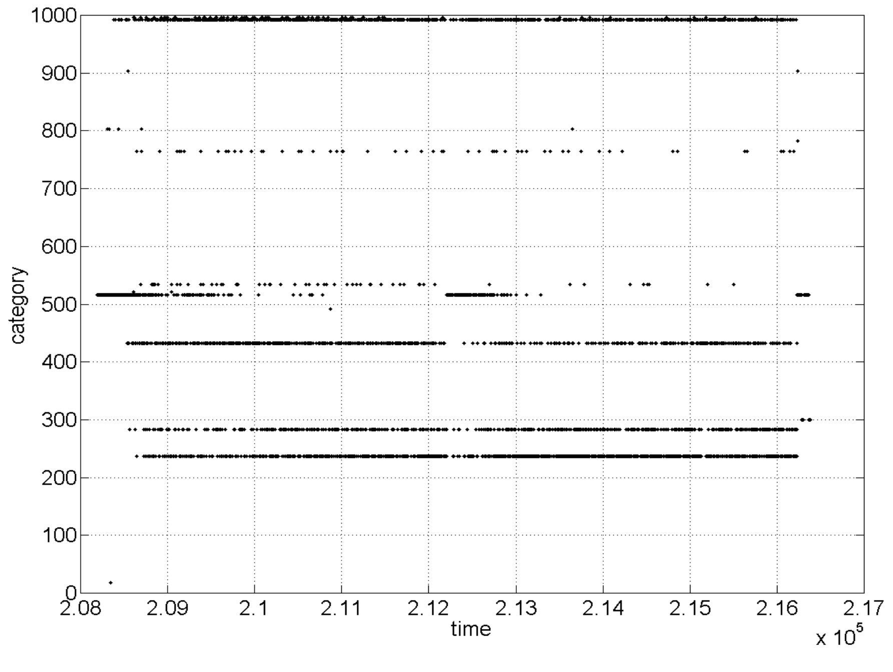

Figure 31. Classification of phenomenon P4 (in time series II) with ART trained with time series III. The phenomenon lasts through the whole figure. The numbering of categories is arbitrary.

One possibility to improve the results of ART prototype is to take the operator interventions into account as was recommended with SOM. Other options in improving could be tuning of the scaling method or network parameters or cluster distribution of recently selected categories. However, for now the monitoring results with the test phenomena P1-P6 are clearly worse than with SOM prototype.

7 Comparison of the Prototypes

The state identification methods used in the prototypes are similar in the sense they both are self-organizing in their nature. This means they both cluster the patterns without supervision or predefined classes. However, they are quite different with respect to many other properties. A summary of the properties of SOM and ART in the case of paper machine monitoring is in Table 2. Each of the properties is considered next in detail.

The quality of taining data is often an important matter with neural network methods. This is also the case with SOM prototype, which needs carefully selected, normal, and steady training data to learn the normal states. In this case, study the selection was made manually due to limitations in availability of data. However, in an actual monitoring tool the selection of the training data should be automatized and be based e.g. on the variances of the signals. ART, on the other hand, does not need any specific training data. It can be trained with the data available, either normal or abnormal.

Due to the training data used, the need for human interaction after the training varies. In SOM, no further human interaction is needed, since all the states on the map are assumed normal. ART, on the contrary, may need post-training interaction in form of labeling the found categories, for the data and the categories found can be either normal or abnormal. This can be very laborious task, since some unstable states can create many categories, as seen in Figure 29. Moreover, every time ART finds a new category during on-line monitoring, the state has to be labeled as "normal" or "abnormal" by a process expert.

The scaling techniques affect the monitoring results considerably in both methods. However, because of the normalization and thresholding ART is somewhat more sensitive to the choice of the scaling method.

Another sensitivity issue is sensitivity to change in method parameters. The comparison of this issue is not so clear, since SOM Toolbox uses well-tried default values for most of the method parameters but same type of guidelines are not available for ART. However, the fact that the number of process states is both undefined and large increases the effect of vigilance parameter of ART. This is because small changes in vigilance cumulate in time as huge changes in the number of categories found. Hence, SOM can be considered more robust with respect to method parameters. The robustness also credits a smaller need for parameter tuning.

The on-line adaptivity of ART is clearly better than that of SOM. In fact, SOM does not adapt at all during the monitoring. ART, as stated earlier, learns as well during the monitoring as during the training period.

SOM's incapability to adapt the changes in the process results a need to retrain the map if the process changes significantly. ART can learn as long as there are uncommitted nodes (i.e. categories) left, but after this it has to be reinitialized. Even if there would be enough memory for several thousands of categories, ART has to be reinitialized for time to time because of the slow down in searching as mentioned in Section 6.2.

The last thing to consider in the methods is their output or the indicators for abnormality. With SOM, the indicator of abnormality is one figure, the QE of the observation. The bounds for abnormality are user defined, so QE can be considered as measure of risk of abnormality instead of a clear binary response. Additionally, QE can be decomposed into featurewise deviations, which tell the most abnormal features. One can further determine the abnormal signals and scales of events from the features to help in isolating and solving the problem.

ART indicates an abnormal state by classifying it to a category of abnormal states or creating a new one. Thus, it is a strict binary indicator of abnormality. The most helpful information got from ART for finding the reasons for abnormality are the other states in the same category. If the reasons for abnormality of the other states are known, also the current observation can be assumed to have the same reasons. If this does not help, the only additional information is the feature vector of the current state. Thus, no most deviant features are possible to extract from ART. In this case differences between the current feature vector and a known normal state can be helpful and give some insight about the abnormality of each feature.

The last and perhaps the most important matter with SOM and ART methods is the overall results of the prototypes. They are considered in Sections 5.3 and 6.2. As a summary of those sections, can be said that the test results of SOM prototype are fair in detecting the test phenomena and with ART prototype they are poor.

The comparison of properties of SOM and ART suggests SOM to be used further in the monitoring. This is because SOM needs less user interaction and it gives more precise information about the current state of the process.

Table 2. Summary of the properties of SOM and ART with respect to paper machine monitoring.

	SOM	*ART*
Requirements for training data	Carefully selected, steady, and normal training data needed	No specific training data needed
Need for post-training interaction	No need for post-training interaction	The categories found have to be labeled to "normal" and "abnormal" ones
Scaling of patterns	Sensitive to scaling	Very sensitive to scaling
Sensitivity to parameter values	Sensitive, but lots of guide-lines available for parame-ter selection	Classification ability very sensitive to parameter val-ues
On-line adaptivity	No on-line adaptation	Adapts on-line, but adapta-tion slows down with very many categories
Need for retraining / reini-tialization	Retraining needed if the process changes signifi-cantly	Reinitialization needed if the adaptation slows down too much for on-line moni-toring
Indicator for abnormality	Scalar measure of abnor-mality; no strict classifica-tion	Classification to "abnormal" or "normal" category or creation of a new category
Aids for finding the reasons for abnormality	The most deviant features	The other states in the selected category; addition-ally only the feature vector of the state
Monitoring performance with test phenomena	Fair	Poor

8 Suggestion for Implementation

The implementation of the monitoring system is suggested to be based on SOM for the reasons stated in Chapter 7. In Section 8.1 the architecture of the system is suggested and in Section 8.2 some practical issues are discussed. Section 8.3 presents some limitations posed by both the SOM algorithm itself and a software implementation of SOM, namely SOM Toolbox.

8.1 Architecture

The suggested architecture for the monitoring system is based on similar principles as the SOM monitoring prototype in Section 5.1. The construction of the monitoring system has two phases. The initial phase is the off-line training of the SOM and the other phase is the actual on-line monitoring phase (Figure 32).

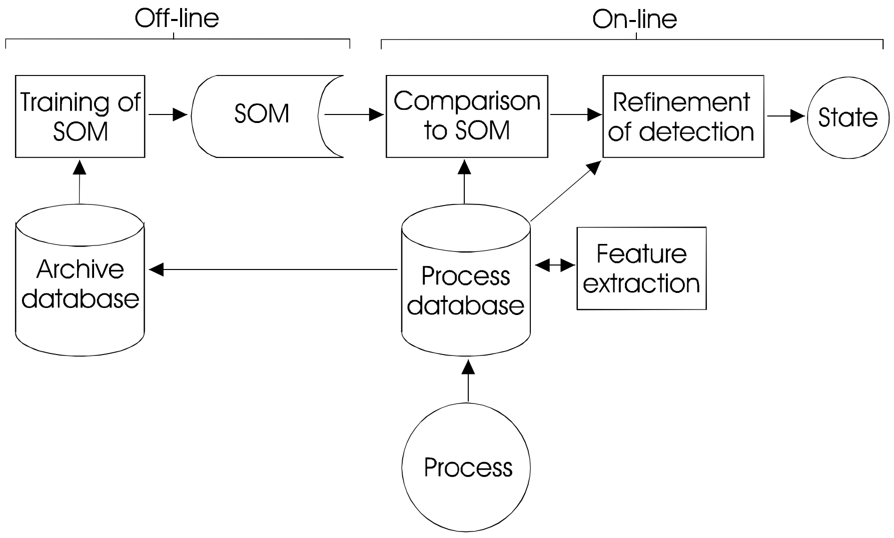

Figure 32. Architecture of the suggested monitoring system.

Training of SOM includes the selection of the training data and the training of the map. The training data has to cover a long enough period of time depending on the process, at least several months. Although the training data has to cover a long period, all the data within this period do not have to be used in training. The important thing is that there are samples from as many different stable states of the process as possible. These samples can be shorter in time compared to the whole period of data collected (Figure 33). The samples should guarantee that SOM learns enough normal states, such that more uncommon normal states are not classified as abnormal. However, since the process cannot be regarded static, new samples have to be collected and the map has to be retrained every now and then.

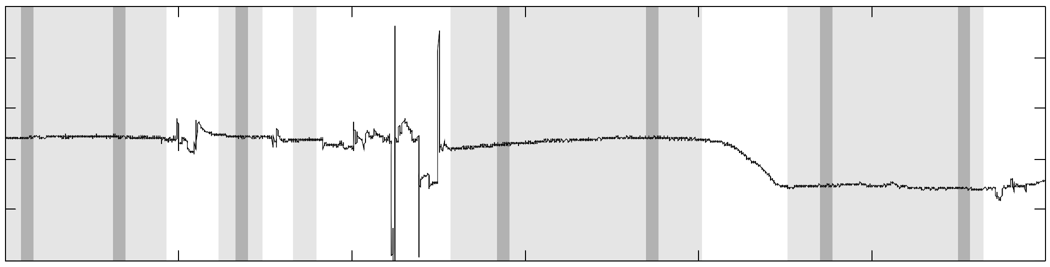

Figure 33. A principle of collecting stable samples from a long period of data. The short stable samples used for training are in the figure marked as dark gray periods along the study period. This way the same amount of variability can be contained in shorter total length data and the training is faster than with all the stable data (marked as light gray periods).

The data for training is retrieved from an archive database and the training is done separately from the on-line part of the monitoring system. The training environment can be e.g. Matlab or a totally separate program made specifically for the task. The information of the trained map is saved in a file to be used later on.

The on-line branch of the monitoring system starts with the map trained off-line. The tasks in on-line monitoring are to retrieve the current information from the process database (e.g. RapidBase [Wolski 2000]), extract the features needed, calculate a rough QE, and finally refine the abnormality detection by taking the control actions into consideration.

The feature extraction is the first task in the chain of on-line monitoring. The extraction is suggested to be done by an external program that retrieves the information needed from the process database, calculates the features and stores them back into the process database. The complexity of the features, like wavelet-components, requires the extraction to be done asynchronously. Thus, the process database does not have to wait for the extraction process to complete before carrying on its tasks.

The on-line monitoring is conducted by presenting the feature vector of a current state to the trained SOM. This is done by a program, which retrieves the definitions of the SOM from the file and then finds the best matching map unit of the current process feature vector at defined time intervals. This part of the system could be based on Matlab and SOM Toolbox, which are compiled to a stand-alone application. However, if the SOM Toolbox is used, one has to remember its use is limited in commercial products by the terms of GNU General Public License [Free Software Foundation 1991].

The comparison to SOM produces a rough abnormality detection. However, since the operator inferences, etc. result in large QE, the abnormality detection has to be refined with the aid of information about the set point changes, openings and closings of the machine sections, etc. stored in the process database. It is important that this kind of information is available in the process database, since SOM trained with stable data is sensitive to any kind of changes in the signals regardless of their normality or abnormality. Thus, the detection of abnormality is eliminated in refining if there is a change in the set-up values of the process.

Finally, if the current state is classified as abnormal after the refining, an announcement is made. Depending on the purpose of the monitoring, the announcement can be written on a log file, to archive database or echo on a screen.

8.2 Practical Issues

The training of SOM is made off-line because it needs some knowledge about the process and the signals. The most crucial part of the off-line training is the selection of the training data. The data has to fulfill the requirements listed in Section 5.1. Otherwise the trained map may respond also to the abnormal states with low QE thus complicating the monitoring.

If the information about the occurrence of the operator interference, process changes, etc. is available, the normal and steady training data can be selected easily. On the other hand without information of the times of occurrence of operator interference and other external events, the selection of proper training data has to be based on exhaustive study of the data available.

8.3 Limitations

The computational and memory capacities pose some limitations for the use of SOM. The computational requirements of SOM algorithm depend on the training parameters. However, the computational complexity of SOM algorithm is essentially $O(nmd)$, where m is the number of map units, n is the number of data samples and d is the dimensionality of the data [Vesanto 2000].

On the other hand, the memory limitations become substantial when using SOM Toolbox. The toolbox uses for batch training algorithm roughly $8\times(5md+4nd+3m^2)$ bytes [Vesanto 2000]. Thus, the training of prototype in Section 5.1 needed about 340 MB of memory. Further information about the properties of SOM Toolbox can be found in [Vesanto 2000].

The memory consumption can be reduced with a more careful selection of the training data. With sampling interval of 0.5 seconds the amount of data is huge for monitoring periods of days, weeks, or months. The long data collection periods are important for acquiring enough variability in the data. However, the steady states of the process do not vary much in the scale of hours, and not every sample is needed for the training set as stated in Section 8.1. By selecting data from variable states, but in shorter periods, a good enough SOM can be trained also with a smaller training set. This was not possible with in case study because of the lack data, but with longer period of study this should not be the problem.

9 Summary

The analysis of the report is based on measurements conducted during six days on a specific paper machine. The data is in three time series and every observation consists of 50 measurements of signals. By using feature extraction methods and expert knowledge, six interesting abnormal phenomena were found in the time series. These phenomena are in general related to periodic properties of the process and they are used to test monitoring system.

In off-line analysis of the data, the process is noticed to be non-stationary. This fact and a requirement for both temporal and frequency resolution led to selection of wavelet components as features to be monitored.

Next some state identification methods are described briefly. This review includes cluster analysis, decision trees, their fuzzy versions, multi-layer perceptron and radial basis function neural networks as well as self-organizing map (SOM) and adaptive resonance theory (ART) neural networks. Most of the described methods are insufficient for the problem on hand, but SOM and ART are more promising and they are prototyped and studied in more detail.

The advantages of SOM in the present case study are self-organization, good interpretability of the results, and considerably easy use. On the other hand, training data for the map has to be selected carefully, because SOM used should be trained with steady normal data. Another weakness of SOM is its inability to adapt changes in the process on-line. The map has to be retrained when the process changes significantly.

The results of prototyping SOM in monitoring are promising. Four of the six test phenomena are detected as increase in QE. However, rapid unmeasured changes in the process, like set-up point changes, openings and closings of machine sections, and other operator interventions, distort monitoring results producing false detections of abnormality. Thus, the operator-made interventions have to be stored with other measurements to eliminate the false detections in future.

In contrary to SOM, the main advantage of ART network is its ability to adapt changes of the process instantaneously. Moreover, ART does not need a specific training data or period before on-line monitoring. The weaknesses of ART are its sensitivity to scaling and noise. These are very apparent in the case of a complex process like paper machine. Against this background, it is not surprising that ART prototype does not perform as well as SOM. The number of detected states is high, but the test phenomena cannot be distinguished well enough within the detected states.

A suggestion for implementation of SOM monitoring systems is made. In the suggestion, SOM is trained off-line with data from an archive database. The training data is recommended to cover at least several months. The trained map is used on-line for monitoring by comparing the current state to the map vectors. The results of the comparison are refined using information about the operator interventions. In refining, the detections of abnormality resulted from operator interventions are eliminated. If the state is after the refining still considered as abnormal, a notion for an operator is made.

References

[Alander et al. 1991]
Alander, J.T.; Frisk, M.; Holmström, L.; Hämäläinen, A. & Tuominen, J. "Process Error Detection Using Self-Organising Feature Maps", in T. Kohonen, K. Mäkisara, O. Simula, J, Kangas (eds.), "Artificial Neural Networks", Vol. II, pp. 1229-1232, North-Holland, Amsterdam, Netherlands, 1991. ISBN 0 444 89178 1

[Alhoniemi et al. 1999a]
Alhoniemi, E.; Himberg, J. & Vesanto, J. "Probabilistic Measures for Responses of Self-Organizing Map Units", International ICSC Congress on Computational Intelligence Methods and Applications (CIMA'99), Rochester, N.Y., USA, June 22-25, pp. 286-290, 1999.

[Alhoniemi et al. 1999b]
Alhoniemi, E.; Hollmén, J.; Simula, O. & Vesanto, J. "Process Monitoring and Modeling using the Self-Organizing Map", Integrated Computer-Aided Engineering, Vol. 6, No. 1, pp. 3-14, 1999.

[Barro et al. 1998]
Barro, S.; Fernández-Delgado, M.; Vila-Sobrino, J.A.; Reguero, C.V. & Sánchez, E. "Classifying Multichannel ECG Patterns with an Adaptive Neural Network", IEEE Engineering in Medicine and Biology Magazine, Vol. 17, No. 1, pp. 45-55, 1998.

[Bishop 1995]
Bishop, C.M. "Neural Networks for Pattern Recognition", Oxford University Press, New York, 1995. ISBN 0 19 853864 2

[Breiman et al. 1984]
Breiman, L; Friedman, J.H.; Olsen, R.A. & Stone, C.J. "Classification and Regression Trees", Belmont, CA, Wadsworth, 1984.

[Carpenter & Grossberg 1987a]
Carpenter, G.A. & Grossberg, S. "A Massively Parallel Architecture for a Self-Organizing Neural Pattern Recognition Machine", Computer Vision, Graphics, and Image Processing, Vol. 37, No. 1, 1987.

[Carpenter & Grossberg 1987b]
Carpenter, G.A. & Grossberg, S. "ART 2: self-organization of stable category recognition codes for analog input patterns", Applied Optics, Vol. 26, No. 23, pp. 4919-4930, 1987.

[Carpenter & Grossberg 1988]
Carpenter, G.A. & Grossberg, S. "The ART of Adaptive Pattern Recognition by a Self-Organizing Neural Network", IEEE Computer, Vol. 21, No. 3, pp. 77-88, 1988.

[Carpenter et al. 1991]
Carpenter, G.A.; Grossberg, S. & Rosen, D.B. "ART 2-A: An Adaptive Resonance Algorithm fir Rapid Category Learning and Recognition", Neural Networks, Vol. 4, pp. 493-504, 1991.

[Chang & Pavlidis 1977]
Chang, R.L.P. & Pavlidis, T. "Fuzzy Decision Tree Algorithms", IEEE

Transactions on Systems, Man, and Cybernetics, Vol. 7, No. 1, pp. 28-35, 1997.

[Daubechies 1992]
Daubechies, I. "Ten Lectures on Wavelets", Society for Industrial and Applied Mathematics, Philadelphia, PA. 1992 . ISBN 0 89871 274 2

[Donoho et al. 1999]
Donoho, David; Duncan, Mark Reynold; Huo, Xiaoming & Levi, Ofer. "WaveLab 802 for Matlab5.x". [http://www-stat.stanford.edu/~wavelab/], 1999.

[Du et al. 1995]
R. Du, M. A. Elbestawi & S. M. Wu, "Computer Automated Monitoring of Manufacturing Processes: Part 1, Monitoring Decision-Making Methods," Trans. of ASME, J. of Eng. for Industry. Vol. 117, No. 2, pp. 121-132, 1995.

[Fernández-Delgado & Barro 1998]
Fernández-Delgado, M. & Barro S. "MART: A Multichannel ART-Based Neural Network", IEEE Transactions on Neural Networks, Vol. 9, No. 1, pp. 139-150, 1998.

[Free Software Foundation 1991]
Free Software Foundation. "GNU General Public License", Version 2, 1991. http://www.gnu.org/copyleft/gpl.html. Referenced on 14th September 2000.

[Friedman & Kandel 1999]
Friedman, M. & Kandel, A. "Introduction to Pattern Recognition: Statistical, Structural, Neural and Fuzzy Logic Approaches", Imperial College Press, London, 1999. ISBN 9810233124

[Fritzke 1991]
Fritzke, B. "Let It Grow – Self-Organizing Feature Maps with Problem Dependent Cell Structure", in T. Kohonen, K. Mäkisara, O. Simula, J, Kangas (eds.), "Artificial Neural Networks", Vol. I, pp. 403-408, North-Holland, Amsterdam, Netherlands, 1991. ISBN 0 444 89178 1

[Gallinari et al. 1988]
Gallinari, P.; Thiria, S. & Soulie, F.F. "Multilayer Perceptrons and Data Analysis", in IEEE Int. Conf. on Neural Networks, Vol. I, pp. 391-399, San Diego, CA, 1998.

[Hamad & Delsert 1995]
Hamad, D. & Delsert, S. "Nonlinear Mapping procedures for Unsupervized Pattern Classification", Engineering Applications of Artificial Neural Networks, Otaniemi, Finland, August 21-23, pp. 457-460, 1995.

[Harris 1993]
Harris, T. "A Kohonen S.O.M. based, machine health monitoring system which enables diagnosis of faults not seen in the training set", in Proc. of 1993 Int. Joint Conf. on Neural Networks (IJCNN'93), Nagoya, Japan, Vol. I, pp. 947-950, 1993.

[Iivarinen & Visa 1998]
Iivarinen, J. & Visa, A. "An Adaptive Texture and Shape Based Defect Classification", 14th International Conference on Pattern Recognition, Brisbane, Australia, August 16-20, Vol. I, pp. 117-122, 1998.

[Jain et al. 1999]
Jain, A.K.; Murty, M.N. & Flynn, P.J. "Data Clustering: A Review", ACM Computing Surveys, Vol. 31, No. 3, pp. 264-323, 1999.

[Janikow 1998]
Janikow, C.Z. "Fuzzy Decision Trees: Issues and Methods", IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics, Vol. 28, No. 1, pp. 1-14, 1998.

[Kaski & Kohonen 1996]
Kaski, S. & Kohonen, T. "Exploratory data analysis by the self-organizing map: Structures of welfare and poverty in the world", in Refenes, A.-P. N.; Abu-Mostafa Y.; Moody, J. & Weigend A. (eds.), "Neural Networks in Financial Engineering", pp. 498-507, World Scientific, Singapore, 1996.

[Kiviluoto 1996]
Kiviluoto, K. "Topology Preservation in Self-Organizing Maps", International Conference on Neural Networks (ICNN'96), Washington, D.C., USA, June 3-6, pp. 294-299, 1996.

[Kohonen 1995]
Kohonen, T."Self-Organizing Maps", Springer-Verlag, Berlin, 1995. ISBN 3-540-58600-8

[Lowe 1995]
Lowe, D. "Radial Basis Function Networks", in Arbib, M.A. (ed.) "The handbook of brain theory and neural networks", pp. 779-782, MIT Press, Cambrigde (MA), 1995. ISBN 0-262-01148-4

[Quinlan 1986]
Quinlan, J.R. "Induction of Decision Trees", Machine Learning, Vol. 1, No. 1, pp. 81-106, 1986.

[Rinta-Runsala 2000]
Rinta-Runsala, E. "Modus-Project Case Study ODT2. Drive System Monitoring: Requirements and Suggestions", Research Report TTE1-2000-19, VTT Information Technology, 2000.

[Sharma 1996]
Sharma, S. "Applied Multivariate Techniques", John-Wiley, 1996.
ISBN 0-471-31064-6

[Simula & Alhoniemi 1999]
Simula, O. & Alhoniemi, E. "SOM Based Analysis of Pulping Process Data", International Work-Conference on Artificial and Natural Neural Networks (IWANN'99), Alicante, Spain, June 2-4, pp. 567-577, 1999.

[Tryba & Goser 1991]
Tryba, V. & Goser, K. "Self-Organizing Feature Maps for Process Control in Chemistry", in T. Kohonen, K. Mäkisara, O. Simula, J, Kangas (eds.), "Artificial Neural Networks", Vol. I, pp. 847-852, North-Holland, Amsterdam, Netherlands, 1991. ISBN 0 444 89178 1

[Vesanto 1999]
Vesanto, J. "SOM-based data visualization methods", Intelligent Data Analysis, Vol. 3, pp. 111-126, 1999.

[Vesanto 2000]
Vesanto, J. "Neural Network Tool for Data Mining: SOM Toolbox", in Proceedings of Symposium on Tool Environments and Development Methods for Intelligent Systems (TOOLMET2000), Oulu, Finland, pp. 184-196, 2000.

[Vesanto el al. 2000]
Vesanto, J.; Alhoniemi, E.; Himberg, J.; Kiviluoto K. & Parviainen, J. "Self-Organizing Map for Data Mining in MATLAB: the SOM Toolbox", Simulation News Europe, No. 25, p. 54, 1999.

[Wolski 2000]
Wolski A. " RapidBase – mittaustiedonhallintajärjestelmä", Prosessiteollisuuden on-line mittaustekniikat, TEKESin teknologiaohjelman vuosiseminaari, 8.2.2000. [http://rapidbase.vtt.fi/]

[Zadeh 1988]
Zadeh, L. "Fuzzy Logic", Computer, Vol. 21, Iss. 4, pp. 83-93, 1988.

[Zhang et al. 1996]
Zhang, S.; Ganesan, R. & Xistris, G.D. "Self-Organising Neural Networks for Automated Machinery Monitoring Systems", Mechanical Systems and Signal Processing, Vol. 10, No. 5, pp. 517-532, 1996.

Appendix A

List of signals measured.

1 2	Power Mains voltage	Power [kW] Mains voltage [V]
3 4	Fan pump	Motor speed [rpm] Motor torque [%]
5 6	2nd press sym roll	Motor speed [rpm] Motor torque [%]
7 8	Dryer section paper lead in roll	Motor speed [rpm] Motor torque [%]
9 10	1st dryer section	Motor speed [rpm] Motor torque [TN%]
11 12	Yankee paper lead in roll	Motor speed [rpm] Motor torque [%]
13 14	Yankee applicator roll	Motor speed [rpm] Motor torque [%]
15 16	1st coating section bottom roll	Motor speed [rpm] Motor torque [%]
17 18	1st coating section top roll	Motor speed [rpm] Motor torque [%]
19 20	One nip calender bottom roll	Motor speed [rpm] Motor torque [%]
21 22	One nip calender top roll	Motor speed [rpm] Motor torque [%]
23 24	Machine calender bottom roll	Motor speed [rpm] Motor torque [%]
25 26	Machine calender top roll	Motor speed [rpm] Motor torque [%]
27 28	Wire turning roll	Motor speed [rpm] Motor torque [TN%]
29 30	Suction couch roll	Motor speed [rpm] Motor torque [TN%]
31 32	Former turn roll	Motor speed [rpm] Motor torque [TN%]
33 34	1st press suction transfer roll	Motor speed [rpm] Motor torque [TN%]
35 36	Yankee cylinder M1	Motor speed [rpm] Motor torque [TN%]
37 38	Yankee cylinder M2	Motor speed [rpm] Motor torque [TN%]
39 40	2nd dryer section	Motor speed [rpm] Motor torque [TN%]
41 42	2nd dryer section cylinder 8	Motor speed [rpm] Motor torque [TN%]
43 44	2nd coating section backing roll	Motor speed [rpm] Motor torque [TN%]
45 46	3rd dryer section	Motor speed [rpm] Motor torque [TN%]
47 48	Reel drum	Motor speed [rpm] Motor torque [TN%]
49 50	Center drive	Motor speed [rpm] Motor torque [TN%]