

ACTA

UNIVERSITATIS OULUENSIS

Perttu Laurinen

A TOP-DOWN APPROACH
FOR CREATING AND
IMPLEMENTING DATA
MINING SOLUTIONS

FACULTY OF TECHNOLOGY,
DEPARTMENT OF ELECTRICAL AND INFORMATION ENGINEERING,
UNIVERSITY OF OULU

C
TECHNICAL



ACTA UNIVERSITATIS OULUENSIS
C Technica 246

PERTTU LAURINEN

**A TOP-DOWN APPROACH FOR
CREATING AND IMPLEMENTING
DATA MINING SOLUTIONS**

Academic Dissertation to be presented with the assent of
the Faculty of Technology, University of Oulu, for public
discussion in the Auditorium TS101, Linnanmaa, on June
22nd, 2006, at 12 noon

OULUN YLIOPISTO, OULU 2006

Copyright © 2006
Acta Univ. Oul. C 246, 2006

Supervised by
Professor Juha Röning

Reviewed by
Professor Heikki Kälviäinen
Professor Heikki Mannila

ISBN 951-42-8125-X (Paperback)
ISBN 951-42-8126-8 (PDF) <http://herkules.oulu.fi/isbn9514281268/>
ISSN 0355-3213 (Printed)
ISSN 1796-2226 (Online) <http://herkules.oulu.fi/issn03553213/>

Cover design
Raimo Ahonen

OULU UNIVERSITY PRESS
OULU 2006

Laurinen, Perttu, A top-down approach for creating and implementing data mining solutions

Faculty of Technology, University of Oulu, P.O.Box 4000, FI-90014 University of Oulu, Finland,
Department of Electrical and Information Engineering, University of Oulu, P.O.Box 4500,
FI-90014 University of Oulu, Finland
Acta Univ. Oul. C 246, 2006
Oulu, Finland

Abstract

The information age is characterized by ever-growing amounts of data surrounding us. By reproducing this data into usable knowledge we can start moving toward the knowledge age. Data mining is the science of transforming measurable information into usable knowledge. During the data mining process, the measurements pass through a chain of sophisticated transformations in order to acquire knowledge. Furthermore, in some applications the results are implemented as software solutions so that they can be continuously utilized. It is evident that the quality and amount of the knowledge formed is highly dependent on the transformations and the process applied. This thesis presents an application independent concept that can be used for managing the data mining process and implementing the acquired results as software applications.

The developed concept is divided into two parts – solution formation and solution implementation. The first part presents a systematic way for finding a data mining solution from a set of measurement data. The developed approach allows for easier application of a variety of algorithms to the data, manages the work chain, and differentiates between the data mining tasks. The method is based on storage of the data between the main stages of the data mining process, where the different stages of the process are defined on the basis of the type of algorithms applied to the data. The efficiency of the process is demonstrated with a case study presenting new solutions for resistance spot welding quality control.

The second part of the concept presents a component-based data mining application framework, called Smart Archive, designed for implementing the solution. The framework provides functionality that is common to most data mining applications and is especially suitable for implementing applications that process continuously acquired measurements. The work also proposes an efficient algorithm for utilizing cumulative measurement data in the history component of the framework. Using the framework, it is possible to build high-quality data mining applications with shorter development times by configuring the framework to process application-specific data. The efficiency of the framework is illustrated using a case study presenting the results and implementation principles of an application developed for predicting steel slab temperatures in a hot strip mill.

In conclusion, this thesis presents a concept that proposes solutions for two fundamental issues of data mining, the creation of a working data mining solution from a set of measurement data and the implementation of it as a stand-alone application.

Keywords: data mining application development, data mining process, similarity measurement, spot welding, trajectory, walking beam furnace

Acknowledgements

This research was carried out in the Intelligent Systems Group (ISG) at the Department of Electrical and Information Engineering between the years 2000 and 2005.

I am deeply indebted to Professor Juha Röning for his able guidance and support throughout this work. I sincerely thank Academy Professor Heikki Mannila from the University of Helsinki and Professor Heikki Kälviäinen from Lappeenranta University of Technology for reviewing this work.

Thanks to all my colleagues with whom I have had the pleasure of working with over these years. Special thanks go to Heli Junno, Eija Haapalainen and Lauri Tuovinen - you are not only great colleagues and friends, but this work would have never been possible without your contribution. Jaakko Suutala, Janne Haverinen and Antti Tikanmäki deserve thanks for the fruitful conversations and good coffee. Last, but not least, thanks to Ilmari Juutilainen for our long-lasting friendship.

I have had the pleasure of cooperating with many partners during this research. The cooperation with Rautaruukki was the seed for this work and I am grateful to Anne Sepänen, Jorma Untinen and Harri Tuomela of Rautaruukki. I also wish to thank the whole consortium that participated in the SIOUX-project, especially Professor Norbert Link, Dietmar Zettel and Daniel Sampaio from the University of Applied Sciences in Karlsruhe, Michael Peschl from Harms & Wende GmbH and Claudia Stöhrle from Steinbeis-Europa-Zentrum. I hope the good cooperation continues in the future as well.

Sincere thanks go to all the financial supporters of this work, the National Technology Agency of Finland (TEKES), the European Union, Tekniikan edistämissäätiö (Foundation for Advancing Technology) and Oulun yliopiston tukisäätiö (Oulu University Scholarship Foundation). I am especially thankful for the time I spent at the Graduate School in Electronics, Telecommunications and Automation (GETA) between the years 2000 and 2004.

My heartfelt thanks go to my whole family. Thanks to my grandparents for being examples of joy and perseverance. Thank you mother, father and sister for so many things, but in particular for encouraging me in making my dreams come true. Thanks to my father-in-law for his invaluable assistance. And above all, thanks go to my two wonderful ladies - my wife Mari and daughter Sara, you are the light of my days.

Oulu, June, 2006

Perttu Laurinen

Abbreviations

API	application programming interface
ARMAX	moving average with exogeneous variable
ARX	auto-regressive with exogenous variable
ATM	asynchronous transfer mode
CRAFT	cooperative research action for technology
CRISP-DM	cross-industry standard process for data mining
DARPA	the defense advanced research projects agency
DBMS	database management system
DM	data mining
DSSA	domain-specific software architecture
DW	data warehouse
EEG	electroencephalography
FIR	finite impulse response
IEEE	institute of electrical and electronics engineers
JDBC	java database connectivity
JDM	java data mining
KDD	knowledge discovery in databases
kNN	k nearest neighbours classifier
LDA	linear discriminant analysis
LVQ	learning vector quantization
mm	millimeter
ODBC	open database connectivity
pc	principal component
PMML	predictive model markup language
QDA	quadratic discriminant analysis
QoS	quality of service
RMS	root mean square
SA	smart archive

SDMA	space division multiple access
SIOUX	intelligent system for dynamic online quality control of spot welding processes for cross-sectoral applications
SQL	structured query language
TEKES	national technology agency of Finland
TWI	the welding institute
TCP/IP	transmission control protocol/internet protocol
XML	extensible markup language

Contents

Abstract	
Acknowledgements	
Abbreviations	
Contents	
1 Introduction	11
1.1 Background	11
1.2 About the complexity of data sets and some examples of DM applications	13
1.3 Scope of the thesis	16
1.4 Contribution of the thesis	18
1.5 Outline of the thesis	19
2 From measurements to a data mining solution	20
2.1 The data mining process	20
2.2 About the interactions in the data mining process	25
2.3 Proposed data mining process for managing interactions	28
2.4 Comparing the semi-open approach to the closed and unclosed approaches	31
2.4.1 Independence between the different stages of the data mining process.	31
2.4.2 The multitude of algorithms easily applicable to the data.	32
2.4.3 Specialization and teamwork of researchers	33
2.4.4 Data storage and on-line monitoring	33
2.4.5 Time savings and computing costs	34
2.5 Case study: A data mining solution for spot welding quality control . . .	35
2.5.1 Pre-processing spot welding data	36
2.5.2 Feature extraction and modeling results	41
2.5.3 Non-destructive analysis of welding spots using Bayesian networks	41
2.5.4 Development of process similarity measurement techniques . . .	44
2.6 Related work	49
2.7 Discussion	50
3 From the solution to a data mining application	51
3.1 Introduction	51
3.2 Functional requirement analysis	52
3.3 The components of Smart Archive	54

3.4	The architecture and operation of Smart Archive	56
3.5	Case study: A data mining application for predicting temperatures of steel slabs	60
3.5.1	Description of the application and the data set	60
3.5.2	Work related to the application	63
3.5.3	Configuring SA for the application	65
3.5.4	The model used for predicting the temperatures	66
3.5.5	Results	67
3.5.6	A comparison of the two ways of implementing the application	73
3.6	Work related to Smart Archive	75
3.7	Discussion	76
4	Similarity detection and an efficient algorithm for the history sink	79
4.1	Principles of similarity and novelty detection	79
4.2	About trajectories and measuring similarities between them	81
4.3	Proposed algorithm	83
4.4	About the complexity of the algorithm	87
4.5	Empirical estimation of efficiency	88
4.5.1	Data from a walking beam furnace	89
4.5.2	Synthetic data set	91
4.5.3	Summary of the performance results	92
4.6	Related work	92
4.7	Discussion	94
5	Conclusions	96
5.1	Discussion	96
5.2	Summary	98
	References	100

1 Introduction

This chapter contains an overview of this work and data mining as a field of research in general. The reader is introduced to data mining in Section 1.1 with a popular overview that does not require much prior information of the field. To illustrate the benefits of applying data mining and the factors that make it challenging, Section 1.2 presents properties of the measured data set making the data mining process more complex and some application examples of recent advances. After these sections the contents of this work are presented. Section 1.3 describes the purpose and scope of this thesis and 1.4 presents the scientific contribution of this work. Finally, Section 1.5 outlines the contents of the rest of this thesis.

1.1 Background

Encyclopaedia Britannica (Britannica 2005) defines data mining as a ‘type of database analysis that attempts to discover useful patterns or relationships in a group of data’. That is an excellent short definition of the subject, but since this whole thesis is an attempt to cover different aspects of data mining, let us take a little more detailed look at it. Data mining has its background in the fields of statistics, databases and machine learning. It emerged as an independent field of research in the 1990’s (Mark 1996) and has matured since then. Typing the search term ‘data mining’ in the World Wide Web search engine Google in August of 2005 returned about 21,700,000 results. Furthermore, a full text search on the publication database IEEE Xplore returns 8,322 documents. A bar chart that presents the yearly amount of these documents is presented in Figure 1. The figure shows a growing trend, the first (and the only one that year) document was published in 1993, while by 2004 the amount had risen to 2,168 publications. Interestingly, both of the previous numbers representing the amounts of documents found were made available using data mining techniques developed for searching document contents.

From the definition given in (Britannica 2005) data mining projects necessarily involve databases¹ and data transformation algorithms. Today, collecting and storing data

¹Here the term database is used to mean an observed collection of data. The storage of the data can be implemented by using ascii text files, for example or an SQL-compliant database management system (DBMS).

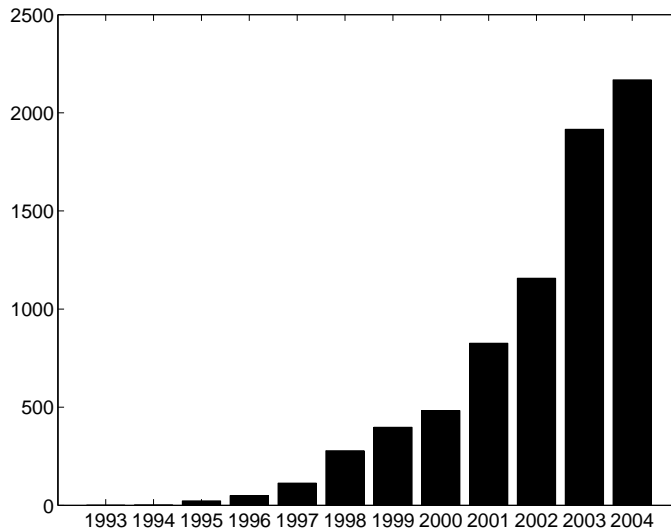


Fig. 1. A bar chart presenting the amount of publications including the phrase 'data mining' found from the IEEE Xplore publication database and sorted by year.

into databases is perhaps easier than ever before, thanks to the advanced measurement and data storage techniques available. As time passes, the measured data cumulates into large masses of raw data and transforming it into useful knowledge can become a very challenging task. However, the motive for measuring is often the desire to be able to extract useful knowledge that may not be directly measurable and thus gain insight into the phenomena under study. As an example, data from the daily variations of weather is collected into meteorological databases in order to anticipate upcoming weather patterns. Data mining is used to refer to the process of refining the measured data into useful knowledge - the process of storing and transforming measurable information with a variety of algorithms into knowledge.

Figure 2 presents a top-level overview of the steps needed to transform the measurements into knowledge. First, measurements are made from the studied phenomena and stored in a database. The data mining algorithms then access the data from the database and transform the measurements into knowledge, if the data mining project is successful. The concept of data mining is not any harder to understand than this - the same logic (with little alternations) can be found behind all studies involving the measurement of information.

The logic behind data mining might raise the question that if research in general is composed of designing experiments and analyzing measurement data and data mining researchers study similar things, what is the difference between data mining and research in some specific science? One answer to this question is that a researcher specialized solely on data mining is seldom an expert in the application that the data originates from. In that case, the data miner often lacks capabilities to develop the physical application behind the data any further, but can improve the application by providing useful knowledge of

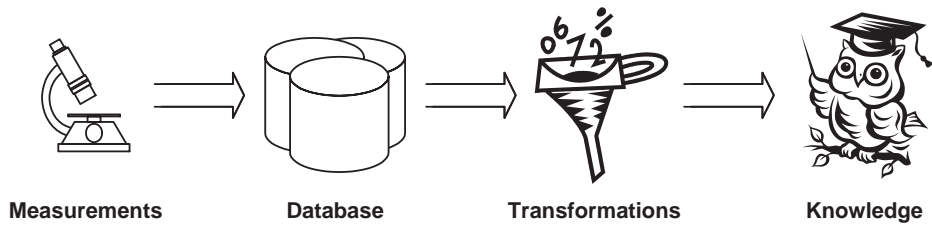


Fig. 2. A top-level overview of the data mining process.

its functioning. On the other hand, the researcher specialized in the application might not have the resources to specialize in the data mining process or algorithms and hence lacks the capability to extract the maximum amount of knowledge from the data. Because of this, data mining is at its best when a fruitful symbiosis between an application specialist and a data mining specialist is formed. During the collaboration process the application expert has the chance to expand his/her knowledge on the capabilities of data mining and the data miner has the interesting opportunity to learn more from an application that generates valuable data. Together they can create an improved version of the application under study. The next section presents some interesting applications in various fields where data mining techniques have been successfully applied.

1.2 About the complexity of data sets and some examples of DM applications

The set of applications producing measurable information is today larger than ever before and is expanding continuously. All measurement data are composed of properties common to most data sets and, in addition to this, application-specific properties, which vary from one application to another. The complexity of the measurement data is directly related to the complexity of the data mining process needed for extracting the desired knowledge from the data. In this section some of the factors that affect the complexity of the measured data, and therefore the data mining solution, are elaborated. The purpose is to be able to understand the challenges posed for finding and implementing data mining solutions for different types of applications. After this, a few examples of data mining solutions developed for real world applications in various fields are given.

The factors affecting the complexity of the measurement data set can be classified roughly into the following categories:

- *Amount of measured variables.* The amount can range from one to several thousands depending on the application. In most cases a large number of stored variables is an indication of multivariate dependencies in the data, which can make finding the solution more challenging. On the other hand, when measurements are available from a large amount of variables, the measurements may cover more data that is

relevant for the application.

- *Amount of data.* It can range from a few observations to practically infinity. A larger amount of data requires a larger storage medium and takes a longer time to process, but makes the results more reliable. If too few observations are available, the reliability of the acquired results is questionable. Also, the amount of measured data should increase with respect to the amount of measured variables.
- *Type of data acquisition.* The solution could be developed for a static data set or for a continuously acquired data set. In general, continuously operating data mining solutions are harder to develop than static ones. Data may originate from one source (for example a database table, a file or a system variable) or from multiple sources. The amount of sources increases the work needed for data integration and thus makes the data mining task more laborious.
- *Reliability of observed data.* Data that are more reliable give more reliable results. In worst case scenarios the data set may contain a lot of missing values, decreasing the quality of the acquired results. Measurement accuracy is also a factor that can have a high influence on the outcome - if the measurements are made in low resolution it is more difficult to generate high resolution results. For example, it is more difficult to predict temperatures in a decimal accuracy if the measurements are only available in decade accuracy.
- *Type of measured variables.* The variables may come in different formats (for example real numbers, textual, images, labels etc.) demanding different amounts of pre-processing.
- *Type of dependency.* In general, it is easier to model dependencies with a few variables than multivariate dependencies. The same applies to modeling linear- and non-linear dependencies; it is usually easier to find a linear dependency in the data than a non-linear.

The above list is not comprehensive in the sense that surely the reader can think of an unlisted factor that affects the complexity. However, hopefully the reader can also get an idea of the factors affecting the complexity of the process without delving deeper. Furthermore, the terms listed contain a lot of cross dependencies. This means that as one of the items is altered, it may also affect other items and therefore also the complexity of the process. For example, if the type of the relationship in the data is non-linear, the minimum amount of collected data should usually be larger than with a linear modeling task and the data mining transformations have to be suitable for modeling non-linear dependencies. These kinds of qualities are reflected throughout the entire data mining chain - different data sets demand different transformation chains.

Moreover, from the viewpoint of this work, data mining applications developed for applications that are producing continuously new measurement data are of special interest. Therefore applications developed for this genre are introduced next in order to motivate the reader on the usability of data mining in various application areas. It would have been interesting to study and present a comprehensive set of state-of-the-art data mining applications in different fields, but that is not in the scope of this thesis. Because of that, the following presentation is limited to an overview of some interesting solutions in different application areas.

In the field of medicine the benefits of data mining can be realized, for example, in faster and more accurate treatments and diagnoses and prevention. Methods that output

linguistic rules are especially useful in this field, since they can be more easily interpreted by the medical personnel (Kwoka *et al.* 2002). A rule-based method regulating the intensive care ventilator (delivering oxygen for the patient) was developed by (Kwoka *et al.* 2002). The method achieves performance comparable to specialized anesthetic controlling the ventilator. For the second example, an extension of a naïve Bayesian classifier was used to accurately recognize dementia patients in (Zaffalona *et al.* 2003). Early diagnosis of dementia can be crucial for the treatment to be effective. Finally, a radial basis function network was used for accurate and automatic detection of the epileptiform pattern (Nurettin 2005) from features extracted from the electroencephalography (EEG).

In the manufacturing of goods the goal is to produce better and more with less. The results of data mining can help in this by optimizing and automating processes. An example of an application area is the packaging of microelectronic devices. One of the critical factors affecting the packaging performance is the processing of stud bumps that interconnect the elements. An optimization of the gold stud bumping process has been obtained by using a neural network approach (Leo & Burce 2005). Another example of a neural network application is a solution developed for the optimization of newspaper color pictures printing process (Verikas *et al.* 2000). The system results in savings in ink and improves the quality of the pictures. Eighteen months of industry experience has proven the effectiveness of the method. In addition to these industrial examples, this thesis presents two applications from the field of manufacturing. One is aimed at improving the quality of resistance spot welding joints and the other the quality of steel slab heating. More information on these two applications is found in the following chapters.

Data mining methods in telecommunication systems have been applied for increasing the quality of data in communication channels among other things. Fuzzy logic based systems have been used to improve the quality of service (QoS) in TCP/IP networks (Chrysostomou *et al.* 2003) and estimating cell loss probabilities in ATM networks (Chandramathi & Shanmugavel 2003). The bit error rate of Space Division Multiple Access (SDMA) networks has been decreased using neural networks in (Benson & Carrasco 2001). Moreover, telecommunication networks form highly complex entities that are continuously processing large numbers of transactions. Maintaining such a system is a demanding task because there are so many possible causes for faulty operation that they cannot be anticipated in the development phase. Using data mining methods, patterns leading to a faulty operation can be identified in a pro-active manner. One of the earliest works on the topic was reported in (Sasisekharan *et al.* 1996). More recent results and treatment of the topic can be found, for example, from (Sandford *et al.* 2005) and (Parish *et al.* 2004) (with an emphasis on developing visualization of results).

Data mining can produce more accurate environmental information helping in defining its state and improving protection and conservation. Remote sensing methods have been developed for measuring the state of a large area with low costs, for example from a satellite or an airplane. Ocean components concentrations are detected using satellite pictures from sunlight reflections by a method utilizing genetic algorithms (Fonlupt 2001). Forestation levels are inventoried remotely by a kNN-clustering application (Haapanen *et al.* 2004). Air pollution levels in cities are predicted in advance using neural networks (Perez & Reyes 2001) and wavelets (Nunnari 2004), if a reliable estimate of the forthcoming pollution level is available, appropriate measures can be taken to act on it.

In service robotics the ultimate goal is to develop robots that act independently and

help humans in everyday tasks, which requires developments in multiple fields. Because of the high requirements set for the autonomous operation of robots in varying operating conditions, the data mining problems in this field are among the most challenging ones. Robot navigation methods have been developed using neural networks (Catarina & Bernardete 2003) and environment recognition by self-organizing maps (Yamada 2004). Robust methods for controlling vehicles in autopilot modes have been developed for example for a fighter aircraft in autopilot mode (Li *et al.* 2001) and ship steering (Yang *et al.* 2003). Among the applications that have received the most publicity in the field is the DARPA (The Defense Advanced Research Projects Agency, USA) grand challenge (Darpa 2005). The Grand Challenge of 2005 was to devise a robot car working in autopilot mode, and was capable of driving autonomously through a challenging 131.2-mile course in the desert in less than 10 hours. The winner (Stanford University's "Stanley") completed the track in 6 hours 53 minutes and was awarded 2 million dollars for it.

Context-aware applications are mobile applications that are capable of reacting upon the information originating from their physical surroundings. The goal of the applications is to derive important pieces of information for adapting to the circumstances and to help the user communicate in a meaningful way with the environment. The field is still young and there are not yet very many applications. Learning the routines of a user has been studied for example in (Pirttikangas *et al.* 2004), where methods for adapting profiles of a mobile phone using association rules are presented. Symbolic maps have been used to recognize the context (walking outside / inside, in office, in coffee room) of a user equipped with a mobile device by (Flanagan *et al.* 2002).

Web mining is the term used when referring to data mining the World Wide Web. Google (<http://www.google.com>) is a well-known example of a web mining system developed for finding relevant information. Current developments in web search engines try to incorporate more intelligence in them, like engines using fuzzy logic (Choi 2003), which do not necessarily contain the key words, but similar words. Another application area is, for example, mining web navigation patterns (Xing & Shen 2004). A report on the topic can be found from (Pal *et al.* 2002).

The examples above presented some interesting data mining applications in various fields of research. It is apparent from the set of examples that data mining techniques are abstracted from the application and can therefore be applied in practically all phenomena that produce measurable information. The common factor in all the applications is the urge to improve the understanding or functioning of the phenomena under study. The examples illustrated a tiny fraction of the existing applications and the potential of data mining. The interested reader can find more examples with very little effort from the World Wide Web or from publication databases.

1.3 Scope of the thesis

This work presents a comprehensive methodology for creating fully working data mining applications. The thesis presents an overview of how to create a data mining application starting from a situation in which a collection of measurement data is available. The work is presented by dividing the task into three subcategories: 1) forming the solution from the

measurement data, 2) implementing the found solution as a data mining application and 3) extending the application to operate in environments where new observations become available continuously. These three subcategories are independent from each other in the sense that any of them can be applied independently. Or alternatively, a complete data mining application can be built from scratch by following the instructions in the listed order.

First a methodology is presented that can be used for organizing the data mining process in such a way that it is more feasible to find a working solution for the phenomena under study. The developed process divides the task into different phases according to the process presented later in Section 2.1. The functionality of the process is then demonstrated with a case study where resistance spot welding signals are analyzed. The case study compares in detail the amount of work needed for utilizing pre-processed resistance spot welding signals using the developed process and the amount of work needed for utilizing the signals using a traditional data mining process. After this, the knowledge acquired using different models developed using the process are demonstrated. The results of the modeling task illustrate how to identify a welding process and how to predict the quality of a welding spot using the shape of the voltage and current signals measured during a welding event. The case study uses a static data set to demonstrate how an appropriate solution can be found before further implementation.

The second part of the thesis describes how to implement the developed solution as an independent application in an environment where new measurements are constantly available. First a component-based application framework, called Smart Archive (SA), designed for implementing data mining applications is presented. The implementation using the framework is based on the data mining process described in Chapter 2 and allows the applier to flexibly apply the data mining algorithms using a variety of tools. In addition to this, the framework includes a component that can be used to detect novel measurements. This component is especially useful when working with continuously expanding data sets. The architecture, the components, the implementation and the design principles of the framework are presented. A case study comparing the implementation of a data mining application build from scratch to an application built using the framework is presented. The application predicts the temperatures of steel slabs as they are heated in a steel slab reheating furnace.

Finally, technology for the component in the framework that makes it possible to utilize past and newly measured observations more efficiently is presented. Using the technology it is possible to form a measure of similarity between two multidimensional measurement series (trajectories). This information is useful when defining the novelty of a new observation, for example, and can help in finding similar observations from the past or when deciding if the observation is novel enough for storing it.

In conclusion, the goal of this thesis is to present a methodology and a framework for building fully working data mining applications from measurement data. Following the methodology it is possible to first efficiently find and test data mining solutions. After a suitable solution is found, it can be implemented as an independent data mining application using the software architecture and framework described in this work. The framework is especially suitable when implementing applications in an environment producing continuously new observations and possibly requiring a high level of customization.

1.4 Contribution of the thesis

This thesis contains a contribution in all the parts it contains and as a whole. This section first lists the contribution in each of the individual parts and then of the work as a whole.

During the data mining process the measurement data is passed through a chain of algorithms that constitute the data mining process. This work proposes a method for better management and implementation of the data mining process and reports a case study of the method applied on the development of a solution for a spot welding application. The approach developed here enables a more systematic processing of data and facilitates the application of a variety of algorithms to the data. Furthermore, it manages the work chain and differentiates between the data mining tasks. The proposed way of managing the data mining process is discovered to especially suit team-oriented data mining tasks in which a group of researchers are forming a solution for a data mining problem.

The case study of applying the method to a resistance spot welding quality estimation project illustrates the advantages of the method compared to the traditional run-at-once approach. Solutions for predicting the quality of a welding spot and solutions capable of identifying the welding process were created using the process. The application-specific results were formed in cooperation with the workers participating in the project (Heli Junno, Eija Haapalainen, Lauri Tuovinen and our colleagues in Karlsruhe) and, to be sure, the sole purpose of presenting them in this work is to present the applicability of the developed data mining process, not to delve into the details of resistance spot welding improvement. Furthermore, these results may be used later in the theses of colleagues. The proposed method of managing the data mining process and the case study have also been partly reported in (Laurinen *et al.* 2004b). The application-specific results acquired using the process have been reported in (Laurinen *et al.* 2004a, Junno *et al.* 2004a,b, 2005, Haapalainen *et al.* 2005). Work reported in (Haapalainen *et al.* 2006) has been submitted for evaluation.

The contribution in the second part of the thesis involves the methods developed for implementing the data mining solution in an environment producing continuously new observations. Using the novel application framework reported in this work, it is possible to build high-quality applications with shorter development times by configuring the framework to process application-specific data. The advantages of a framework-based implementation are demonstrated in a case study which compares the framework approach to implementing a real-world application with the option of building an equivalent application from scratch. A data mining application that is able to accurately predict the post-roughing mill temperatures of steel slabs while they are being heated is developed. The application anticipates the core temperature of a steel slab before it exits a steel slab re-heating furnace, which is a solution that has not been previously available on this scale. This information can be used to help heat the slabs more accurately to preset temperature targets. The results of the model have also been reported in (Laurinen & Rönning 2005, Laurinen *et al.* 2001) and the application framework has been partly reported in (Laurinen *et al.* 2005).

The contribution of the component of the developed framework that enables the utilization of past measurement data is an efficient algorithm developed for calculating the similarity of two measurement trajectories. The algorithm is useful for calculating the distance between trajectories where the measurements have been observed in varying in-

tervals and contain one increasing measurement dimension, for example time. The algorithm outperforms the existing algorithms under these conditions. The algorithm has been partly reported in (Laurinen *et al.* 2006).

The previous paragraphs reported the contribution of this thesis on the parts that make up this work. However, the most significant contribution of the work is the compilation of the developed and applied techniques into a single entity. The novelty of the entity lies in the fact that, arguably, no consistent methodology for creating data mining applications has been put forward in such detail to date. This contribution is significant, especially in developing data mining applications applicable to on-line measurement data. Following the presented methodology it is possible to build a highly customized data mining application working in an on-line environment.

1.5 Outline of the thesis

The contents of this thesis are organized in such a way that it forms a logical entity. The reader can follow through the thesis starting from Chapter 2 and get an overview of how to build a data mining application starting from the acquired measurement data.

Chapter 2 elaborates on how to form a data mining solution from a collection of measurement data. It explains the method for managing the data mining process developed in this work, a case study in applying the process on a resistance spot welding application and the results generated using the process. After this, Chapter 3 presents the software architecture and framework developed for implementing a solution as a data mining application.

In Chapters 2 and 3 the tools required for creating a fully working data mining application were presented. In addition to this Chapter 3 reports an example of a data mining application that has been implemented using the developed framework for implementing data mining solutions. The application is based on data collected from a steel mill and the results of the application are presented in detail. A comparison between implementing a solution from scratch to implementing it using the framework is presented. After that, Chapter 4 describes a technique that can be used to utilize past measurement data, the presentation focuses on describing the developed algorithm and illustrates the efficiency of the method using the steel slab data set and an artificial data set. Finally, Chapter 5 discusses the presented concept and summarizes the work.

2 From measurements to a data mining solution

This chapter introduces a systematic way for finding a data mining solution from a set of measurement data. The developed approach, called semi-open data mining process, enables more systematic processing of data. It verifies the accuracy of the data, facilitates the application of a variety of algorithms to the data, manages the work chain, and differentiates between the data mining tasks. The method is based on the storage of the data between the main stages of the data mining process. The different stages of the process are defined on the basis of the type of algorithms applied to the data. The stages defined in this research consist of the measurement, preprocessing, feature extraction, and modeling stages. An easy-to-apply method for implementing and managing the work flow of the data mining process is presented, which should make it more feasible to find a properly working solution to a given problem.

Section 2.1 covers the general properties of the concept of obtaining a data mining solution and implementing it as an application. The section presents the commonly used classification of algorithms for refining the measured data into knowledge and relates it to the solution and implementation tasks. Furthermore, the section presents work related to the proposed concept. After that, different ways of organizing the algorithms that constitute the DM process are presented in Section 2.2 and the proposed process is presented in Section 2.3. The proposed method is compared to the other approaches of managing the solution formation phase in a comparative analysis in Section 2.4 and a case study in Section 2.5. Finally, work related to the data mining process is presented in Section 2.6, with a discussion in Section 2.7.

2.1 The data mining process

This section discusses the general properties of the process of finding a solution to a given data mining problem and implementing it as an application. A data mining solution consists of transforming the measurement data with a variety of algorithms in order to discover useful knowledge, as stated in the first chapter. The process of creating and implementing the solution is all about organizing this process so that it becomes more manageable and can be controlled more easily. In order to create an organization that is

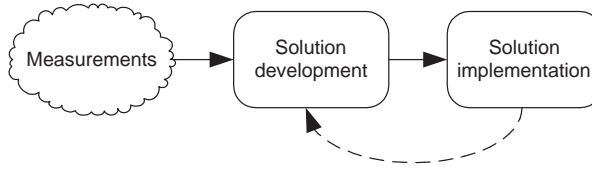
suitable for most data mining processes, a process that is sufficiently abstract, with common properties for all data mining projects, needs to be defined. Thereafter the process can be tailored to the specific needs of individual data mining projects.

Figure 3(a) shows one way of organizing the application development process at the topmost level. The figure presents a straightforward flow of tasks where a data mining solution is first created based on the measurement data and thereafter the found solution is implemented. It is desirable that the implementation can be updated based on further research results (marked with the dashed line in the figure). The work spent on seeking the solution consist mainly of determining the most suitable algorithms needed for extracting knowledge from the data. The implementation work consists of designing and implementing software that contains the functionality of the best solution. In some data mining projects it is enough just to extract the knowledge from the measurement data and report the results, ignoring the implementation. In these cases the topmost view of the process consists only of the solution development phase.

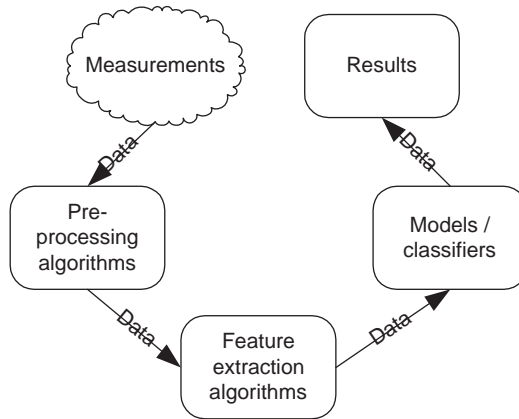
When expanding the process to the next level, it has been observed that the quality and success of the solution is determined by the outcome of the chain of algorithms that the measurements are passed through. A common practice is to categorize the individual data transformation algorithms into different classes based on the stage of the data mining process that they are applied in. The typical stages of the processing chain are identified to be the acquisition of measurements, pre-processing, feature extraction and modeling. Figure 3(b) presents the layout of this transformation chain. From now on, this basic layout of the different phases of the data mining process is called the "reference architecture" because it can be found behind most data mining solutions. Moreover, the reference architecture can be found behind both phases of the application development process, the solution and the implementation. This is because the algorithms that form the data mining solution are organized using the reference architecture and thereafter the implementation is based on the found solution. Now it is evident that the reference architecture is an abstraction that can be used for both, creating a data mining solution and implementing it as an application. This thesis presents and develops this concept. This chapter studies the process of searching the data mining solution in more detail and Chapter 3 continues from there, elaborating how the found solution can be efficiently implemented. It is argued that the concept has not been presented so far anywhere else to the level of detail exhibited in this work.

There exist previous research results in all of the subfields this work is composed of, but it is hard to find anything similar that would integrate them into a concept similar to this. In order to be able to report other concepts in the field, the term "concept" should be first defined in this context. Unfortunately the term is so abstract that it is somewhat ambiguous. However, a sufficient overview can be accomplished by studying previous work reporting approaches for performing data mining projects and presenting implementation principles for some parts of it. There were hardly any concepts found from the previous work in the field falling in the same class as the ones presented, presenting the complete process, taking the project from data to a deployable application.

There are many studies presenting data mining approaches at a level where descriptions on the implementation of the DM application development process are not given, but a more general organization of the tasks of the DM solution formation process is elaborated. Many of these studies include the reference architecture and some additional



(a)



(b)

Fig. 3. An overview of data mining solution development. (a): Topmost view of the steps in data mining application development. (b): The data mining process characterized using the classification of algorithms into different stages. This layout is called the reference architecture.

phases in them, depending on the study. One of the earliest and best known references on the subject is Brachman & Anand (1996). They define the process of knowledge discovery in databases (KDD) to include the phases of data cleaning, model development, data analysis and output generation. A domain model for the analyzed phenomena is developed using a variety of tools assigned to these stages. These results are important, especially at the time of their publication, but the work does not take a more detailed stand on the organization of algorithms or their interactions and is not related to application development. Pyle (1999) defines the data exploration process to include the stages of exploring the problem space, exploring the solution space, specifying the implementation method and mining the data. The primary contribution of the work is in data pre-processing and it does not contribute to the organization of the process on a level more detailed than that (nor on the application deployment phase). Chapman *et al.* (2000) defines the CRISP-DM reference model to include the phases of business understanding, data understanding, data preparation, modeling, evaluation and deployment. This concept comes closest to the one presented in this thesis. The largest difference to the other approaches reported in the literature is that CRISP-DM also gives ideas on the deployment phase, noting that the

creation of the model is generally not the end of the project and that in many cases it is the customer who carries out the final implementation. CRISP-DM provides guidelines for the deployment phase in a very general form, in four different stages: planning the deployment, planning monitoring and maintenance, producing a final report and reviewing the project. The work does not present guidelines or detailed views on the practical implementation of any of the subjects, but the advice and guidelines are of high quality. What further distinguishes this work is that this work (Smart Archive) gives detailed guidelines on the implementation of the application, but not on the quality assurance of the deployment. These important advice can be found from CRISP-DM, although on so general a level that they are hardly of any practical use - the work just briefly explains "what", not "how" or "why".

As can be seen, these studies have a view into the DM process that is extended from the reference architecture-centric view presented in this thesis. This means that they also include phases like problem definition as part of the concept, whereas the approach presented in this thesis concentrates on the core parts of the process in which the actual DM algorithms are applied, which are the parts that can be concretely implemented and deployed. The other parts are more application dependent and less technical, in the sense that they are directly involved with the application in the definition phase of the DM project where more human interaction is needed between the different parties managing the project. After the project definition is ready, it is the task of the data miner to find a working solution and implement it - this is where the reference architecture is used. Therefore this thesis did not comment on the tasks in the application interface outside the reference architecture. It would certainly be interesting to study application independent methodologies for the other phases as well, but as has been stated, even though they are very important, they are not in the reference architecture that is the backbone of this thesis and are hence excluded from this work.

It is logical to define the term "approach" or "concept" in an application- or algorithm-independent manner. In spite of this, some authors define the term as an approach using some specific modeling technique for extracting knowledge. An example can be found from Man & Kwong (2000) where approaches using decision trees, genetic programming and some other methods are presented. This differs quite fundamentally from the way the term approach is viewed and presented in this thesis, where the approach is abstracted from specific data transformation techniques. It is also surprising to see that some studies even manage to neglect to report the concept of data mining and concentrate merely on case studies or individual methods even though the title would imply otherwise, as for example Parag (2003) and Kudyba (2004). Could this be a sign that the general concept is not yet strong or uniform enough? Is the field still so young or diverse that the concept is not yet well established or standardized?

There are on-going attempts to creating standard notations for presenting DM models and data. Two notable projects are the Predictive Model Markup Language (PMML) by Savasere *et al.* (2005), and Java Data Mining (JDM) by Hornick (2005). PMML is an effort to standardize the components of a DM process using XML-format to describe models and data. Based on the examples provided with PMML Version 3.0, the approach seems to be very suitable for presenting models, but it is debatable how well it suits for presenting data sets i.e. how well XML works for presenting large sets of data. JDM is an Application Program Interface (API) developed specifically for Java for interacting

with DM algorithms. JDM provides interfaces and some implementations of algorithms working through these interfaces. The API offers the possibility of using algorithms implemented by third parties, as long as they support the API. Neither of these projects deal with the DM process or requirements related to the DM application development. Nevertheless, it is obvious that these methods would be very suitable tools to be used at the practical implementation level.

The roots of data mining probably lie more in statistics and computer science than in mathematics. But then again, mathematical notation can be used for formalizing the DM process. A substantial contribution to the subject using a mathematical approach has been developed under a school of research studying rough sets. Rough sets based approaches to different aspects are available in several sources. Grzymala-Busse & Ziarko (2003) relate the concept to data mining and define the term as a formal mathematical theory modeling knowledge of a domain of interest in a form of equivalence relations. They note that the main application area is in creating computer-processible models from data - which sounds promising. The technique is based on presenting information using approximately defined sets in which the approximations are the upper (the smallest possible set containing the target set) and lower (the largest possible set contained in the target set) approximations of the target set in question. If a method, for example a classifier, is presented using rough sets and the approximations are good enough, the prediction error will be small. There are plenty of tools for implementing DM processes based on rough sets. An extensive list of tools and research papers is available in the rough set database system Suraj & Grochowalski (2005), maintained by the rough sets society. Examples of these are "KDD-R: rough sets-based data mining system" (Ziarko 1998) and "LERS: A knowledge discovery system" (Grzymala-Busse 1998). The first is a prototype system aimed mostly at calculating probabilistic rules and the latter is designed for inducing rules from data sets. Some of these systems could be extended to present the concept presented in this thesis. Rough sets are suitable for modeling knowledge and transformations formally, but may not be that good for designing and presenting the organization of the sub-tasks of the DM process, architecture, application framework or elaborating the implementation. This is largely due to the fact that this is not the purpose of rough sets. Also, parts of this thesis could be reported using the terms and language adapted from the rough sets theory and it might be interesting to make further studies in this area. However, rough sets theory is a mathematical formalism and a language of its own and it is good to keep in mind that a description written in that language is understandable by a limited audience only.

Other techniques related to this work include, for example, an approach based on flow graphs and approaches based on granular computing. Flow graphs are used for modeling information streams between entities. That is also a high level abstraction of what the DM process efficiently is. Zdzislaw (2005) is a somewhat vague, bottom-up attempt at presenting the concept of data mining using flow graphs. The work does not relate the individual algorithms or variables it presents to the DM process more generally, but there could be a lot of potential in it. Granular information processing and its relation to data mining are explained in Pedrycz (2000), for example. The approach is based on presenting the information with granules abstracted from the data. The size of the granules is dependent on the application. Basically, the DM process could be explained also using this notation.

As can be seen, there exist many methods and techniques studying the field from dif-

ferent perspectives. It would have been interesting to find a study presenting a concept describing the process of moving from measurement data to an application at the same level of detail as this work. In spite of extensive searches, such a concept was not found. Therefore, this work was also related to what has been done and what could be done in this area, too, using the existing methods and techniques. At the conceptual level the CRISP-DM (Chapman *et al.* 2000) is one of the few and maybe the most similar in this sense to this work.

2.2 About the interactions in the data mining process

The process presented by the reference architecture starts by pre-processing the measurement data with algorithms designed, for example, to identify missing values, combining data originating from multiple sources and extracting idle periods from time-series signals. The feature extraction algorithms are then used to transform the data set into a selective subset of variables. The features formed can be for example information that is difficult to measure directly or can be formed only after all the measurements that an observation is comprised of has been made. Examples include descriptive values of signals (e.g. minimum, maximum), variables formed based on other variables (e.g. body mass index) and reduced dimension of the observation space in general. After forming the features meeting the requirements set for the application under study, the modeling (also called classifying in some contexts) algorithms are applied on the feature data. The modeling algorithms implement the final step in the process and output the knowledge that can be further exploited. Examples of modeling algorithms are neural networks, statistical classifiers and regression methods, among others. Good and extensive descriptions of the various methods and principles of applying them in data mining projects can be found from textbooks and articles, for example Hand *et al.* (2001), Hastie *et al.* (2001) and Martinez & Martinez (2001).

It has been observed that in some applications it is enough to apply the modeling algorithms directly on the measurement data and ignore the pre-processing and feature extraction steps. In the context of the reference architecture this can be described as a special case where the pre-processed and feature data equal the measurement data. It has also been noticed that this thesis does not delve into the field of acquiring or planning the collection of measurement data, since it is highly dependent on the application under study and is quite independent from the rest of the steps in the data mining process. The assumption is made that the data is readily available; it can be stored, for example, in a relational database or flat file system.

The reference architecture can be found in the underlying process of obtaining most of the existing data mining solutions. Researchers are organizing their data mining chain (whether on purpose or unintentionally) according to this process. From the viewpoint of the process defined by the reference architecture, there are two factors that affect the quality of the outcome when seeking a suitable data mining solution (ignoring the effect of the quality of the observation data):

1. the selection of the data mining algorithms,
2. the method of managing the interactions between the algorithms.

The existing research has concentrated largely on studying the algorithms that filter the data into knowledge, while less attention has been paid to the methods of managing the interactions between these algorithms. However, the management of the interactions between the algorithms should not be neglected. The way of interaction can be thought of as an infrastructure on which the algorithms run and when it is well managed, it gives the practitioner better possibilities for developing the data mining process. Using a well-managed interaction method, the practitioner can test extensive sets of data mining algorithms and better assess the quality of individual algorithms; hence the overall quality of the solution can be expected to be higher.

One of the traditional approaches of implementing the interaction has been to combine the algorithms developed for the different stages, run the data through the chain in a single run and observe the output. This approach has its advantages when everything functions smoothly, but may lead to suboptimal performance if some of the algorithms in the chain fail.

In this section the algorithms and interactions that constitute the data mining process are formalized using notations familiar from set theory for being able to present and study the effect of interactions between the algorithms more clearly. Let us start by marking the measurement data set with X_0 and the set of all possible data mining functions transforming data from one form to another with \mathbf{F} . Now the data mining task is to select an ordered tuple \mathcal{F} (where $\mathcal{F} \subset \mathbf{F}$) so that it gives the most satisfying mapping from X_0 to X_n , where X_n marks the result of the data mining process. Because there are $n - 2$ intermediate results between X_0 and X_n the cardinality of \mathcal{F} is n and the individual functions transforming data can be marked with f_1, f_2, \dots, f_n , where $f_i \in \mathcal{F}, i = 1, \dots, n$. In order to define the data mining process using this kind of notation, one more symbol is introduced - the special operation of storing the transformed data is represented with the symbol s_i , where i is an index showing the number of the data storage operation. The data mining process can be stopped and continued from a storage location, in this sense the storage location is like a "pause" button for the process. Furthermore, the practitioner is assumed to be able to observe the (intermediate) results of the process only from a storage location.

The reference architecture (presented in Figure 3(b)) can be described in this context by categorizing the ordered tuple \mathcal{F} into smaller sets of ordered tuples, $\mathcal{P}, \mathcal{FE}, \mathcal{C}$, where the symbols represent the tuples for pre-processing, feature extraction and modeling transformations, respectively. Figure 4(a) presents an implementation of the data mining process using the developed notation. The circles in the figure signify operations and the arrows indicate data transfers between the operations. The first algorithm in the chain, f_1 , accesses the stored measurement data, which is then transformed using the algorithms f_i ($i = 1, \dots, n - 2$), until the last algorithm in the chain f_n outputs its results, which are stored for later use and observation (s_1). For clarity, the intermediate transformation results (X_1, \dots, X_{n-1}) have not been drawn in the figure. The classification of the functions based on the reference architecture is also presented in the figure, although the results of the pre-processing and feature extraction stages cannot be observed, since the only storage location is the end of the process.

Using this method of processing measurement data, it is assumed that the data is processed from the storage location, without intermediate storage points, until the end of the chain has been reached. Consequently, in order to obtain the results of the function f_i in the chain, all the functions prior to it have to be calculated. This makes the results of the

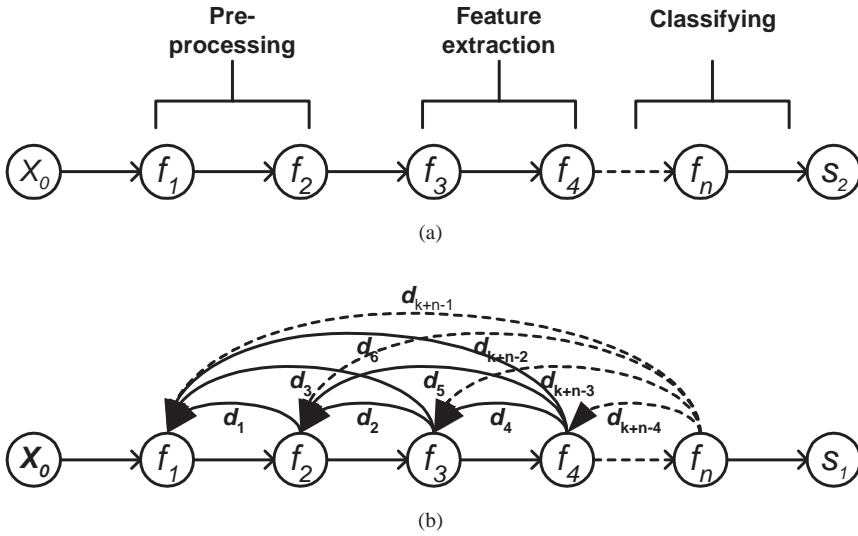


Fig. 4. The closed data mining process and its dependencies. (a): The closed data mining process. The transformed observations are not stored until the end of the chain. (b): The dependencies between transformations in the closed data mining process are illustrated using the arrows pointing backwards.

functions highly dependent on the performance of the previous functions in the chain. As a result, for the chain to be able to produce a correct output, all the algorithms have to function properly. Therefore, when the data miner is using this approach and wishes to observe results of the process, only the final results X_n (stored using the operation s_1) of the process can be observed. Because of this, it can be a very challenging task to identify functions in the chain that are possibly not functioning correctly. For obvious reasons, this method of processing the data shall hereafter be referred to as the "closed data mining process".

Figure 4(a) does not comment on the amount of dependencies between the functions. Figure 4(b) illustrates the dependencies between the algorithms when using the closed data mining process for managing the connections. The dependencies between operations are marked with the curved arrows with black tips. As we can see, the function $f_i (i = 1, \dots, n - 1)$ is dependent on the output of the previous $i - 1$ functions in the chain. In general, the cumulative number of dependencies (the amount of arrows pointing backwards) at the i th function in the chain is $\sum_{j=1}^{i-1} j$.

What could help make the functions less dependent on the direct output of earlier functions in the chain and give the data miner the possibility of observing intermediate results? One solution is to add intermediate storage points in the data mining chain. Figure 5 presents the other extreme of approaching the implementation of the data mining process, hereafter referred to as the "open data mining process". Here, the data is stored after the application of each function in the process. Now the operation f_i in the chain is

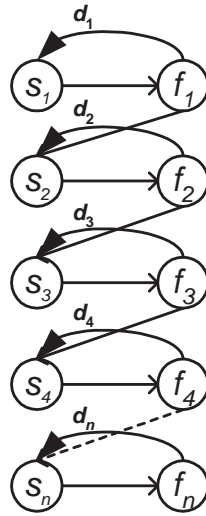


Fig. 5. Open data mining process. The transformed observations are stored after every transformation.

directly dependent only on the data read from the storage point s_i , and the data miner can evaluate the performance of each function in the chain. As a consequence, in order to be able to calculate the function f_i ($i = 1, \dots, n$), one does not need to calculate all the functions f_1, \dots, f_{i-1} prior to f_i but just to retrieve the data, X_{i-1} , stored using the operation s_i and to apply the function f_i on that data. The cumulative amount of dependencies for function f_i in the chain using the open data mining process is i .

The obvious difference between these two processes is that in the latter the result of the function f_i is dependent only on the data, X_{i-1} , while in the former it is dependent on X_0 and the transformations f_1, \dots, f_{i-1} . The difference between the interaction methods, or approaches, might seem small at this point, but the following sections will demonstrate how large a difference it can make in practice.

The third alternative for implementing the data mining process using this notation would be to develop categorizations of the functions and place storage locations between the sub-chains of functions. Using this approach the question would be how to divide the process into sub-chains of functions? However, the reference architecture already provides a logical categorization of the function chain. That is the topic of the next section, which proposes a solution for this challenge.

2.3 Proposed data mining process for managing interactions

The data mining process proposed in this thesis is a cross-over of the closed and open interaction methods presented in the previous section, therefore it is called "semi-open

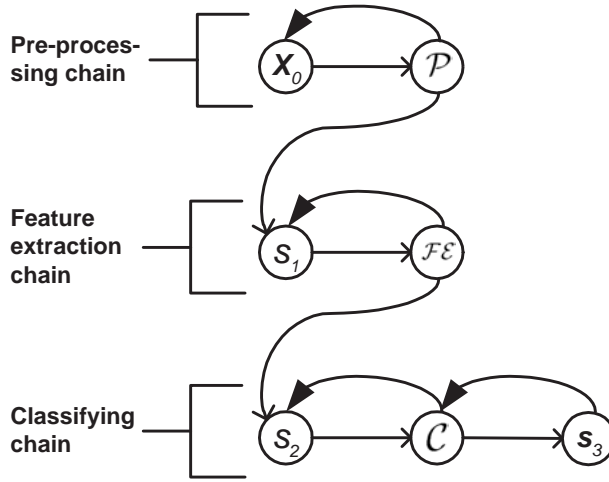


Fig. 6. The proposed semi-open data mining process. The transformed observations are stored after the main phases of the reference architecture.

data mining process". The approach groups the functions into logical order reflecting the reference architecture and lets the applier observe intermediate results of the process.

Figures 6 and 7 present the semi-open data mining process. The functions transforming data are ordered into sub-chains, denoted by \mathcal{P} , \mathcal{FE} and \mathcal{C} . These sub-chains present the categorization of functions in the reference architecture, that is, pre-processing (\mathcal{P}), feature extraction (\mathcal{FE}) and classification (\mathcal{C}) operations. The data storage operations, too, are now presented in two categories: local and global storage operations. The data stored using a local operation are visible to the functions inside a chain of functions and the data stored using a global operation is visible to all functions (including functions in other chains). After the data have been processed in a sub-chain it is stored (s_1 , s_2 and s_3) and the results are viewable globally¹. The storage points defined in this process are placed after the pre-processing, feature extraction and classifying chains.

Each sub-chain in the process is organized as shown in Figure 7. An arbitrary number of functions is laid in a chain processing the data (X_i) input into the sub-chain. The sub-chain is connected to the world outside by connections to the data it processes (X_i) and to the storage location (s_i) where it stores the results. The sub-chain contains 0 to $(n_1 - 1)$ local storage operations (l_i), which enables the observation of the results of the sub-chain, but are not visible to functions outside the chain.

The arrows with black triangle tips are marking again the dependencies between the operations in Figures 6 and 7. The sub-chains of functions are dependent only on the data stored after the previous sub-chain in the process has processed its output. That is, the pre-processing stage is dependent only on the measurement data, the feature extraction stage

¹Here the term "globally" resembles the same term used in programming i.e. the data is accessible globally by all appliers.

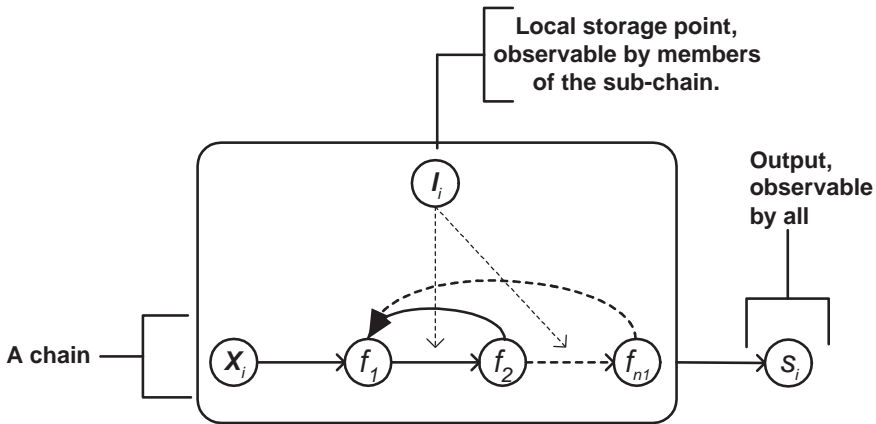


Fig. 7. Intermediate results from the sub-chains can be observed by placing a temporary data storage point in an appropriate position.

on the pre-processed data, and the classification stage only on the feature data. Therefore, the number of global dependencies is always four and is independent of the amount of dependencies inside the sub-chains. The sub-chains contain local dependencies, similar to the dependencies in the processes presented in Figures 4(b) and 5 depending on the amount and placement of the local storage operations (I_i). If a local storage operation is placed after each function in the chain, the sub-chain is organized as the open data mining process (Figure 5) and if no local storage points are present, the sub-chain is like the closed data mining process (Figure 4(b)).

Why is this layout of the data mining process any better than the closed or unclosed layouts? It can be considered better for a number of reasons. First of all, the reference architecture is clearly present in this layout. All the researchers involved in data mining are familiar with the reference architecture because it is the de-facto procedure for organizing the data mining chain. Reflecting it in the layout lowers the threshold for getting familiar with the data mining process in hand. That is, a person not familiar with the underlying application can still understand the management of the data mining process with a little effort when studying a process implemented with the familiar reference architecture in mind. Furthermore, the semi-open process supports the application of a variety of algorithms on the data and differentiates between the data mining tasks. In particular, the ease of application of a variety of data mining algorithms is among the most important properties when seeking for a data mining solution. That is because the quality of the solution is highly dependent on the quality of the set of algorithms applied on the data. In that sense the selection of the data mining process is analogous to selecting the training algorithm of a classifier - with a properly designed data mining process, the optimal (or near optimal) solution can be found in less time than with a less effective one.

2.4 Comparing the semi-open approach to the closed and unclosed approaches

In this section, the proposed semi-open approach is compared to the closed and unclosed approaches. In general, the open and semi-open processes resemble each other more than the closed process. They both contain intermediate storage points and, in practice, the semi-open process can be made almost similar to the open process by placing local storage points after all the functions in the sub-chains. The main difference between the open and semi-open processes is that the semi-open process is more transparent. It categorizes the functions in clear order according to the reference architecture, while when using the open process, it is harder to draw lines between the different stages of the process. On the other hand, the closed process is a kind of black box, measurements go in and results come out, without the possibility of observing intermediate actions. The following subsections present a more detailed comparison using different factors. The comparison is especially suitable for considering the practical applicability of the processes. The closed approach is marked using (I), the open approach with (II) and the semi-open approach with (III).

2.4.1 *Independence between the different stages of the data mining process.*

The level of independence between the functions comprising the data mining chain varies according to the applied data mining process, as was explained in the Section 2.2. Here the topic is treated once more, with a more practical point of view. The properties of the closed approach are first related to the issue of independence under the item marked with I, then the properties of the open approach under item II and finally the semi-open approach under item III.

I In the closed approach, the output of a function is directly dependent on each of the functions applied prior to it - in this case the chain is as weak as its weakest link. In other words, if one of the transformations does not work properly, none of the transformations following it can be assumed to work properly either, since each of them is directly dependent on the output of the previous transformations. On the other hand, if the functions in the chain all work as they should and the chain can be easily implemented, using this approach demands the least administration.

II In the open approach, all of the functions are independent entities, in the sense that they are only indirectly dependent (through the storage point) on the output of the previous functions in the chain. From the standpoint of the applier it might be good that the functions are so independent from each other. On the other hand, it takes much more work to manage the storage locations than in the other approaches, since all the storage locations are visible to all the functions in the chain. More effort has to be put toward developing the naming conventions and making it clear which data belong to which part of the process, especially if the number of functions in the chain is large.

III In the semi-open method, a function is directly dependent only on the data stored in the local or global storage location immediately prior to the function, not on the

functions inside the previous chain of functions. The operations prior to a certain function do not necessarily have to work perfectly, it is enough that the data stored in the global storage location are correct. The transparency of the process is high and the naming convention is clear. One difference compared to the open approach is the management of interactions, in the semi-open approach the management becomes more natural because the logic behind the reference architecture (and hence, behind the solution under study) is familiar for practitioners.

2.4.2 The multitude of algorithms easily applicable to the data.

The multitude of algorithms that can be applied and tested on the data with a little effort in the development phase is a very important factor from the standpoint of being able to find the best possible solution. Here it is emphasized that the algorithms are *easily* applicable, because in theory any algorithm can be implemented on any tool if enough resources are given. The efforts needed for experimenting with a certain algorithm or a chain of algorithms is highly dependent on the method used for managing the interactions between the algorithms. The amount of work needed to change parts of the algorithm chain is elaborated in the following for the closed (item **I**), open (item **II**) and semi-open approaches (item **III**).

I In the closed procedure, the functions must be implemented in a way where the data can flow directly from one function to another. This can be challenging from a software engineering point of view, especially if the algorithms have been implemented using different tools. Because of this, changing the chain of functions is generally more demanding than with the two other approaches and the applier might not want to test so extensive a set of functions on the data.

II Using the open procedure, it is easy to insert new functions in the chain. The chain works like a linked list, where the addition of a new function is equivalent to adding a new function and storage location between the functions where the new function is to be added. The number of functions is not limited to those implemented in a certain tool, but is proportional to the number of tools that implement an interface for accessing the storage medium where the data has been stored. For example, one of the most frequently used interfaces is the database interface for accessing data stored in a SQL-compatible database. Therefore, if a SQL-compliant database is used as a storage medium, the number of algorithms is limited to the number of tools implementing an SQL interface - which is numerous.

III Also, in the semi-open approach, the number of functions is not limited to those implemented in a certain tool, but is proportional to the number of tools that implement an interface for accessing the storage medium. What distinguishes it from the open approach is that the different organization of the storage clearly groups the functions into pre-processing, feature extraction and classification, where each type of a function is naturally added to a respective section in the process. Furthermore, because the applier has the freedom of adding a local storage point between any two functions inside a sub-chain of functions in the reference architecture, the individual functions are as easily replaceable as with the open procedure.

2.4.3 Specialization and teamwork of researchers

The different phases of the data mining process require a great deal of expertise. Therefore, in general it is harder to find persons who would be experts in all of them, than to find an expert specialized in some of the stages of the DM process. Despite this, even if the researcher would be specialized in a certain field he / she must also apply or know details of many, if not all, of the other steps in the data mining chain, to be able to understand the process and to conduct the research work. If the process is managed carelessly, the applier has to delve too deeply into details, which results in wasted resources, since it takes some of her / his time away from the area she / he is specialized in. Furthermore, when a team of data miners is performing a data mining project, it might be that everybody is doing a bit of everything. This results in confusion in the project management and de-synchronization of the tasks. It is perhaps this feature that makes the biggest difference between the open (contrasted under item **II**) and semi-open approaches **III**, while the closed approach (**I**) remains a black-box.

I A good question is how to manage the data mining process when using the closed approach. In particular, in situations when more than one researcher is participating in the project, there is no obvious solution for it. One alternative is to give each data miner the freedom of developing his or her own processing chains. In this case, each of them has to possess the functions needed in all the phases of the process. Or, alternatively, the researchers can try to develop a method for maintaining the function chain collaboratively. In any case, it will be hard to distribute the data mining project.

II When using the open method, it might be hard to stay fully informed of the stage of development of the process. Because there are no clear limits between the stages of the data mining process, it is harder to allocate resources on the different stages. It will be also harder to tell who is involved in the different stages of the process. When functions are added to or removed from the process, everybody involved with the process has to stay informed on the changes.

III The proposed method facilitates the natural management of the data mining process. Researchers can be allocated to work on the data relevant to their specialization. Each stage of the data mining project is clearly allocated to its own independent sub-chain. When a team of data miners are working on a project, the work can be naturally divided between the workers by allocating the data stored after global storage points to suit the expertise and skills of each person. Furthermore, the experts can freely add and remove functions, as long as the data in the global storage point after the respective sub-chain remains unchanged.

2.4.4 Data storage and on-line monitoring

Data storage and monitoring of the results are important issues when observing the quality of and utilizing the results of the data mining process. The data acquired in the different phases of the data mining process can be stored in a coherent way when, for example, a standard database is used to implement the data storage. When the data can be accessed through a standard interface after the transformations, one can peek in on the data at any

time during the storage points specified in the process. When using a SQL-database, for example, as a storage medium, one can select the monitoring tools from a set of readily available software. The closed-, open- and semi-open approaches contain different amounts of storage points by definition, which is a factor that directly affects the monitoring of the results. With the closed approach the monitoring differs largely from the open- and semi-open approaches. The following items relate the approaches to the monitoring issue.

- I** The process has only two data storage points, the measurement data and the results. So, by definition, these are the only stages that can be monitored when applying the closed data mining process. This can be convenient in simple applications, but in the development stage of the application it is hard to observe any intermediate results.
- II** The results of each function in the data mining process are stored for observation. This will increase data storage demands, but gives the freedom of observing the results of each function in the chain. Again, possible problems might be caused by improperly designed naming conventions, making it hard to distinguish the stage which the functions belong to.
- III** The results of each stage of the reference architecture are stored for global observation and a selective number of results (decided by the applier) of individual functions in the chains are stored for local observation. This makes it possible to observe the quality of the process at a glance in the different stages presented in the reference architecture or in detail through the local storage points. Individual observation points can be placed after functions that need further development.

2.4.5 Time savings and computing costs

Executing the data mining chain can require a lot of processing power, especially when the number of observations or dimensions in the measurement data grows or the amount or complexity of the functions in the chain increases. Therefore, it is important that the data mining process is designed in such a way that it decreases *redundant* calculations. Here the term *redundant calculation* refers to an identical execution of a function in the chain i.e. the same input data is given for the function, it has the same parameters and outputs the same results in two or more executions of the chain. When working with large data sets, this issue may result in significant computational performance differences. The properties of the three approaches with relation to the issue vary highly and are described next.

- I** Using the closed method, all the functions in the chain must be calculated when one wants to observe the output of the process. This results in unnecessary waste of resources and a lot of redundant calculations if only a part of the processing chain has been changed.
- II** The data mining chain does not need to contain any redundant calculations when using the open approach. Only the functions in the chain posterior to the changed data have to be recalculated.

III When the data in the global storage locations have been calculated once in the semi-open process, it does not need to be re-calculated unless data prior to it has changed. Depending on the amount and location of the local storage points, further time savings can be made.

Now that the numerous benefits of the proposed method have been presented, we could ask what are the drawbacks of the proposed method? The obvious thing that needs more resources is the care and effort one has to put into defining the interface for transferring the intermediate results to the database. On the other hand, if this work is left undone, one may have to put twice as much time in tackling with the flaws in the data mining process. It might also seem that the calculation of the whole data mining chain using the closed process is faster than using the semi-open process, since the data do not need to be loaded from a permanent storage media. That is true, but it is known from practice that the data mining chain needs multiple executions before finding the desired solution. When using the developed method it is necessary to run only part of the chain. Finally, it can be said that the selection of the management method is a project-wise decision. For some projects the closed approach might be appropriate, for example when creating early mock-ups of data mining solutions. In general, when moving towards larger and more complex data mining projects with many research workers involved in them, the advantages of the proposed process become clearer.

2.5 Case study: A data mining solution for spot welding quality control

This section illustrates the benefits of the proposed method in practice. The semi-open process is applied to a data mining project analyzing the quality of spot welding joints, and a detailed comparison to the closed approach is made concerning the amount of work required for acquiring pre-processed data.

The spot welding quality improvement project (SIOUX) was a two-year, EU-sponsored CRAFT project aimed at creating non-destructive quality assessment methods for a wide range of spot welding applications. Spot welding is a welding technique widely used in the electrical and automotive industries, for example, where more than 100 million spot welding joints are made daily in the European vehicle industry alone (TWI 2005). Non-destructive quality estimates can be calculated based on the shape of the signal curves measured during the welding event (Laurinen *et al.* 2004a, Junno *et al.* 2004b). The method results in savings in time, material, environment, and salary costs - which are the kind of advantages that the European manufacturing industry should have in their competition against outsourcing work to cheaper countries.

The data collected consist of information regarding the welded materials, the quality of the welding spot, the settings of the welding machine, and the voltage and current signals measured during the welding event. To demonstrate the data, Figure 8(a) displays a typical voltage curve acquired from a welding spot and Figure 8(b) shows the resistance curve obtained after pre-processing the data.

The project was conducted by two teams of researchers, the other one consisted of three to four members (based in Oulu) and the other one of two to four members (based

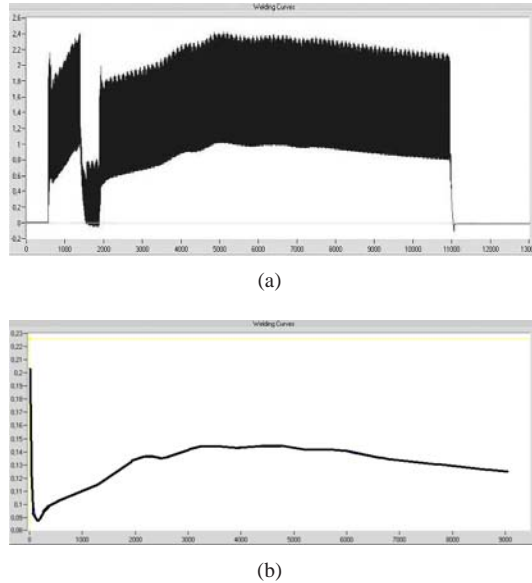


Fig. 8. Examples of measured and pre-processed data. (a): The voltage signal of a welding spot measured during a welding event. The high variations and the flat regions are visible in the curve. (b): The resistance curve acquired after pre-processing the voltage and current curves (current curve is not displayed here).

in Karlsruhe, Germany). The major responsibility of the German team was in developing the pre-processing algorithms, a method for computationally estimating the nugget size of a welding spot and the platform where the final implementation of the project prototype was implemented. The responsibility of the Finnish team was more on developing feature extraction and classification methods for process identification and a database system for storing the data. The fact that the project was distributed geographically and many researchers participated in it made the good management of the data mining process especially important.

2.5.1 Pre-processing spot welding data

The data transformations needed for pre-processing signal curves consist of removal of the flat regions from the signal curves (welding machine inactivity), normalization of the curves to a predefined interval, smoothing of the curves using a filter, and calculation of the resistance curve based on the voltage and current signals.

The transformations were implemented in software written specifically for this project, called Tomahawk. The software incorporates all the algorithms required for calculating the quality estimate of a welding spot, along with a database for storing the welding data. The software and the database are closely connected, but independent. The basic

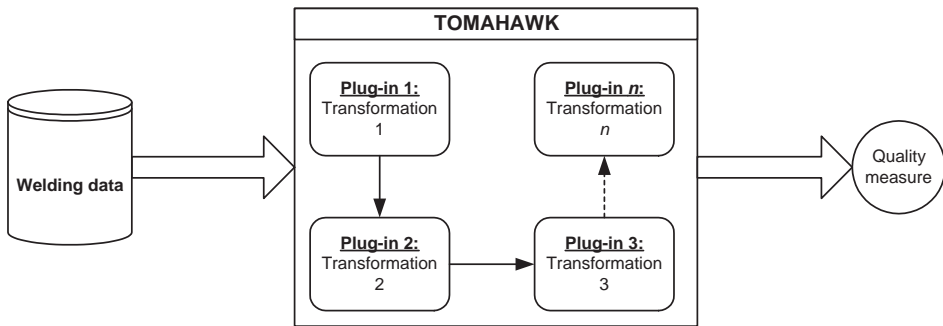


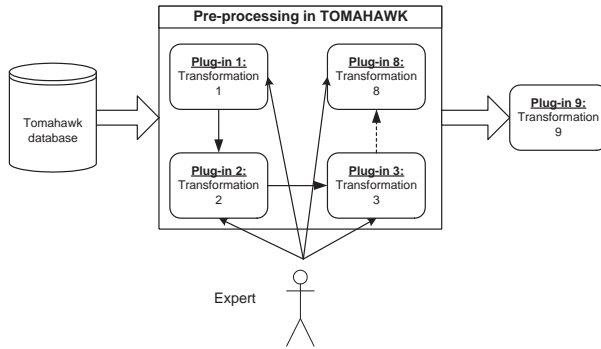
Fig. 9. The operating principle of the original implementation of the Tomahawk software. The architecture is a realization of the closed data mining process.

principles of the system are presented in Figure 9. The algorithms in Tomahawk are implemented as a connected chain. The algorithms are called plug-ins and the processed data is transferred from one plug-in to another, until the end of the plug-in chain has been reached. Hence, the result of applying all the algorithms is the desired output of the data mining process. When the program is executed, the chain of plug-ins is executed at once. This is an implementation of the definition of the closed data mining process.

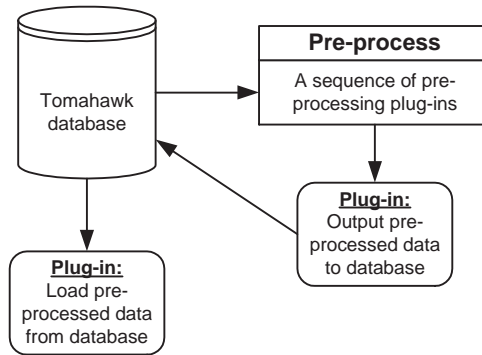
The ultimate goal of the project was to have all the plug-ins ready and working for all kinds of welding data as seamlessly as presented in Figure 9. However, in the production phase of the system, when the plug-ins were still under active development, three major issues that interfere with the daily work of the local development team were identified based on the criteria presented in Section 2.4 as follows:

- *Independence.* It cannot be guaranteed that all parts of the pre-processing algorithms would work as they should for all the available data, at least during the development stage. However, the researcher extracting features from the pre-processed data is dependent on the results output by the pre-processing sequence. Because of this, if the data pre-processed using the closed data mining process is used in feature extraction, the persons developing the feature extraction algorithms cannot be certain that the features are based on correctly pre-processed data.
- *Specialization and teamwork.* The expert developing features based on the pre-processed data might not have the expertise to correctly pre-process the raw data in the context of Tomahawk, which would make it impossible for him/her to perform her/his work correctly.
- *The multitude of algorithms* easily applicable to the data. In the development phase, it is better if the range of algorithms tested on the data is not exclusively limited to the algorithms implemented in Tomahawk, since it would require a lot of effort to implement algorithms that are also available elsewhere as plug-ins just in order to be able to test them.

The solution was to develop Tomahawk such that it would also support the semi-open data mining process - a plug-in capable of storing and delivering pre-processed data was



(a)



(b)

Fig. 10. The closed- and semi-open data mining processes in the context of Tomahawk. (a): The application of the closed data mining process on the pre-processing of the raw data using Tomahawk. (b): Tomahawk after the modifications that made it support the developed, semi-open, data mining process for pre-processing data.

implemented. Figures 10(a) and 10(b) present the influence of these developments. Figure 10(a) displays the pre-processing sequence prior to the adjustments. In this phase of development, all the plug-ins were calculated at once, and they had to be properly configured to obtain properly pre-processed data. Figure 10(b) shows the situation after the adoption of the semi-open data mining process. The pre-processing can be done in its own sequence, after which a plug-in that inserts the data into the database (a global storage operation) is applied. Now the pre-processed data has been stored in the database and is available for further use at any given time.

It is easy to see how the first and second issues are resolved using the proposed approach. The pre-processing expert of the project takes care of properly configuring the pre-processing plug-ins. If the plug-ins need to be re-configured or re-programmed for different data sets, the expert has the required knowledge to do it. After the application of the re-configured plug-ins the data can be stored in the developed database. If it is not

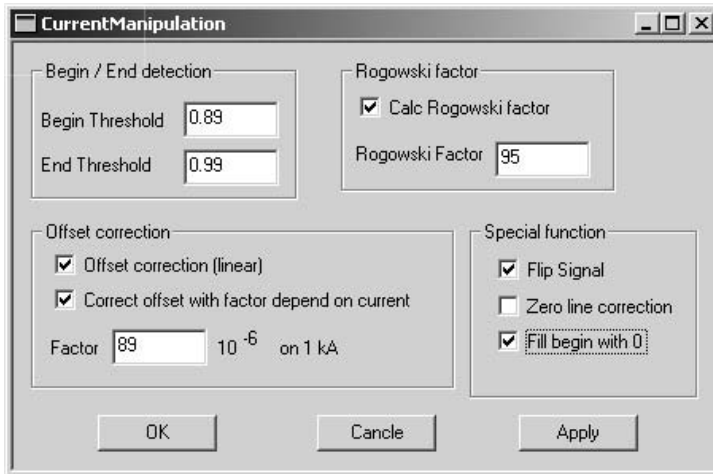
possible to find a working combination of plug-ins at the current state of development, the data can still be pre-processed manually, which would not be feasible when using the closed process. After this, the expert working on the pre-processed data can load the data from the database and be confident that the data the expert is working on has been correctly pre-processed. The third issue is also easy to solve; after the modifications, the set of feature extraction algorithms that can be applied to the data is no longer limited to those implemented in Tomahawk, but is extended to tools containing a database interface implemented in them, for example Matlab and most statistical software packages. This drastically expands the range of available algorithms, which in turn makes it also faster to find an algorithm suitable for a given task. As soon as a suitable algorithm has been found from the set of readily available choices, it can be implemented in the Tomahawk framework.

The case study is finished by presenting a comparison of the steps required for acquiring pre-processed data in the SIOUX project using the closed and semi-open approaches. The purpose of the comparison is to demonstrate how large a task it would be for the researcher working on the pre-processed data to pre-process the data using the closed approach before the actual work could be started.

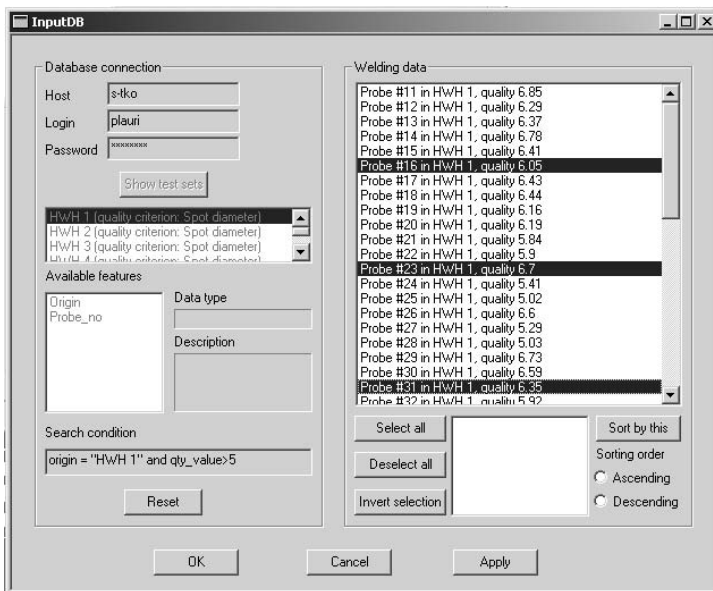
If one wants to acquire pre-processed data using the closed approach, it requires taking the application and configuration of eight plug-ins. Figure 11(a) shows one of the configuration dialogs of the plug-ins. This particular panel has four numerical values that must be set correctly and the option of setting six check boxes. The total number of options the researcher has to set in the eight plug-ins for acquiring correctly pre-processed data is 68. The 68 options are not the same for all the data sets gathered in the project, and it requires advanced pre-processing skills to configure them correctly. Therefore, it is a rather complicated task to pre-process the data, and it is even more difficult for a researcher who has not constructed the pre-processing plug-ins. The need to configure the 68 options of the pre-processing sequence would take a lot of time and expertise away from the work done in feature extraction and classification phases, and still gives poor confidence that the data is correctly pre-processed.

To acquire the pre-processed data using the semi-open approach, one only needs to fetch the pre-processed data from the database. Figure 11(b) shows the configuration dialog of the database plug-in, which is used to configure the data retrieved for analysis from the database. Using the dialog, the researcher working on the pre-processed data can simply choose the pre-processed data items that will be used in further analyzes. The researcher can be sure that all the data loaded from the database has been correctly pre-processed by the expert who is responsible for pre-processing. From the standpoint of the pre-processing expert, it is good to know that the sequence of pre-processing plug-ins does not have to be run every time that pre-processed data is needed, and that correctly pre-processed data will surely be used in the further steps of the data mining process.

In conclusion, when using the closed process, a researcher responsible for the feature extraction could not always be certain that the data has been correctly pre-processed, or that all the plug-ins have been configured the way they should, which resulted in confusion and uncertainty about the quality of the data. The semi-open process, on the other hand, allowed a notably simpler way to access the pre-processed data, resulted in time savings, and ensured that the analyzed data was correctly pre-processed.



(a)



(b)

Fig. 11. Dialogs used for configuring the data mining chain of Tomahawk. (a): Applying the closed data mining process on the pre-processing of the raw data in Tomahawk, an example of a dialog that has to be configured in order to pre-process the measurement data. (b): Fetching pre-processed data using a plug-in developed for supporting the semi-open data mining process.

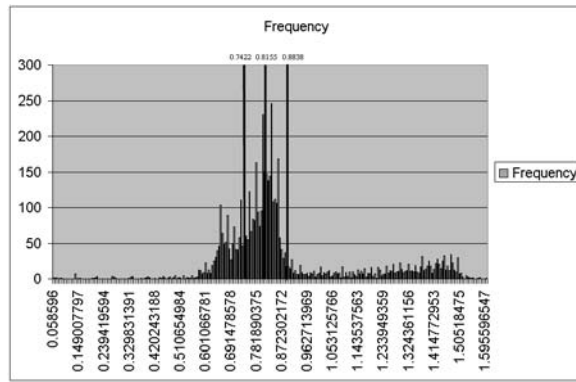
2.5.2 Feature extraction and modeling results

This section of the study presents the fruits of the application of the developed process. Because of limited resources, it was not possible to conduct a detailed comparison of the proposed process to the other approaches in the feature extraction and classification stages. The conditions in which the data mining process was applied would have also made a thorough comparison quite difficult - the data set was expanding throughout the project and different sets of features were formed and experimented upon as the project progressed and a varying number of researchers were participating in the project. The fine results that were obtained, presented in the following two subsections, should be enough to convince the reader on the efficiency of the proposed process on the feature extraction and modeling parts of the data mining chain. Subsection 2.5.3 presents the preliminary results of the project acquired using Bayesian networks. Subsection 2.5.4 presents the results when the data set was expanding to its full scale and some of the steps taken for finding the solution to this data mining task.

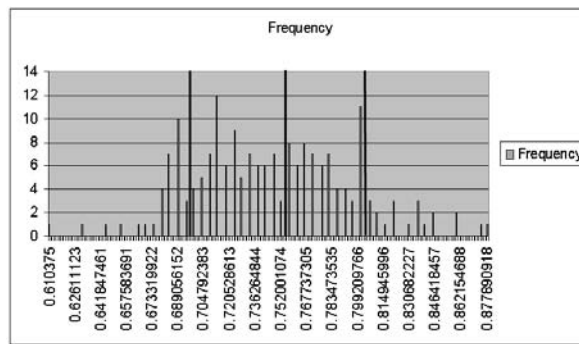
2.5.3 Non-destructive analysis of welding spots using Bayesian networks

At the start of the project only a small set of welding spot data from a previous project done by the project partners was available. The data set consisted of 192 welding experiments, where the signal curves from voltage-, current- and compression force measurements and the diameter of the welding nugget were available. At this phase a very simple approach was used to study the nature of the interaction between features extracted from the signals and the welding spot size, in order to get a grasp of working with and analyzing spot welding data. The reasons for the variation in the welding spot diameters were studied using Bayesian networks and the features used in the study were extracted from histograms calculated from the available signal curves (Laurinen *et al.* 2004a).

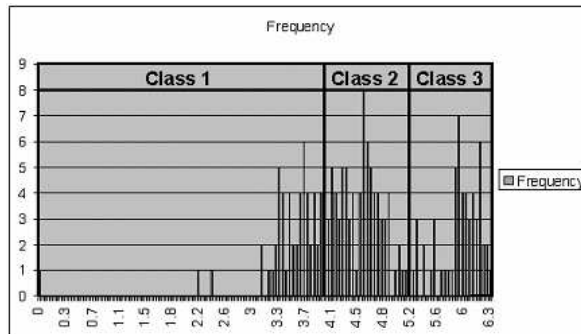
The features used in this study were extracted from the histograms calculated based on pre-processed voltage-, current- and compression force signals. Unlike the data sets gathered later on, a measurement of the compression force was also available. Figure 12(a) presents an example of a voltage histogram of a welding spot. The lower and upper quartiles and median are plotted in the figure using vertical lines. After calculating the quartiles and median for all the signal curves, a histogram of these values was calculated. In order to be able to use the tool used for creating Bayesian networks (Ramoni & Sebastiani 1997a,b) the data had to be classified, because the tool only worked on classified data. The classification was based on the distributions of the quartiles and medians of the signal curves. As an example, the histogram presenting the distribution of the lower quartiles of the voltage curves is plotted in Figure 12(b). Based on this histogram, the quartile values were assigned to four distinctive classes, marked again in the figure using vertical lines. Figure 12(c) presents the distribution of the response variable, the diameter of the welding nugget. An expert in welding technology gave the class limits for the nugget size - diameters smaller than 4.0 millimeters (49 observations) are of poor quality, diameters ranging from 4.0 millimeters to 5.275 millimeters (88 observations) are of good quality and diameters larger than 5.275 millimeters (55 observations) are of excellent quality. The



(a)



(b)



(c)

Fig. 12. Some of the histograms used with the Bayesian network. (a): Histograms generated from the voltage curve of a welding spot. The three tall vertical lines in the plot mark the lower and upper quartiles and the median. (b): Histogram of the lower fractiles of the voltage curves and their classification. (c): Histogram of the diameters of the welding spots and their classification.

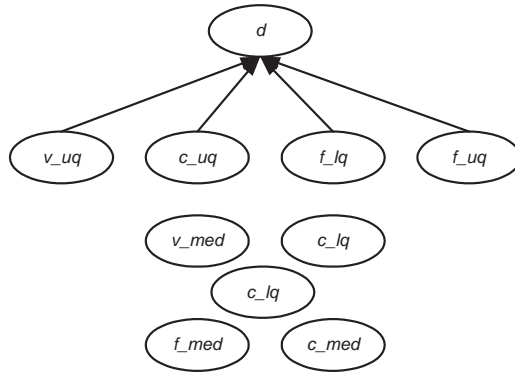


Fig. 13. The structure of the first Bayesian network used in the study. d = welding spot diameter, v = voltage, c = current, f = compression force. lq = lower quartile, med = median, uq = upper quartile.

expert also agreed on the classification of the quartiles and median used as features.

Bayesian networks were used to study the relationships between these features. It proved to be a suitable tool for this task, since it enabled the study of the effects of the different variables on the diameter in the form of easily understandable probabilities. Figure 13 shows one of the Bayesian networks applied. The nodes denote the variables and the edges between the nodes the interactions between the variables. The network structure was generated using an automatic search algorithm implemented in the software (Ramoni & Sebastiani 1997a,b). The algorithm established the connections between the variables based on the data measured from the welding events and the features calculated from the data. In this particular network structure the welding spot diameter is interacting with the median of voltage, the upper quartile of current and the upper quartile of compression force. The disconnected nodes did not have an effect on the welding spot diameter according to the search algorithm.

The welding spot diameters can be studied by examining the class probabilities of the diameter from the Bayesian network. These probabilities are calculated conditionally on the values observed from the other variables. Table 1 shows some of the most interesting class probabilities of the welding spot diameter based on the configuration of Figure 13. It can be read from the table for example, that the combination $v_{med} = 2$, $c_{uq} = 2$ and $f_{uq} = 1$ leads almost certainly to a high quality welding spot. Or that, if $v_{med} = 2$, $c_{uq} = 1$ and $f_{uq} = 2$, it is questionable if the welding spot is of good quality. The rest of the configurations did not contain a significant number of observations in class number one (poor quality), which is why they are not listed in the table.

The rules presented in the table can easily be implemented in practical applications because of their simplicity. For the same reason it is also easy for humans to understand them. The more equally distributed probabilities are cases in which the quality assignment of the welding spot is not certain. These situations can be identified using these results and appropriate actions can be made.

The probabilities and results presented above were used to present some of the most

Table 1. Probabilities associated with the Bayesian network. v_med = voltage median, c_uq = current upper quartile, f_uq = force upper quartile, d = welding spot diameter, $n(obs)$ = number of observations in the configuration.

Variables and their classes			Welding spot diameter			n(obs)
v_med	c_uq	f_uq	$d = 1$	$d = 2$	$d = 3$	
1	1	3	0.986	0.007	0.007	1
1	2	3	0.993	0.003	0.003	1
2	1	1	0.002	0.002	0.995	3
2	1	2	0.739	0.087	0.174	23
2	1	3	0.498	0.498	0.003	2
2	2	1	0.001	0.001	0.999	10
2	2	2	0.588	0.294	0.118	34
2	2	3	0.499	0.499	0.001	6
2	3	1	0.001	0.001	0.998	6
2	3	2	0.25	0.375	0.375	16
2	3	3	0.498	0.498	0.003	2
2	4	2	0.001	0.499	0.499	6
3	1	2	0.001	0.997	0.001	5
3	2	2	0	0.904	0.095	41

interesting results acquired in this small experiment. Presentation and analysis of all combinations of variables and their respective probabilities would not have served the purpose of this work. However, these results show the idea and some results of the study. Using the probabilities assigned with Bayesian networks, it is possible to discover combinations that lead to different sizes of the welding spot and therefore also to different quality classes. The problem with classified data is that, if there are many classes but few observations, not enough observations may fall into the different configurations of the classes to allow reliable conclusions to be drawn. Fortunately this was not a serious issue in this study. The main contribution of this experiment was to demonstrate how feature data were utilized using the developed process in devising a method that is applicable in the field. The results of another experiment, where the feature data set was utilized in a similar manner, but with self organizing maps Kohonen (2000), can be found from (Junno *et al.* 2004b).

2.5.4 Development of process similarity measurement techniques

The ultimate goal of the SIOUX project was to develop a process identification method capable of recognizing resistance spot welding processes. Different processes are applied for manufacturing different products and each process requires different settings to be applied in the controller of the welding machine. The logic behind the data mining task was to first collect data sets representing different spot welding processes and to store them in a central database. After this, when starting a production of a new product, possibly at a

different site, and initializing the process, the database could be remotely queried using data from a few sample joints from the new process. Based on measurements from these sample joints the database should then return controller settings from the most similar process that it contains. Furthermore, the controller settings should be returned from a controller configuration resulting in high quality welding joints. The new process could then be initialized based on these settings. This decreases the set-up time of new processes, because the controller settings leading to good quality welding joints do not have to be searched manually, only fine tuning is necessary.

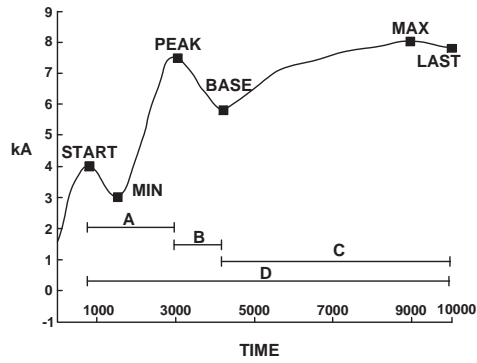
Before delving further into process identification, it is important to understand what exactly a welding process is in this context. All resistance spot welding applications can be considered different processes. In this project, data sets were divided into different processes based on three conditions:

1. *Type of application.* Examples of applications are welding car parts or constructing electrical switches.
2. *Type of materials.* For example joining two objects made of 1.0 mm thick uncoated steel is considered to be a different process from welding together two 1.25 mm thick objects made of aluminium.
3. *Type of welding controller.* Manufacturing products using different welding controller model / welding controller combinations.

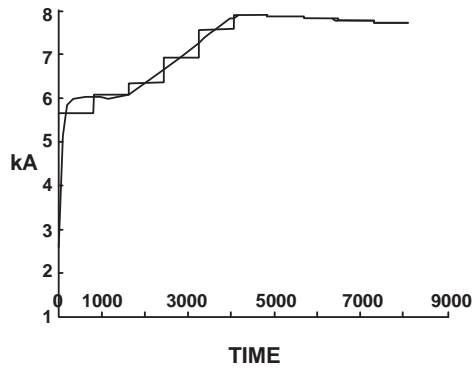
All combinations of these conditions were considered different processes. During the project, samples from different processes were gathered and stored in a database developed for storing welding data. The process data set kept increasing to the very end of the project, which made it more and more important to use a systematic data mining process for managing the data mining task. The data mining process developed here made it possible to analyze the new data as it became available and to update the results and algorithms developed.

As the project progressed, the set of features and applied classifiers developed also kept increasing, alongside the fact that the available data set was continuously extended with new batches of data from new welding experiments. At the end of the project, altogether 54 geometrical and statistical features were extracted from the two signal curves and the data set contained 20 processes used in the project identification task. The data set consisted of measurements from 3,879 welding spots, with the quality assured using a destructive test.

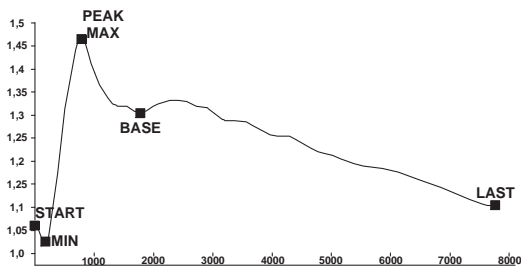
The geometrical features were extracted from pre-processed voltage and current signals. These features were developed in order to identify the transition points of the curves as precisely as possible. These features are marked in the signal in Figure 14(a), which shows an artificial curve simulating the real data. Figure 14(c) shows an example of these features calculated on a real signal curve - as can be seen the features are often overlapping in reality, which can be considered a characteristic quality of the curve. The statistical features included the median of the signal, and the arithmetic means of the signal values calculated on four different intervals based on the transition points. These features are marked using the horizontal lines in Figure 14(a). In addition, the means of the signal values inside ten intervals of equal length were used as features, as demonstrated in Figure 14(b). This adds up to 12 geometrical and 15 statistical features that were extracted from both of the signal curves.



(a)



(b)



(c)

Fig. 14. Illustrations of features extracted from the signals relevant to the resistance spot welding application. (a): The geometrical features on an artificial voltage curve. The line segments A-D below the curve demonstrate the intervals based on the transition points on which means were calculated. (b): Ten means of a current curve calculated on intervals of equal length. (c): An example of how the geometrical features often partially overlap in practice. On this voltage curve, the features 'peak' and 'max' overlap.

Table 2. Comparison of the classification accuracy for the 11 HWH processes with different classifiers and feature sets **using features extracted from the voltage and current signals**. LDA = linear discriminant analysis, QDA = quadratic discriminant analysis, Mahalanobis = Mahalanobis discrimination, LVQ = learning vector quantization and KNN = k nearest neighbors.

Method / feature set	LDA	QDA	Maha-lanobis	LVQ, 200 code-books	kNN, k = 5	kNN, k = 10
All features	94.33	-	-	64.78	73.43	74.03
All features, normalized	94.33	-	-	92.84	92.45	90.45
10 means	87.16	96.12	95.22	85.07	98.51	97.91
10 means, normalized	87.16	96.12	95.22	93.43	97.01	93.13

Table 3. Comparison of the classification accuracy for the 11 HWH processes with different classifiers and feature sets **using features extracted from the resistance signal**. LDA = linear discriminant analysis, QDA = quadratic discriminant analysis, Mahalanobis = Mahalanobis discrimination, LVQ = learning vector quantization and KNN = k nearest neighbors.

Method / feature set	LDA	QDA	Maha-lanobis	LVQ, 200 code-books	kNN, k = 5	kNN, k = 10
All features	77.61	-	-	45.97	56.72	57.31
All features, normalized	77.61	-	-	85.37	86.87	86.27
10 means	70.15	71.94	72.84	80.9	86.87	84.78
10 means, normalized	70.15	71.94	72.84	77.31	84.48	79.7

During the creation of the feature set alternative features were also tested and process classification experiments were run using the expanding feature set, for example using self organizing maps (Junno *et al.* 2004a). At the point when the final set of features was available, 11 different processes welded at Harms+Wende (Germany) were obtained for classification. Using the developed data mining process, the researchers could retrieve feature data directly from the global feature storage point and apply different classification methods on the feature data.

The applied approach made it easier to test a representative set of classifiers. The

set of tested classifiers contained linear discriminant analysis (LDA) (McLachlan 2004), quadratic discriminant analysis (QDA) (McLachlan 2004), Mahalanobis discrimination (similar to LDA and QDA, but uses Mahalanobis distance), learning vector quantization (LVQ) (Hastie *et al.* 2001) and k-nearest neighbors (kNN) (Hastie *et al.* 2001) classifiers. Uniform randomly selected two thirds of the data was used as training data and the remaining one third as an independent test set. In later studies, cross-validation was used to confirm the results. During the experimentation with these classifiers and features, a feature / classifier combination capable of classifying the processes correctly with a high level of accuracy was found. The results are presented in Tables 2 and 3. The percentages in the cells mark the ratios of correctly classified processes; the cells left empty mark nonworking classifier and feature set combinations. The kNN-classifier was the most accurate one, with a classification accuracy of 98.51%. At this stage it was also discovered that the features calculated from current and voltage curves outperformed the features calculated from resistance curves. Using the resistance curve based signals the maximum accuracy of classification was only 86.87%. Because of these results the process classification study was carried out using only the current and voltage signals, whereas the project partners kept using resistance curves for other purposes.

After these intermediate results, the next significant milestone in the task was to update the results with a batch of data consisting of nine more processes supplied by another manufacturer (Stanzbiegetechnik, Austria). After this point, the set of processes was extended no more and thus the final data set consisted of 20 different processes. Classification tests were continued using the same set of classifiers as with the successful tests with the 11 processes from HWH. The difference compared to the previous tests was that this time the feature set was extended with five and ten principal components formed from the original features. The results of these tests are displayed in Table 4. The kNN-classifier using three closest neighbours and the ten means of the signal intervals again outperformed the other classifier, with a classification accuracy of 98.53%.

The results obtained exceeded clearly the requirements set for classification accuracy, and the method was now taken for further implementation. Further studies were and are still being actively continued, even though the project has already been completed. It is not a big effort to utilize the features calculated using the developed data mining process in these studies, because they remain in the global storage point. They are now utilized in studies developing the application even further and in projects where data sets are needed for testing new classification methods. An example of this is a study that presents a more detailed comparative study of performance (Haapalainen *et al.* 2005). The results of a feature selection study in which a feature set was found that contains approximately only half the amount of features of the kNN-classifier presented in these results, but achieves higher classification accuracy is reported in (Haapalainen *et al.* 2006). A third example is a kNN-based method that does not only classify the process, but also gives an estimate on the similarity of the closest matching process (Junno *et al.* 2005).

Table 4. Comparison of the classification accuracy for the 20 processes with different classifiers and feature sets. LDA = linear discriminant analysis, QDA = quadratic discriminant analysis, Mahalanobis = Mahalanobis discrimination, LVQ = learning vector quantization and KNN = k nearest neighbours classifier, pc = principal component.

Method / feature set	LDA	QDA	Maha-lanobis	kNN, k=3	kNN, k=5
All features	92.96	-	-	84.13	84.52
All features, 5 pc's	62.46	75.23	72.37	83.2	82.51
All features, normalized	92.96	-	-	94.74	94.89
All features, normalized, 5 pc's	71.05	85.45	86.3	93.5	92.41
10 means	90.87	96.36	97.14	98.53	98.07
10 means, 5 pc's	82.12	94.27	94.35	97.76	97.06
10 means, normalized	90.87	96.36	97.14	95.43	96.13
10 means, normalized, 5 pc's	76.16	89.32	88.31	94.58	94.12

2.6 Related work

Extensive searches of scientific databases and the World Wide Web did not bring to light similar approaches applied to the implementation of the data mining process. However, there are studies and projects on the management of the data mining process, as was noted already in Section 2.1. These studies identify the main phases of the process in a manner resembling the reference architecture and give a general outline of the steps that should be kept in mind when carrying out the process. It was also noted earlier that CRISP-DM is a process model proposed to serve as a standard reference for applicers of data mining (Chapman *et al.* 2000). Several studies testify to the usefulness of CRISP-DM as a tool for managing data mining ventures (Hotz *et al.* 2001, Liu & Han 2002, Silva *et al.* 2002). The approach proposed in CRISP-DM was expanded in RAMSYS (Moyle & Jorge 2001), which proposed a methodology for performing collaborative data mining work. That study also comments on the implementation of the DM process, which in that case resembles the open DM process. Other proposals, with many similarities to CRISP-DM, for the data mining process were presented in (Pyle 1999) and (Brachman & Anand 1996). Nevertheless, these studies did not take a stand on what would be an effective implementation of the data mining process in practice. This study proposed an effective approach for implementing the data mining process and presented its relation to alternative ways of implementing the process, pointing out the obvious advantages of the method proposed here.

2.7 Discussion

This part of the thesis proposed a semi-open approach for managing the data mining process. It is based on the stages present in the reference architecture that is found behind most data mining solutions. Using the proposed method it is possible to differentiate the steps of the data mining process logically and to test different data mining chains with lesser efforts. This is important when seeking a solution, especially in situations where the solution needs to be sought from among many alternatives. A case study presenting how the developed data mining process was applied on a resistance spot welding project illustrated the applicability of the developed process.

Furthermore, the question this part of the thesis tried to answer is: "what is a good practice or procedure for conducting the transformations a data mining process is composed of?". As it is known, there are as many ways of practicing data mining as there are practitioners. In that sense the researchers applying data mining tools are like artists, at least they have a lot of artistic freedom in conducting the research, as they must have, for being able to find the transformation chain leading to a satisfying solution. Nevertheless, it was discovered that there are common practices of managing the interactions between the different data mining algorithms - arguably the best known is the one referred to as the "closed approach". And as we also know, and as the results of the study show, managing the steps of a data mining project is very important for the successful outcome of the project.

How useful is the presented contribution in practice when implementing a data mining process? First of all, it is important to acknowledge the understated fact that some kind of data mining process management would be necessary in most data mining projects. After that it is the task of the practitioner(s) to select a method suitable for their project. For some projects the closed approach might be the appropriate solution and for others the open method might work better. In this thesis it was discovered that the developed semi-open approach is highly usable, at least based on the comparative analysis and the resistance spot welding project in which it was applied. In the welding project, the semi-open approach was compared in detail to an alternative approach (the closed approach) on the data pre-processing part of the transformation chain and its applicability on the rest of the process was illustrated with the fine results acquired in the project. It was not possible to present a detailed comparison on all of the stages, for two reasons. The first is that it was thought that it is enough to present the comparison only on data pre-processing, and based on that, the reader could see the analogy to the other parts of the processing chain and draw the same conclusions as in this thesis - the proposed approach was more suitable for conducting the research. The second reason, and perhaps more important one from a practical point of view, is that as in most projects, including this one, resources are limited. In order to be able to determine the most optimal data mining process for a data mining project, one should conduct the project using a representative set of different approaches for managing the data mining chain - which would mean conducting the project several times. That is only rarely possible. However, the fact that the proposed approach is based on the transparent reference architecture and that its usability was presented from multiple viewpoints and a case study should support the selection of the proposed process, and hopefully lowers the threshold of adapting it to other projects as well.

3 From the solution to a data mining application

The previous chapter presented an overview of data mining processes and the data mining process developed, called semi-open data mining process, for the purpose of implementing the work flow required for finding a solution for a given data mining problem. Implementation of the solution as an independent data mining application is a challenging and complicated task, and the applications are often built from scratch. This chapter presents a component-based application framework, called Smart Archive (SA), designed for implementing data mining solutions as independent DM applications. SA provides the functionality common to most data mining applications and the components for utilizing history information. It is especially suitable for implementing DM applications processing continuously acquired measurement streams. Using SA, it is possible to build high-quality applications with shorter development times by configuring the framework to process application-specific data. A detailed case study of a data mining application predicting the post-roughing mill temperatures of steel slabs is presented and a comparison of implementing the application from scratch and using Smart Archive is given.

The chapter is organized as follows. A general introduction to the topic is given in Section 3.1. Functional requirements analysis of the properties that the developed framework must be capable of managing is presented in Section 3.2. The principles of the components used in the architecture are described in Section 3.3 and the architecture itself in Section 3.4. The case study is then presented in Section 3.5. Finally, related work and discussion are presented in Sections 3.6 and 3.7.

3.1 Introduction

A data mining (DM) application without an underlying framework is like a computer program without an operating system. Without a proper framework a considerable portion of the implementation time of the application will be spent implementing functionality common to all applications in the DM domain. However, the redundant work would not be necessary if a sufficiently generic framework that could be easily tailored to meet application-specific needs were available. The need to develop a component-based data mining framework is emphasized in (Berzal *et al.* 2002). The authors note that decision

support systems have specific needs that cannot be properly addressed by conventional information systems. This part of the thesis introduces an application framework, called Smart Archive (SA), for implementing data mining applications utilizing continuously measured data streams.

SA is a domain-specific but application-independent framework. This means that the framework supports operations required by most DM applications, but is not tied to any particular application. Creating an application using SA requires implementing application-specific algorithms using the interfaces offered by the framework and configuring the framework to use the measurement data the application is designed to process. The framework takes care of transferring and storing data between application components and implements some of the basic functionality required of application-specific filters. The full potential of SA can be realized in on-line applications generating multivariate time series data simultaneously from multiple entities. It has been observed after practical experiments that this set of applications is the most demanding to implement.

The benefits of using a framework for application generation are numerous because a typical DM application has to handle a large number of variables and transformations. For one thing, the development time spent implementing functionality common to most or all similar applications can be significantly decreased to the benefit of increased resources for application-specific development. The quality of the application is likely to increase, since the code of the framework is already tested and the application-specific code is called through well-defined interfaces. More advantages of using a framework are described in the case study presented in Section 3.5.

The architecture of SA is an instance of domain-specific software architecture (DSSA) (Hayes-Roth *et al.* 1995). The principle of DSSA development is to create a software architecture based on a functional requirements analysis of the target application domain. Components common to potential applications within the domain are identified based on the requirements analysis. After that, software architecture that is sufficiently abstract for modeling the interoperability of the components is formed. This version of the architecture is called the reference architecture.

The overall principle of data mining is the measurement-pre-processing-feature extraction-modeling cycle, as was observed in the previous chapter. The generic reference architecture for data mining applications can therefore be identified as the one presented in Chapter 2 (page 22), Figure 3(b) and this was also the reason for naming the figure "reference architecture".

3.2 Functional requirement analysis

As was stated in the introduction, the development of the architecture is based on an analysis of functional requirements. Functional requirement analysis is used to specify what the developed framework must be capable of achieving. Therefore, it is important to state what the framework is developed for. The analysis was performed for the domain of data mining applications that are implemented for processing continuously observed measurements. Furthermore, much of the analysis was inspired by the application under study (the walking beam furnace, results of the application are presented later in Section

3.5).

First the types of requirements set by the application were analyzed. After that, the requirements were abstracted so that they could still be adapted to the needs of the application, but also to any other data mining application belonging to the same class or subclass. Because the application under study was so complex, it is safe to say that most of the data mining applications are in the same class of complexity or in a subclass of it. Therefore, the framework developed could be used to implement any data mining application in the same class (or in a subclass).

The following description of requirements set for the types of applications the framework should be capable of handling was created by abstracting the requirements set by the walking beam furnace application. The framework has to be capable of receiving and processing multivariate measurements from multiple observations and their environment in the pre-processing phase inside a given time cycle. The application has to be able to keep track of the live objects continuously producing measurement data e.g. monitoring when an observation sequence is complete. After that, the feature selection component has to be capable of selecting data only from the variables that the application is actually using. Because different models might contain different features and variables, the feature selection component must give the user the freedom of configuring the variables that are included in the feature formation phase of the reference architecture. The applier must also be free to use and experiment with different feature extraction methods, therefore the feature extraction methods of the component should be configurable. The model of the framework must also be easily replaceable by another because it may be necessary to update the parameters, the model, or even to replace the model with a new one. Finally, because the types of applications the framework is developed for are producing data continuously, the framework must contain a mechanism for handling and storing cumulatively acquired history information from the process under study.

After this description of the requirements a list of guidelines for the framework was drawn up. The list is based on two general guidelines recommended for component-based data mining frameworks: transparency and usability Berzal *et al.* (2002). The five most important requirements that were identified are:

1. The architecture should implement, and, preferably, extend the reference architecture and preferably be easily extendable with new components.
2. The architecture should be able to utilize history information.
3. The architecture should be customizable to suit application-specific needs. The components used for tailoring it to application-specific needs should be separate from the core architecture.
4. The architecture should be suitable for processing continuously observed multivariate time series data from multiple entities.
5. The architecture should be transparent and easily understandable to practitioners of data mining.

These instructions were kept in mind when shifting towards designing the architecture and implementing the framework, as described in the next section.

3.3 The components of Smart Archive

Before going into the presentation of the overall architecture of SA the individual components that make up the architecture are discussed. The components are divided into three categories:

1. components common to software architectures in general
2. components specific to data mining architectures
3. components specific to SA.

Data processing units common to most software architectures are ones that store, transform and transfer data. Storage components are also referred to as *storage units*, *data sinks* or *data warehouses*. The terms *filters*, *transformations* and *operators* are used for transformation components and components transferring data are referred to as *pipes* and *data streams*. A component that is present in fewer architectures, but in most frameworks, is the *interface* component that makes the components less dependent on application. For the sake of coherence, the remaining part of this description uses the term filter to refer to transformation, the term pipe to components transferring data and the term sink to refer to components storing data.

Components specific to data mining architectures are the ones present in the reference architecture. The data pre-processor was responsible for performing elementary operations, such as data cleansing and integration, ensuring that only quality data is fed into subsequent components. The feature extractor filtered the data to extract information that is not necessarily directly measurable, but may increase the performance of the model. Examples of commonly used features are averaged time series, principal components and Fourier coefficients of signals. The model was a component fit on the feature data for extracting desired knowledge. The model is often used to anticipate the future behavior of a phenomenon (predictive models); examples of modeling methods include statistical classifiers, neural networks and regression models.

The components specific to the architecture of SA handle the storage and utilization of information of the history of the phenomena being studied. The incremental history component archives completed measurement series in a non-redundant way. In this context, the non-redundant data storage means storing data in such a manner that no two observations (or series of observations from an entity) resembling each other too closely are stored in the database. In practice, when observing quantities that are measured using real number precision, it is very unlikely that any two multidimensional observations turn out to be exactly the same, but a level of similarity can be calculated. The selective data storage component provides filters for determining the similarity between measurements already archived in the incremental history component and completed measurements considered as candidates for archival. If the candidates are observed to resemble existing history observations too closely they will not be archived. The algorithm for determining similarity can be, for instance, the k-nearest neighbors (kNN) algorithm, which has been developed for the resistance spot welding application (Junno *et al.* 2005), or the trajectory similarity method presented in Chapter 4 of this thesis and in (Laurinen *et al.* 2006). The component returning similar data compares on-going measurement data with archived data and pipes the most similar data found in the incremental history back to the component calling it.

The basic components presented so far are organized into larger units that are used to build the SA architecture. The components (units) of SA are implemented according to

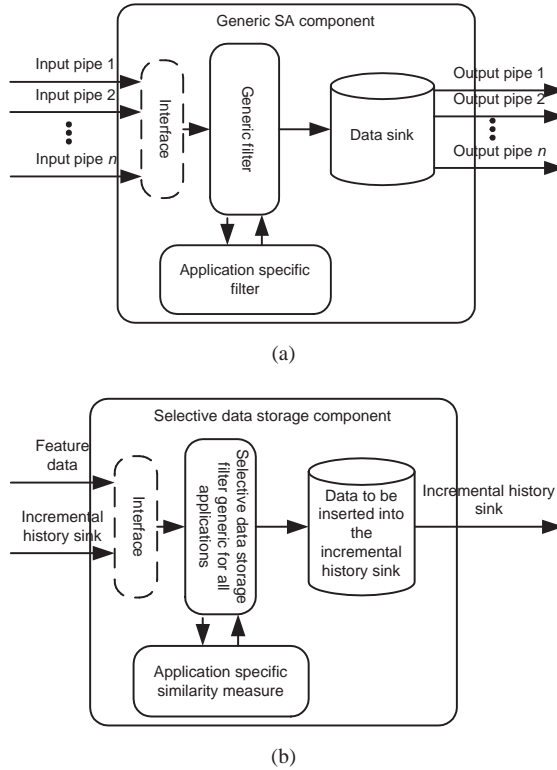


Fig. 15. Components of Smart Archive. (a): The implementation principle of a Smart Archive component at the generic level. (b): The implementation of the component for storing historical data selectively.

the pattern shown in Figure 15(a). Each component consists of input and output pipes, an interface, a generic filter, an application-specific filter and a data sink. The data is input into the component using the input pipes, after which it is fed into the filter specific to the component through an interface common to all components. The interface gives the applicator the freedom to customize the application-specific filter for application-specific purposes. From the filter the data is piped to the data sink and from the sink to the other components of the architecture. Using standard data storage technology for implementing the sink allows the applicator to access the data in a manner independent of SA. Implementing the sink using, say, a table in an SQL-compatible database enables direct access to the data through an ODBC / JDBC interface.

Finally, an example of adapting the generic component for the implementation of an application-specific component is presented in Figure 15(b). The example shows the implementation of the selective data storage component; the other components can be implemented in a similar manner. The data is entered into the selective data storage filter using pipes originating from the feature and history components. The filter provides the generic functionality of the selective data storage algorithm, such as retrieving the data being

currently processed from the feature component and archived data from the incremental history component. If the similarity measure implemented in the application-specific filter notices that the incremental history does not contain data resembling the feature data being processed, the data is piped to the sink. Finally, the output pipe transfers the data to the incremental history sink.

3.4 The architecture and operation of Smart Archive

A software architecture extending the reference architecture was developed using the components presented in Section 3.3. The component architecture is based loosely on the pipes and filters architectural pattern (Buschmann *et al.* 1996), which is suitable for organizing the cooperation of separate components highly dependent on data flow. The pattern divides data processing tasks into a number of sequential processing steps using filters to transform data and pipes to transfer the data between steps. The data processed by SA typically originates from entities producing sequences of observations. There are therefore three kinds of data in the system that the pipes transfer: on-going measurements, completed measurement series (a completed observation) and archived measurement series.

The architectural layout of Smart Archive is shown in Figure 16. The architecture is divided into live and history sections. Units in the live section handle the processing of data from on-going entities, that is, entities that can be expected to produce more measurement data. The organization of the data flow in the live section follows the reference architecture. The history section processes completed and archived measurement series. When measurements from an entity are completed, the completed measurements are transferred from the feature component to the selective data storage component. If the incremental history component does not contain measurements that are too similar, the completed measurements are archived. Archived data can be retrieved to be utilized in models using the component returning similar data.

The components of SA are updated in sequential order. The data flow and the order in which data are transferred during an update cycle are explained using the numbers and shapes above the pipes in Figure 16. The pipes transferring data from on-going measurements are marked with circles, the pipes transferring completed measurements with boxes and the pipes transferring archived measurements with diamonds.

The processing starts by checking the measurement component to see if there are new entities or new data available from the on-going entities in the system. If so, the new data are transferred to the pre-processing unit (pipe no. 1) on to the feature extractor (2) and the model (3). In order to be able to utilize archived data the model also needs data from the history sink. Therefore the feature data is piped to the component returning similar history data (4) for the purpose of determining the subset of archived data most similar to the on-going data. Archived data is retrieved (5) and returned to the model (6 & 7). After that, the model gives its analysis based on on-going measurements and knowledge from archived measurements. The results of the model are then piped to the results unit (8).

The sequence so far was all about analyzing data from on-going entities. The last three steps of the sequence perform the archiving of data. When the measurement sequence

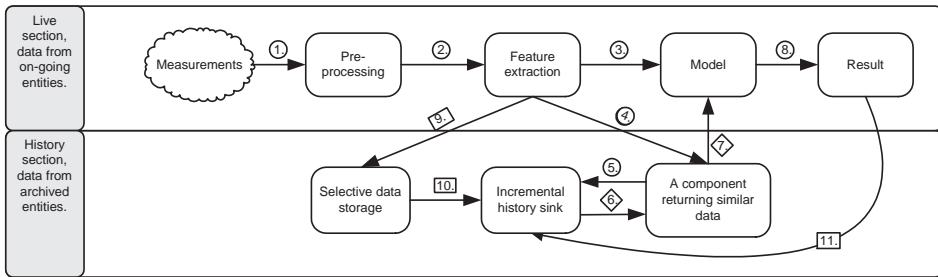


Fig. 16. The architectural layout of Smart Archive. The components are shown as rounded boxes and the pipes connecting them as arrows. The sequential order in which data is processed during an update cycle is shown as numbers above the pipes.

from an entity is finished, pipe no. 9 transfers the completed measurement series to the selective data storage component. The component then decides if the data will be piped (10) into the incremental history sink. The last pipe in the sequential order transfers the results of the completed measurements into the history sink (11).

One more property of the operation of Smart Archive needs to be described in order to understand and replicate its functioning: a basic explanation of the data model that was used to implement it. Figure 17 presents the entity relationship diagram of this data structure. The figure contains entities representing the components found in the architecture, attributes¹ of the entities and the relationships between the entities. Entities that would not exist without other entities are called weak entities and are drawn using double boxes (features, model and history data). Attributes that are derived from other attributes are distinguished with dashed circles. Each entity, except for the measurements, is composed of application-dependent attributes and system attributes needed for the operation of the system. The application-specific attributes are drawn below the entities and the system attributes above them. Three of the system attributes constitute a set of key attributes (denoted by underlined attribute names) that are used to distinguish the processed data items from each other: the identity code of the observation (id), time stamp that tells the time when the measurement was entered in the system (time_stamp) and a measurement number (m_no). The identification code and time stamp are self-explanatory; the measurement number is used to distinguish measurements in the event that two or more measurements from the same observation occur at the same time. The last system attribute (is_last) tells if the measurement is the last one from the measurement series of an observation, meaning that the observation can be considered completed and transferred to the history data, for example. The application-specific attributes are used to store the results of the various stages of transformations applied to the measurements made by the application (var 1, var 2, ..., var i). The diagram contains attributes for pre-processed data items (pp 1, pp 2, ..., pp i), features (feature 1, feature 2, ..., feature i), outputs of the model (output 1, output 2, ..., output i) and features stored in the history data (feature 1, feature 2, ..., feature i). The relationships between the entities are self-explanatory, but a few words on

¹In statistics an attribute is usually called a variable, in database terminology it may be called a column and in engineering in some cases a feature.

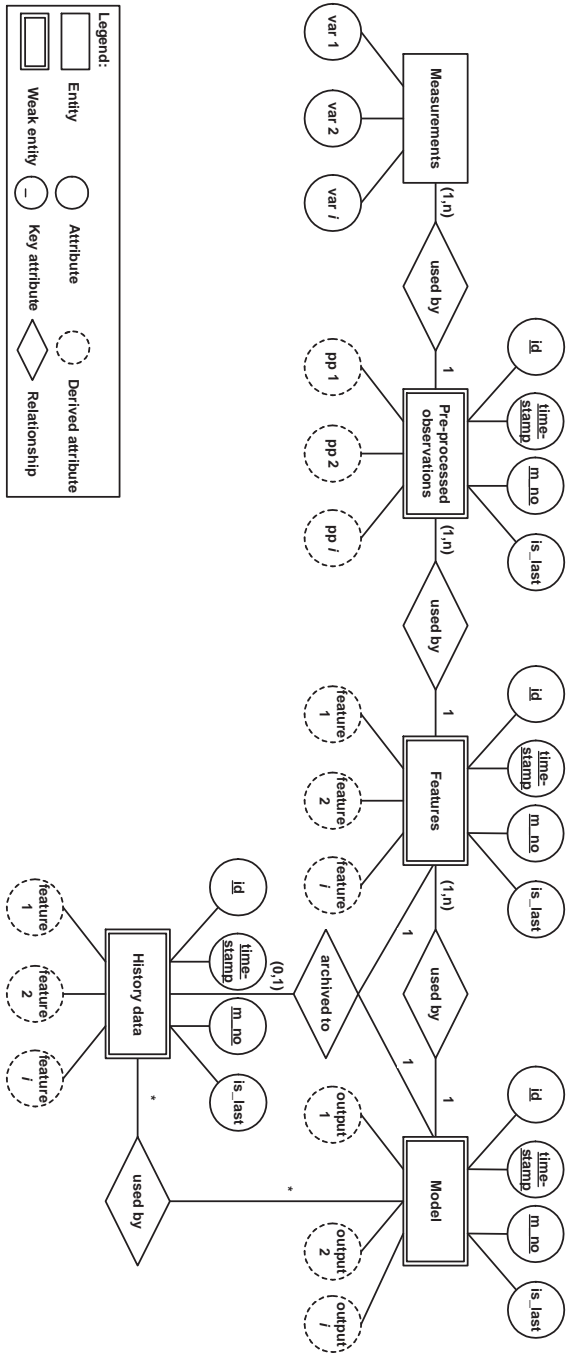


Fig. 17. Entity-relationship diagram showing the data structure of Smart Archive.

Table 5. An example presenting the format in which the feature data could be stored.

id	time_stamp	m_no	is_last	feature 1	feature 2	feature 3
261174	154656000000	0	0	13,1	18,9	A
151275	187833600000	0	0	10,1	19,1	quick
261174	1062241200123	0	0	18,1	18,1	brown
261174	1062241200123	1	0	11,1	19,20	fox
151275	1062241200456	0	0	1,14	20,5	jumps
190304	1079654400789	0	0	9,20	28,1	over
...
261174	3310416000000	0	1	9,14	1,0	the
151275	3343593600000	0	1	3,14	15,92	lazy
190304	3373574400000	0	0	65,35	89,79	dog.

the cardinalities of the relationships are necessary. The cardinality and ordinality of the relationship are marked on both sides of it and are used for describing the maximum and minimum number of entities the relationship consists of. For example, pre-processing a measurement may involve using data from 1 or more measurements (marked by (1,n) in the diagram), but one measurement is pre-processed precisely to one pre-processed measurement (marked by 1). The history data does not necessarily store any information from a feature (marked by (0,1)), but if it does, it stores the feature as it is (marked by 1). Also, the model does not have to utilize any history data but may utilize an arbitrary amount of it (marked by *) and an arbitrary amount of history data may be related to each output of the model (marked by the other * in the relation).

Finally, to make the idea clear, the data structure of an individual entity filled with a few measurements is demonstrated in Table 5. In this case the data structure is the one used with the feature and history components, but the data presented originates from the feature component. The table contains data from three observations, with id codes 261174, 151275 and 190304. The id code can either be acquired from the attributes of the measurement entity, or if the entity does not have an identification attribute, it is given by the system. The time stamp in this case is a millisecond value based on the UNIX time of the current date. The table shows the measurements sorted in ascending order using the time stamp. At the time 1062241200123 two measurements from the observation with id code 261174 have been made and thus they are distinguished using the m_no attribute. The last measurements from the observations with id codes 261174 and 151275 take place at the end of the table and therefore the measurement series from those observations form a completed observation and moving them to the history data can be considered. Measurements from the observation with id-code 190304 are still to be expected because the is_last column does not contain the value 1 for any of the measurements. Finally, the number of features is defined by the user who configures SA and in this case it is three. The types of features are not limited by the system, they can be for example in numerical, date or textual format.

The architecture meets the requirements laid down in the functional requirements analysis in Section 3.2. It instantiates and extends the reference architecture. The history

section stores data selectively and provides data most similar to on-going measurements to be utilized in models. The architecture can be configured to fulfill application-specific needs with its facilities for creating application-specific filters. It is suitable for processing continuously observed data from multiple entities and the workings of the architecture and its components are easy to understand.

3.5 Case study: A data mining application for predicting temperatures of steel slabs

This section presents a rather elaborate study on an application designed for predicting the post-roughing mill temperatures of steel slabs while they are heated in a walking beam furnace. This information can be used in fine tuning the heating of the steel slabs more accurately into their predefined goal temperature. The creation and results of the application are described here. A feedforward-type neural network is fitted on the measurement data measured from the furnace in an adaptive fashion.

In addition to presenting a promising data mining application, the presentation illustrates the motivation and importance behind developing a framework that can be used for implementing continuously operating data mining applications. The results presented in this section can be acquired by implementing the data mining solution from scratch or by using a framework. As is presented in the case study, the implementation based on the framework developed here is more efficient than that of implementing the application in an independent stand-alone style. These results show how a real application can be constructed using the more theoretical results of this thesis.

The application, the data set and work related to the application are described in Subsections 3.5.1 and 3.5.2. Subsection 3.5.3 describes how Smart Archive was configured for this application. The general properties of the neural network method that was selected for predicting the temperatures is then described in Subsection 3.5.4 and the results of applying the method are discussed in Subsection 3.5.5. Subsection 3.5.6 illustrates the advantages of using SA for implementation by comparing the implementation of the application from scratch and using Smart Archive.

3.5.1 Description of the application and the data set

Steel strips are produced from steel slabs in hot rolling mills. One part of the production process consists of reheating the steel slabs in a slab reheating furnace to a predefined temperature, usually between 1200°C and 1300°C. After this heating, the steel strips are formed by rolling the slabs. The first mill the steel slabs go through is the roughing mill. After the roughing mill, the temperature of the slabs, now called transfer bars, varies between 1050°C and 1170°C. The transfer bars then go through a few more procedures, the most important ones being rolling in the finishing mills and cooling. The end products are steel strips hundreds of meters in length. The strip can still be subjected to further treatments, such as cold rolling. Figure 18 presents these steps schematically.

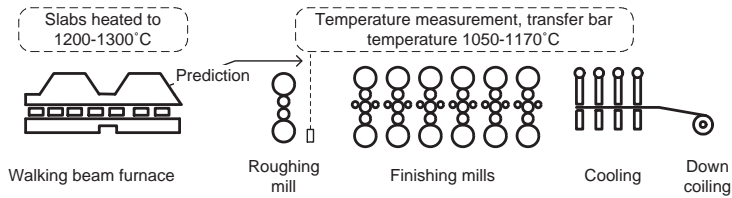


Fig. 18. Hot strip mill. Slabs are heated in the furnaces, roughed in the roughing mill, transformed into steel strip in the finishing mills, cooled and coiled into a roll. The figure illustrates temperatures of interest and the location where the post-roughing mill temperature is measured.

The two most commonly used types of reheating furnaces are the walking beam furnace and the pusher type furnace. This work looked at the walking beam furnace. The main difference between these furnace types is the way the slabs move in them. In the pusher type furnace, the slabs are in constant contact with the floor of the furnace and each other. The slabs that are inserted into the furnace push the slabs in front of them forwards. In the walking beam furnace, the slabs move on top of rails that convey them forward and do not allow them to touch each other.

The walking beam furnace is divided into independently controllable zones. The first zones of the furnace work on the heat produced in the next zones, i.e. the active heating zones where propane or carbon monoxide is burnt to heat the slabs. The last zones, called soaking zones, are used to fine tune the temperatures to the desired goal temperature set for each slab. It is important to adjust the parameters of the zones in such a way that the slabs are heated to the predefined temperature as accurately as possible. This will lead to a higher rolling quality of the slabs and to a higher quality of the finished product, the steel strip.

The high operating temperatures of the furnace make the collection of measurement data difficult, and it is simply impossible to collect information about some measures. One such undetectable measure is the inner temperature of a steel slab. However, the post-roughing mill temperature measurement of the surface temperature of the transfer bar can be used as an estimate of the inner temperature. In this work, a neural network model was developed to predict the mean temperature of the transfer bar based on the initial information of the slabs and on the measurements made from the furnace and its operating environment while the slabs are still in it. The precision of heating control can be increased when a prediction of the transfer bar temperature is available while the slabs are still being heated. When heating control is more precise, the temperature deviations of the slabs in the different heating categories decrease and tuning of the further treatments, including the finishing trains, is easier. The time it takes for the slabs to pass through the furnace possibly also decreases.

The data used in the work consists of two data sets measured from a hot strip mill. The first one was used to test a prototype and the second to test the software implementation of the model. The data set for the prototype was collected in the summer of 1999 and consists of observations from 3,200 steel slabs. The data set used in software development consists of observations on 200 slabs recorded in the summer of 2001.

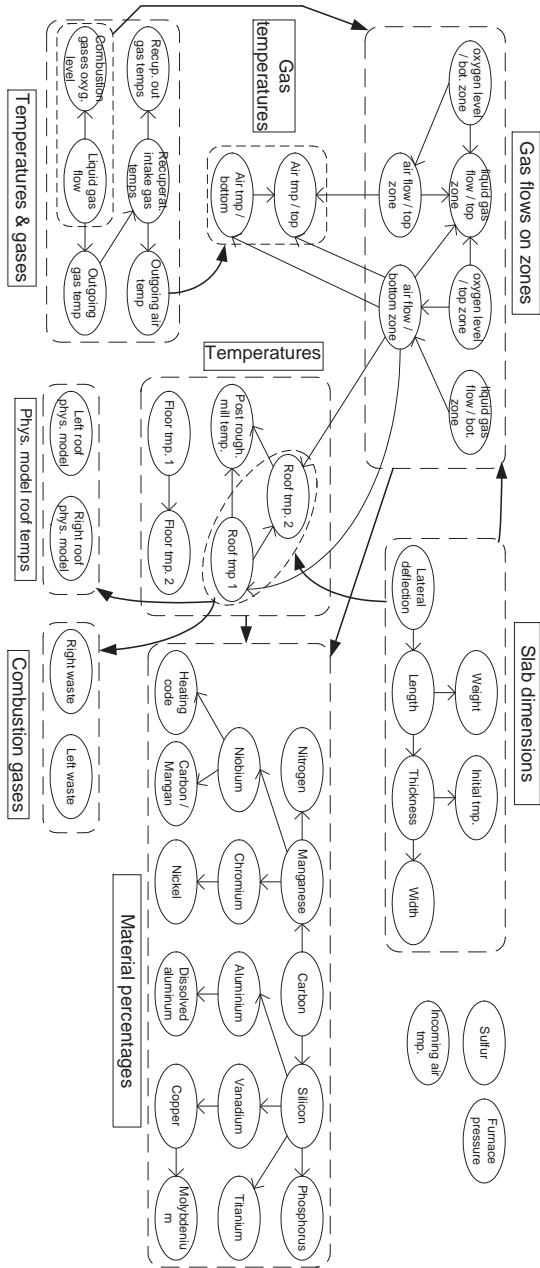


Fig. 19. The Bayesian network constructed to illustrate the interactions between the variables measured from the furnace and the slabs.

The preliminary variables used in the model were selected on the basis of expert information. In the later modeling phases, the set of variables was reduced even further. The total number of variables measured at the hot strip mill that are relevant to this application is about 150-200. This number was reduced first by carefully selecting the most important variables from the production database together with an expert. At this stage, there were still about 50 variables left. To gain more understanding of the relationships between the different variables, the interactions were studied using a Bayesian network. First, the connections of the network were identified using a search algorithm that generates the network structure automatically (Ramoni & Sebastiani 1997a,b), like in Section 2.5.3, but this time the network contained a considerably larger set of variables. Then, the final model was formed with an application expert based on the automatically generated model. Figure 19 shows the constructed network. The nodes in the figure stand for the different variables, while the arrows indicate the interactions between them. The variables were also grouped into larger units to portray higher level abstractions, such as the interaction between the slab dimensions and gas flows. The connections between these abstractions show how the abstractions interact. The network was especially useful for us researchers, who are not experts on the production process, but on modeling. More details about the work with the Bayesian network can be found from (Laurinen *et al.* 2001) and (Laurinen 2000).

The number of data points was reduced by including only the soaking zones in the model, since the opinion of an expert was that this is a good starting point for producing a working model. Moreover, slabs with post-roughing mill temperatures less than 1100°C were excluded because these slabs may have been standing on the roller table too long before roughing. This waiting time cannot be anticipated while the slabs are in the furnace. Twenty percent of the slabs fell into this category. Furthermore, the application deals with possible missing values of variables by checking whether the measurement from that variable is within the range set for the variable. If not, it is replaced by the lower limit of the variable if the measured value is lower than the lower limit or with the upper limit value if the measured value is higher than the upper limit. If the measurement is completely missing, it is replaced by the median of the range. There are also more complex methods for replacing missing values, including methods based on conditioning the missing value on the existing values (Ramoni & Sebastiani 1997b). Finally, the input and target variables of the neural network were scaled to a range from -1 to 1, which can make the training of the network faster and help in initializing the weights Masters (1995b).

3.5.2 Work related to the application

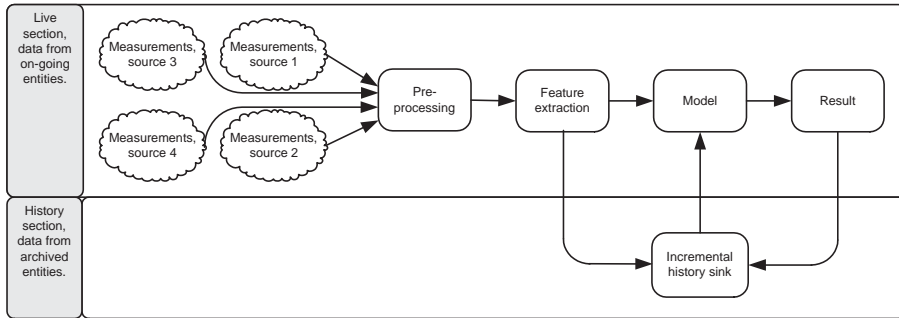
In theory, a model based on physics and mechanics could be constructed to predict the post-roughing mill temperature instead of the proposed neural network model. Unfortunately, this alternative is far from reality. The construction of a physical model to predict even the simplest process can be very difficult (Gong & Yao 2001, Lennox *et al.* 2001). Taking into account the fact that the operating conditions in the furnace are not constant and that steel slabs with varying metal concentrations are heated in the same furnace, it would be risky and very demanding to construct a physical model. Neural networks, on

the other hand, are a suitable tool for process modeling tasks requiring a highly non-linear approach (Lennox *et al.* 2001). Other alternatives for neural network models include the finite impulse response (FIR), auto-regressive with exogenous variable (ARX) and moving average with exogenous variable (ARMAX) models and other statistical classifiers, like for example those used in Chapter 2. The disadvantage of these alternatives is that they are perhaps not capable of capturing non-linearities equally well as neural networks (Gong & Yao 2001, Lennox *et al.* 2001).

An extensive survey of the existing applications in this area did not reveal any applications of this kind. However, there exist quite a few neural network applications in the field of steel production. The research most similar to this has been done on slab temperature prediction studies by Gorni (1997) and Nilsson (1998). The approach proposed by Gorni built a neural network model from information obtained from thermometers installed inside special slabs run through the furnace. The use of these thermometers is, however, so expensive that not many slabs can be run through the furnace, and excessive observations are not economically feasible. It may also be a false assumption that the data gathered with these few slabs would be representative. Moreover, the approach hardly allows for longer term changes in the environment, since these special slabs cannot be run through the furnace continuously. Nilsson (Nilsson 1998) predicts the same post-roughing mill surface temperature of transfer bars in her paper as this research does, including the neural networks. The purpose of her models is to use the prediction to set the parameters of the mills, which is different from the goal of this solution (more accurate heating of steel slabs). Her model is based on a data set available only after the slab has exited the furnace, including surface temperature and other measurements, and hence it cannot be used to control the heating of the slabs while they are inside the furnace, which is acquired using the proposed model. Her results are, however, comparable to the presented results at some level, since they predict the same temperature. The prediction accuracy, i.e. the root mean squared (RMS) error reported by Nilsson was 13. The RMS for the presented neural network prediction was 7.9 and that for the median-filtered prediction 7.5. For the predictions of the last observations, the respective values were 8.5 and 7.3. The benefit of the developed model in comparison to hers is that it does not use any data recorded after the slabs exit the furnace. It should be noted, however, that the data set and the process used are different, which makes the results mutually quite disparate.

The proposed solution is based on sensor data gathered on-line from the furnace and stored in a production database. The data used by the model can be acquired from the database at the production line, and the use of the model does not require the installation of any additional instrumentation. The model can be run as part of the existing plant data system, which makes it affordable.

Other neural network applications at hot strip mills include an application controlling the heating furnaces (Kim *et al.* 1998) and applications developed for predicting the rolling forces at finishing mills (Lee & Lee 2002). Applications for predicting the temperature after the last finishing stand based on information measured from the finishing mills and the transfer bar have been developed in Vermeulen *et al.* (1997) and applications controlling the shape of the strips in Fechner *et al.* (1994) and Yao *et al.* (1995). More information about data mining applications in hot rolling processes can be found from review papers like Jämsä-Jounela (2001), Takahashi (2001) and Schlang *et al.* (2001).



(a)

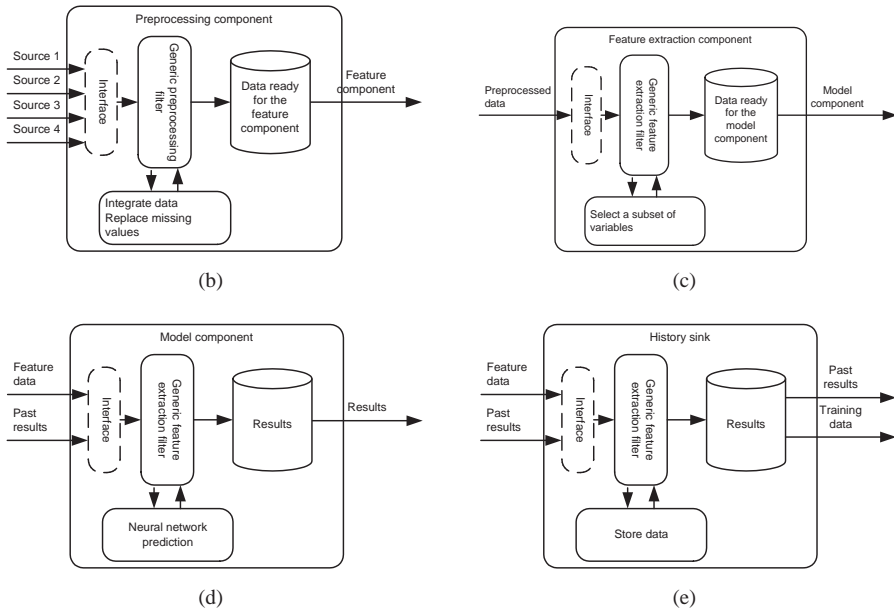


Fig. 20. Configuring Smart Archive for the steel slab temperature application. (a): Configuration of SA at the architectural / operational level. (b): The pre-processing component. (c): The feature extraction component. (d): The model component. (e): The history sink.

3.5.3 Configuring SA for the application

A short description of the application-specific coding of Smart Archive is given because it is not necessary to go too deep into application-specific details that cannot be applied anywhere else. The outline of the configuration is easy to understand by looking at the architecture and components presented in Figure 20. The application-specific data originating from the four SQL-compliant database tables is marked with clouds in Figure 20(a). The first source contains temporal measurements from the environment, the second and third spatio-temporal measurements from the environment and the slabs, and the

fourth contains static measurements from the slabs. Thereafter the measurements pass through the pre-processing, feature and model components of the live-section of Smart Archive. The work required for configuring SA for the application after constructing the processing chain was to configure the application-specific filters of the components. The first pre-processing task performed by the application-specific filter of the pre-processing component is data integration (marked in Figure 20(b)). Data originating from the four sources is integrated so that it forms measurement series in which variables are targeted on the positions where measurements from the slabs have been acquired in the furnace. After this the application-specific filter detects and processes possible missing values. Now each measurement (row) from the slab contains approximately 150 dimensions. The filter in the feature extractor is used to reduce this dimensionality (Figure 20(c)), so that measurements only from selected variables enter the model. The filter also handles the scaling of the data. From the feature extractor the completed observations (containing on average 81 rows) are output to the history sink (Figure 20(e)), so that data from past observations can be used for training the model. The data from the on-going observations are passed to the model component (Figure 20(d)), which applies the neural network implemented in its filter and gives a prediction (result), which is then output to the history sink. The prediction accuracy of the model is observed using the prediction data stored from the processed observations, and if the prediction accuracy decreases below a certain threshold, the model parameters are re-trained.

3.5.4 The model used for predicting the temperatures

The development of a data mining solution for this kind of application is a non-trivial task for a number of reasons, as was described in the functional requirement analysis in Section 3.2. However, the model selection method was not the best possible one, since the model that was applied was selected solely on the basis of intuition and previous experience with similar problems. A feedforward-type neural network was used on the prediction, since it was known that it resembles regression models and is capable of forming non-linear prediction surfaces for multivariate data mining tasks.

A feedforward neural network consists of connected data processing units called neurons with each connection having an adjustable weight. The neurons are organized into layers called the input, hidden and output layers. The data is fed into the input layer, further transformations are done in the hidden layer(s), and the result of the transformation is read from the output layer. The number of hidden layers and neurons, as well as the design of the connections between the neurons, defines how well the network can adapt to the data. More detailed descriptions of the functioning of a feedforward network can be found from various text books, such as Bishop (1995) and Press (1999).

The weights of the network are estimated from the data. The performance of the estimation technique determines how well the weights capture the behavior of the phenomena under study. The techniques are divided into two general categories: deterministic and stochastic methods. Deterministic methods have been designed to find a local minimum from the initial settings of weights in the network. The best known deterministic method is the back-propagation of error using derivatives calculated according to the error of the

network to tune the weights to optimal values. More sophisticated methods include the conjugate gradient technique. Stochastic algorithms have been designed to find a global minimum from the search space. They incorporate a form of randomness, allowing the optimized parameters, i.e. the weights, to change into new values that may be quite dissimilar from the previous values. This ultimately leads to the most optimal settings for the weights, but may require considerable computing time. Examples of stochastic algorithms include simulated annealing (description available for example in Masters (1995a)) and genetic algorithms (description in Press (1999)).

The method of estimating the weights in this work was a compromise between the fast deterministic algorithms and the computationally intensive stochastic algorithms. A mixture of these two methods, called hybrid learning, was used. In hybrid learning, a stochastic algorithm is used to estimate a good global starting point, and, after that, the deterministic algorithm is used to estimate the parameters more accurately using these starting points. After estimating the weights using the deterministic algorithm the final set of weights is assigned to the set of values giving the best performance (possibly on a validation data set).

When working with continuously acquired data, the estimated weights of the neural network can be kept constant or they can be re-estimated within suitable periods. Models in which the weights are re-estimated as the data changes are called adaptive models. The two basic ways of estimating the weights adaptively are batch and on-line learning. In batch learning, more than one observation is collected and the weights are re-estimated using this data set. In on-line learning, the weights are updated after every new observation. The advantage of using adaptive estimation is obvious: the parameters of the model are kept up to date.

A completely different question is the question of when an adaptive approach is needed. Applications have shown that an adaptive model can outperform a static model, but the decision to use adaptive modeling must be made based on the phenomena under study. If the operating environment is not likely to remain constant, or it is too difficult to collect a representative data set, then the use of adaptive modeling is justified. An example is a model that predicts the temperature of steel strips after the finishing stands on a hot strip mill Schlang *et al.* (2001). An adaptive neural network achieved a mean prediction error of 1°C. When re-training was discontinued and the static version was adopted, the error grew to 8°C. The cost of adaptivity is the more complex implementation of the models because of the larger number of parameters to tune. A decision must be made regarding when the parameters are re-estimated and how much data is used for the estimation, and there might also be constraints on the time available for re-estimating the parameters.

3.5.5 Results

In order to use the feed forward neural network model with time series data, a windowing function for the observations was used. The observations within the time window were then used to train the network parameters and to predict the temperature of the slabs in the soaking zones of the furnace. The prototype and the software implementation used somewhat different time windows, as will be described later.

The actual modeling work was started with a prototype made with Matlab and its neural network toolbox. The neural network was selected to include 29 input variables and one hidden layer with 12 neurons with *tanh* activation functions. The input variables were selected at this stage after experimentation with different input variable sets. They were the same variables as shown in Figure 19, except that the material percentages and unconnected variables were left out. A deterministic training algorithm (the conjugate gradient method) was used to estimate the network parameters. The time window that makes up the training data was selected to contain information from so many slabs that it contained at least 150 data points, which meant information from about fifteen slabs (time series measurements of the slabs contain different amounts of measurement points because the heating time of slabs is not constant). The network parameters were re-estimated every time a slab went through the roughing mill. The predictions were filtered using a mean filter. For a single slab, many observations were available in the soaking zones, and hence many predictions, too, were given. The cumulative average of these predictions was used as the predicted temperature, meaning that the first prediction was used as such, while the second prediction was the mean of the first and second predictions and so on.

The performance of the model was estimated by calculating different error statistics for the predictions. The statistics are calculated from the absolute values of the last prediction errors of the slabs before they exit the furnace. The mean error was 8.0°C and the median error 5.6°C. The median error was considerably smaller, because the prediction error for some slabs was large, and this increased the mean. The percentage statistics showed the proportion of predictions that were closer than the respective temperature. Predictions within 5°C (47% of the slabs) can be considered very good, predictions within 10°C (73% of the slabs) good and predictions outside the range of 15°C (14% of the slabs) poor. The product expert considered the prediction accuracy of the prototype good, since it is sufficient for setting up additional treatments of the products, and further studies, including pilot implementations of the model, should be made. Finally, the Figures 21(a) and 21(b) show examples of the predicted (dashed line) and measured (continuous line) values.

Software implementation of the model was started after successful implementation of the prototype. Transferring the Matlab model to the production line was not considered, as it would have been hard to fit it together with the information system, and the robustness of the solution would hence have been questionable. A feedforward-type neural network library with hybrid training algorithms was written, and an environment in which data from the production line database could be tested off-line was developed.

The structure and the parameters used by the training algorithm of the network were tested more extensively at this stage. Since there is no proven theory for constructing the network structure, a semi-automated empirical approach was used. The approach is based on testing a large number of neural network model candidates generated by using a set of rules. The parameters chosen this way were the input variables, the number of hidden layers, the number of neurons in each hidden layer and the parameters of the training algorithm.

All of the tested models included a particular set of input variables. In addition to this set, every model contained input variables selected uniform randomly from a set of candidates. After selecting the input variables, the number of hidden layers was selected,

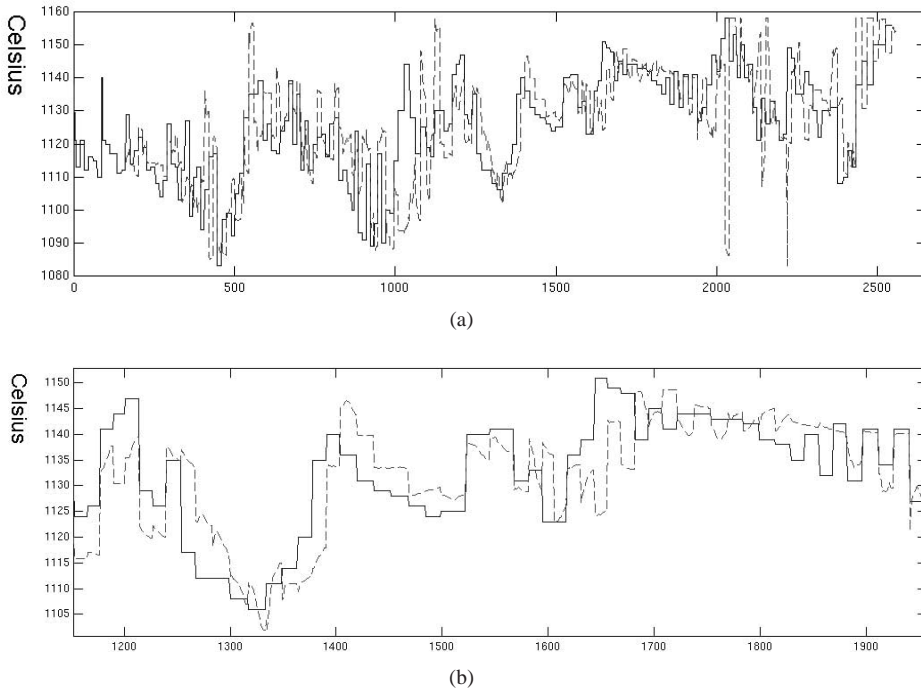


Fig. 21. Prediction of the prototype (dashed line) plotted against the measured post-roughing mill temperatures (solid line). (a): The most erroneous data set used with the prototype. The vertical axis shows the post-roughing mill temperature in Celsius and the horizontal axis shows the number of record in the data set. The slabs are in chronological order. (b): Enlargement of a set of data from Figure 21(a).

and either one or two hidden layers were used. Then, the number of neurons in the hidden layers was defined. The first hidden layer was randomly selected to contain a number of neurons from the range of $0.33n - 1.2n$, where n is the number of inputs. If the network also contained a second hidden layer, it was selected to contain a number of neurons in accordance with the first hidden layer in the same manner. Figure 22 shows the structure of the final neural network model with the input variables.

Finally, threshold rules were used to decide when to retrain the network and when to stop the training. It was not necessary to retrain the network unless prediction accuracy decreased below a certain value. Because of this, the weights were re-estimated after the average absolute prediction error of five consecutive slabs exceeded 9°C . These values were determined by testing different rules for updating the weights. The training of the network was completed after the training algorithm had applied the deterministic training algorithm to a certain number of starting points initialized using the stochastic algorithm. The number of starting points varied from model to model, as the models and training parameters were randomly generated. After finding suitable training parameters, it was not necessary to use cross-validation or other techniques for studying the training error

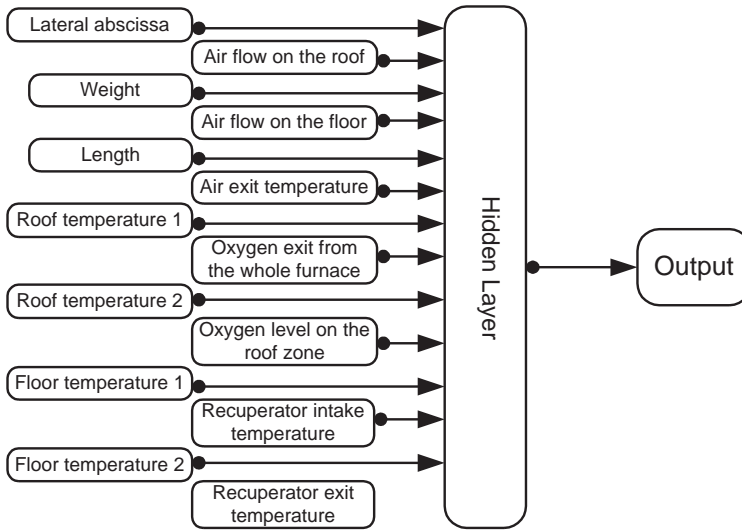


Fig. 22. The structure of the neural network with the input variables.

because the large number of tested models (>2000) ensured sufficient parameter quality of the best performing models. Parameters resulting in over-fitting or over-generalization of the model would result in poor prediction accuracy and an inadequate model, since each model was tested using the time series data, where each prediction can be considered to be a prediction for validation data.

The models were calculated on multiple clients connected to a database containing pointers to uncalculated models. After a client finished its calculations, the prediction statistics for the calculated model were entered back into the database. The best performing models were then selected from the database, and the results were analyzed in more detail.

The subfigures in Figure 23 show the predictions of the neural network plotted against the measured values. In a perfectly performing model, the dots in the figure should lie in the diagonal line. The two outer lines are 15°C away from the theoretical optimum, representing the limits of acceptable deviation. As we can see from Figure 23(a), most of the predictions are within the 15°C limits, but some clearly exceed them. These points outside the limits are mostly predictions from single slabs. Figure 23(b) presents the predictions for the last observations on the slabs before they exit the furnace.

Cumulative median filtering ² of the predictions of each slab was applied to find out if it would help to bring the predictions outside the limits closer to the optimum. Figures 23(c) and 23(d) show these points plotted in the same manner as in Figures 23(a) and 23(b). The median filtering brought the predictions of each slab clearly closer to each other and removed most of the large errors in the lower part of the plot. Figure 23(c)

²Cumulative median filter means in this context the application of a function that returns the median of the predictions available at a point in time. As the slab closes the exit of the furnace, the value of the filter approaches the median of the predictions for that particular slab.

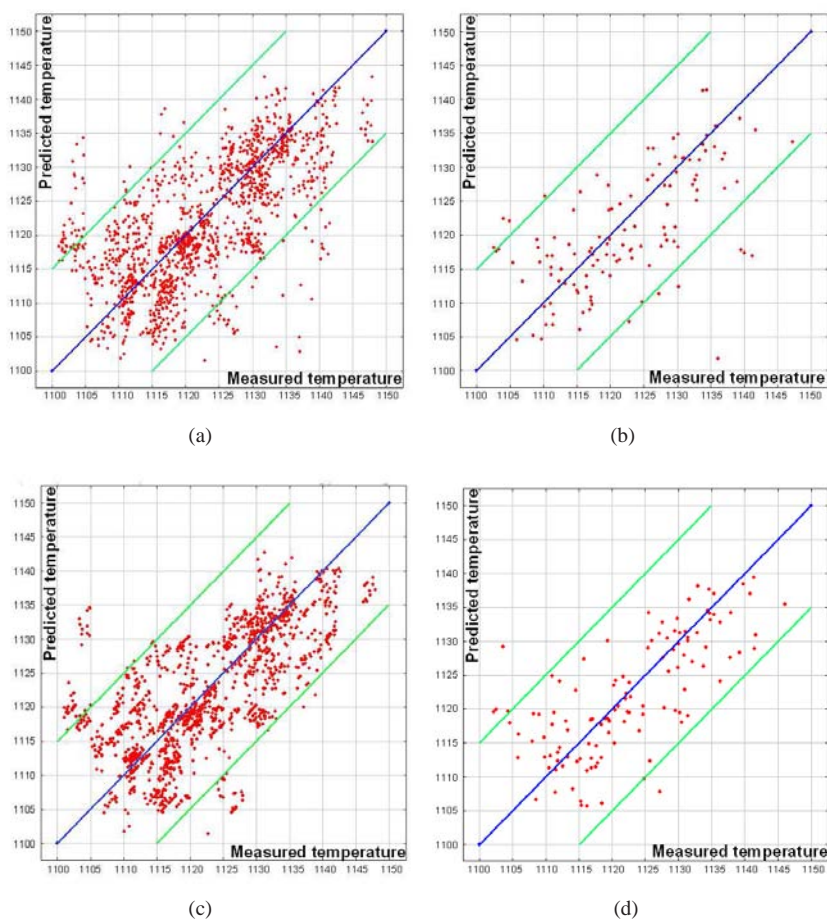


Fig. 23. The measured values of the post-roughing mill temperature have been plotted on the x -axis and the predicted values on the y -axis, and some random noise has been added to the measurements to distinguish the points more clearly. Plots 23(a) and 23(b) are plots for the neural network prediction of the best performing model. In plots 23(c) and 23(d), median filtering has been applied to the neural network prediction. Figures 23(a) and 23(c) contain all the observations from the soaking zones, while the Figures 23(c) and 23(d) only contain the last observation from each slab.

shows that the points outside the limits are grouped together and present an observation on one slab. It is likely that the errors originate from a slab that has remained on the roller table for an abnormally long time after exiting the furnace and has therefore lost some of its temperature before entering the roughing mill. The time the slabs stay on the roller table cannot be anticipated while they are heated, but it is usually standard. In the operation on the production line, the model is informed of the delays, and the effect can hence be controlled. At the time of this work, the information was not available off-line at

Table 6. Statistics for the prediction error of the software implementation of the neural network model predicting the post-roughing mill temperature. The column titled "normal" shows the statistics for the neural network prediction, the column "median-filtered" shows the corresponding statistics for the median-filtered prediction, and the last two columns show the same statistics for the last observations of the slabs.

	Normal	Median-filtered	Last measurement only	Last measurement only, median-filtered
Mean error	5.9°C	5.6°C	6.3°C	5.5°C
Median error	4.5°C	4.0°C	4.9°C	3.9°C
Standard deviation	5.2°C	5.0°C	5.7°C	4.9°C
RMS	7.9	7.5	8.5	7.3
< 5°C	54 %	55 %	52 %	56 %
< 10°C	80 %	80 %	80 %	79 %
> 15°C	6 %	5 %	9 %	5 %

the set of data extracted from the plant information system for constructing these models.

Table 6 shows the same statistics for the prediction results as were calculated for the prototype, the only difference being that these statistics were also calculated for predictions from all of the observations instead of merely the last values. The average of the absolute error values of the unfiltered neural network model was 5.9°C and that for the filtered model 5.6°C. The median error was 4.5°C for the neural network predictions and 4.0°C for the filtered values. The statistics calculated from the last observations of the slabs are comparable to the prototype statistics.

The prediction accuracy of the model is sufficient for a tentative implementation at the production level. The large prediction errors in 5% of the predictions are problematic, however, and studies are being made to eliminate this problem using prior information, for example. The presented results have been also reported in Laurinen & Rönning (2005).

Furthermore, although the application is functioning as planned, there are a lot of ideas for future development as well. Firstly, among the most important ones is the usage of spatio-temporal information from the slabs, which is very limited at the moment. Chapter 4 discusses an interesting possibility of using trajectory data and provides a brief introduction on how data formed by the slabs could be used for retrieving most similar heating trajectories. Secondly, the method that is used for adapting the neural network into the data could be further developed. At the moment the adaptation is done in a batch learning mode after the prediction error exceeds a certain threshold. Using the incremental learning mode could be a viable alternative. Last, but certainly not least, is the usage of the history component for augmenting the training set. The technique presented in Chapter 4 can be used for this task as well.

3.5.6 A comparison of the two ways of implementing the application

Two versions of the application were implemented. Both versions give similar results, but are implemented in different styles. The first version of the production line implementation of the model was started by building it from scratch and the second version was based on SA. With the first version, most of the implementation of the algorithms and communication between the software and data sinks was highly application-specific. In its most advanced form the version used an SQL-compliant database to communicate data between the application-specific algorithms. The algorithms connected to the database retrieved the data processed by the previous algorithm in the chain and output their results to the database. The implementation of this version did not follow any existing framework. The second version, which used SA for the generic operations, was implemented as already described in Subsection 3.5.3. In the second version the application-specific parts were implemented in the application-specific filters and the overall structure was implemented using the framework offered by SA. The tools used for implementing both versions included Java for implementing the filters; MySQL and Oracle were the supported data sinks, and SQL queries (through a JDBC connection) were used to pipe the data.

Although software components for the pre-processing-feature extraction-modeling cycle could be identified (with careful examination) from the source code of the first implementation, almost all of the code was fully application tailored. During the laborious implementation of the model, the need for a generic framework for constructing individual data mining applications became more and more evident. Unfortunately (or not), the tailored implementation was finalized before the development of SA was started and only after that the application was re-implemented using it. The upside of carrying out the implementation twice with different approaches is that it offers a chance for comparison. The two approaches are compared using criteria reflecting the guidelines of the requirements analysis of Section 3.2. The following categorization presents the evaluation, with the implementation from scratch denoted with **I** and the implementation using SA with **II**.

The presence of the reference architecture:

I The implementation is an application-specific adaptation of the reference architecture where the reference architecture is not particularly present. Feature extraction filters are hard-coded as part of the base application. The model is hard-coded as part of the application.

II The reference architecture is the backbone of the system and is clearly present. The amount of work needed for data pre-processing is approximately the same as in the first implementation, but the implementation of the application-specific filters can be kept separate from the framework using the interfaces. Feature extraction and model filters are adapted to application-specific needs using the interfaces.

Use of history information:

I There is no support for history information. The user must implement his or her own components for utilizing history information, which raises the threshold for utilizing it.

II The architecture of SA supports the utilization of history information. The user has the possibility to take advantage of the facilities for identifying similar measurements from long observation periods.

Level of transparency:

I Almost no transparency. It is quite difficult, even for a person familiar with the reference architecture, to understand how the components of the system relate to each other. The documentation of the software requires documenting all operations used in transforming data (about 10,000 lines of code) to be explained.

II The level of transparency is high. A person familiar with the reference architecture can understand the workings of the system at a glance. Only application-specific filters need to be documented.

Usability:

I The implementation meets the application-specific needs and works well as long as no major changes are made to the furnace data system. When upgrading becomes necessary, it must be carried out by hard coding the changes in an application-specific manner.

II The framework is tailored to the application-specific needs and works well. It is easy to adapt to new requirements, with the option of using third party components developed for the SA framework.

Implementation time:

I It is hard to tell exactly how much time it would take to implement the application, since different versions of the implementation built from scratch have been made in an iterative cycle. If the implementation was started from scratch again, it would take about six months to implement the solution.

II It takes about one and a half months to implement the application-specific filters and a week to configure the information flow of the architecture for the application.

Software quality:

I Bugs are equally likely to appear in all the parts of the code. Because of this, all of the code has to be tested.

II Bugs are more likely to be present in the application-specific filters because the application independent framework has been extensively tested in its development phase. Therefore, only application-specific filters need to be tested, which is a much smaller task than testing all of the code.

From the standpoint of implementation, the most important advantage was the decrease in implementation time and the increase in quality. Almost no time was spent in designing the architecture and operation of the software, since SA is based on the reference architecture. In fact, one of the largest design tasks was to decide in which filters to place the application-specific code, which is not a very hard decision (using the reference architecture). Creating the application did not require coding an application-specific

code for handling the data flows and hard coding the set of variables as part of the application. These time-consuming tasks could be implemented by properly configuring SA. The amount of time spent tracking bugs decreased and the bugs could be traced by default to an application-specific code. It is also much easier to explain the application logic to the customer using the framework than with the fully-tailored approach.

The ability to use history data is an important feature from the standpoint of modeling. In this particular application the production conditions vary so much that using a training set containing data from a limited time frame does not necessarily lead to optimal results. When the training data could be extended with similar measurements from longer periods of history it would be more likely that measurements resembling the current production conditions will be present. However, the studies on augmenting the training set with similar data are still in an early phase and will be continued after the completion this thesis. During this work techniques that can be used in the history component for selective data storage and retrieval were developed (as is described in Chapter 4), but not yet utilized in the application.

3.6 Work related to Smart Archive

The field of data mining architectures is still rather unexplored, since very few earlier architectural studies reporting the reference architecture as part of them were found in the literature. However, architectures that can be adapted to implement the reference architecture do exist, along with architectures and frameworks in other fields overlapping in some aspects with data mining architectures. A data mining architecture supporting the reference architecture is reported in Ahmed *et al.* (1998). The work approaches the problem from a data warehouse (DW) perspective and does not describe a framework for implementing DM applications. Other DW-centric studies are presented in Mariño *et al.* (2000), Chan *et al.* (2000).

Architectures for processing data streams or data feeds have been developed in Roodyn & Emmerich (1999), Lui *et al.* (2000), Fernandez (1998), Hsiung *et al.* (2002). Roodyn & Emmerich (1999) presents an architectural style for the integration of multiple real-time data feeds on Windows NT platforms. The design includes a live data object manager and a historical data object manager and the idea of applying filters on the data feeds for "filtering those information from the data feeds so as to reduce the information overload". These properties reflect the ideas behind data mining applications, but it is obvious that the authors have not had data mining applications in mind when designing the architecture. For example, the design does not mention the stages found in the reference architecture and the history component has no intelligence for selective data storage or similarity detection. The proposal by Lui *et al.* (2000) (called iFlow) is also a component-based framework for streaming data processing. What is common to this work is the property of utilizing continuously observed data streams and the architectural style of using pipes to connect components. Fernandez (1998) has a quite different design in his architectural style for object-oriented real-time systems. However, he notes that "design decisions can be evaluated based on mathematical analysis of real-time behavior previous to testing activities" and applies his system for "real-time data acquisition and alarm mon-

itoring of industrial processes". Finally, Hsiung *et al.* (2002) present a framework called VERTAF for embedded, real-time systems development. The framework shares some of the motivation of this work, offering components and interfaces for implementing embedded systems. None of the authors mentioned above have developed their designs for data mining applications. They do not include the reference architecture in their descriptions nor do the designs contain mechanisms for selectively processing history data. These contributions are valid in their respective fields of application, but contributions similar to the contributions presented in this work were not identified. On the other hand, any of these designs could be further developed for presenting an architectural style and framework for implementing data mining applications. However, even if these designs were developed further to make them suitable for data mining applications, it is questionable whether they would achieve the same quality as a framework built solely for data mining applications. It is clear that a consistent comparison is not possible.

3.7 Discussion

This part of the thesis presents an application framework and architecture for implementing data mining applications and an application for predicting the post-roughing mill temperatures of steel slabs. The main points of the functional requirements analysis were presented in order to motivate the design solutions behind the framework. The components and architecture of the SA framework were explained at such a level that people interested in experimenting with the architecture can implement it and adapt it to their applications and tools. The benefits of the SA framework in application implementation were outlined in a case study in which a method for predicting the post-roughing mill temperature of steel slabs was developed. The prediction was done using neural networks while the slabs were heated in the walking beam furnace. A Bayesian network was first used to visualize and clarify the interactions between the various variables affecting the heating process. After that, an adaptive neural network model was applied to the data, with a mean prediction error of 5.6°C. The results were accurate enough for a tentative application of the model on the production line. A comparison of implementing the application using a fully-tailored approach and implementation using SA was analyzed. The major benefits of using SA for implementation were a reduction in development time, higher quality, extensibility and a more transparent system structure.

In the previous chapter of this thesis the process of finding a data mining solution from a measurement data was discussed - which is a challenging task in itself. Nevertheless, finding the solution is often not enough - for example in data mining projects developing on-line applications the project is not completed until the solution has been implemented as a data mining application capable of on-line utilization of measurement data. The search for the solution and implementation of the found solution are two different tasks linked by the data mining solution. They also require very different specialization: the data mining process requires advanced knowledge of the data transformation functions, while the implementation phase requires advanced knowledge of software engineering. It is often the case that people skilled in data mining are not as skilled in software engineering. This may be part of the reason why there exist a lot of studies wherein a good data

mining solution for an issue has been found, but it has not been implemented as a practical application. On the other hand, if the challenge of finding a data mining solution is posed to skilled software engineers capable of implementing the solution, they might not have the skills for finding the solution. This part of the thesis presented an application framework and architecture that can hopefully be used to lower the threshold for implementing the solution found as a real world application.

A lot of the contribution presented in this thesis was inspired by the steel slab temperature application and the needs it has imposed for the data mining process and framework. These results illustrated the capabilities of a real world data mining application that can be implemented by using the developed approach. This is of utmost importance for justifying the usability of the contribution of this thesis. The purpose of the application-independent, theoretical work presented here is to facilitate the creation and implementation of data mining applications. These results have shown that the methodology is valid and applicable in practice - based on the results acquired after implementing the presented solution.

The development of the presented SA framework that was used for implementing the solution was an iterative process in which the application had a central role. The work was started by hard coding the solution as an independent application reading the measurement data from flat files, which was a task that consumed both time and patience. Shortly after that implementation, it was clear that the requirements (see Section 3.2) set by the application were not being met using the approach. After rewriting the application to meet some of the requirements it became evident that the application should give more freedom for changing the algorithms and variables it is using. This led to re-thinking the application and changing the way it is designed. The new approach was to design and implement the application in a top-down approach instead of the bottom-up approach applied up to that point. The application framework presented in this chapter was created for that purpose, using the reference architecture as a backbone. Finally, the application was implemented using the framework. After this, the application had grown to meet all the requirements set for it. It had evolved from a platform-dependent, hard-coded application that is reading flat files, to a platform-independent, re-configurable application that can be tailored to be run both in the information system of the steel plant as well as in the simulation environment run in the research laboratory. Of course, it would have been much better if the framework would have been available already when starting the implementation of the first version of the application. That would have avoided a lot of unnecessary implementation work. But, on the other hand, the framework would not exist if its importance had not been learned the hard way.

Furthermore, the data mining process presented in Chapter 2 of this thesis was not available during the creation of the steel slab temperature application because it was created in the project studying the quality of the spot welding joints only after the slab temperature application had moved to the implementation and prototyping phase. There is no doubt that using the semi-open data mining process for feature selection and model testing would have been very fruitful in this application as well. Partly because of that, the features and model of this application might not be optimal - there are possibilities for improvement in feature and model selection that are left for future work. However, the parameters of the neural network were selected with care and the performance of the application meets the requirements set for it.

Future development work on the topic has already begun. Emphasis is given to devel-

oping the incremental history component of the framework and making it easier to link it to the other components. The history component of the current development version contains the base technology needed for detecting similarity (or novelty) from the measurement data. Future work also consists of developing an improved, more product-like, version of the framework. The current version is easy to apply and configure for a person experienced with the framework - the results and principles are easy to understand by any data miner. The future version should also be easy to apply and configure by persons not so experienced with the framework. Finally, the framework will be made easier to reconfigure. At the moment, applications can be constructed by configuring the framework to application-specific needs and then running them on top of the platform provided by the framework. Future research and development work will be addressed on runtime configuration of the framework. This will allow for the changing of the components and filters of the framework runtime, which will result in a greater capability to react to changing conditions. The current version supports only runtime reconfiguration of the parameters of the application-specific filters, such as the parameters of the neural network. The current version, its capabilities and future development ideas were considered so promising that funding for a new research project developing these properties of Smart Archive, and new applications, has been granted by the National Technology Agency of Finland (TEKES).

4 Similarity detection and an efficient algorithm for the history sink

This chapter describes an efficient algorithm for similarity detection. Before presenting the algorithm, Section 4.1 gives a short overview of what similarity (and novelty) detection is and what it can be used for. The algorithm developed is applicable for *trajectory* data and can be used for increasing the performance of the history component of Smart Archive. Section 4.2 explains what trajectories are and what has to be considered when calculating similarities between them. Section 4.3 then presents the algorithm, which can be applied on all kinds of trajectory data in which one dimension of the data is increasing. Section 4.4 shows that the complexity of the algorithm is linear when the measurement dimensions are increasing and Section 4.5 presents an empirical estimation of efficiency under more general conditions. Finally Section 4.6 presents work related to the algorithm, and a discussion of the algorithm is given in Section 4.7.

4.1 Principles of similarity and novelty detection

Similarity and novelty detection are rather broad terms. People face situations in which they apply them throughout their everyday lives. Recognizing the events of meeting new people, hearing new songs or visiting new places are natural examples of everyday applications. Another example is the capability of the human body to recognize organisms that do not belong there, for example viruses. For living organisms, similarity and novelty detection and the capability of responding to novel events is more or less natural. However, the basic principle behind all similarity and novelty detection, be it human or computational, is the same - the level of similarity of a new item is based on a comparison of the item to an existing collection of items. The degree of similarity of the new item can be evaluated based on the evidence acquired using the comparison. After this, the item is considered novel if it fulfills some application-specific rule. In some cases the query item could be defined as novel if it is different from all the items in the collection. In the case where all new observations are defined to be novel, items are defined as novel even if an exactly similar item is already in the collection. This leads to noticing the connection between similarity and novelty detection. Novelty detection is obviously a special case of

similarity detection. It is a two-class problem in which the observed item is either novel or not. Because novelty detection methods incorporate similarity measurement and in many cases similarity measurement methods enable output in continuous precision, the degree of novelty (or similarity) can be calculated. This degree of similarity is then transformed into binary output (novel or not) or could be used for other purposes as well. This touches on one of the key problems in novelty detection - where is the line drawn with regard to how dissimilar an observation has to be before it can be considered novel?

What are some typical applications of similarity and novelty detection? The general logic behind the applications is to compare a query observation to the observations in a database, as stated above. For example a data storage application could compare a new observation to the observations in a database and decide to store the new observation only if it is novel. The same principle could be used in retrieving the most similar observations from the database and to use them for example to expand the training data set of a classifier. Information about the novelty of the new observation could also be used to calculate a statistic describing the confidence of the classification of the new observation - a novel or dissimilar observation may lead to unexpected behavior of the classifier.

Real world applications of novelty detection can be found for example in the fields of fault detection or inspection. Marsland *et al.* (2005) have developed a neural network application that controls a mobile robot capable of novelty detection by ignoring neural network inputs that present normal operation. The mobile robot can be used for example on remote inspection tasks, where its task is to observe anything that deviates from the usual. This can be useful in environments where humans cannot enter or in repeatedly activated inspection tasks, for example guard work. Pontoppidan *et al.* (2005) present an application monitoring the condition of large-scale diesel engines. A mean field independent component analysis model is trained on normal signals measured from the engine and when a fault is induced in it, the model is capable of detecting it. Furthermore, an overview of intelligent fault detection can be found from Worden & Dulieu-Barton (2004). Singh & Markou (2004) have used novelty detection methods to recognize unknown objects from video streams. The application lets the user manually label the objects that have not been recognized using a neural network model trained for identifying the objects known so far. After manual labeling the novel objects are added to the training data of the network. Finally, Tarassenko *et al.* (1995) have developed a novelty detection approach for identifying masses in mammograms. They report that breast cancer is a major cause of death among women aged from 35 to 55 years and if it can be recognized in an early stage by means of a mammogram, it is easier to cure. The method was based on comparing new mammograms to the point density functions of earlier mammograms using local Parzen estimators. The system was capable of correctly drawing attention to the regions of mammograms that needed further analysis from a human expert. A good review of both theoretical and application developments of novelty detection can be found from review papers Markou & Singh (2003a) and Markou & Singh (2003b).

The next sections introduce the contribution of this thesis in the field of similarity / novelty detection. An algorithm that is developed for a special case of similarity measurement is presented. The algorithm can be applied to observations that are composed of finite measurement series (trajectories) measured possibly from more than two quantities. The algorithm presented is substantially more efficient than the practice currently used for calculating the similarities and can be used for example in the history sink of SA.

4.2 About trajectories and measuring similarities between them

The Merriam-Webster dictionary defines "trajectory" as: 'a path, progression, or line of development resembling a physical trajectory'. In other words, trajectories are series of measurements made on an entity of interest and interconnected in the measurement space. Trajectories of measurement values varying over time (temporal) or location (spatial) or both (spatio-temporal) are observed in many application areas. Examples vary from studying the variation of temperature over time (and its possible effects on the climate) to predicting stock prices. The development of computers and digital data storage devices has made it feasible to process and store trajectories automatically. There are tools for efficiently producing answers to such questions as: "given a trajectory $traj_q$ and a collection of trajectories T , what are the x trajectories most similar to the query trajectory $traj_q$ in the collection T ?". In order to find an answer, it is necessary to formulate a measure of similarity between two trajectories, to calculate the values for the similarity measure between $traj_q$ and the trajectories in the set T , and to return the x trajectories with the smallest value of the measure, in case smaller is equivalent to more similar.

In order to be able to measure the similarity between two trajectories, it is necessary to define a way for measuring the similarity between the values that make up the trajectories. The values are measurement points in a m -dimensional measurement space, where m is the number of observed quantities. The L_p norm is a popular point-to-point measure of distance, and it is defined for two m -dimensional points a and b as

$$L_p(a, b) = \left[\sum_{i=1}^m (|a_i - b_i|^p) \right]^{1/p}. \quad (1)$$

Setting p to 1 gives the Manhattan distance and, the Euclidean distance is obtained when $p = 2$.

The process of calculating the similarity between two points is more trivial than calculating the corresponding similarity between two trajectories. There are three major features that make the calculation more complicated for trajectories:

1. **Measurement intervals.** The values may be observed in equidistant or varying distance intervals.
2. **Number of values.** Two trajectories may contain different numbers of measurement points.
3. **Dimensionality of measurement space.** Trajectories may be measured in two- or multi-dimensional spaces.

Figure 24 illustrates these differences. The trajectories $traj_a$ and $traj_b$ contain the same number of points ($n(traj_a) = n(traj_b) = 6$), are of equal length in the horizontal direction (time in this case) and are measured in equidistant intervals ($s_{i+1} - s_i = c, \forall i = 1, \dots, n - 1$). It is obvious, but still worth pointing out, that the measurement values (in the vertical axis) may increase and decrease over time, but the value of time between two consequent values (horizontal axis) is always increasing. In other words, $t_i < t_{i+1}, \forall i = 1, \dots, n - 1$. The increasing nature of one of the dimensions is emphasized here because it is the key point in the development of the similarity measurement algorithm presented in the next section. The L_p norm can be trivially extended to measure the similarity between the types of trajectories where each point has a parallel counterpart in the other trajectory,

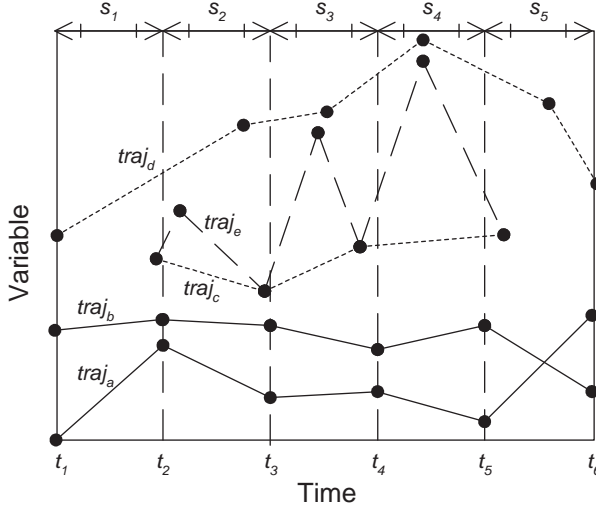


Fig. 24. Trajectories *a* and *b* are observed at equidistant intervals with the same number of measurement points, while trajectories *c*, *d* and *e* are observed in varying intervals and varying amounts of measurement points.

like for example $traj_a$ and $traj_b$, which are plotted using the solid line in Figure 24. Since all the points of each trajectory have matching points (in the vertical direction) in the other trajectory, the L_p norm can be calculated using the formula

$$L_p(traj_1, traj_2) = \sum_{i=1}^{n(traj_1)} \left[\sum_{k=1}^m (|traj_{1,ik} - traj_{2,ik}|^p) \right]^{1/p}, \quad (2)$$

which is the L_p norm between the matching points of two trajectories summed over all the points of the trajectories. This is one of the simplest examples showing the calculation of similarity between two trajectories. But what if the assumptions of equidistant intervals, equal numbers of measurement values, and two dimensions were relaxed? The trajectories $traj_c$ and $traj_d$ in Figure 24 are an example of trajectory data where the formula 2 could not be applied to the calculation of distances, or at least it using it would not make sense. Although the points are presented in two dimensions, both trajectories contain a different numbers of values unequally distributed in time. To be able to calculate the measure of similarity for these two trajectories, one needs to define how to find the matching points closest to each other in the two trajectories and how to take into account the varying numbers of points in the trajectories.

The proposed algorithm is a natural extension for calculating the similarity measure for trajectories of this kind, where the assumptions are more relaxed. The algorithm is efficient in terms of usage and calculations. Firstly, the increasing nature of one of the dimensions is used to optimize the performance of the algorithm - a result which can always

Algorithm 1: The intuitive algorithm for calculating the similarity between two trajectories.

input : trajectories $traj_a$ and $traj_b$ of size $n(traj_a)$ and $n(traj_b)$
output: the distance between the trajectories, $trajectorydistance$

set $trajectorydistance$ to 0;
 set $smallestdistance$ to ∞ ;

for $i \leftarrow 1$ to $n(traj_a)$ **do**
 for $j \leftarrow 1$ to $n(traj_b)$ **do**
 if $d(traj_{a,i}, traj_{b,j}) < smallestdistance$ **then**
 | set $smallestdistance$ to $d(traj_{a,i}, traj_{b,j})$;
 end
 end
 increment $trajectorydistance$ by $smallestdistance$;
 set $smallestdistance$ to ∞ ;
end

set $trajectorydistance$ to $(trajectorydistance/n(traj_a))$;
return $trajectorydistance$;

be used with temporal trajectories, for example. Secondly, the similarity measure it calculates nearly fulfills the requirements of a metric space, which makes it more credible. The following sections present the algorithm, its computational efficiency and its usability.

4.3 Proposed algorithm

The algorithm described in this section can be used to measure the similarity between two trajectories that contain values observed at varying intervals, possibly containing different numbers of values, with one of the dimensions increasing. This algorithm outperforms the intuitive algorithm currently used.

What exactly is the "intuitive" algorithm? If one were given the task of devising an algorithm that calculates the similarity for trajectories fulfilling the above-mentioned conditions, the intuitive idea would be to use the kind of algorithm represented by the pseudo code in Algorithm 1. The algorithm starts from the first point of the trajectory $traj_a$, scans all the points of the trajectory $traj_b$, adds the distance between the closest pair of points in the two trajectories to the overall distance between the trajectories, and repeats this until all the points in the trajectory $traj_a$ have been processed. Finally, the distance is divided by the number of points in $traj_a$ to neutralize the effect of varying numbers of points in the different trajectories.

Why is use of the intuitive algorithm insufficient? After all, this ensures that the distance is always calculated using the smallest distance between the points in the two trajectories. To begin with, the performance of the intuitive algorithm is far from optimal. The number of distance calculations needed to obtain the similarity measure is $n(traj_a)n(traj_b)$, i.e., the complexity of the algorithm can be classified as $O(n^2)$. Secondly, the usability and reliability of the algorithm are questionable, as is explained after

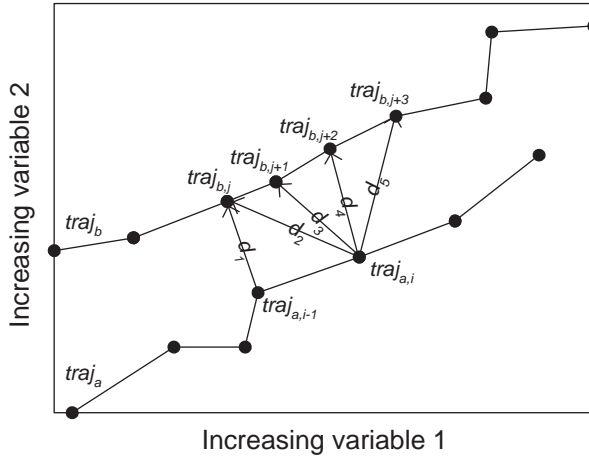


Fig. 25. Illustration of the algorithm when all dimensions are increasing.

the presentation of an improvement for the performance issue.

The performance increase of the presented algorithm is based on optimization leading to a drastic decrease in the number of necessary distance calculations when applied to trajectories containing an increasing dimension. The idea of the algorithm is explained using Figure 25. This Figure presents a special case, where the two trajectories are increasing in both dimensions, which makes it easier to understand the idea. The increasing property is utilized to limit the number of distance calculations needed to define the distance between the trajectories. In order to find the point in $traj_b$ closest to the point $traj_{a,i}$ it is enough to start the search from the point closest to $traj_{a,i-1}$ (found in the previous iteration of the algorithm, marked $traj_{b,j}$ in the figure). Because the trajectories are increasing, all the points observed before the j :th point of $traj_b$ must be farther from $traj_{a,i}$ than the point $traj_{b,j}$ and can therefore be ignored. The distance is then calculated from $traj_{a,i}$ to the consequent points in $traj_b$ starting from the point $traj_{b,j}$, until the distance between the points in the two trajectories begins to increase. After this, it is certain that the distances to the other points in $traj_b$ are greater than the minimum distance that was found and can be ignored, for example, in Figure 25 $d_2 > d_3 > d_4 < d_5$; hence the search can be stopped after calculating d_5 . The algorithm therefore makes the search for the pairs with the shortest distance independent of the number of points in $traj_b$ and dependent only on the neighborhood of the points following the point $traj_{b,j}$.

The previous explanation described the first version of the algorithm that could be applied only to a scenario in which all the dimensions of the trajectories are increasing. Shortly after this, it was noticed that the algorithm can be extended to any scenario in which only one of the dimensions of the trajectories increases. Figure 26 visualizes the operation of the algorithm in this more general setting, using a two-dimensional case where the values of the trajectories are increasing on the horizontal axis. The aim is again to find the bounding points in $traj_b$, to ensure that the closest point in distance lies between them (or is one of them). The search for the closest point for the point $traj_{a,i}$ is

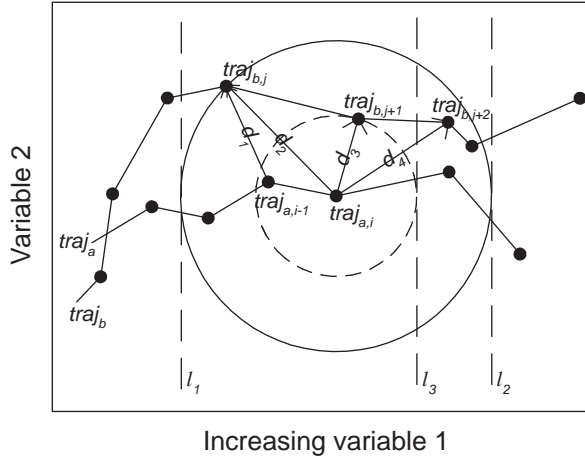


Fig. 26. Operation of the algorithm when only one of the dimensions is increasing.

started again by calculating the distance to the point in tra_j_b closest to the point $tra_j_{a,i-1}$. In Figure 26 this distance is marked as d_2 . Now, it is certain that the point closest to $tra_j_{a,i}$ must be within or on a circle with a radius of d_2 . The search is then continued by traversing the points in tra_j_b backward, starting from $tra_j_{b,j}$, until the point is reached that is farther away on the increasing dimension than the radius. In Figure 26, this limit is denoted with a vertical line marked with l_1 . Now, it is certain that the points behind this line are farther away than the closest point discovered and can be ignored. The same procedure is repeated onward from the point $tra_j_{b,j}$. During the search for the closest pair of points, the search radius is further decreased by updating it to be the distance from the point found so far to be closest to tra_j_b . In Figure 26, this happens after calculating d_3 ($d_3 < d_2$), and the search radius is updated. After the update, it is necessary to traverse the points only until the line marked with l_3 is crossed. The procedure is repeated for all the points in tra_j_a . Again, the algorithm is not dependent on the number of points in the other trajectory, but only on its local behavior. The pseudo code for both of these algorithms is presented in Algorithm 2. The notation $tra_j_{a,i,k}$ is used to denote the value of the k :th-dimension of the i :th-point of the trajectory a , where k is the index of the increasing dimension.

Let us continue by examining the usability of the algorithm compared to the intuitive algorithm. First of all the distance calculated using the intuitive algorithm does not fulfill three of the four requirements set for a *metric space* (see the requirements in Table 7). The first condition is more dependent on the point-to-point distance measure and holds whenever the L_p norm is applied. Trajectories tra_j_c , tra_j_d and tra_j_e in the Figure 24 (on page 82) demonstrate an example of three trajectories violating the latter three requirements when using the intuitive algorithm. The second condition does not hold because all the points in tra_j_c overlap with a point in tra_j_e and hence $d(tra_j_c, tra_j_e) = 0$ and $tra_j_c \neq tra_j_e$. Furthermore, the algorithm is not invariant to the order in which the trajectories are given to it (violation of the third condition). In the figure, this means that $d(tra_j_c, tra_j_e) \neq$

Algorithm 2: The algorithm developed for calculating the similarity between two trajectories.

input : trajectories tra_j_a and tra_j_b of size $n(tra_j_a)$ and $n(tra_j_b)$, index of the increasing dimension k

output: the distance between the trajectories, *trajectorydistance*

set *trajectorydistance* to 0;
 set *smallestdistance* to ∞ ;
 set *pointdistance* to ∞ ;
 set *index* to 1;
 set *scroll* to *true*;

for $i \leftarrow 1$ to $n(tra_j_a)$ **do**

for $j \leftarrow index$ to $n(tra_j_b)$ **do**

while ($tra_{j_b,j,k} > (tra_{j_a,i,k} - pointdistance)$) and ($scroll == true$) and ($j > 1$) **do**

| subtract 1 from j ;

end

set *scroll* to *false*;

if ($tra_{j_a,i,k} - pointdistance < d(tra_{j_a,i}, tra_{j_b,j})$), $< (tra_{j_a,i,k} + pointdistance)$ **then**

| set *pointdistance* to $d(tra_{j_a,i}, tra_{j_b,j})$;

| **if** *pointdistance* $< smallestdistance$ **then**

| set *smallestdistance* to *pointdistance*;

| set *index* to j ;

| **end**

| **end**

else if $tra_{j_b,j,k} > (tra_{j_a,i,k} + pointdistance)$ **then**

| break;

end

end

if $i < (n(tra_j_a) - 1)$ **then**

| set *pointdistance* to $d(tra_{j_a,i+1}, tra_{j_b,index})$;

end

increment *trajectorydistance* by *smallestdistance*;

set *smallestdistance* to ∞ ;

set *scroll* to *true*;

end

set *trajectorydistance* to (*trajectorydistance*/ $n(tra_j_a)$);

return *trajectorydistance*;

$d(tra_j_e, tra_j_c)$. Finally, the fourth condition (triangle inequality) does not hold, because $d(tra_j_c, tra_j_d) > d(tra_j_c, tra_j_e) + d(tra_j_e, tra_j_d)$.

From a practical point of view, violation of the third requirement is the most serious flaw of the intuitive algorithm. It not only makes the algorithm unusable, but may also lead to the practitioner using very unreliable results of analysis. On the other hand, the algorithm could be made nearly compliant with metric spaces, and the problem fixed, by a slight modification that is now applied to the algorithm developed. Instead of applying the

Table 7. Requirements of a metric space.

No.	Requirement
(1)	$d(traj_a, traj_b) \geq 0, \forall traj_a, traj_b \in T$
(2)	$d(traj_a, traj_b) = 0 \Rightarrow traj_a = traj_b$
(3)	$d(traj_a, traj_b) = d(traj_b, traj_a)$
(4)	$d(traj_a, traj_c) \leq d(traj_a, traj_b) + d(traj_b, traj_c), \forall traj_a, traj_b, traj_c \in T$

algorithm only to calculate $d(traj_a, traj_b)$, two distances, $d_1 = d(traj_a, traj_b)$ and $d_2 = d(traj_b, traj_a)$, are calculated. After this, the ultimate distance between the trajectories is defined to be $\max(d_1, d_2)$. The procedure makes the algorithm fully compliant with the first three properties of a metric space and *nearly* compliant with the triangle inequality. Although in most cases the algorithm also fulfills the triangle inequality, under certain circumstances the procedure is only nearly compliant with it.

The procedure can be further optimized by storing the point-to-point distances calculated at the first pass (d_1) in memory and, on the second pass, by calculating (d_2) only the distances not calculated during the first pass. For the sake of simplicity, this optimization is not presented in the pseudo code in Algorithm 2. It is emphasized that the performance studies presented in the next section are applied to this two-pass version of the developed algorithm and to the one-pass version of the intuitive algorithm.

4.4 About the complexity of the algorithm

This section demonstrates that the complexity of this algorithm is linear in the case in which all measurement dimensions are increasing. The property is shown by studying the number of point-to-point comparisons required by the algorithm.

Let z_i be the number of point-to-point comparisons needed for finding the closest point in $traj_b$ to a point $traj_{a_i}$ and let k_i be the index of the closest point to $traj_{a_i}$ in $traj_b$. The total number of calculations needed for finding all the closest matching pairs in the two trajectories is $\sum_{i=1}^{n_a} z_i$, where n_a is the number of points in $traj_a$. The number of calculations, z_i , can also be written as $z_i = k_i - k_{i-1} + 2$, assuming that the comparisons are always started from the point $(k_{i-1} - 1)$ and ended at the point $(k_i + 1)$. This means the assumption is made that at least three comparisons are needed for every point in $traj_a$. In some cases this number can be smaller (one or two), but it is enough to show that the algorithm is linear under this more demanding condition. If so, then the same result holds for cases when the number is one or two.

Figure 27 gives a concrete example of these notations. The point-to-point comparisons needed for finding the closest matching point to $traj_{a_1}$ are drawn using the dashed lines. Three comparisons are needed and therefore $z_1 = 3$. The closest point to the first point of $traj_a$ is the second point of $traj_b$ and hence $k_1 = 2$. Now, the task is to determine the complexity of this sum with respect to the number of points in the two trajectories. In order to do so, the sum is written as

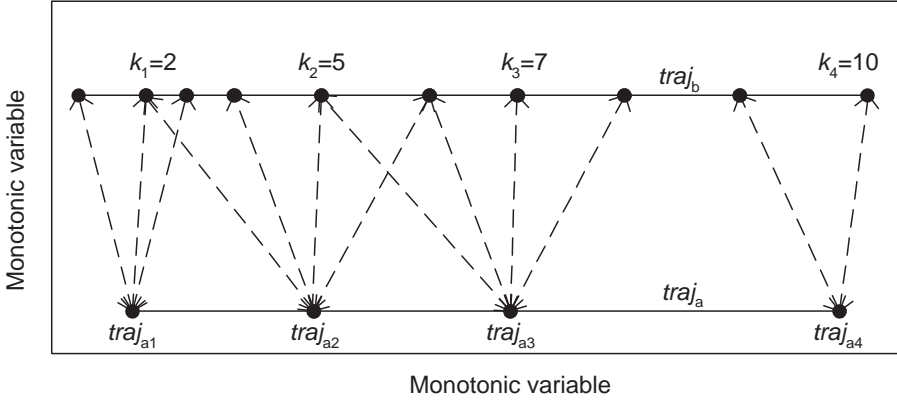


Fig. 27. Illustrating the calculations related to the complexity of the algorithm. Dashed line = a distance calculation between a pair of points, k_i = index of the closest point in $traj_b$.

$$\begin{aligned}
 \sum_{i=1}^{n_a} z_i &= & (3) \\
 \sum_{i=2}^{n_a} z_i + z_1 &= \\
 \sum_{i=2}^{n_a} (k_i - k_{i-1} + 2) + k_1 - 1 &= \\
 2(n_a - 1) + \sum_{i=2}^{n_a} (k_i - k_{i-1}) + k_1 - 1 &= \\
 2(n_a - 1) + (k_{n_a} - k_1) + k_1 - 1 &= \\
 2n_a + k_{n_a} - 3, &
 \end{aligned}$$

which is clearly linear and hence the complexity of the algorithm in this case is in the class $O(n)$. The calculation of the accurate complexity of the version of the algorithm that is also applicable in situations when all of the dimensions are not increasing is left for future work.

4.5 Empirical estimation of efficiency

The performance of the algorithms developed here was evaluated using two data sets. The first experiment 4.5.1 shows the efficiency and usability of the algorithm in a real world application, and the second one 4.5.2 demonstrates the efficiency under more general circumstances, using synthetically generated data. The results are finally summarized in

Subsection 4.5.3. The performance of the algorithms was evaluated in an implementation-independent way. This was accomplished by measuring the performance with the number of point-to-point distance calculations needed to obtain the value of the similarity measure.

4.5.1 Data from a walking beam furnace

In this part of the comparison the performance of the algorithms was evaluated using trajectory data formed from the steel slab data set collected from the walking beam furnace described in Chapter 3.5. In addition to the static measurements made from the steel slab, the data set contains the location of the slabs in the furnace and 29 other temporal measurements stored at approximately one-minute intervals as the slabs pass through the furnace. An important factor affecting the post-roughing mill temperature is the spatio-temporal trajectory of the slab, which is formed by the elapsed distance of the slab from the entrance of the furnace and the elapsed heating time of the slab. In addition to this, the trajectory can be augmented with extra (temporal) dimensions, such as the roof and floor temperatures measured from the locations the slab has been in.

The data set used in this study consists of 5,532 trajectories recorded from steel slabs between the years 1999 and 2003. Each trajectory contains on average 81 values, and the heating times vary from 7,283 to 53,390 seconds (from ≈ 2 hrs to ≈ 14 hrs). These values are observed at approximately one-minute intervals, as stated earlier, and it is not unusual that occasional values are missing. These conditions were the impetus for the development of the algorithm presented. The trajectories need to be scanned for finding the closest matching pair of points in them (because matching pairs do not occur in parallel time). Furthermore, the varying lengths of trajectories make the intuitive algorithm unusable, mostly because of the property $d(traj_c, traj_e) \neq d(traj_e, traj_c)$ when using it.

Three different scenarios were used to evaluate the performance of the algorithm developed in this study compared to the intuitive algorithm. The data set used in the first scenario contains the type of trajectories presented in Figure 25, where all the measurement dimensions are increasing. The observed dimensions are elapsed time and elapsed distance, and both are thus increasing. The second scenario reflects the properties of Figure 26, where the first dimension is elapsed time (increasing measurement) and the second dimension is the temperature of the roof measured above the slab at each location the slab has been in (measurement varying according to time). The third scenario presents the performance of the algorithm when the first dimension is increasing and the number of varying dimensions increases to four (roof temperature, floor temperature, liquid gas flow, and coke gas flow, with values observed with respect to time).

Data from 100 trajectories were used as a test set to query the most similar trajectories from the training set (collection of stored trajectories). The training set was limited to trajectories from slabs that were not in the furnace at the same time as the query trajectory. This was done to make the experiment a little more realistic, since in a real usage scenario the data from an on-going measurement would be compared to the trajectories in the history sink (finished observations). Therefore, the training set was a bit different for most slabs, with the size varying between 5,457 and 5,508 slabs. After calculating the

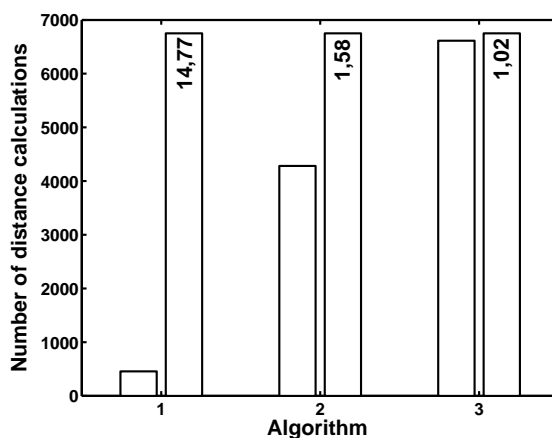


Fig. 28. A comparison of the performance of the algorithm using measurement data from a steel mill. 1 = Data with two dimensions, both increasing. 2 = Data with two dimensions, the first increasing. 3 = Data with five dimensions, the first increasing.

similarity between the query slab and the slabs in the training set, the average number of calculations needed to compare the query slab to a slab in the training set was stored. Finally, the reported performance metric for each scenario is the average number of these averages.

Figure 28 presents a bar chart where the results of the comparison are grouped using the different scenarios as labels. Each scenario contains two bars. The one on the left displays the number of calculations needed by the developed algorithm, and the one on the right shows the corresponding number for the intuitive algorithm. In addition to this, the bar on the right contains a number showing how many times taller it is than the left bar. It should also be noted that if the intuitive algorithm were used to apply the two-pass strategy (making it more compatible with the requirements of metric space), the number of calculations needed would double.

The algorithm developed was expected to outperform the intuitive algorithm under the first scenario where all, albeit only two, dimensions are increasing. This was indeed true, as shown by a comparison of the first two bars of Figure 28. It took 457 calculations on average to define the similarity between the two trajectories using the algorithm developed in this study and 6,750 calculations using the intuitive algorithm. When the other dimension was allowed to vary freely over time (bars labeled 2), it took 4,281 calculations to define the similarity using the algorithm developed in this study and 6,750 using the intuitive algorithm, which is still a significant increase in performance. Finally, when the number of freely varying dimensions was increased to four (bars labeled 3), almost the same number of calculations was needed (6,614 vs. 6,750). This is because the larger number of dimensions increases the radius in which the algorithm searches for the closest point.

4.5.2 Synthetic data set

The trajectories used in the previous section contained an average of 81 measurement points. To be able to see how the algorithm performs under varying circumstances, the performance was studied using synthetically generated data formed by subsets of data with trajectories of different lengths.

The generated data consists of ten subsets of trajectories, each consisting of 30 trajectories of a certain length. The measurement points of each trajectory contain two increasing dimensions (location and elapsed time) and four varying dimensions. Data from a trajectory were generated until the length limit of the trajectories in the respective subset was reached (10, 25, 50, 100, 200, 250, 500, 1,000, 2,000 or 4,000 locations). During each iteration of the data set generation, the location was increased with a probability of 0.80 and a uniformly generated random value between 0 and 2,000. The elapsed time was increased with a uniformly generated value between 1 and 60, and the varying dimensions were assigned uniformly generated double values between 0 and 10.

Three scenarios similar to those of the previous subsection were studied. The first was applied to a subset of data consisting of two increasing dimensions. The second was applied to a subset of data containing elapsed time and one varying dimension. Finally, the third one was applied to the elapsed time and four varying dimensions. Each scenario was applied to data from the ten trajectory subsets. Furthermore, the algorithms were tested in each subset with the leave-one-out method, which means that each trajectory in turn was used as prediction data and the rest as the training data set. The number of calculations needed for acquiring the similarity measure between two trajectories within each data set was recorded for each pair of trajectories in the respective data set, and the average of this value was used as the measure of performance.

The results of the comparison are presented in Figure 29. These results illustrate how the lengths of the trajectories affect the number of calculations needed. The horizontal axis shows the number of points in the trajectories in each data set, and the vertical axis shows the logarithm of the average number of the point-to-point distance calculations needed for calculating the similarity between two trajectories. The curve formed by the first scenario is marked with an asterisk, the curve of the second with squares and the curve of the third with triangles. In comparison to this, the performance of the intuitive algorithm is plotted using circles. The algorithm developed in this study again clearly outperforms the intuitive algorithm in a situation where both dimensions are increasing (the curve marked with asterisks vs. the curve marked with circles) as was expected, because the complexity of the algorithm is linear in this case. The performance under the second scenario is also clearly better with all trajectory distances. The performance under the third scenario is comparable to the intuitive algorithm up until trajectory lengths of less than 100 measurement points, but after that, a slight boost of performance is evident. As an example of the precise number of calculations, when the intuitive algorithm is applied to the longest trajectories used in this study (4,000 locations), 16 million calculations are needed, whereas the algorithms developed in this study require 23,898, 1,619,000 and 6,875,000 calculations (relative to the data set).

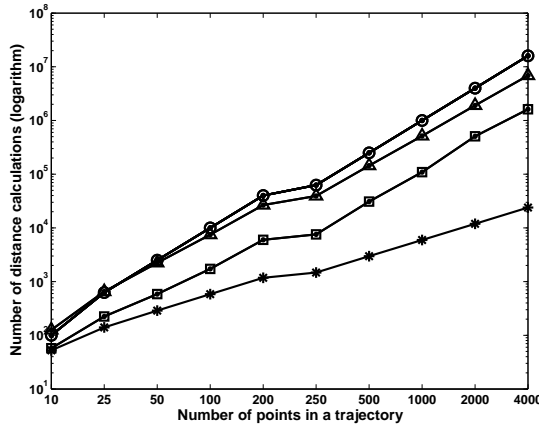


Fig. 29. A comparison of the performance of the algorithm using generated data. Curve marked with *asterisk* = two increasing dimensions, *square* = one increasing and one varying dimension, *triangle* = one increasing and four varying dimensions, *circle* = intuitive algorithm.

4.5.3 Summary of the performance results

Based on the results of these two performance tests, the conclusion can be drawn that the algorithm developed in this study is computationally most usable in situations in which all of the dimensions are increasing, such as spatio-temporal trajectories, where the spatial location can be measured as a distance from the starting point. It also performs well in situations where the data contains only a few varying dimensions. As the number of varying dimensions grows, the performance becomes more comparable with the intuitive algorithm. However, even if the number of freely-varying dimensions were increased to infinity, the number of calculations needed would never exceed the limit, which is twice the number of calculations needed using the intuitive algorithm. This is because the search radius of the algorithm cannot expand beyond the first and last points in the candidate trajectory (which is the search radius of the intuitive algorithm), and the distance is calculated from the query trajectory to the candidate trajectory by using the two-pass strategy. Although the computational performance would be similar to that of the intuitive algorithm, the usability of the algorithm developed in this study is superior to that of the intuitive algorithm because of the important properties it fulfills (Section 4.3).

4.6 Related work

Many recent developments have been made in the field of trajectory similarity measurement. The research can be divided into two distinct branches, one of which focuses on advancing indexing methods and the other on developing methods for measuring similarity between two trajectories, as in this work. Moreover, it is not unusual that work from

both branches is reported in the same paper. Some of the recent studies that present the current status of the field are reported here.

Similarity measures and techniques for measuring similarity between two trajectories have been developed for various purposes. kNN-type queries that return the x most similar trajectories from a set of T trajectories have been reported, for example, in Yanagisawa *et al.* (2003). Methods used for searching trajectories similar in shape (after shifting and transforming operations) are called "minimum distance" search methods. A "bounded similarity" method is reported in Goldin *et al.* (2004). It defines the trajectories $traj_s$ and $traj_q$ to be boundedly similar if a trajectory $traj_{s'}$ can be formed from $traj_s$, which is within the distance ε from $traj_q$. "Time warping" similarity measures allow trajectories to accelerate and decelerate along the time dimension. The idea behind the method is to extend the trajectories with repeating elements and then to calculate the Euclidean distance for these transformed trajectories (Berndt & Clifford 1996). Techniques that develop similarity measures based on the longest common subsequences found in a pair of trajectories are presented in Bollobás *et al.* (2001) and Vlachos *et al.* (2003), among others. In this approach the distance between two trajectories is based on the length of subsequences having the most measurement points approximately similar. Bollobás *et al.* (2001) also present complexity estimates of the many algorithms presented in the work. The version that returns exact similarity can be computed in $O(n^3)$ -time, an approximate algorithm in $O(n^2)$ -time and an approximate algorithm based on randomization in $O(n)$. The authors also note that even though the randomized version of the technique does not give exact values of similarity, the approximations are close to the correct one and the performance is far better than with the exact version. Finally, another approach to the measurement of trajectory similarity is to first apply transformations, reducing the number of data points needed for presenting the trajectory, and then to apply a similarity search to the compressed presentation. One popular approach is to present the trajectory as a limited set of its Fourier co-efficients Rafiei & Mendelzon (2000).

Methods utilizing different approaches for similarity definition can be useful in application areas where numeric data cannot be gathered. An approach developed for processing genomic data sequences is reported in Muthukrishnan & Sahinalp (2002). It uses the number of edit operations needed to transform one genome into another as the similarity measure. A method based on presenting time sequences as a number of intervals instead of trajectories of measurement points is presented in Yi & Roh (2004). The method is especially useful in situations where the exact time of measurement has not been determined.

When searching for the closest matching trajectory in a collection of trajectories, it may be computationally expensive to iterate through all the trajectories stored. Therefore, indexing schemes that guarantee that the closest matching trajectory is in a smaller subset of trajectories with a certain probability have been developed. A fast time sequence indexing for arbitrary L_p norms is presented in Yi & Faloutsos (2000). Using this method, it is possible to build a one-dimensional index that supports similarity queries using any of the L_p point-to-point distance measures. The work is based on first averaging the trajectory within s segments of equal length and then building the index based on these averages. Indexing methods for minimum distance queries are presented in Lee *et al.* (2004), including an indexing scheme that does not require vertical shifting of trajectories. An efficient indexing method for situations where similarity is measured using the

time warping measure is presented in Kim *et al.* (2004).

Algorithms developing the efficiency of the fundamental algorithm (referred to here as the "intuitive algorithm", see the pseudo code in Algorithm 1 for details) for calculating the similarity between two trajectories were not found in the literature. However, it should be noted in the literature that the complexity of the intuitive algorithm is $O(n^2)$, and that this is a major computational issue when defining the exact similarity between trajectories Yanagisawa *et al.* (2003), Meretnia & de By (2002). The results presented in this work bring the computational efficiency of the fundamental algorithm to a completely new level and are also suitable for boosting many of the above-mentioned algorithms, as long as the algorithms iterate through an increasing measurement dimension.

4.7 Discussion

This chapter started by introducing the logic behind novelty and similarity detection in general. After that, an improved algorithm for calculating the level of similarity between trajectory data was presented. Both the computational efficiency and the usability of the algorithm were evaluated in comparison to the currently used "intuitive algorithm". It was determined that, depending on the usage scenario, the computational efficiency of the algorithm developed in this study varies from a significant improvement up to twice that of the intuitive version, depending on the kind of data it is applied to. The usability of the algorithm developed in this study was discovered to be superior to the currently used version.

The similarity measurement technology presented can be used in the history sink of Smart Archive for defining the level of novelty of observations. For example, the component enables the accumulation of representative data into the history sink without storing redundant data. This information can be taken into use in e.g. forming confidence estimates for classification results - when an observation having closely matching observations in the training data set of the model is classified, more confidence can be given for the classification accuracy and vice versa.

The core technology of the history sink is based on the algorithm developed in this study for similarity detection of trajectory data. To justify the use of the algorithm presented here, it was observed that often, in practice, trajectories do not contain naturally matching pairs of points. Therefore the trajectories need to be scanned for identifying the closest matching points in them. Furthermore, trajectories often contain a measurement dimension that is increasing, as for example temporal trajectories. The algorithm developed in this study utilized this property and reduced the number of calculations needed for calculating the similarity measure by orders of magnitude at best and, in the worst case, by two times when compared to the current alternative, the intuitive algorithm. In the empirical performance experimentation the algorithm developed here never performed more poorly than the intuitive version. The algorithm is especially suitable for the history component because it can process trajectory data of any kind, especially the kind of trajectories observed in many real world applications, that is, applications that produce trajectories with varying observation intervals or trajectories of varying lengths. The algorithm can, of course, also be applied in cases where the data consist of matching pairs

of points or only individual measurement points.

In addition to this, this work analyzed the usability of the similarity measurement algorithms. It was discovered that the logic of the intuitive algorithm incorporates serious flaws and that it does not fulfill three out of four conditions set for metric spaces. Why should the distance metric fulfill the conditions of a metric space? That is a good question, one for which it is difficult to find an answer in the literature. Part of the reason might be that the distance should be unambiguous. That is, the distance measured using the metric should not differ significantly from the distance observed by the human eye. For example, it is natural to think that the distances $d(a,b)$ and $d(b,a)$ should be equal. This leads to a question resembling the question "how similar must an item be to be similar? " That is, how well must a distance metric fulfill the conditions set by the definition of metric space to be of practical use? It is difficult to answer, but it is safe to say that as well as possible. The algorithm presented in this thesis fulfills the conditions much better than the intuitive version of the algorithm and corrects the flaws in the usability of the intuitive algorithm. The properties of the intuitive algorithm are so far from the conditions that it was easy to observe how strongly it is also reflected in the usability of the algorithm. On the other hand, the algorithm developed in this study almost fulfills the conditions. Therefore, it is up to the practitioner to decide if the gap is small enough for his / hers application. But it is certainly safe to say that the algorithm developed here is more usable than the intuitive algorithm currently used.

Future research topics include developing and adapting indexing schemes for the history component and applying it in the applications built on top of Smart Archive. The currently developed algorithm can be applied for finding the most similar data from the history data, but limiting the set of search candidates using an indexing technique would result in time savings.

5 Conclusions

5.1 Discussion

Data mining and data mining application development are challenging research areas. The methods can be applied to benefit practically all kinds of applications creating measurable information. Part of the challenge is due to this diversity - diversity not only in the field of applications, but also within the discipline itself. Data mining incorporates expertise from many different fields and the ability to create sophisticated DM applications, especially as stand-alone installations, requires advanced knowledge of many of these fields. The perfect data miner should have expertise in statistics and mathematics (data transformations), information processing (data storage and processing), software engineering (implementing and designing applications) and in work sciences (management and distribution of the DM tasks). Furthermore, in order to be able to create a DM application one should have vision and expertise in organizing and allocating these resources and skills into a concept that can be used for moving from data to knowledge.

This work tried to respond this challenge by presenting a top-down approach for creating and implementing DM applications. It discussed and proposed solutions for two fundamental issues of data mining, the creation of a working DM solution from a set of measurement data and the implementation of it as a stand-alone application. Using the presented approach, or concept, it is possible to create a fully-working application from the stage when measurement data have been observed. The concept is especially useful for creating applications that are working on continuously observed measurements.

This thesis presented contribution in all the subjects it treated. To summarize this shortly, the main contribution of the first subject (presented in Chapter 2) was the results of the study focusing on the phase in which a working solution for a given set of measurement data is sought. An efficient DM process suitable for this purpose was introduced and the efficiency was demonstrated with case studies. The main contribution of the second subject (presented in Chapter 3) was the results of a study presenting a methodology for implementing the DM solution as an independent application. The thesis reported the developed application framework (Smart Archive) and a case study in which a DM application was created using it. The main contribution of the third subject (presented in Chapter 4) was an efficient algorithm developed for calculating the level of similarity between two trajectories. The algorithm was developed with the history component of

Smart Archive in mind, but it can be used with any application. A more detailed presentation of the contribution and a discussion on the subjects comprising this work can be found from the respective chapters of this thesis.

However, the most important contribution of this thesis is not considered to be within any of the particular subjects it advances, although those contributions are important, but the work as a whole. This thesis has been an effort to create a top-down approach for building DM applications. Following the guidelines given in this work, one should be able to get insight into building deployable, continuously operating DM applications. Adopting the approach to DM application development should first of all ensure that the applier can understand the different phases of development along the path from the point when data have been acquired to the point at which a DM application is ready for deployment. It is argued that this concept has not been previously presented anywhere else with the level of detail presented in this work. Secondly, based on the results presented in this thesis the applier should understand how this path can be made more efficient. It is argued that employing the research results of this work, the applier can make the application development process significantly more efficient than using for example an arbitrary DM process and building the application from scratch. Finally, based on the comparisons and case studies presented, the applier should understand what some of the pitfalls of creating DM applications are and avoid stepping into them. For example, one should try to employ some form of methodicalness when seeking for a solution that best fits the DM problem at hand. After reading this thesis one can get a deeper understanding of what the important qualities affecting the selection of the method for managing the DM process are. Another important lesson that can be learned is the use of an application framework when implementing the solution as an independent application. By studying the application framework proposed here, one should be able to get ideas for the application implementation phase. Finally, the presented approach should give novel ideas on the complete process of moving from a set of measurement data to a finished DM application. Hopefully the results of this thesis can strengthen the vision of the applier on how to manage this whole ensemble and give more confidence in creating new applications.

Both of the applications in this thesis were from the manufacturing industry, the first one concerned joining metal objects together (resistance spot welding) and the second one producing the metal objects (hot rolling of steel). The applications were not chosen purposefully, but it was determined that the approach developed in this study suits the qualities of these applications very well. Both of them required the application of a rather sophisticated knowledge extraction process and both operate in an environment in which new observations are produced continuously. A lot of work was spent on studying both applications before the top-down approach for creating DM applications was formed. Therefore, it is interesting to note that the development of the approach was not conducted in a top-down manner, but rather in a bottom-up manner, where the applications gave inspiration to the formation of the methodology. In this sense the whole work has been blessed - if another set of applications had been studied, an approach this good might not have emerged. Nevertheless, the approach is application-independent and developed for a class of applications having the most demanding qualities; therefore there is no reason why it would not be applicable in most of the applications in the data mining field.

In this thesis the DM solution for the resistance spot welding quality control project was acquired using the DM process developed here and the application predicting the

steel slab temperatures was implemented using the application framework (SA) also developed in this study. Neither of these were completely developed using the presented concept. It would have been interesting to develop a complete application using the concept developed here. There is no reason, other than scheduling, why the concept could not have been applied to the creation of both of the applications from the beginning. The steel temperature project had proceeded to the implementation phase when the DM process was created. In the spot welding project it was not the task of Intelligent Systems Group to implement the solution and in addition to this, Smart Archive was available only after the project was completed. It is also questionable if it would have been possible to present the implementation of two applications from the beginning within one thesis without changing the perspective of the thesis significantly. In this work the reporting of the results of these two applications is closely tied to the development of the more abstract methods (the DM process and the application framework). Finally, applying the concept to the creation of a solution for one application area and on the implementation stage of another showed the applicability of the results for two separate applications.

In conclusion, so far there have existed high-level descriptions of what the DM process is and what is included in it. On the one hand, a lot of descriptions and developments are available for DM algorithms on the implementation level, but these descriptions may be quite separate from the bigger picture. It may have been hard to determine a good approach for creating a complete DM application on the basis of the existing knowledge. This work had the ambitious goal of presenting a concept that ties together the higher-level view of the DM process and the implementation-level view. The use of the case studies demonstrated that the developed concept can be efficiently used for this task.

5.2 Summary

The first chapter of this thesis introduced data mining (DM) as a field of research in general and presented some interesting applications that are designed for processing continuously measured observations. The presentation of the real contribution of this thesis started in the second chapter, which presented the semi-open DM process developed in this study for creating a DM solution. The treatment of the topic started by introducing the reference architecture existing behind most DM solutions. After that, a formalism for presenting the DM chain was developed in order to study the interactions between the functions that make up a DM process. The closed and open approaches for implementing the DM chain were presented, which led to the question of whether higher-order categorizations of functions could be used in managing the interactions. The natural answer to this question was to categorize the functions according to the reference architecture - this categorization was then named the semi-open DM process. The work contrasted the semi-open process to the closed and open ones and deduced that the open and semi-open approaches are related to each other more than the closed one, but the DM process becomes better managed with the semi-open one. After that a case study of a DM project in which solutions for the quality control of resistance spot welding joints were developed was presented. The case study presented a detailed comparison of the pre-processing phase, where a software system built for the application was used for pre-processing the

welding signals using the closed- and semi-open approaches. The results of the feature extraction and modeling phase, which were acquired using the presented process, were also presented. It was concluded that the approach developed in the study is well-suited for projects where the solution needs to be developed by evaluating a comprehensive set of methods. Furthermore, the approach supports the distribution of the DM chain especially well, which enables e.g. easier distribution of responsibilities among a group of experts.

The topic of the third chapter continued logically from where the second chapter had ended by presenting an architecture and framework, called Smart Archive, designed for implementing the solution that was discovered. The architecture, motivated by a short functional requirements analysis, and operation principle of Smart Archive were reported on. The architecture contained the components found from the reference architecture and a component for storing and processing history information from the observed phenomena. Also, the principles of the data structure needed for implementing SA were elaborated. After that, an application that processes and analyzes continuously observed measurements from steel slabs heated in a walking beam furnace and predicts their exit temperatures was presented. The application used a feedforward-type neural network for predicting the temperatures. The selection of the training method of the network, the structure and parameters were explained and after that the results were analyzed. The prediction accuracy of the model was considered very good for most of the slabs and overall good enough for starting implementation on a production line. After that, the implementation of the solution using SA was contrasted to a previous implementation, where the software had been implemented from scratch. The framework-based implementation was determined to have obvious advantages over the tedious way of implementing the solution from scratch.

Chapter 4 focused on presenting an improved algorithm for calculating similarities between two measurement trajectories. The subject was motivated by the requirements of the history component of SA, which needs to detect the level of the similarity of observations for data storage and for retrieving the most similar trajectories from the storage. The algorithm presented was suitable for data that contains at least one increasing measurement dimension. The performance of the algorithm proved to be linear in the case where all measurement dimensions are increasing. The version of the algorithm also capable of handling varying measurement dimensions was evaluated empirically and the results indicated that the computational performance was generally better than that of the version currently used and, at its worse, was the same as with the current version. The usability of the algorithm was superior to the alternative, mostly because the similarity calculated with the algorithm previously used did not fulfill the requirements of a metric space as well as the version developed in this work.

References

- Ahmed, K., El-Makky, N. & Taha, Y. (1998) Effective data mining: a data warehouse-backed architecture. In: 1998 Conference of the Centre for Advanced Studies on Collaborative Research. Toronto, Ontario, Canada, p. 11 pp.
- Benson, M. & Carrasco, R. (2001) Application of a recurrent neural network to space diversity in sdma and cdma mobile communication systems. *Neural Computing & Applications* 10:136–147.
- Berndt, D. J. & Clifford, J. (1996) Finding patterns in time series: a dynamic programming approach. *Advances in knowledge discovery and data mining* pp. 229–248.
- Berzal, F., Blanco, I., Cubero, J.-C. & Marin, N. (2002) Component-based data mining frameworks. *Commun. ACM* 45(12):97–100.
- Bishop, C. (1995) *Neural Networks for Pattern Recognition*. Oxford University Press.
- Bollobás, B., Das, G., Gunopulos, D. & Mannila, H. (2001) Time-series similarity problems and well-separated geometric sets. *Nordic J. of Computing* 8(4):409–423.
- Brachman, R. J. & Anand, T. (1996) The process of knowledge discovery in databases, American Association for Artificial Intelligence, Menlo Park, CA, USA, pp. 37–57.
- Britannica, E. (2005) The online version of the encyclopaedia britannica, <http://www.britannica.com/>, referenced 1.4.2006.
- Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P. & Stal, M. (1996) *Pattern-Oriented Software Architecture, Volume 1: A System of Patterns*. John Wiley & Sons.
- Catarina, S. & Bernardete, R. (2003) Navigating mobile robots with a modular neural architecture. *Neural Computing & Applications* 12:200–211.
- Chan, M., Leong, H. & Si, A. (2000) Incremental update to aggregated information for data warehouses over internet. In: 3rd ACM International Workshop on Data Warehousing and OLAP (DOLAP '00). McLean, Virginia, United States, pp. 57–64.
- Chandramathi, S. & Shanmugavel, S. (2003) Estimation of cell loss probability for self-similar traffic in atm networks—A fuzzy approach. *Applied Soft Computing* 3:71–83.
- Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T. & Wirth, C. S. R. (2000) *Crips-dm 1.0: Step-by-step data mining guide*. Tech. rep., NCR, SPSS, DaimlerChrysler.
- Choi, D. (2003) Enhancing the power of web search engines by means of fuzzy query. *Decision Support Systems* 35:31–44.
- Chrysostomou, C., Pitsillides, A., Rossides, L. & Sekercioglu, A. (2003) Fuzzy logic controlled red: congestion control in tcp/ip differentiated services networks. *Soft Computing - A Fusion of Foundations, Methodologies and Applications* 8:79–92.
- Darpa (2005) Darpa grand challenge 2005, available on-line at <http://www.grandchallenge.org/>, referenced 1.4.2006.
- Fechner, T., Neumerkel, D. & Keller, I. (1994) Adaptive neural network filter for steel rolling. *IEEE International Conference on Neural Networks* 6:3915–3920.
- Fernandez, J. (1998) An architectural style for object oriented real-time systems. In: *Fifth International Conference on Software Reuse, 1998. Proceedings*. Victoria, BC, pp. 280 – 289.

- Flanagan, J., Mäntyjärvi, J. & Himberg, J. (2002) Unsupervised clustering of symbol strings and context recognition. In: In Proceedings of the International IEEE Conference on Data Mining (ICDM). Maebashi, Japan.
- Fonlupt, C. (2001) Solving the ocean color problem using a genetic programming approach. *Applied Soft Computing* 1:63–72.
- Goldin, D. Q., Millstein, T. D. & Kutlu, A. (2004) Bounded similarity querying for time-series data. *Information and Computation* 194(2):203–241.
- Gong, J. & Yao, B. (2001) Neural network adaptive robust control of nonlinear systems in semi-strict feedback form. *Automatica* 37(8):1149–1160.
- Gorni, A. (1997) The application of neural networks in the modeling of plate rolling processes. *JOM-e* 49.
- Grzymala-Busse, J. (1998) LERS: A knowledge discovery system, Physica-Verlag, Heidelberg, Germany, pp. 562–565.
- Grzymala-Busse, J. W. & Ziarko, W. (2003) Data Mining: Opportunities and Challenges, Idea Group Inc., chap. Data Mining Based on Rough Sets, pp. 142–173.
- Haapalainen, E., Laurinen, P., Junno, H., Tuovinen, L. & Röning, J. (2005) Methods for classifying spot welding processes: A comparative study of performance. In: The 18th International Conference on Industrial & Engineering Applications of Artificial Intelligence & Expert Systems. Bari, Italy.
- Haapalainen, E., Laurinen, P., Junno, H., Tuovinen, L. & Röning, J. (2006) Feature selection for identification of spot welding processes. In: submitted to IEA-AIE 2006, notification of acceptance due 9.1.2006.
- Haapanen, R., Ek, A., E., M. B. & Finley, A. (2004) Delineation of forest/nonforest land use classes using nearest neighbor methods. *Remote Sensing of Environment* 89:265–271.
- Hand, D. J., Smyth, P. & Mannila, H. (2001) Principles of data mining. MIT Press, Cambridge, MA, USA.
- Hastie, T., Tibshirani, R. & Friedman, J. H. (2001) The Elements of Statistical Learning. Springer.
- Hayes-Roth, B., Pflieger, K., Lalanda, P., Morignot, P. & Balabanovic, M. (1995) A domain-specific software architecture for adaptive intelligent systems. *IEEE Transactions on Software Engineering* 21(4):288–301.
- Hornick, M. (2005) Java Specification Request 73: Java Data Mining (JDM). Oracle Corporation, 1st edn.
- Hotz, E., Grimmer, U., Heuser, W., Nakhaeizadeh, G. & Wieczorek, M. (2001) Revi-miner, a kdd-environment for deviation detection and analysis of warranty and goodwill cost statements in automotive industry. In: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining. San Francisco, California, pp. 432–437.
- Hsiung, P.-A., Lee, T.-Y., See, W.-B., Fu, J.-M. & Chen, S.-J. (2002) Vertaf: an object-oriented application framework for embedded real-time systems. In: Proceedings of the Fifth IEEE International Symposium on Object-Oriented Real-Time Distributed Computing, 2002. (ISORC 2002). Washington, DC, pp. 322 – 329.
- Jämsä-Jounela, S.-L. (2001) Current status and future trends in the automation of mineral and metal processing. *Control Engineering Practice* 9(9):1021–1035.
- Junno, H., Laurinen, P., Tuovinen, L., Haapalainen, E., Röning, J., Zettel, D., Sampaio, D., Link, N. & Peschl, M. (2004a) Resistance spot welding process identification and initialization based on self-organising maps. In: 1st International Conference on Informatics in Control, Automation and Robotics (ICINCO 2004), August 25–28, Setubal, Portugal. Setubal Portugal, vol. 1, pp. 296–299.
- Junno, H., Laurinen, P., Tuovinen, L. & Röning, J. (2004b) Studying the quality of resistance spot welding joints using self-organising maps. In: Fourth International ICSC Symposium on Engineering of Intelligent Systems (EIS 2004). Madeira, Portugal.
- Junno, H., Laurinen, P., Haapalainen, E., Tuovinen, L. & Röning, J. (2005) Resistance spot welding process identification using an extended knn method. In: IEEE Int. Symp. on Industrial Electronics. Dubrovnik, Croatia.
- Kim, S.-W., Park, S. & Chu, W. W. (2004) Efficient processing of similarity search under time warping in sequence databases: an index-based approach. *Information Systems* 29(5):405–420.

- Kim, Y., Moon, K., Kang, B., Han, C. & Chang, K. (1998) Application of neural network to the supervisory control of a reheating furnace in the steel industry. *Control Engineering Practice* 6(8):1009–1014.
- Kohonen, T. (2000) *Self-Organizing Maps*. Springer.
- Kudyba, S. (ed.) (2004) *Managing Data Mining: Advice from Experts*. Idea Group Inc.
- Kwoka, H., Linkens, D., Mahfoufa, M. & Millsb, G. (2002) Rule-base derivation for intensive care ventilator control using anfis. *Artificial Intelligence in Medicine* 29:185–201.
- Laurinen, P. (2000) Modelling the temperature of a steel strip after roughing mill with bayesian networks and neural networks. Master's thesis, Department of Mathematics, Statistics, University of Oulu, Finland.
- Laurinen, P. & Röning, J. (2005) An adaptive neural network model for predicting the post roughing mill temperature of steel slabs in the reheating furnace. *Journal of Materials Processing Technology*.
- Laurinen, P., Röning, J. & Tuomela, H. (2001) Steel slab temperature modelling using neural and bayesian networks. In: *Fourth International ICSC Symposium on Soft Computing and Intelligent Systems for Industry*. Paisley, Scotland, UK.
- Laurinen, P., Junno, H., Tuovinen, L. & Röning, J. (2004a) Studying the quality of resistance spot welding joints using bayesian networks. In: *Artificial Intelligence and Applications (AIA 2004)*. Innsbruck, Austria, pp. 705–711.
- Laurinen, P., Tuovinen, L., Haapalainen, E., Junno, H., Röning, J. & Zettel, D. (2004b) Managing and implementing the data mining process using a truly stepwise approach. In: *Proceedings of the Sixth International Baltic Conference on Databases & Information Systems (DB&IS2004)*. Riga, Latvia, pp. 246–257.
- Laurinen, P., Tuovinen, L. & Röning, J. (2005) Smart archive: A component-based data mining application framework. In: *Proceedings of the 5th International Conference on Intelligent Systems Design and Applications*. IEEE Computer Society Press, Wroclaw, Poland, pp. 20–26.
- Laurinen, P., Siirtola, P. & Röning, J. (2006) Efficient algorithm for calculating similarity between trajectories containing an increasing dimension. In: *accepted to the proceedings of Artificial Intelligence and Applications (AIA 2006)*.
- Lee, D. & Lee, Y. (2002) Application of neural-network for improving accuracy of roll-force model in hot-rolling mill. *Control Engineering Practice* 10(4):473–478.
- Lee, S., Kwon, D. & Lee, S. (2004) Minimum distance queries for time series data. *Journal of Systems and Software* 69(1-2):105–113.
- Lennox, B., Montague, G., Frith, A., Gent, C. & Bevan, V. (2001) Industrial application of neural networks – an investigation. *Journal of Process Control* 11(5):443–559.
- Leo, C.-K. L. & Burce, S.-C. C. (2005) Process optimization of gold stud bump manufacturing using artificial neural networks. *Expert Systems with Applications* 29(2):264–271.
- Li, Y., Sundararajan, N. & Saratchandran, P. (2001) Neuro-flight controllers for aircraft using minimal resource allocating networks (mran). *Neural Computing & Applications* 10:172–183.
- Liu, J. & Han, J. (2002) A practical knowledge discovery process for distributed data mining. In: *In Proc. ISCA 11th International Conference on Intelligent Systems: Emerging Technologies*. pp. 11–16.
- Lui, A., Grigg, M., Au, T. & Owen, M. (2000) Component based application framework for systems utilising the streaming data passing semantic. In: *37th International Conference on Technology of Object-Oriented Languages and Systems (TOOLS-Pacific 2000)*. Sydney, Australia, pp. 328–339.
- Man, L. W. & Kwong, S. L. (2000) *Data Mining Using Grammar-Based Genetic Programming and Applications*. Kluwer Academic Publishers, Norwell, MA, USA.
- Mariño, P., Sigüenza, C., Nogueira, J., Poza, F. & Dominguez, M. (2000) An event driven software architecture for enterprise-wide data source integration. In: *International Conference on Information Technology: Coding and Computing (ITCC 2000)*. Las Vegas, Nevada, United States, pp. 140–145.
- Mark, B. (1996) Data mining - here we go again. *Expert, IEEE* 11(5):18–19.
- Markou, M. & Singh, S. (2003a) Novelty detection: a review - part 1: statistical approaches. *Signal Process.* 83(12):2481–2497.

- Markou, M. & Singh, S. (2003b) Novelty detection: a review - part 2: neural network based approaches. *Signal Process.* 83(12):2499–2521.
- Marsland, S., Nehmzow, U. & Shapiro, J. (2005) On-line novelty detection for autonomous mobile robots. *J. Robotics and Autonomous Systems* 51:191–206.
- Martinez, W. L. & Martinez, A. R. (2001) *Computational Statistics Handbook with MATLAB*. Chapman & Hall.
- Masters, T. (1995a) *Advanced algorithms for neural networks a C++ sourcebook*. John Wiley and Sons.
- Masters, T. (1995b) *Neural, Novel & Hybrid Algorithms for Time Series Prediction*. John Wiley and Sons.
- McLachlan, G. J. (2004) *Discriminant Analysis and Statistical Pattern Recognition*. Wiley-Interscience.
- Meretnia, N. & de By, R. A. (2002) Aggregation and comparison of trajectories. In: *Proceedings of the 10th ACM International Symposium on Advances in Geographic Information Systems ACM-GIS*. pp. 49–53.
- Moyle, S. & Jorge, A. (2001) Ramsys - a methodology for supporting rapid remote collaborative data mining projects. In: *ECML/PKDD'01 workshop on Integrating Aspects of Data Mining, Decision Support and Meta-Learning: Internal SolEuNet Session*. pp. 20–31.
- Muthukrishnan, S. M. & Sahinalp, S. C. (2002) Simple and practical sequence nearest neighbors with block operations. *Lecture notes in computers science* 2373:262–278.
- Nilsson, A. (1998) Predicting the mean temperature of the transfer bar after rolling in the rougher using a neural network. *Journal of Materials Processing Technology* 80-81:469–474.
- Nunnari, G. (2004) Modelling air pollution time-series by using wavelet functions and genetic algorithms. *Soft Computing - A Fusion of Foundations, Methodologies and Applications* 8:173–178.
- Nurettin, A. (2005) Automated system for detection of epileptiform patterns in eeg by using a modified rbfn classifier. *Expert Systems with Applications* 29(2):455–462.
- Pal, S., Talwar, V. & Mitra, P. (2002) Web mining in soft computing framework: relevance, state of the art and future directions. *Neural Networks, IEEE Transactions on* pp. 1163–1177.
- Parag, P. (ed.) (2003) *Managing Data Mining Technologies in Organizations: Techniques and Applications*. Idea Group Inc.
- Parish, D., Pagonis, A., Barnet, D., Sandford, J. & Phillips, I. (2004) Presentation of real-time communication network measurement information. *Science, Measurement and Technology, IEE Proceedings-* 151(5).
- Pedrycz, W. (2000) *Knowledge Discovery for Business Information Systems*, Kluwer Academic Publishers, chap. The Role of Granular Information in Knowledge Databases, pp. 294 – 305.
- Perez, P. & Reyes, J. (2001) Prediction of particulate air pollution using neural techniques. *Neural Computing & Applications* 10:165–171.
- Pirttikangas, S., Riekkki, J. & Röning, J. (2004) Routine learning: Analyzing your whereabouts. In: *International Conference on Information Technology (ITCC 2004)*. Las Vegas, NV, USA.
- Pontoppidan, N. H., Sigurdsson, S. & Larsen, J. (2005) Condition monitoring with mean field independent components analysis. *Mechanical Systems and Signal Processing In Press, Corrected Proof*, Available online 15 September.
- Press, O. U. (1999) *Neural Smthing: Supervised Learning in Feedforward Artificial Neural Networks*. The MIT Press.
- Pyle, D. (1999) *Data preparation for data mining*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.
- Rafiei, D. & Mendelzon, A. (2000) Querying time series data based on similarity. *IEEE Transactions on Knowledge and Data Engineering* 12(5):675–693.
- Ramoni, M. & Sebastiani, P. (1997a) *Bayesian Knowledge Discoverer: reference manual*. KMI, Open University, 1997.
- Ramoni, M. & Sebastiani, P. (1997b) *Learning bayesian networks from incomplete databases, knowledge media institute technical report, kmi-tr-43*. Tech. rep., Knowledge Media Institute.
- Roodyn, N. & Emmerich, W. (1999) An architectural style for multiple real-time data feeds. In: *21st International Conference on Software Engineering (ICSE '99)*. Los Angeles, California, United States, pp. 564–572.

- Sandford, M., Parish, D., Sandford, P. & Phillips, I. (2005) Automated detection of unexpected communication network performance changes. *Communications, IEE Proceedings-* 152(5):743–748.
- Sasisekharan, R., Seshadri, V. & Weiss, S. (1996) Data mining and forecasting in large-scale telecommunication networks. *Expert, IEEE* 11(1):37–43.
- Savasere, A., Hanley, D., Raspl, S., Grossman, R. & Michal, S. (2005) Predictive Model Markup Language (PMML), 3rd edn.
- Schlang, M., Lang, B., Poppe, T., Runkler, T. & Weinzierl, K. (2001) Current and future development in neural computation in steel processing. *Control Engineering Practice* 9(9):975–986.
- Silva, E., do Prado, H. & Ferneda, E. (2002) Text mining: crossing the chasm between the academy and the industry. In: *In Proc. Third International Conference on Data Mining*. pp. 351–361.
- Singh, S. & Markou, M. (2004) An approach to novelty detection applied to the classification of image regions. *Knowledge and Data Engineering, IEEE Transactions on* 16:396–407.
- Suraj, Z. & Grochowalski, P. (2005) The rough set database system: An overview. *Transactions on Rough Sets III* 3:190–201.
- Takahashi, R. (2001) State of the art in hot rolling process control. *Control Engineering Practice* 9(9):987–993.
- Tarassenko, L., Hayton, P., Cerneaz, N. & Brady, M. (1995) Novelty detection for the identification of masses in mammograms. In: *Artificial Neural Networks, 1995., Fourth International Conference on*. pp. 442–447.
- TWI (2005) Twi knowledge summary: Resistance spot welding, available on-line at: http://www.twi.co.uk/j32k/protected/band_3/kssaw001.html, referenced 1.4.2006.
- Verikas, A., Malmqvist, K. & Bergman, L. (2000) Neural networks based colour measuring for process monitoring and control in multicoloured newspaper printing. *Neural Computing & Applications* 9:227–242.
- Vermeulen, W., Bodin, A. & van der Zwaag, S. (1997) Prediction of the measured temperature after the last finishing stand using artificial neural networks. *Steel Research* 68:20–26.
- Vlachos, M., Hadjieleftheriou, M., Gunopulos, D. & Keogh, E. (2003) Indexing multi-dimensional time-series with support for multiple distance measures. In: *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM Press, New York, NY, USA, pp. 216–225.
- Worden, K. & Dulieu-Barton, J. (2004) An overview of intelligent fault detection in systems and structures. *Structural Health Monitoring* 3(1):85–98.
- Xing, D. & Shen, J. (2004) Efficient data mining for web navigation patterns. *Information and Software Technology* 46:55–63.
- Yamada, S. (2004) Recognizing environments from action sequences using self-organizing maps. *Applied Soft Computing* 4:35–47.
- Yanagisawa, Y., ichi Akahani, J. & Satoh, T. (2003) Shape-based similarity query for trajectory of mobile objects. In: *MDM '03: Proceedings of the 4th International Conference on Mobile Data Management*. Springer-Verlag, London, UK, pp. 63–77.
- Yang, Y., Zhou, C. & Rena, J. (2003) Model reference adaptive robust fuzzy control for ship steering autopilot with uncertain nonlinear systems. *Applied Soft Computing* 3:305–316.
- Yao, X., Tieu, A., Fang, X. & Frances, D. (1995) Neural network application to head & tail width control in a hot strip mill. *IEEE International Conference on Neural Networks* 1:433–437.
- Yi, B.-K. & Faloutsos, C. (2000) Fast time sequence indexing for arbitrary l_p norms. In: *VLDB '00: Proceedings of the 26th International Conference on Very Large Data Bases*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 385–394.
- Yi, B.-K. & Roh, J.-W. (2004) Similarity search for interval time sequences. In: *Database Systems for Advances Applications, 9th International Conference, DASFAA 2004, Jeju Island, Korea, March 17-19, 2004, Proceedings*. Springer, vol. 2973 of *Lecture Notes in Computer Science*, pp. 232–243.
- Zaffalona, M., Wesnes, K. & Petrini, O. (2003) Reliable diagnoses of dementia by the naive credal classifier inferred from incomplete cognitive data. *Artificial Intelligence in Medicine* 29:61–79.
- Zdzislaw, P. (2005) Flow graphs and data mining. *Lecture Notes in Computer Science* 3400:1–36.
- Ziarko, W. (1998) Rough sets in knowledge discovery, Part II. *Studies in Fuzziness and Soft Computing*, Physica-Verlag, Heidelberg, Germany, vol. 2, chap. KDD-R: rough sets based data mining system, pp. 598–601.

ACTA UNIVERSITATIS OULUENSIS
SERIES C TECHNICA

230. Kansanen, Kimmo (2005) Wireless broadband single-carrier systems with MMSE turbo equalization receivers
231. Tarkkonen, Juhani (2005) Yhteistoiminnan ehdoilla, ymmärryksen ja vallan rajapinnoilla. Työsuojeluvälineet ja -päälliköt toimijoina, työorganisaatiot yhteistoiminnan areenoina ja työsuojelujärjestelmät kehittämisen kohteina
232. Ahola, Timo (2005) Intelligent estimation of web break sensitivity in paper machines
233. Karvonen, Sami (2006) Charge-domain sampling of high-frequency signals with embedded filtering
234. Laitinen, Risto (2006) Improvement of weld HAZ toughness at low heat input by controlling the distribution of M-A constituents
235. Juuti, Jari (2006) Pre-stressed piezoelectric actuator for micro and fine mechanical applications
236. Benyó, Imre (2006) Cascade Generalized Predictive Control—Applications in power plant control
237. Kayo, Olga (2006) Locally linear embedding algorithm. Extensions and applications
238. Kolli, Tanja (2006) Pd/Al₂O₃ -based automotive exhaust gas catalysts. The effect of BaO and OSC material on NO_x reduction
239. Torkko, Margit (2006) Maatilakäytäntöjen yritysten toimintamalleja. Laadullinen tutkimus resursseista, kehittämisestä ja ohjaustarpeista
240. Hämäläinen, Matti (2006) Singleband UWB systems. Analysis and measurements of coexistence with selected existing radio systems
241. Virtanen, Jani (2006) Enhancing the compatibility of surgical robots with magnetic resonance imaging
242. Lumijärvi, Jouko (2006) Optimization of critical flow velocity in cantilevered fluid-conveying pipes, with a subsequent non-linear analysis
243. Stoor, Tuomas (2006) Air in pulp and papermaking processes
244. György, Zsuzsanna (2006) Glycoside production by in vitro *Rhodiola rosea* cultures
245. Özer-Kemppainen, Özlem (2006) Alternative housing environments for the elderly in the information society. The Finnish experience

Book orders:
OULU UNIVERSITY PRESS
P.O. Box 8200, FI-90014
University of Oulu, Finland

Distributed by
OULU UNIVERSITY LIBRARY
P.O. Box 7500, FI-90014
University of Oulu, Finland

S E R I E S E D I T O R S

A
SCIENTIAE RERUM NATURALIUM
Professor Mikko Siponen

B
HUMANIORA
Professor Harri Mantila

C
TECHNICA
Professor Juha Kostamovaara

D
MEDICA
Professor Olli Vuolteenaho

E
SCIENTIAE RERUM SOCIALIUM
Senior Assistant Timo Latomaa

E
SCRIPTA ACADEMICA
Communications Officer Elna Stjerna

G
OECONOMICA
Senior Lecturer Seppo Eriksson

EDITOR IN CHIEF
Professor Olli Vuolteenaho

EDITORIAL SECRETARY
Publication Editor Kirsti Nurkkala

ISBN 951-42-8125-X (Paperback)

ISBN 951-42-8126-8 (PDF)

ISSN 0355-3213 (Print)

ISSN 1796-2226 (Online)

