JYVÄSKYLÄ STUDIES IN COMPUTING 32

Jouni Raitamäki

An Approach to Linguistic Pattern Recognition Using Fuzzy Systems

Esitetään Jyväskylän yliopiston informaatioteknologian tiedekunnan suostumuksella
julkisesti tarkastettavaksi yliopiston vanhassa juhlasalissa (S212)
joulukuun 18. päivänä 2003 kello 12.

Academic dissertation to be publicly discussed, by permission of
the the Faculty of Information Technology of the University of Jyväskylä,
in the Auditorium S212, on December 18, 2003 at 12 o'clock noon.

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2003

An Approach to Linguistic Pattern Recognition Using Fuzzy Systems

JYVÄSKYLÄ STUDIES IN COMPUTING 32

Jouni Raitamäki

An Approach to Linguistic Pattern Recognition Using Fuzzy Systems

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2003

Editors
Tommi Kärkkäinen
Department of Mathematical Information Technology, University of Jyväskylä
Pekka Olsbo, Marja-Leena Tynkkynen
Publishing Unit, University Library of Jyväskylä

URN:ISBN 9513916537
ISBN 951-39-1653-7 (PDF)

ISBN 951-39-1377-2 (nid.)
ISSN 1456-5390

Jyväskylä University Printing House, Jyväskylä
and ER-Paino Ky, Lievestuore 2003

ABSTRACT

Raitamäki, Jouni
An Approach to Linguistic Pattern Recognition Using Fuzzy Systems
Jyväskylä: University of Jyväskylä, 2003, 165 p.
(Jyväskylä Studies in Computing
ISSN 1456–5390; 32)
ISBN 951-39-1653-7
Finnish summary
Diss.

The present work is devoted to the study of fuzzy logic systems and to enhance their applicability to real-world problems. The goal is to find techniques for extracting readable linguistic representations from numerical data. The problematic relationship between interpretability and accuracy of a model is considered throughout the thesis. Fuzzy logic systems are applied to both modeling and knowledge acquisition types of problems.

Modeling is viewed as a functional mapping from the input to the output. A fuzzy logic system that is suitable for this kind of task and methods to train such a system are sought for. Also ways to reduce the number of rules and dimensionality are studied.

In knowledge acquisition the goal is to find relevant features from high-dimensional data and to form rules which can be presented as a quick overview of the data for the user. Thus, this application could be described with information compression or summarization. The basic idea behind the proposed method is to evaluate the importance of features by using a (fuzzy) entropy criterion. The method is tested with real-world examples.

Keywords: fuzzy logic systems, feature selection, fuzzy rule extraction, fuzzy if–then rules, fuzzy modeling

Author Jouni Raitamäki
Department of Mathematical Information Technology
University of Jyväskylä
Finland

Supervisor Dr. Pasi Koikkalainen
Department of Mathematical Information Technology
University of Jyväskylä
Finland

Reviewers Dr. Vesa Niskanen
Department of Economics and Management
University of Helsinki
Finland

Professor Olli Silvén
Department of Electrical and Information Engineering
University of Oulu
Finland

Opponent Professor Juha Röning
Department of Electrical and Information Engineering
University of Oulu
Finland

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my instructor, Dr. Pasi Koikkalainen. Without his extensive knowledge, criticism, assistance and guidance this work would have never become ready. He has helped me to clear many misunderstandings and problems encountered in this work. His contribution to make the overall structure of the thesis better has been significant. I am deeply indebted to him for his help.

I am thankful to Dr. Vesa Niskanen from the University of Helsinki and Professor Olli Silvén from the University of Oulu for their comments and criticism. I am also thankful to Edward M. Riseman from the University of Massachusetts for his valuable comments.

I thank my present and former colleagues in our research group, especially Anssi Lensu who has been a good workmate and friend during these several years. I would also like to thank Professor Pekka Neittaanmäki and COMAS graduate school for giving me the opportunity to start my postgraduate studies.

The seminars organized by the Centre for Mathematical and Computational Modeling (CMCM) have provided an excellent way to familiarize myself with the scientific work done outside our research group. I thank Professors Antti Penttinen, Pekka Orponen and Pasi Koikkalainen for this. I have also enjoyed the "book seminars for postgraduate students" which were organized by Pasi Koikkalainen and Antti Penttinen. These seminars have proved to be the most efficient way to broaden my view and learn new things.

I am grateful to language consultants Gary Littler and Robert Shakspeare (who read the earlier version of this thesis) for their careful reading and linguistic comments.

This work was financially supported by COMAS, Graduate School of Computing and Mathematical Sciences, at the University of Jyväskylä and The Academy of Finland under grant MaDaMe/72497. Early development in Fiber Analysis were supported by TEKES under project "NDA ympäristön laajentaminen konenäkösovelluksiin". The support from the Jyväskylä Science Park (JSP) to the Laboratory of Data Analysis is also acknowleged.

I wish to thank Pekka Pakarinen and Marko Avikainen (Metso Paper Corporation, Finland) for providing the paper image data.

Finally, I want to express my appreciation to my family and friends for their support and patience.

Jyväskylä, December 8, 2003

Jouni Raitamäki

CONTENTS

LIST OF SYMBOLS AND ABBREVIATIONS

General symbols used throughout the thesis

$\emptyset$	Empty set
$\mathbf{1}_A(\cdot)$	Indicator function
$\mathbb{A}$,$\mathbb{B}$,...	Matrices
a_{ij}	The ijth element of matrix $\mathbb{A}$
$\lvert A \rvert$	Cardinality of set A
$\tilde{A}$, $\tilde{B}$	Fuzzy sets
$\tilde{A}_i^k$, $\tilde{B}_i^k$	Fuzzy sets on axis x_i, k indices the fuzzy set on that axis
$b_j(x\lvert\theta_j)$	The jth basis function
C_j	The jth class or cluster
$\mathbb{C}$	Covariance matrix
d	Dimension, number of inputs
$\mathcal{D}$	Data set
$D(\cdot,\cdot)$	Distance
E	Error
$\mathcal{E}$	Expectation
$\hat{f}(\cdot)$	Model
$f_X(\cdot)$	Probability density function
$f_X(\cdot\lvert\cdot)$	Conditional probability density function
$g(\cdot)$	Sigmoidal function
i, j, k, l, n	Indexes, i is mostly used as input label
$i(r), i_j(r)$	Index functions that can be used to indice the $i(r)$th variable for the rule r
I_j	Implication that is indexed by j
$\mathcal{I}_j$	The jth image
$J(\cdot)$	Feature selection criterion
$k(l), k_i(l)$	Index fuctions that indice the fuzzy set (on axis i) for class l
$K_j(\cdot,\cdot)$	The jth kernel function
M	Number of basis functions or membership functions
m_i	Number of fuzzy sets on input axis x_i
m_y	Number of fuzzy sets on output axis y
N	Number of patterns
$\mathrm{P}(\cdot)$	Probability
r	Index of rule
$\mathbb{R}$	Real line, $(-\infty,\infty)$
$\mathbb{R}^d$	d-dimensional Euclidean space
SSE	Sum of squared errors
v^j	Strength for output set $\tilde{B}^j$
$\mathbf{w}$	Weight vector
$x_{i_j}(r)$	Variable used in rule r
$\mathbf{x}$	Column vector whose elements x_i correspond to different variables
$\mathbf{X}(j)$	The jth data point (pattern)
$\{x_i\}$	Set of variables (features) x_i
$\bar{\mathbf{x}}$	Mean vector, center point

$\|\|\mathbf{x}\|\|$	Norm, Euclidean length of vector $\mathbf{x}$
$\|x\|$	Magnitude of scalar x
$x\|\mathcal{P}$	All elements x which possess property $\mathcal{P}$
$\tilde{x}$	Fuzzified variable
y	Output variable
$y_l, y_{l(r)}$	Output: class
$\mathbf{\Theta}_j$	Parameter vector
$\mu_{\tilde{A}_i^k}(\cdot)$	Fuzzy membership function that corresponds to fuzzy set $\tilde{A}_i^j$
τ	Threshold
λ	Coefficient, often the regularization coefficient
ω_r	Truth for rule r

1 INTRODUCTION

The main objective of this work is to find techniques for extracting meaningful linguistic information from numerical data. In order to do this in practice we try to find a balance between two, often contradictory, requirements: the interpretability and the accuracy of a model.

Although the final answer cannot be provided, an attempt will be made to challenge one general problem of data analysis: an automatic interpretation of observations. The reason why fuzzy methodology could provide the answer is that it combines the benefits of three important domains:

1. *Syntactical and logical information processing*, that allows us to describe the world via rules and structures. This is the traditional approach in computer science.

2. *Approximate and probabilistic information processing* which is required when building systems that interact with a noisy and inexact real world. This is also believed to be closer to human information processing than traditional computer logic with its deterministic procedures.

3. *Linguistic and semantic information processing* We emphasize that fuzzy systems are often more readable by humans than traditional syntactic or probabilistic systems. They provide linguistic interpretations and rules that support human interaction with computational machinery.

An attractive property of the fuzzy set theory is the definition of the linguistic (fuzzy) variable. It assigns a linguistic label, e.g. small, large, heavy or far, to each fuzzy set. When all inferences are carried out by using the linguistic variables, the fuzzy methods can be made readable in the sense that all the actions can be explained by rules that consist of human understandable words. This linguistic representation does not have to rely entirely on words. Numeric values are also acceptable as shown throughout the thesis. This is demonstrated in practice when a rule extraction method is applied to a fiber image data. Before that is done a motivation to the chosen methodological approach is given and the construction of the thesis is explained.

1.1 Questions and answers about knowledge extraction

The amount of information available in modern society is overwhelming. With increased storage and computing capability of computers, data collection has become easier. This has resulted almost in an explosive increase in the number and size of various databases. For example, information is gathered from our consumer habits. Our ways to surf the internet, loan books, travel, vote, and make movements on the account may be monitored or questioned by using gallups. Data acquisition systems in factories gather information about the performance of different engines and devices. The list could be continued.

Unfortunately, it is often the case that although valuable information is contained in these databases it may never be found and translated into a more easy human understandable form. The reason might be the lack of expertise, available amount of time, money, or a combination of them. A solution to this unsatisfactory situation is often searched for by using data mining techniques. However, the problem with many of these techniques is that they produce results that are difficult to understand. Although the results are accurate (e.g. classify new observations well) they might be presented in an overly complicated form.

Due to extracting knowledge from raw data is time-consuming there is a clear need for automated methods. This leads us to one of the main questions of the current thesis:

Is it possible to make automated systems that can extract important human readable knowledge from given data?

Although the exact answer is apparently negative, it still gives us an objective that we can try to get closer to. Before we can take steps to develop automated methodology we must consider the second part of the question more carefully.

What is important knowledge?

The answer is problem dependent; that is, it depends on what we aim to achieve. In "simple" cases it could be the information about how to classify or predict something with the help of background information, but this kind of knowledge is very weak kind of knowledge. It does not tell us about the reasons or causalities about the phenomenon, which we prefer as stronger (more important) information. The dilemma, implied by this characterization, is that simple "black box" systems like neural network classifiers can be automated, but systems that would give the causalities can not. Therefore we must look for intermediate solutions.

Generally we may regard knowledge as important if it reveals something about the true generator of the data, especially if it is something that is not known beforehand or if it is not known for sure. It could be just a few parameters associated with some distribution, if sufficient statistics are available for an identifiable

problem. It may also consist of information about relationship between different variables, statistically significant structures, or about patterns in data. For example, we might try to discover a connection between different variables in the generating process of fiber networks and the formation of spatial patterns in paper. If it is our aim to reveal this knowledge such that humans can learn to produce better quality paper, then it is not arguable whether this kind of information is important or not. The research questions regarding this kind of data will be stated in Chapter 2.

We may or may not have prior information about the data. If the data is unknown, the extraction of important knowledge could consist of uncovering patterns, changes or associations. Abnormalities in data might also reveal something important. We can summarize our requirement by asking for a representation that is quantitatively accurate enough (for prediction or classification) but also readable by humans. To do this we must combine quantitative and qualitative aspects of data analysis and decide:

What is good representation of knowledge?

We note that information may be represented in a qualitative or quantitative form. Quantitative data consists of numerical data, i.e. values that can be counted or measured. For qualitative data there is no unique way to define a distance metric, although one can develop subjective criterions for it. Examples of qualitative data are images, videos, music and words. Statisticians like Hastie et al. (2001) are usually more conservative. They divide the variable types into

1. Quantitative (numerical).
2. Qualitative or categorical.
3. Ordered categorical.

Statisticians regard all qualitative data as categorical, which is a somewhat narrow viewpoint on the matter. However, it can be justified in its context, i.e. statistical learning, where the majority of the methods work with numbers and identifiers, or must be coded in such a format.

Respect for numbers and exactness has always had a priority in science. That is, if someone can measure what he is speaking about, he can express it with numbers and he knows something about it. The problem with quantitative methods and numerical results is that there might not be an easy interpretation. Thus, supporting qualitative information and expert knowledge is required to interpret the meaning of numbers. This is in conflict with the requirement of analyzing new types of data, where such knowledge might not exist. This leads us to the next question:

How about human knowledge building?

The human way of problem solving has been studied in psychology. Humans often use simple mental representations to build an understanding of the subject.

Eysenck and Keane (1999) divide the representations into two parts: external representations (e.g. writing, pictures, and diagrams) and internal mental representations (images and propositional representations). All of these seem to be of a qualitative nature rather than a quantitative one. They write about the organization of knowledge as follows:

Cognitive economy is achieved by dividing the world into classes of things to decrease the amount of information we must learn, perceive, remember, and recognise (Collins & Quillian, 1969). Once concepts have been formed they can, in turn, be organised into hierarchies...

The obvious conclusion is that we need representations that humans are familiar with, like graphs, pictures and words. The use of graphical representations has been widely accepted, especially in the context of explorative data analysis (EDA). The use of textual representations, however, has mostly been limited to human sciences, outside of more exact scientific disciplines. This disrespect for words (qualitative information) in science is discussed in (Zadeh, 1999), where some arguments are given in the favor of alternative linguistic representations that are more readable by humans. Zadeh (1999) emphasizes the importance of "computing with words" (CW), where perceptions are expressed as propositions in a natural language. He notes that the justification for computing with words is as follows:

In essence, the rationale for computing with words rests on two major imperatives: 1) computing with words is a necessity when the available information is too imprecise to justify the use of numbers and 2) when there is a tolerance for imprecision which can be exploited to achieve tractability, robustness, low solution cost and better rapport with reality.

He also notices the relation to compactness:

In effect, fuzzy information granulation (fuzzy IG) may be viewed as a human way of employing data compression for reasoning and, more particularly, making rational decisions in an environment of imprecision, uncertainty and partial truth.

In this work we assume that the problem of understanding new types of data meets Zadeh's rationale. At the beginning of a data analysis we allow ourselves a vague idea of the real world. We further assume that this can be expressed with readable (linguistic) rules that are not necessarily exact but close enough for human inference about the causalities behind the data.

We note that in science the use of imprecise words only is never enough, rather we need exact knowledge. Therefore we must have a way to say exactly what is behind the linguistic representation that we have made. This leads to the proposed fuzzy architecture of this thesis, which is depicted in Figure 1.

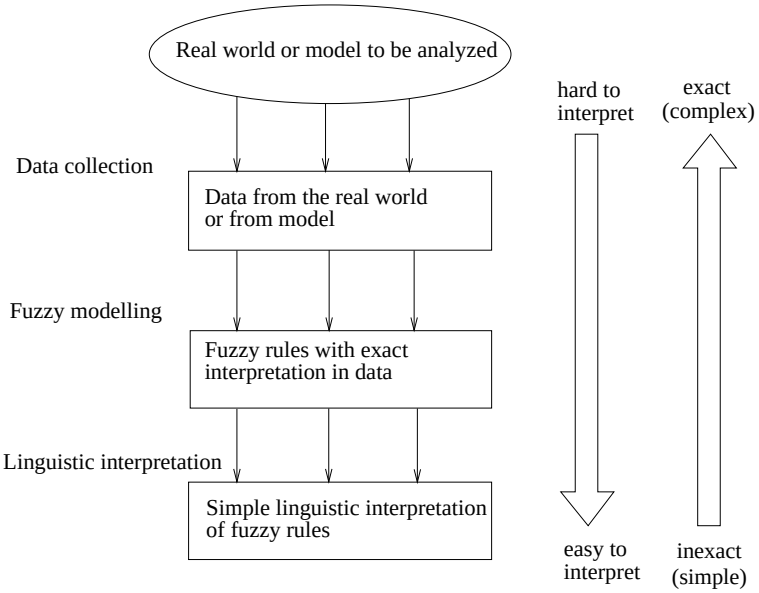

FIGURE 1 A fuzzy architecture where two levels of fuzzy rules are used: precise but complex and simple but readable.

The idea is that one fuzzy set of rules can be made such that its performance in classification or prediction type of tasks is high, while another set of rules gives a rough but close enough linguistic interpretation of the more accurate rule set. Using this idea we allow ourselves to make close connections between fuzzy systems and statistical learning when building the fuzzy representation of the data. This leads to more accurate fuzzy systems, which can be hard to interpret, but are more respectful from a science point of view. The human interpretation is supported, if needed, with another set of simplified rules that try to follow the "logic" of more accurate rules but are simpler for humans to interpret.

Often the differences between simple and complex rules is in the way how the fuzzy sets are created and named. Simple rules are fixed and human labelled, while complex rules use adaptive methods to optimize the number of rules, their shapes and locations. This adaptivity easily leads to a loss in human interpretation.

It is obvious that we must expect that the user understands the difference between simple and complex rules. Simple rules might be misleading and should only be used to help the data-analysis process, unless the exact nature of the rules

can be provided in some way.

1.2 How is fuzziness applied in this work?

Fuzzy methods are presented in this work in such a way that the leap from other methodologies (neural networks, wavelets, different interpolation and approximation techniques, statistical classifiers etc.) is not as large as what it could be when deduced by reading the basic fuzzy logic textbooks. The benefit of fuzziness over other methodologies is that, even when it is applied to function approximation types of problems, linguistic representations can be formed, but there is a problem. Most fuzzy methods work best when the number of variables is quite small and they are independent such that there is no need to construct large rule sets for all variable combinations. In our case this assumption does not hold.

The problem with curse of dimensionality (term introduced by Bellman, 1961) and the problem of combinatorial rule explosion (Combs and Andrews, 1998) are considered in this work. The following quote confirms the essentiality of this viewpoint. Duda et al. (2001) regard some limitations of fuzzy methods as "formidable":

Fuzzy methods are cumbersome to use in high dimensions or on complex problems or in problems with dozens or hundreds of features.

This claim was in the first place on their list and thus we may consider it as one of their main complaints about fuzzy methods. In general, we accept the claim as truth to some extent, and this will be discussed later. However, the dimensionality problem can be greatly reduced by careful design and using greedy divide-and-conquer strategies. For example, instead of using all features, only the best features are selected. Hundreds of features may be evaluated by dividing the set of features into smaller sets. The smaller group of good features may then be used to form linguistic representations for the classes.

In this thesis we divide the problem of fuzzy rule extraction into several subtasks that are studied in their own chapters. Each chapter adds one property to the system in such a way that the most automated system is presented last. The organization is depicted in Figure 2 and it is explained more carefully in the next section.

1.3 The content of this thesis

In **Chapter 2** the problem of fuzzy rule extraction is concretized with a real world example, the analysis of fiber image data. The chapter includes the presentation of the data set as well as a list of concrete questions that we hope to find answers to with fuzzy computing. Other data sets used in this thesis are also described.

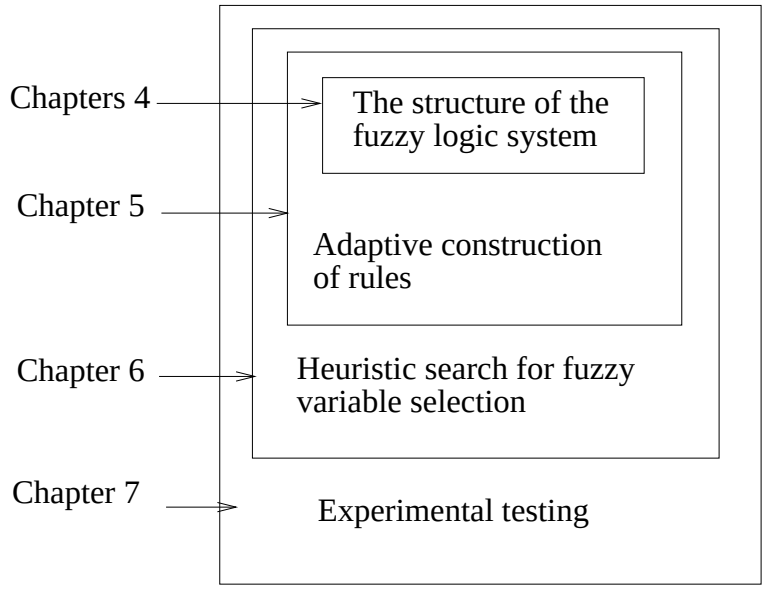

FIGURE 2 Organization of the chapters.

Chapter 3 is a review of feature selection and statistical methodology. Feature selection is our main tool to minimize the effects of curse of dimensionality. The statistical methodology is used in the thesis in two ways. First, it gives a background for some of the adaptive fuzzy methods that are developed in this thesis. Secondly, some of the methods are used as baselines to measure the performance of the developped fuzzy systems.

Chapter 4 begins the methodological content of this thesis. The used fuzzy logic system, and some standard notations are introduced here. **Chapter 4** also contains an additional evaluation of the chosen fuzzy system architecture. It is shown that the selected system has many advantages when compared to other possible architectures. This is provided for the interest of a fuzzy logic community. It gives reasons for the choices that are made.

Chapter 5 examines different possibilities to learn rules in a supervised way from the data. The aim is to favour those techniques that produce interpretable results. Regularization is proposed as a way to prune the rule set. The rule reduction is achieved by a penalized optimization strategy, which retains only those rules which are the most crucial for the modeling task.

Chapter 6 introduces a fuzzy feature selection and rule extraction algorithm. It is designed to solve multidimensional problems with tens of variables. The most important aspect of the method is that it can be used to explain data in readable rules. Each cluster or class in data is explained with features that separate it from the other clusters/classes such that the linguistic interpretation is possible. The features are class-specific.

Chapter 7 contains experimental testing of the method proposed in Chapter 6. The method is compared to well-known statistical learning techniques, like linear and quadratic discriminant analysis.

1.4 Contribution of the author

Chapter 2 presents the example problem data sets. The extraction of descriptive features of fiber image data has been carried out as teamwork with DrTech Pasi Koikkalainen and MSc Heikki Niittylä.

After giving a review of a variable and feature selection Chapter 3 describes different statistical techniques. The Tree-Structured Self-Organizing map (TS-SOM), Sammon mapping and Principal Component Analysis (PCA) have been implemented into a Neural Data Analysis (NDA) environment by DrTech Pasi Koikkalainen. More information about NDA and TS-SOM can be found from (Häkkinen, 2001). The author has implemented the other techniques used in this thesis.

Chapter 4 presents the fuzzy logic system used in this thesis. This chapter also gives some empirical justifications for the chosen system. The work done in this chapter is completely the author's work.

Chapter 5 introduces an adaptive construction of rules. An algorithm which utilizes the chosen fuzzy logic system and fixed membership functions to learn rules is introduced. Also a technique known as lasso shrinkage in statistical learning has been applied to the new task, the selection of rules in fuzzy systems. The viewpoint and some concepts used in this chapter are more common to the statistical learning community; this differs to some extent from the main viewpoint of a fuzzy community. This chapter mainly presents the author's work.

Chapter 6 proposes a new way to search the set of all features and to select the most suitable subset of them. This subset is then used to construct a set of interpretable linguistic rules. Chapters 6 and 7 are mainly the author's work.

The analyzation of fiber image data using SOM, principal component analysis and Sammon mapping as reference methods has been carried out as teamwork with DrTech Pasi Koikkalainen. All other experimental work is done by the author.

Finally, DrTech Pasi Koikkalainen has had a strong influence on all aspects of this work.

2 DATA SETS AND RESEARCH PROBLEMS

One of the most difficult questions in methodological research, and especially regarding the fuzzy methodology, is

How to evaluate the developed methodology?

The question is almost overly difficult in our case since we do not have one simple and clear criterion for the evaluation. Instead, we want the methods that support easy interpretation and reflect the real world accurately enough. The difficulty is that there is often a trade-off between these two criteria: something accurate may not be interpretable and vice versa.

If considering the question in more general terms, we notice that there are, in principle, two possibilities to do the evaluation. The first possibility is to analytically prove certain properties of the developed method like the computational complexity, robustness (e.g. breakdown point), asymptotic accuracy, etc. This way is difficult and often some simple assumptions must be made to succesfully carry out the analysis. These assumptions may be too restrictive and therefore a danger exists that a gap between a "theoretical clean world" and the real world may be established. Hence theoretical justification of some method(s) does not necessarily ensure that it works the best or even adequately in some real world problems.

The second possibility is empirical testing, where the aim is to show how well the methodology works in a real world setting. Also this approach has some problems:

- Real world phenomena is difficult to simulate or understand even if there is "enough" real world data available.
- All empirical tests give a limited and optimistic view of the methodology when compared to the large variation of a true real world phenomena. Diverseness of real world cannot be captured into simple data sets.

Yet, many people prefer that the methodological research is carried out using empirical tests, because it requires full implementation of the method. This kind

of "I believe when I see it" type of reasoning is also behind the evaluations done in this thesis.

Finally, one should note that the evaluation of a fuzzy linguistic data analysis is especially challenging because of the qualitative aspect of the methodology. By definition there is no exact measure for qualitative research. One can only try to express qualitative findings in a readable way. This can be done by studying parts of the whole problem with exact methods and combining them with qualitative interpretation.

2.1 Research problems

To concretize the evaluation of the developed fuzzy methodology, six real world data sets are used and they are summarized in Table 1. The data sets are described more carefully in the following sections. The default fiber image data set used in this thesis is the fiber image data set 1. The fiber image set 2 is analyzed only in section 7.6.2, where the objective is to find linguistic rules for clustered data.

TABLE 1 Summary of data sets.

Data set	Records	Variables	Categories	Description
Fiber image set 1	52	48	4	see section 2.4
Fiber image set 2	40	40	?	see section 2.4
Iris	150	4	3	see section 2.5.1
Sonar	208	60	2	see section 2.5.2
Vowel	990	10	11	see section 2.5.3
Wine	178	13	3	see section 2.5.4
Glass	214	9	6	see section 2.5.5

It is typical to these data sets that the number of observations in respect to the number of variables is small. In an extreme case, In Fiber image set 1, there are only four observations for the second class, which does not allow reliable modeling of the class boundaries even with low-dimensional feature spaces. In all data sets (except in fiber image set 2) there are categorical variables, which may or may not be easy to interpret. Depending on the data set there can be a large number of other variables, which are usually not easy to explain, but which might hold important information. We can now formulate a common research question for all data sets:

In terms of classification performance, what are the most important variables and criteria?

This is just one example of possible questions, but it serves us well for the evalu-

ation purpose, because we may compare the results with other variable selection methods and classifiers (see Chapter 3 for an introduction of comparative methodologies). We can evaluate the methods according to

i) Classification performance;

ii) Computational complexity; and

iii) Interpretation of the classification rules:

   a) Are the rules readable?

   b) Is the importance of individual variables clearly expressed?

   c) What is the complexity of the rules?

   d) What do we learn about the data from the rules?

Although all questions about item iii) are qualitative and subjective, there are many characteristics that are quite apparent. For example, if considering the methods in Chapter 3 some algorithms are clearly black-box type, giving almost no qualitative information about the logic of the classifier. Some other might be expressed in a great number of detail, but without any simple and readable explanation. Hence one important research problem regarding the developed methodology in this work is

How to increase the interpretability without decreasing the accuracy?

One problem of fuzzy methodology is that when it is applied to some nontrivial real world data set, an overwhelming number of linguistic rules may result. Unfortunately it is often the case that the rules consist of a large number of variables. Therefore, we seek ways to limit the number and complexity of the rules. Furthermore, one aim of this work is that the developed methodology provides a way to form a quick overview of the data. For example, the methodology should provide answers to the questions:

- Which features describe each class the best?
- How difficult the data is from the rule extraction point of view? (e.g. How much the classes overlap each other?)
- How well the produced rules describe the data? (e.g. Can we rely on the result?)

2.2 The generalized variable selection problem

In general, the aim of this thesis is not limited to classifier construction. Rather, the wider objective is linguistic data analysis using fuzzy systems. Due to the fact

that the presentation and evaluation of the methodology is made from the perspective of classifiers, some discussion is required to bridge low level objectives to the bigger picture.

In general we may regard the classifier performance as a measure $E(\{\mathbf{x}\})$ over the data set $\{\mathbf{x}\}$ with variables $\mathbf{x} = \{x_1, x_2, \ldots, x_d\}$. For example, if x_k is a categorical variable, that we want to predict, $E(\cdot)$ could be the total hard classification error

$$E = \sum_{j=1}^{N} \begin{cases} 1, & \text{if } X_k(j) \neq g(\mathbf{X}^{(-k)}(j)), \\ 0, & \text{otherwise,} \end{cases}$$

between x_k and classifier output g, where $\mathbf{x}^{(-k)}$ denotes variables $\mathbf{x}$ with x_k excluded. The generalized objective for the developed methodology can now be expressed for any measure $E(\cdot)$ with variable subsets $\mathbf{x}^* \subset \mathbf{x}$. The aim is to find the smallest subset of variables $\mathbf{x}^*$ such that the loss of information is less than ϵ with some probability $1 - \delta$. Formally this can be expressed as

$$\min_{|\mathbf{x}^*|} \mathrm{P}(|E(\{|\mathbf{x}\}) - E(\{|\mathbf{x}^*\})| > \epsilon) < \delta, \tag{1}$$

where we assume that $E(\mathbf{x})$ is a perfect classifier with conditions

$$E(\{\}) = 0 \tag{2}$$

and

$$E(\{\mathbf{x}^A\}) \leq E(\{\mathbf{x}^B\}) \text{ if } \mathbf{x}^A \subset \mathbf{x}^B. \tag{3}$$

In real world the condition (3) is very difficult to guarantee, because perfect classifiers are hard to find. The objective (2) is therefore only a theoretical guideline that allows us to express what the desired properties of our methodology are. Informally this can be written as:

- We want to obtain a minimal set of variables $\mathbf{x}^*$.
- We accept information loss ϵ.
- We require that this is achieved with certainty $1 - \delta$.

2.3 Wider application possibilities

There is a difference between evaluation and the actual use of methodology. In evaluation we must quantify our findings as well as possible, while the actual use may be an interactive process, where only the final result might be measurable. This is also the case with current methodology.

The tools that are presented in this thesis are well applied to work with explorative data analysis. For example, it is often customary to use unsupervised clustering to examine a new data set. The clusters $C_1, C_2, \ldots, C_m$ are data subsets $\{\mathbf{x}|C_l\}$ of original data set $\{\mathbf{x}\}$, and the aim is to interpret data via similarities

and differences between the subsets. In many methods, like the self-organizing map (SOM) (see Kohonen, 1997), one usually tries to use visualization or simple labeling rules to see the differences.

The fuzzy rule systems that are presented in this thesis provide a generalization of labeling rules, which can be used with unsupervised clustering methods like the SOM, classification trees (see for example, Duda et al., 2001; Hastie et al., 2001), or the K-means algorithm. Given a set of clusters, one can ask to present the clusters as rules of type

What rule set best explains the difference between C_l and the rest of data $\cup_{l \neq k} C_k$?

If there is a simple IF-THEN type of rule with easy to interpret variables, then this is clearly what the data analysis is trying to do.

In practice this type of analysis works best when the actual subsets C_l are hand-picked from a larger number of prototypes (cluster centroids), as it is often done with SOM-assisted data analysis. Good examples of this are presented in the work of Kohonen (1997) and in the recent PhD theses by Häkkinen (2001) and Lensu (2002).

2.4 An introduction to fiber image data

In this chapter a challenging real world data analysis task, a fiber image analysis problem, is explained. This and some other data sets will be used in the thesis to evaluate various aspects of the proposed fuzzy methodology. While other data sets are publicly available and well documented, this is not the case with fiber data. Therefore a brief introduction to this data set and its problems is given in the following text.

As in many real world problems, the understanding of the structure of paper is a complex problem, for which there is no "true answer" nor is there any single methodology for the problem. Rather there are several ways to collect information by doing physical measurements (see for example, Mark et al., 2002).

In the paper and pulp industry the regularity of fiber mass distribution is called formation. Formation is an important property since it affects the strength, printability, visual appearance, and other properties of paper. The full description of the arrangement of fibers, however, cannot be done using on-line devices or everyday laboratory measurements (Kajanto et al., 1989). This is because there is no measuring system that is both fast and accurate. For example, optical measurement is fast but its accuracy is known to depend on the optical properties of the paper. Measurement based on radiation gives a more accurate estimation of formation (although not accurate enough to distinguish individual fibers), but it is quite slow. Thus, a more coarse representation of fibers must be used in practice.

One way to evaluate formation is to use β-radiography combined with digital image analysis. The β-radiogram images are produced by placing the object, a paper sheet, on top of a β-radiation source. An X-ray film is then laid on the paper. Radiograph exposes the film and the result is a β-radiogram image (see Figure 4). This is an indirect method of measuring transmittance of radiation through the paper. The direct method would require a detector which would replace the X-ray film. The benefit of the indirect method is that the resolution is better than any modern CCD devices can produce.

Image gray-levels indicate how much the radiation has penetrated the paper. The lighter the film is the more radiation has been absorbed by the paper, which means that lighter areas contain more fiber mass. The advantage of the method is its immunity to variations in optical properties. Thus, the β-radiogram gives a good basis for off-line formation measurements.

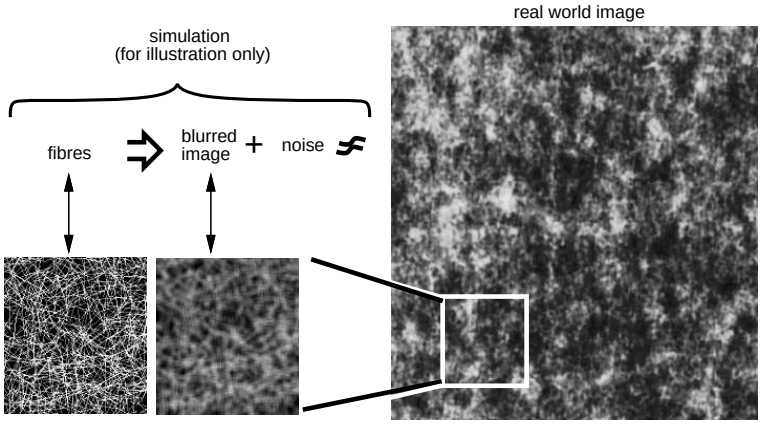

FIGURE 3 The figure illustrates a process from individual fibers to a fiber network. A real world image could be similar to a noised and blurred simulation of fibers. In practice the process is more complex.

The process from an ideal fiber distribution to the final fiber image is illustrated in Figure 3 by a rough artificial example. Individual fibers cannot be distinguished in the final image—only the aggregation of them can. In a real world paper making process the situation is more complicated, since the structure and properties of the paper machine, raw materials and control points, etc., affect the fiber distribution. A flow of paper mass, which consists of a mixture of pulp and water, through the paper machine changes the fiber orientation distribution from the ideal. Flow vortices gather fibers into dense accumulations, called fiber flocs. Furthermore, different substances used in paper making blend the individual fibers to a uniform compound that is, hopefully, as uniform as possible.

Examples of real β-radiogram images can be seen in Figure 4. The brightness of the β-radiogram images may vary a lot, even if they are obtained from the same

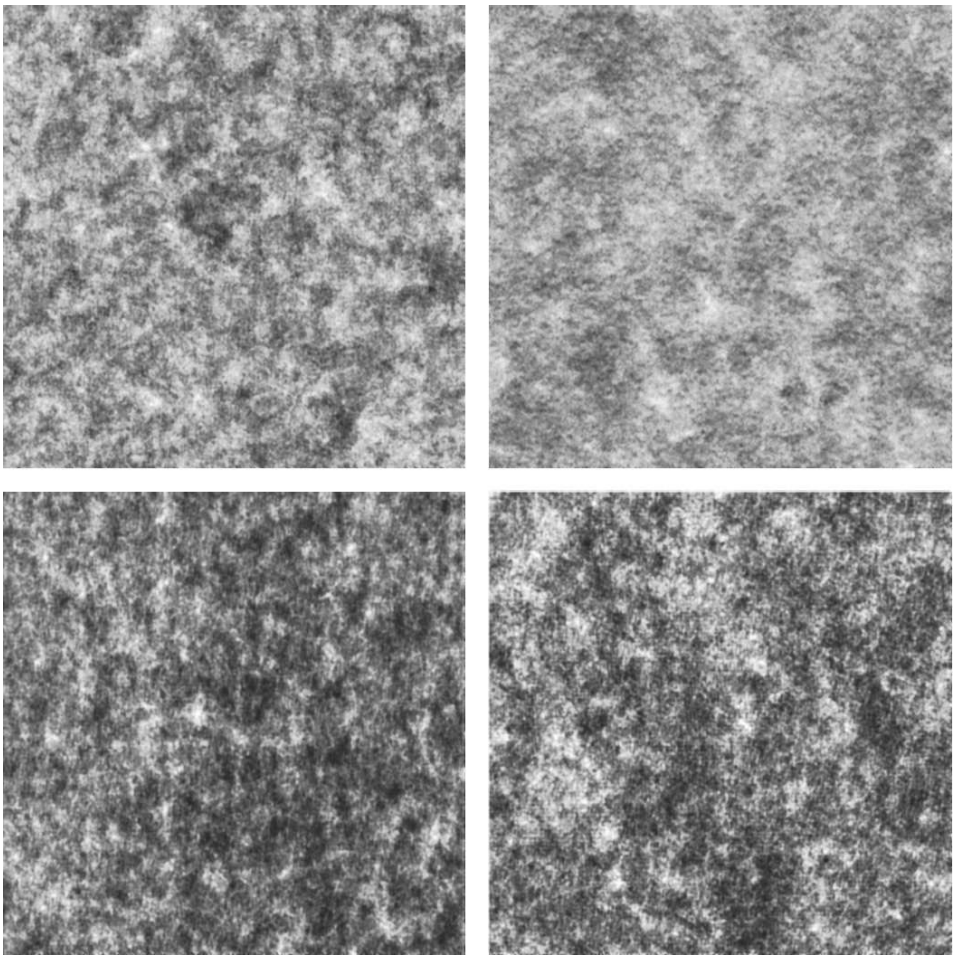

FIGURE 4 Examples of typical β-radiograms of paper (image brightness has been altered and contrast enhanced). Fiber clusters (flocs) can be seen as lighter areas. Each image corresponds to one paper class.

paper making process. This is due to the fact that image acquisition is sensitive to the power of the radiation source and the radiation time. One should also note that the gray-level histograms of the images have different mean intensities but their shapes are almost identical Gaussians.

From an application point of view fiber clusters, flocs, are essential for the analysis of the physical properties of paper (Kajanto et al., 1989). Therefore, it is most important to find their representative features.

2.4.1 The research problem

There are several research questions in the analysis of β-radiogram images. In the following, all questions are related to situations where we have extracted a set of features $\mathbf{x} = \{x_1, x_2, \ldots, x_d\}$ from a set of example images $\mathcal{I} = \{\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_N\}$. Furthermore, we may assume that there are some labels $\mathbf{y} = \{y_1, y_2, \ldots, y_m\}$ that can be used to describe the quality of the paper. In the simplest case y_l is a quality class

$$y_l = \begin{cases} 1, & \text{if class is } l, \\ 0, & \text{otherwise,} \end{cases}$$

but it could also be some other kind of quality indicator. There are also cases where a series of images is taken to investigate how a change of process parameters changes the quality or other characteristics of the paper. This can be denoted by additional time indexes $\mathcal{I}^t, \mathcal{I}^{t+1}, \ldots, \mathcal{I}^{t+n}$ on images, features and quality labels, if needed.

Some of the concrete research questions can now be stated as

1. Given a set of images that represent classes $l = 1, 2, \ldots, m$, what are the best or sufficient features from each class of observations

   $$\mathcal{D}^l = \{\mathbf{X}(j) | Y_l(j) = l\}?$$

2. Assume, that there is a ranked indicator Y such that samples can be ordered according to their quality

   $$\text{Quality of } \mathcal{I}_k > \mathcal{I}_j \text{ if } Y(k) > Y(j).$$

   Given a set of observations $\mathcal{D} = \{\mathbf{Y}(j), Y(j)\}$, what are the features $\{x_i\}$ that best preserve the quality ordering of samples (without knowing y)?

3. Given a series of changing process parameters $\theta^{t_1}, \theta^{t_2}, \ldots$ (about the paper manufacturing process) and related measurements

   $$\mathcal{D}^{t_1} = \{\mathbf{x}^{t_1}\}, \mathcal{D}^{t_2} = \{\mathbf{x}^{t_2}\}, \ldots,$$

   what are the features $\{x_i\}$ that are most affected by the changes in θ^{t_i}? We could also ask for a linguistic explanation of these changes. Due to contentual aspects and lack of good time-dependent paper manufacturing process data this type of research has been omitted from this thesis.

All these questions (and more) are addressed in a project *Fiber Image Analysis* that is funded by the Academy of Finland under the MaDaMe research programme. In the current work we limit ourselves to the first question, classification of images.

In the experiments we have used 52 real-world images that represent actual products from the paper manufacturing process. The process was run with four

different parameter settings, which give us four classes of images. The differences between the classes at the image level are not known.

Although our setting of the problem does not tell the reader anything about the relationship between the images and the physical quality properties of the paper, it is believed that this example demonstrates the methodological aspects of the work well. The real world images of this case were provided by Metso Paper, Inc., a member of the Metso Corporation.

2.4.2 The image analysis

The image analysis is divided into two phases. The first stage calculates measures of texture properties (features) and forms a high-dimensional feature vector of them (Figure 5). The second stage evaluates the vector components with respect to their discriminative power and reduces the dimension of the vector. Also linguistic if–then rules will be formed (Figure 6) in this work to give the user an idea of how the system makes the decision about the class of the sample vector. Depending on the actual application, the result can be used to achieve a better understanding of the data, to classify different textures or to find out the function between the image and the material properties of the physical paper (strength, tension, etc.). For related work in other applications see, for example Greenspan et al. (1994) and Ramze Rezaee et al. (1999).

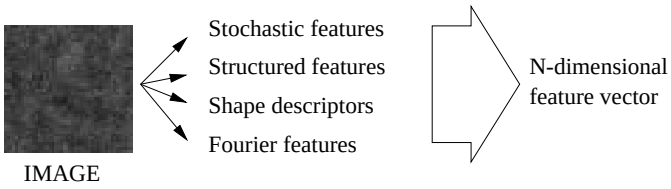

FIGURE 5 Feature extraction step.

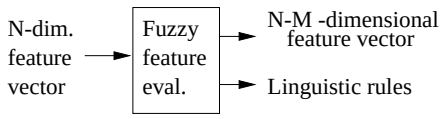

FIGURE 6 From features to rules.

The actual data consists of dozens of calculated features. In the current work 48 features have been used. Our idea is that the user does not have to specify *a priori* what type of measures should be used. Rather the selection procedure should only be based on the discriminative power of the features.

2.4.3 The features

The images of Figure 4 are typical realizations of a random process (Lantuéjoul, 2002), which in theory can be characterized by the spatial covariance function. Unfortunately we cannot assume any functional form for the covariance, and the Fourier representation of its nonparametric shape is not very useful for data analysis. Therefore we decided to use simple features that might (or might not) characterize these images well.

Features are similar to those that one can find from the basic image analysis textbooks (Gonzalez and Woods, 1992; Jain, 1989; Sonka et al., 1996). From our perspective there are four categories about them:

- stochastic co-occurence matrix based features,
- Fourier features,
- features of the spatial distribution of shape descriptors calculated from thresholded and segmented images,
- structured features of random graphs, which are not used in this work.

Stochastic and Fourier features work directly with the random process, while the shape descriptors and structured features are based on preprocessed data. For example, for the shapes that are obtained via thresholding, we assume that the underlying stochastic process is visible as a random set or a boolean process (see Lantuéjoul, 2002; Stoyan et al., 1995).

The reason for such a large number of features is that there are many causes for observed changes in paper formation. For example, Fourier analysis reveals periodic patterns caused by rotating machinery, while stochastic and geometric distributions may characterize how fluid flow behaves in the wet end of the paper machine.

In this work we use only shape descriptors (blob features) and Fourier features, because in various experimental test runs they seemed to provide better characterization of fiber images than the other types of feature sets, containing more conventional stochastic co-occurence matrix based features and fractal dimension. A more thorough explanation of the features used in this work is given in the next subsections.

Fourier features

A spectral approach brings a new viewpoint to the analysis when compared to spatial methods. The Fourier spectrum is ideal for describing the directionality of periodic or almost periodic patterns in image (Gonzalez and Woods, 1992). Fourier features are obtained by forming 1-D angular and radial power spectrums

from the 2-D FFT-spectrum of the image. The final features are signal energies of sub-bands of 1-D spectrums selected by using user supplied fuzzy membership functions.

The formal description of the approach is as follows. First the image is transformed with 2-D FFT into a frequency domain

$$F(u,v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \mathcal{I}(x,y) e^{-2\pi i(\frac{ux}{M}, \frac{vy}{N})},$$

where M and N are the x and y resolutions of the image. The power spectra of the image is

$$|F(u,v)| = \sqrt{[\mathrm{Re}\{F(u,v)\}]^2 + [\mathrm{Im}\{F(u,v)\}]^2},$$

which is visualized for a typical β-radiogram image in Figure 7.

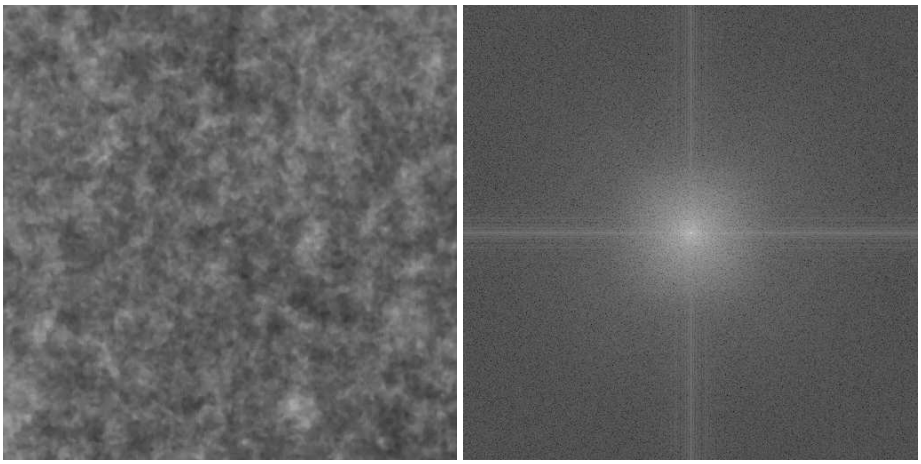

FIGURE 7 β-radiogram and the corresponding 2-D FFT that is aligned such that the lowest frequencies are in the centre of the image.

The radial V_r and angular V_θ spectrums are obtained as marginal distributions from the 2-D FFT such that

$$V_\theta = \sum_{r=0}^{R} |F(r\cos\theta, r\sin\theta)|,$$

$$V_r = \sum_{\theta=0}^{2\pi} |F(r\cos\theta, r\sin\theta)|,$$

where R is the maximum radius and 2π corresponds to the maximum angle (360 degrees). These are illustrated in Figure 8. The corresponding fuzzified features are obtained as

$$\tilde{x}_i^{rf} = \int V_r \mu_{\tilde{A}_{r,i}}(r)\, dr$$

and

$$\tilde{x}_i^{\theta f} = \int V_\theta \mu_{\tilde{A}_{\theta,i}}(\theta)\, d\theta.$$

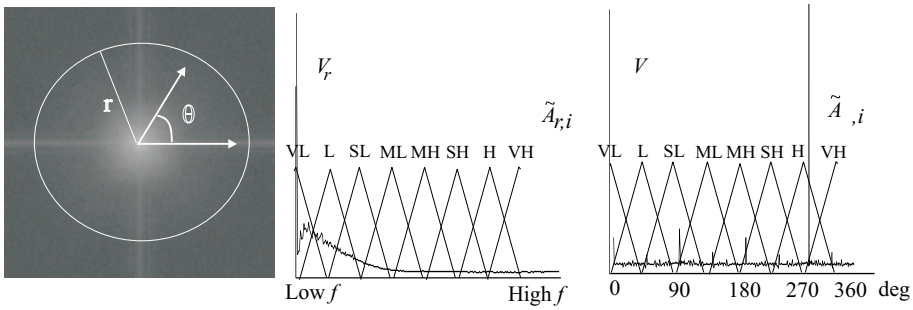

FIGURE 8 Angular and radial Fourier spectrums for image 7 and the corresponding fuzzy labels: VL = very low, L = low, SL = slightly low, ML = medium low, MH = medium high, SH = slightly high, H = high, and VH = very high.

Blob features

The third group of features consists of shape descriptors that require a binary image. Once the gray-level image has been thresholded to binary values, it is segmented to achieve a set of blobs. This can cause some extra degree of difficulty, since the threshold must be chosen *a priori* and different thresholds yield different kinds of distributions. Usually, the threshold is set to be the median of the gray-level histogram (Kajanto et al., 1989; Trepanier et al., 1998). In contrast, in our approach the threshold is selected using the example images by maximizing the discriminative power of the blob shapes. Formally the blob extraction procedure is as follows.

0. Remove noise, e.g. via median filtering with 3×3 window.

1. Using a threshold level τ^*, mark the pixels of the image $\mathcal{I}(x, y)$ either to black or white. Let this binary image be denoted by

$$B(x, y) = \begin{cases} 1, & \text{if } \mathcal{I}(x, y) > \tau^*, \\ 0, & \text{otherwise.} \end{cases}$$

2. Segment the image into blobs $\omega_i^{\text{blob}} = \{(x, y)|i, B(x, y) = 1\} \in \Omega^{\text{blobs}}$ such that all (x, y) points in the blob i are connected in the 4-neighborhood:

$$\text{If } (x, y) \in \omega_i^{\text{blob}} \text{ then also all } (x', y') \in \omega_i^{\text{blob}},$$

where

$$(x' = x{+}1,\, y' = y) \vee (x' = x{-}1,\, y' = y) \vee (x' = x,\, y' = y{+}1) \vee (x' = x,\, y' = y{-}1)$$

TABLE 2 Blob shape measurements, where function $h(a)$ is the gray-level $\mathcal{I}(x, y)$ at spatial point $a = (x, y)$ and A denotes the region covered by all points $\{(x, y)|i, B(x, y) = 1\}$ of the blob.

Name	Definition
Area	$are = \int_{a \in A} da$
Perimeter	$per = \int \sqrt{x^2(t) + y^2(t)}dt$ where t parameterizes the border line.
Volume	$vol = \int_{a \in A}(h(a) - h_{\min})da$
Roundness	$rou = \frac{per^2}{4\pi are}$
Radii	$rad = \frac{r_{max}}{r_{\min}}$, where $r_{\max}$ and $r_{\min}$ are the maximum and minimum diameters of the blob, respectively.
Eccentricity	$ecc = \frac{(\mu_{2,0}-\mu_{0,2})^2+4\mu_{1,1}}{are}$, where $\mu_{p,q}$ are (p, q)-degree central moments (see Jain, 1989).
Steepness	$ste = \frac{1}{are}\int_{a \in A} \arctan(\frac{h(c)-h(a)}{\|\|c-a\|\|})da$, where c is the point with lightest gray level in blob.
Orientation	$ori = \frac{1}{2}\arctan(\frac{2\mu_{1,1}}{\mu_{2,0}-\mu_{0,2}})$

and

$$\omega_{i_1}^{\text{blob}} \cap \omega_{i_2}^{\text{blob}} = \{\}, \qquad \text{if } i_1 \neq i_2, \text{ and}$$
$$\bigcup_i \omega_i^{\text{blob}} = \{(x, y)|B(x, y) = 1\}.$$

This is easy to implement with a standard line scan algorithm (Jain, 1989).

3. Extract shape measures from all the blobs. The measures that were used here are discrete versions of those that are presented in Table 2. For more information see Ohser and Mücklich (2000). This gives us a set of feature observations

$X^{\mathrm{fea}}(i|j)$ inside each image $\mathcal{I}_j(x,y)$:

$$\mathbb{X}(j) = \begin{bmatrix} X^{\mathrm{Area}}(1|j), & X^{\mathrm{Area}}(2|j), & \dots, & X^{\mathrm{Area}}(N_j|j) \\ X^{\mathrm{Per}}(1|j), & X^{\mathrm{Per}}(2|j), & \dots, & X^{\mathrm{Per}}(N_j|j) \\ \vdots & \vdots & & \vdots \\ X^{\mathrm{Ori}}(1|j), & X^{\mathrm{Ori}}(2|j), & \dots, & X^{\mathrm{Ori}}(N_j|j) \end{bmatrix},$$

where N_j is the number of blobs inside the image.

4. The final features of the image $\mathcal{I}_j(x,y)$ are extracted from the blob-shape distribution of the image. Thus they are features of the distribution, like

$$f_{X^{\mathrm{Area}}}(x^{\mathrm{Area}}|j).$$

In most cases the features (moments) can be estimated directly from the image as explained in the following. The features that are used in this thesis are

expectation: $X^{\mathrm{fea\_mea}}(j) = \frac{1}{N_j}\sum_i X^{\mathrm{fea}}(i|j) = \bar{x}^{\mathrm{fea}}$,

variance: $X^{\mathrm{fea\_var}}(j) = \frac{1}{N_j - 1}\sum_i (\bar{x}^{\mathrm{fea}} - X^{\mathrm{fea}}(i|j))$,

skewness: $X^{\mathrm{fea\_ske}}(j) = \frac{\sum_i (\bar{x}^{\mathrm{fea}} - X^{\mathrm{fea}}(i|j))^3}{N_j (X^{\mathrm{fea\_var}}(j))^{3/2}}$,

kurtosis: $X^{\mathrm{fea\_kur}}(j) = \frac{\sum_i (\bar{x}^{\mathrm{fea}} - X^{\mathrm{fea}}(i|j))^4}{N_j (X^{\mathrm{fea\_var}}(j))^2} - 3$,

and **entropy**: $X^{\mathrm{fea\_ent}}(j) = -\int f_{X^{\mathrm{fea}}}(x^{\mathrm{fea}}|j) \log f_{X^{\mathrm{fea}}}(x^{\mathrm{fea}}|j)\, dx^{\mathrm{fea}}$,

where $f_{X^{\mathrm{fea}}}(x^{\mathrm{fea}}|j)$ is the estimated distribution of blob shapes. Typically kernel or nearest neighbour estimators are used for this. In this thesis, however, a simple histogram estimation and corresponding discrete implementation $-\sum p_j \log p_j$ of entropy was found reliable enough. The whole blob-shape-distribution-feature process is illustrated in Figure 9

2.4.4 Other approach to fiber data

One approach which is not considered in this work is the estimation of random functions that characterize fiber processes. The methodology offers tools for describing the statistical and stereological properties of spatial random models. For example, Lantuéjoul (2002) introduces a wide range of spatial models, which consist of random sets and functions, and point processes, which could be applicable not only to geostatistical estimation but also to fiber processes. We know that there is this kind of random function behind the images, and it is often quite simple in mathematical terms (defined by a couple of parameters). The problem with this approach is that the processes are hard to interpret, and the obtained process model does not directly indicate the final quality of the product. The current approach is more user-friendly.

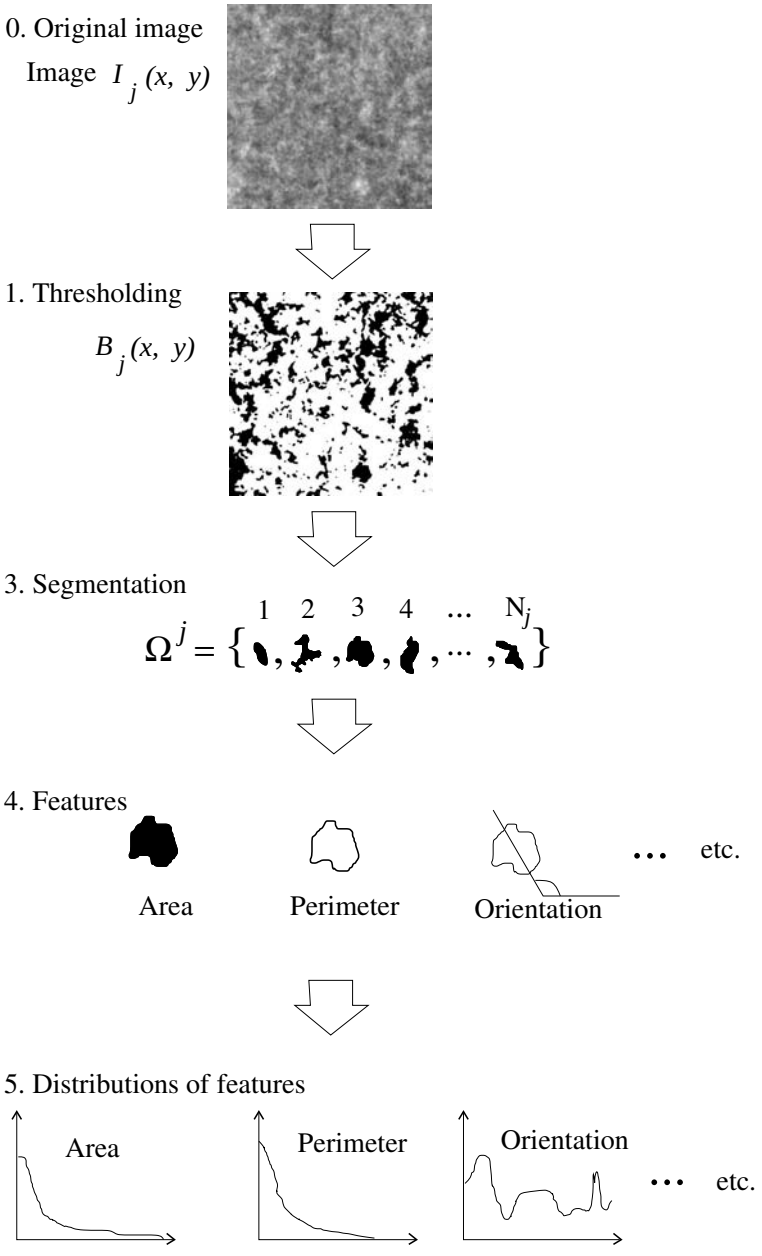

FIGURE 9 The blob feature extraction process.

2.5 Other data sets

Some well-known and publicly available data sets are used in this thesis. Numerous references to these data sets can be found in literature. The common aspects

of these data sets are that they are numeric and do not have missing attribute values. The lack of missing feature values was one criterion when selecting the data sets, since we do not consider that problem in this theses. In our methodology it is supposed that the missing value problem is solved in pre-processing of the data.

2.5.1 Fisher's iris plants database

Fisher's iris data (Fisher, 1936) is a well-known and highly referenced database. It contains 150 data points consisting of an equal number (50 in each class) of examples from three iris flower species classes; setosa, versicolor and virginica. Data points are measurements of sepal length, sepal width, petal length and petal width. Setosa can be easily separated from the other classes while the two other classes, versicolor and virginica, overlap each other. The predicted attribute is the class of iris plant. The data set was obtained from UCI Machine Learning Repository. The classification results reported are usually between 90–99%.

2.5.2 Sonar data

Sonar data set (obtained from UCI Machine Learning Repository) contains measurements of sonar signals bounced off a metal cylinder and a rough cylindrical rock. This data set was used by Gorman and Sejnowski (1988) in their study of the classification of sonar signals using a neural network. The set contains 111 patterns for the metal cylinder and 97 patterns for the rocks. The data set has been formed by bouncing sonar signals (frequency modulated chirps, which rise in frequency) off the objects at various angles.

Each pattern consists of 60 features and each feature represents the energy within a particular frequency band. The values are in the range 0.0 to 1.0. Gorman and Sejnowski (1988) reported classification results that varied between 73.1 and 90.4%, depending on the network structure and the way to divide the data into training and test sets.

2.5.3 Vowel data

A vowel data set consists of measurements of vowels in British English. The thorough description of the data and several references can be found from UCI Machine Learning Repository site. The data set contains 11 classes which correspond to 11 vowel sounds (i:, E, a:, O, U, 3:, I, A, Y, C: and u:), each contained in 11 different words (heed, head, hard, hod, hood, heard, hid, had, hud, hoard and who'd). The vowels were spoken by fifteen speakers and each of them said the vowels six times. Each sample consists of 10 floating-point values. Thus the total number of samples is $15 \cdot 6 \cdot 11 = 990$.

The original problem was to train a neural network as well as possible only using data from the training set, and then to test the network on a separate test set. 528 of the samples were used for training and the remaining 462 for the testing. The classification results (the per cent of test vowels classified correctly) range

30% to 60%, depending on the classifier. The low percentage values indicate that a vowel data set corresponds to a hard example problem. Hastie et al. (2001) have also used this data in their experiments. They show results for a wide variety of techniques.

2.5.4 Wine data

The wine data set contains the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. These three cultivars correspond to the classes which must be predicted using 13 continuous features. The features are quantities of constituents found in each of the three types of wines. The 13 features are: alcohol, malic acid, ash, alcanity of ash, magnesium, total phenols, flavanoids, nonflavanoid phenols, proanthocyanins, color intensity, hue, OD280/OD315 of diluted wines and proline. The number of instances are:

- Class 1: 59,
- Class 2: 71,
- Class 3: 48.

The data set has been used by many researchers and it was submitted to UCI Machine Learning Repository by Stefan Aeberhard (for a comparison of classifiers see Aeberhard et al., 1992). Classification percentages of 95–100 are typical results.

2.5.5 Glass data

Glass data set (from UCI Machine Learning Repository) has 214 instances and 9 continuously valued features. The origin of the data is USA Forensic Science Service and was created by B. German (more information with contact addresses can be found from UCI Machine Learning Repository). The problem is to identify different glasses. The study of classification of glasses was motivated by criminological investigation, since the glass left at the scene of the crime can be used as evidence. The features are refractive index, sodium, magnesium, aluminum, silicon, potassium, calcium, barium and iron. There are six classes (actually seven, but one of the classes has no samples). The classes are

- building windows: float processed,
- building windows: non float processed,
- vehicle windows: float processed,
- vehicle windows: non float processed (none in this database),
- containers,
- tableware,

- headlamps.

Often this data has been used to classify the glasses either to float processed or non float processed glasses. The data description file associated with the data set file tells that in this case the classification error rates obtained were, depending on the method, between 10% and 22%. In this work we try to classify each sample to one of the original 6 classes.

2.6 Conclusions

We started this chapter by considering an important question: "How to evaluate the developed methodology?" It was noted that there are two possibilities, an analytical and an empirical approach. The choice of empirical approach was then explained.

Our main example data set that consists of numerous features computed from fiber images was described next. Also research questions regarding this data were stated. The most convenient for this work is the question: "What are the best or sufficient features from each class of observation, given a set of images that represent different classes?" A detailed explanation of the different features was given.

Other data sets used in this thesis were also described. The data sets which are widely known and easily available were selected from the wide data repository.

3 SHORT REVIEW OF VARIABLE SELECTION AND MODELING

As discussed in the previous chapter, the developed methodology has been evaluated using empirical testing. In this chapter the comparative (or reference) methods are introduced.

In the case of high dimensional problems, dimensionality reduction can often improve the performance of the chosen model. In addition, it allows one to focus on the most important variables of the data sets. To achieve the objectives of good performance and easy interpretation, we shall study the problem of a feature selection. Before explaining the comparative methodologies, we shall have an overview of the general problem, where the main question is

Given d variables what is the best subset $p \leq d$ that describes the data well enough?

Datasets with tens or hundreds of thousands of variables is not unusual, for example in gene analysis, text categorization and chemistry. In such problems the feature selection has many benefits:

- Prediction accuracy of the model may be improved.
- Smaller subset of features is easier to interpret than the original large set.
- Models may be made faster.

The first two points are interesting from a fuzzy logic viewpoint, since the feature selection promises to improve both accuracy and interpretability.

Feature selection is a challenging problem. Especially if we need to do it without any *a priori* knowledge. There are many non-obvious decisions. For example, we must decide which criterion to use for determining whether some feature is important or not.

3.1 Taxonomy of the problem

In the following the problem considered consists of three parts: the model, the performance criterion and the search strategy (see Figure 10).

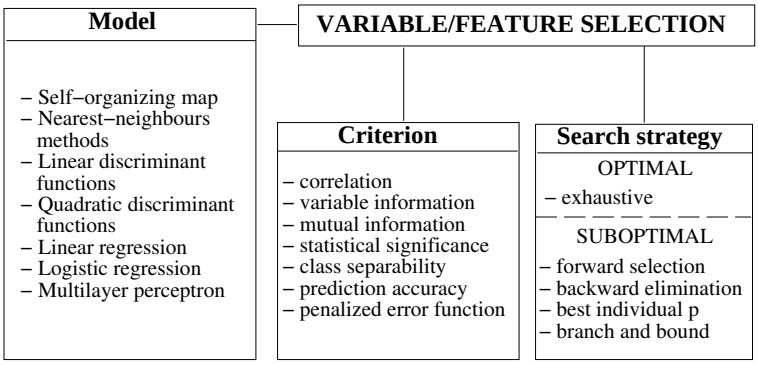

FIGURE 10 The taxonomy of variable/feature selection problem.

Let us assume that we have some model selected from the list on the left in Figure 10, and we want to increase the performance of the model by pruning the redundant or by selecting the most important features. We have several options for the criterion, which gives an estimate of which features should be selected or discarded for the model.

The choice of criterion may depend on the model. For example, the penalized error function can be most easily applied to linear or logistic regression, and multilayer perceptron. Prediction accuracy is the most general of the criteria since it can be applied without any complications to any model. Class separability criterion is suitable for classification tasks or clustering type of models. Statistical significance criterion is often combined with linear regression.

The relationship between performance criterion and search strategy is more obvious than the relationship between model and criterion. The type of criterion may have a direct effect on the search strategy. For example, if the criterion assumes that variables are independent, then a very simple search strategy can be used. If the criterion is computed over the whole data set, then the search may need a complex optimization algorithm (e.g. a genetic or a gradient based search algorithm).

Besides the criterion the choice of search strategy depends on the actual task, which we would like to carry out. If the task is to prune redundant features, backward elimination might be more suitable than forward selection. In general, different models, criteria and search strategies can be combined quite freely. However, it is unrealistic to assume that any chosen combination could work; some combinations are better than others (depends on the actual application). Without

an extensive study it is difficult to judge which combinations are generally the most effective.

3.1.1 Variables or features

Before giving the descriptions of the parts, shown in Figure 10, we define some central terms. **Variable** refers to the original input variable, which has not been changed in any way. It consists of direct measurements of some object(s) or event(s). **Features** are new variables which have been constructed from the original variables, e.g., by pre-processing. The selection algorithms can be applied to both types of problems, which explains why the terms feature and variable are often used without distinction.

Variables are usually easier to interpret than features, especially if the features are complex combinations of the original variables. The use of features is justified if they can extract useful and simple information from complex real world patterns, like digital images. If full interpretation is not required, then any approach is justifiable if it leads, for example, to improvement in class separability.

This is also the assumption in this thesis. We do not claim that automatic variable selection can replace human intuition. Rather it is a tool that helps us to work with features that are hard to interpret and select by human reasoning. This is clearly an engineering point of view.

For the above mentioned reasons, in the following we prefer to use the term feature, although term variable could be used as well.

3.2 Criterion

To carry out the selection, a criterion J is required to determine which feature subset is better. Analyzing the nature of the criterion we may also try to understand why some specific features are considered more important than others.

If possible, the criterion J is optimized over the set of all subsets of features. Let $\mathcal{X}$ be the power set of d features (the set of all possible subsets). A set of features $\mathbf{x}^*$ (an element of $\mathcal{X}$) is sought for which

$$J(\mathbf{x}^*) = \max_{\mathbf{x}' \in \mathcal{X}} J(\mathbf{x}'),$$

where the higher value of J indicates a better feature subset. The criterion usually measures the capability of the subset in regression, classification or clustering. Depending on the type of criterion the selection may be supervised or unsupervised.

In the following, only some commonly used criteria will shortly be introduced:

- prediction accuracy,
- correlation criterion,

- information theoretic criterion,
- class separability,
- statistical signifigance,
- penalized cost function.

Most of the criteria is based on some type of dependency between features x_i and some response variable y, which in our case is usually the classification accuracy.

In the simplest case, we may assume that features are independent, which allows us to evaluate the importance of each feature x_i separately with partial criteria $J(x_i)$. If the assumption is not true, we have to face new problems like the collinearity, where two or more variables (features) $x_{i_1}, x_{i_2}, \ldots$ show strong dependency between the response y, but do not bring any new information for the system. Therefore most of the criteria tries to minimize dependencies between the features x_i while maximizing their dependency with y.

In summary, the criteria may be categorized as

1. $J(x_i)$ (evaluates only single features),
2. $J(x_i|\mathbf{x})$ (evaluates single features but takes into account dependencies between the features),
3. $J(\mathbf{x}' \subset \mathbf{x})$ (true multivariate criteria).

Correlation, mutual information criterion, statistical significance and receiver operating characteristic curve (see Theodoridis and Koutroumbas, 2003) are examples of category 1. Prediction accuracy, class separability and penalized cost function are true multivariate criteria (category 3). The categorization is not distinct, since there are examples of information theoretic criteria which are multivariate, and multivariate criteria which are often applied to individual features. In the following, we do not have direct examples from category 2, but these can be formed easily from multivariate criteria. One should also note that most single feature criteria is checked against variable dependencies by human made data analysis. This also implements criteria 2.

3.2.1 Prediction accuracy of features

Among the simplest ideas is to build a predictor or a classifier with a single feature. The *predictive power* of the feature can be expressed as an error rate (or misclassification rate). The problem of this approach is that it assumes the full independence of variables, which is often in conflict with modeling criteria.

A natural generalization of the idea is to use subsets of feature combinations and then see how the model works with respect to *prediction accuracy*. Due to the fact that this is a computationally heavy approach, it is often impractical. Especially if we need to use computationally intensive techniques such as the bootstrap

for the estimation of true prediction power. Often a different choice of model gives different feature sets, i.e. the features that best suit the chosen model are selected.

3.2.2 Correlation criterion

The estimate of Pearson correlation coefficient

$$J(x_i) = \frac{\sum_{j=1}^{N}(X_i(j) - \bar{x}_i)(Y(j) - \bar{y})}{\sqrt{\sum_{j=1}^{N}(X_i(j) - \bar{x}_i)^2 \sum_{j=1}^{N}(Y(j) - \bar{y})^2}}$$

may be used to construct a feature ranking criterion for feature x_i. The notations $\bar{x}_i$ and $\bar{y}$ stand for the averages of the feature x_i and y. The problem with the correlation criterion is that it assumes linear independency between variables x_i. Therefore the problem of collinearity is quite severe with this method.

3.2.3 Information theoretic criterion

Information theoretic criteria is based on entropies and can be used to measure the information shared by different features. *Mutual information* (see Cover and Thomas, 1991) between the input feature x_i and the target variable y may be obtained by

$$J(x_i) = I(x_i, y) = \int_{x_i}\int_{y} f_{X,Y}(x_i, y)\log\frac{f_{X,Y}(x_i, y)}{f_X(x_i)f_Y(y)}\, dx_i dy,$$

where $f_{X,Y}(x_i, y)$ is the estimated joint probability density, and $f_X(x_i)$ and $f_Y(y)$ are the probability densities of x_i and y. If x_i and y are statistically independent, $I(x_i, y)$ becomes zero. If x_i is deterministically related to y, $I(x_i, y)$ gives a large value.

Feature selection based on correlation or mutual information between the candidate feature and the target does not take into account the correlations between the candidate features. Bekkermann et al. (2003) use a multivariate variation of mutual information criterion, where the redundancy can be avoided by maximizing

$$J(\mathbf{x}') = I(\mathbf{x}', y) - \beta I(\mathbf{x}', \mathbf{x}),$$

where β is a coefficient (Lagrange multiplier) that determines how much the mutual information between the original feature vector $\mathbf{x}$ and the subset of features $\mathbf{x}'$ affects the selection criterion. This type of criterion is known as *Information bottleneck* (Tishby et al., 1999).

3.2.4 Class separability

In class separability criterion one measures the discrimination power J^{class} of feature vectors. The separability measure may be seen as a computational shortcut

for suboptimal feature selection (Ripley, 1996). To apply these criteria to the feature selection, one must use them with different feature sets $\mathbf{x}'$ and look for the the set $\mathbf{x}^*$ with best performance

$$J(\mathbf{x}^*) = \max_{\mathbf{x}'} J^{\text{class}}(\mathbf{x}').$$

This may require the use of resampling methods like the bootstrap, because the overlearning problem is quite severe with sparse, high dimensional data sets.

A good example is the *divergence*

$$d_{kl} = D_{kl} + D_{lk} = \int (f(\mathbf{x}|C_k) - f(\mathbf{x}|C_l)) \log \frac{f(\mathbf{x}|C_k)}{f(\mathbf{x}|C_l)} d\mathbf{x},$$

where

$$D_{ij} = \int f(\mathbf{x}|C_i) \log \frac{f(\mathbf{x}|C_i)}{f(\mathbf{x}|C_j)} d\mathbf{x},$$

and $f(\mathbf{x}|C_r)$ are estimated density functions for classes C_r, can be used as a separability measure for the classes C_k and C_l. It must be computed for every class pair, and some square-integrable model $f(\mathbf{x}|C_r)$ must be estimated for each class. The average class separability may be obtained by

$$J^{\text{class}} = \sum_k \sum_l P(C_k) P(C_l) d_{kl},$$

where $P(C_k)$ and $P(C_l)$ are the class prior probabilities. Divergence is related to the Kullback-Leibler (KL) distance, which is measured between two probability density functions. Unlike the KL distance, divergence is symmetric. Divergence depends explicitly on both the difference of the class means and the respective variances (for more details see Theodoridis and Koutroumbas, 2003). Another example of class separability is the widely used Bhattacharyya distance (see, for example, Theodoridis and Koutroumbas, 2003; Ripley, 1996).

Perhaps the most known class separability criteria is based on scatter matrices, which bear the information about how the data classes are scattered over the input space. Often these criteria utilize covariance matrices in some form. An example of such criteria is

$$J^{\text{class}} = \frac{\text{trace}\{\mathbb{C}_m\}}{\text{trace}\{\mathbb{C}_w\}},$$

where matrix $\mathbb{C}_m = \mathbb{C}_w + \mathbb{C}_b$. $\mathbb{C}_w$ is a within-class covariance matrix and $\mathbb{C}_b$ is a between-class covariance matrix. A number of different criteria may be formed by combining these matrices. The famous Fisher's discriminant uses two types of covariance matrices:

$$J^{\text{Fisher}}(\mathbf{w}) = \frac{\mathbf{w}^{\text{T}} \mathbb{C}_b \mathbf{w}}{\mathbf{w}^{\text{T}} \mathbb{C}_w \mathbf{w}},$$

where the direction of the best separability $\mathbf{w}$ is a vector of adjustable weight parameters. The final separability J^{class} is then computed along direction $\mathbf{w}$ using some simple variable criterion. The characteristics of these criteria is that they assume that the shape of distribution is elliptic.

3.2.5 Statistical significance

Statistical significance criteria is based on statistical hypothesis testing. Usually these techniques consider the features independently and it is the role of the user to eliminate strongly dependent variables. The idea is to test if two sets of observations come from two different populations. The assumption that the values of the feature does not differ significantly is taken to be the *null hypothesis* H_0. The reverse hypothesis is known as an alternative hypothesis H_1. Then experimental evidence is used to find either support or reason to reject the null hypothesis.

Statistical significance can be used e.g. with forward selection types of techniques, where the aim of the test is to show if an added variable improves the result. Hastie et al. (2001) use F statistic, which can be computed for linear regression as

$$F = \frac{\mathrm{SSE}_0 - \mathrm{SSE}_1}{\mathrm{SSE}_1/(N-p-2)},$$

where N is the number of samples, p is the number of features in current model, SSE_0 is the sum of squared errors when using the current set of features and SSE_1 is the sum of squared errors of the bigger model (with more features). If adding a new feature does not produce an F-ratio greater than, e.g. 0.95 of the F-distribution, then the feature is discarded.

3.2.6 Penalized cost function

Some methods weight features but do not select a subset of them. For example, ridge regression shrinks the effect of unimportant features by imposing a penalty on their size. The idea of penalizing the sum-of-squares error is also used in neural networks and it is known as *weight decay*. The criterion may be written as

$$E(\{\theta_i\}) = \sum_{j=1}^{N} (Y(j) - \hat{f}(\mathbf{X}(j)|\{\theta_i\}))^2 + \lambda \sum_{i=1}^{d} \theta_i^2,$$

where $\hat{f}$ is the model, $\{\theta_i\}$, $i = 1, \ldots, d$, is a set of parameters, that weight features and $\lambda \geq 0$ is a coefficient that controls how much the parameters are shrunk toward zero. If the relationship between parameters θ_i and features x_i is expressed via linear dependency

$$\hat{f}(\mathbf{x}|\{\theta_i\}) = \sum_i \theta_i x_i,$$

then we may interpret ridge regression as a variable selection method. Note also that the sum-of-squares error $E(\{\theta_i\})$ is minimized, not maximized. However, minimization problem is easy to convert to the maximization problem by

$$J(\{\theta_i\}) = -E(\{\theta_i\}).$$

The *least absolute shrinkage and selection operator* (lasso) method (see Tibshirani, 1996; Hastie et al., 2001) resembles ridge, but its penalty term

$$\lambda \sum_{i=1}^{d} |\theta_i|$$

forces some of the parameters to be exactly zero. Thus the lasso is even better for variable subset selection, because the parameters approach smoothly zero as the penalty is increased. We may then interpret that the most important variables x_i are those with large non-zero weights θ_i.

3.3 Search strategies

Once we have a number of criteria for evaluating individual features or feature sets, the next problem is

How to do the search of the feature sets effectively?

As the criteria $J(x_i)$ and $J(\mathbf{x}')$ already suggest there are two general options to do the search: features are treated individually, or they are treated as vectors. Treating features individually is simple and computationally effective, but it is often inadequate for complex problems.

First we consider the simple "best individual p" method which is an example of scalar search strategies. Then multivariate strategies, which accept feature vectors are reviewed, starting from optimal and ending in suboptimal strategies.

3.3.1 Best individual p

The best individual p selection (or rank-order-based selection) (see e.g. Schürmann, 1996; Webb, 1999) is the simplest suboptimal scalar search strategy. As the name tells, the features are treated individually. The value of the criterion $J(x_i)$ is computed for each feature $i = 1, \ldots, d$ and the features are then ranked according to the criterion and sorted in a decreasing order. The best p features are selected to form the final set of features. *Variable ranking* or *filtering* is independent of the choice of model (fuzzy logic system, MLP, etc.), but it forces one to use them in a univariate way: one model $f(x_i)$ for each variable x_i. Treating features individually is computationally simple, but it is not accurate if there are dependencies between the features. However, if the features are reasonably independent, the method may produce reasonable subsets.

3.3.2 Exhaustive search for p

Exhaustive search for the best p features is an optimal strategy. Generally, it is the only technique which can be guaranteed to produce an optimal feature set (with

respect to criterion). All vector combinations for p out of the d original features are formed. For each combination some criterion (e.g. separability criterion or prediction accuracy) is used, and the best feature vector combination according to this criterion is selected. The disadvantage of the approach is that

$$\left(\begin{array}{c} d \\ p \end{array} \right) = \frac{d}{p!(d-p)!}$$

evaluations must be carried out. Also, we often do not have any prior information about p, and therefore we must carry out the search using different values for p. Thus, the computational complexity is the limiting factor.

An exhaustive search is suitable only if p is relatively small, say $p < 10$, but despite these limitations it is a good alternative for finding small sets of features. For example, Jain et al. (1999) wrote:

It has been argued that since feature selection is typically done in an off-line manner, the execution time of a particular algorithm is not as critical as the optimality of the feature subset it generates.

They continue that this is true for feature sets of moderate size. However, in large problems suboptimal heuristics may be the only realistic option.

From a rule extraction point of view exhaustive search is interesting. It can produce very efficient sets of features (and thus rules), provided that the criterion function peaks or achieves a high value with small feature subset size.

3.3.3 Branch-and-bound

Branch-and-bound techniques (Narendra and Fukunaga, 1977; Fukunaga, 1990) expect a monotonic criterion function. It is a top-down procedure that begins with d features and forms a tree where each level has one feature less than the previous level. Monotonic criterion states that

$$\mathbf{x}' \subset \mathbf{x} \Rightarrow J(\mathbf{x}') < J(\mathbf{x}),$$

where $\mathbf{x}$ and $\mathbf{x}'$ are subsets of features and J is the criterion function. This guarantees that the procedure finds the subset of a given size without needing to evaluate all the subsets. While maximizing $J(\mathbf{x}')$, if any node has the criterion smaller than the best one found so far, there is no need to evaluate the nodes below this node. So the tree can be easily pruned. The problem with the monotonicity requirement is that it is difficult to guarantee with real world modeling, since a smaller number of features may outperform the larger set.

3.3.4 Stochastic search

Monte Carlo methods, which rely on stochastic search can sometimes lead to a globally optimal solution. Genetic algorithms (GA) introduced by Holland in 1975

(see Holland, 1995) and stochastic simulated annealing (SA) (Ackley et al., 1985) are examples of Monte Carlo methods. By incorporating the number of features into the fitness function (criterion) the genetic selection may be made to provide smaller feature (and rule) sets. This can be accomplished, for example, by using a fitness function

$$J(\mathbf{x}) = -(\mathrm{SSE}(\mathbf{x}) + \lambda h(p)),$$

where $\mathbf{x}$ is the subset with p features, SSE is the sum of squred errors between the predictor output and the target, λ controls the trade-off between error and model complexity, and $h(\cdot)$ is some function of a number of features. Both GA and SA are expensive from a computational standpoint. Despite this, they have been successfully applied to many polynomial complete (NP-complete) problems. Ferri et al. (1994) show by using experimental results that GA produces reasonable results when the selection problem is small or of moderate size (less than about 50 features). As the dimensionality increases the GA performs worse on average. Very contradictory results regarding the comparison of GA with other methods (sequential search strategies) can be found from literature. Different from Ferri et al., Kudo and Sklansky (2000) recommend the GA for large-dimensional problems. According to Kudo and Sklansky GA sometimes found better solutions that the best sequential floating search strategies could not find. They also found that GA becomes faster than the competitors when the dimensionality increases.

There are two problems with the GA approach. First, many GA algorithms expect that the problem can be divided into independent subproblems, which leads to suboptimal solutions. Secondly, there are many variations and parameters of the general idea. Therefore, a trial and error type of testing must be carried out to make the algorithm work satisfactorily.

3.3.5 Sequential search strategies

Sequential forward selection (SFS) (see, for example, Devijver and Kittler, 1982) does a bottom-up search. It adds new features to a feature set in a stepwise manner. That is, it finds the best feature, one at a time, that improves the performance (according to criterion) the most, and adds it to the set of selected features. SFS is suboptimal because it is a typical greedy algorithm. Many good features may be discarded because of the earlier feature choices.

Sequential backward elimination (SBE) (see, for example, Devijver and Kittler, 1982) proceeds in the opposite direction. Like branch-and-bound it begins with the set of all features. At each stage the feature which reduces the performance criterion the least is the one deleted from the current set of features. Although backward elimination can find a good set, it is also a greedy algorithm, not an optimal strategy. It requires more evaluations than the SFS. Guyon and Elisseeff (2003) argue that the SFS is often claimed to be more efficient than SBE. They show by an example that this is not always the case. Kudo and Sklansky (2000) claim that "*Usually backward algorithms are better than their counterparts*." They, however, admit that the supremacy depends on the specific problem.

In addition to pure forward selection and backward elimination strategies, combinations of these have been proposed. Often the combined algorithms require a threshold parameter to decide whether to use forward (add feature) or backward step (delete feature) at each stage. *Plus l–take away r* method adds l features to the current set by using SFS, and then removes the worst r features by using SBE. *Floating search* methods provide a variation to this strategy. The values of l and r are allowed to change (float) at different stages of the selection (Pudil et al., 1994). The floating search also works well with nonmonotonic search criteria, which is not true for SFS and SBE. This comes, naturally, with the cost of higher computational complexity.

Jain and Zongker (1997) infer, relying on experiments, that the sequential forward floating selection (SFFS), proposed by (Pudil et al., 1994), dominates SFS, SBE and their generalizations. Ferri et al. (1994) compared SFS, SFFS and GA with data sets where the number of features was 100 at maximum. Their results show that SFFS works the best, GA the second best and the SFS the worst on average. They also tested the methods on a data set whose optimal recognition rate was known for each subset size. Both SFFS and GA could obtain recognition rates very close to the optimal rates. However, in that particular test the number of features was "only" 20.

3.3.6 Optimization in general

Almost all optimization techniques can be used in feature selection. For example, in ridge regression and lasso the search is carried out by linear optimization or quadratic programming, respectively. The objective function may consist of a term, which represents the goodness of fit, while another term tries to minimize the effective number of features. With criteria like ridge and lasso, this moves the weights of the features toward zero. If the solution is found, where one or more of the weights become zero, the procedure has accomplished feature selection (or actually: feature pruning). The problem with some penalization methods is that while they may increase predictor performance, they do not necessarily reduce the number of features.

3.4 Models

The following is a summary of models that are used for comparison in this thesis. Most of them are applied to feature selection and supervised learning type of problems.

The methods are presented by proceeding from linear to nonlinear approaches. Simple approaches are given first (the order is only suggestive). The selected methods may be categorized as follows:

- regression models: linear regression model, multilayer perceptron,

- linear methods for classification: linear discriminant analysis, quadratic discriminant analysis, logistic regression,
- prototype methods: k-nearest neighbors and self-organizing map.

The categorization is not definite since, for example, linear regression model and k-nearest neighbours may be used in classification and logistic regression in regression problems. Although the number of methods is not large we can see that they form quite a representative sample of different approaches.

3.4.1 Linear regression

A linear regression model assumes that the regression function is linear in the input. Although the model is simple, it can often provide both adequate and interpretable description of the data (how input affects the output). It may also outperform more sophisticated models, which will be seen later in this work.

Linear regression model predicts the output y using d-dimensional input vector $\mathbf{x} = (x_1, x_2, \ldots, x_d)^{\mathrm{T}}$ and parameters $\mathbf{w} = (w_0, w_1, \ldots, w_d)^{\mathrm{T}}$. The form for model is

$$y = \mathcal{E}[Y|\mathbf{x}] + \epsilon,$$

where

$$\mathcal{E}[Y|\mathbf{x}] = f(\mathbf{x}) = \sum_{i=1}^{d} w_i x_i + w_0,$$

and ϵ is zero mean noise. Let N be the number of samples. The parameters may be estimated by using a least squares method

$$\hat{\mathbf{w}} = (\mathbb{X}^{\mathrm{T}}\mathbb{X})^{-1}\mathbb{X}^{\mathrm{T}}\mathbf{y},$$

where $\mathbb{X}$ is the model matrix with $d + 1$ columns corresponding to d inputs, and N rows corresponding to the observations. The first column consists of 1's for the intercept. Vector $\mathbf{y}$ is the target and it has N elements, one element for each training sample.

In the case of classification as many linear regression models are fitted as there are classes; one model is fitted for each class. Let the number of classes be K. The target vectors $\mathbf{y}_l$, $l = 1, \ldots, K$ consist of only 0's and 1's. The element in lth target vector is 1 if the data sample corresponding to that particular element belongs to the lth class, otherwise it is 0. The classification of a sample is carried out by selecting the class whose linear regression model gives the largest value.

3.4.2 Linear discriminant functions

Linear discriminant analysis (LDA) can be explained via Bayes theorem which gives posterior probability for class l

$$\Pr(C = l|\mathbf{X} = \mathbf{x}) = \frac{f(\mathbf{x}|l)\Pr(l)}{\sum_{l'=1}^{K} f(\mathbf{x}|l')\Pr(l')},$$

where the functions $f(\mathbf{x}|\cdot)$ are Gaussian densities, $\Pr(l)$ is the prior probability of class l and K is the number of classes.

LDA assumes that the classes have a common covariance matrix $\mathbb{C}_l = \mathbb{C}$. It can be shown (for more details see for example, Hastie et al., 2001) that the linear discriminant function for class l becomes

$$\delta_l(\mathbf{x}) = \mathbf{x}^{\mathrm{T}}\mathbb{C}^{-1}\bar{\mathbf{x}}_l - \frac{1}{2}\bar{\mathbf{x}}_l^{\mathrm{T}}\mathbb{C}^{-1}\bar{\mathbf{x}}_l + \log\Pr(l),$$

where $\bar{\mathbf{x}}_l$ is the mean of class l. Mean, class prior and covariance matrix are estimated from the data.

The decision boundary between classes l and k is described by $\{\mathbf{x}|\delta_k(\mathbf{x}) = \delta_l(\mathbf{x})\}$ and the classification G for point $\mathbf{x}$ is obtained from decision rule

$$G(\mathbf{x}) = \mathrm{argmax}_j\delta_j(\mathbf{x}).$$

3.4.3 Quadratic discriminant functions

Quadratic discriminant analysis (QDA) is similar to LDA. The difference is that the covariance matrices are assumed not to be equal for the classes. The discriminant function for QDA is (for more details see Bishop, 1995; or Hastie et al. 2001):

$$\delta_l(\mathbf{x}) = -\frac{1}{2}\log|\mathbb{C}_l| - \frac{1}{2}(\mathbf{x} - \bar{\mathbf{x}}_l)^{\mathrm{T}}\mathbb{C}_l^{-1}(\mathbf{x} - \bar{\mathbf{x}}_l) + \log\Pr(l),$$

where $\mathbb{C}_l$ is the class specific covariance matrix and $\bar{\mathbf{x}}_l$ is the mean vector of the class l. While LDA gives linear decision boundaries, the boundaries found by QDA are quadratic. The classification is done in the same way as in LDA.

The reason for using LDA and QDA in this thesis is that both of these methods are supposed to work well in classification problems. Quoting Hastie et al. (2001):

Both LDA and QDA perform well on an amazingly large and diverse set of classification tasks. For example, in the STATLOG project (Michie et al., 1994) LDA was among the top 3 classifiers for 7 of the 22 datasets, QDA among the top 3 for 4 datasets, and one of the pair were in the top 3 for 10 datasets. Both techniques are widely used, and entire books are devoted to LDA. It seems that whatever exotic tools are the rage of the day, we should always have available these two simple tools.

They try to explain the success of these methods as follows:

The reason is not likely to be that the data are approximately Gaussian, and in addition for LDA that the covariances are approximately equal. More likely a reason is that the data can only support simple decision boundaries such as linear or quadratic, and the estimates provided via the Gaussian models are stable.

3.4.4 Logistic regression

A logistic regression model models the posterior probabilities of the K classes and it has the form (see Hastie et al., 2001)

$$\begin{aligned}
&\log\tfrac{\Pr(G=1|\mathbf{X}=\mathbf{x})}{\Pr(G=K|\mathbf{X}=\mathbf{x})} = w_{10} + \mathbf{w}_1^T\mathbf{x} \\
&\log\tfrac{\Pr(G=2|\mathbf{X}=\mathbf{x})}{\Pr(G=K|\mathbf{X}=\mathbf{x})} = w_{20} + \mathbf{w}_2^T\mathbf{x} \\
&\vdots \\
&\log\tfrac{\Pr(G=K-1|\mathbf{X}=\mathbf{x})}{\Pr(G=K|\mathbf{X}=\mathbf{x})} = w_{(K-1)0} + \mathbf{w}_{K-1}^T\mathbf{x},
\end{aligned}$$

where $\Pr(\cdot|\cdot)$ are the posterior probabilities, G is the categorical variable, $\mathbf{x}$ is the input vector and w_{j0} and $\mathbf{w}_j$ denote the intercept (bias) and the weights for the input variables. It can be shown that

$$\begin{aligned}
&\Pr(G=l|\mathbf{X}=\mathbf{x}) = \tfrac{\exp(w_{l0}+\mathbf{w}_l^T\mathbf{x})}{1+\sum_{l'=1}^{K-1}\exp(w_{l'0}+\mathbf{w}_{l'}^T\mathbf{x})},\ l=1,\ldots,K-1, \\
&\Pr(G=K|\mathbf{X}=\mathbf{x}) = \tfrac{1}{1+\sum_{l'=1}^{K-1}\exp(w_{l'0}+\mathbf{w}_{l'}^T\mathbf{x})}.
\end{aligned}$$

The regression problem is then to estimate the parameters

$$\{w_{10},\ldots,w_{(K-1)0},w_1,\ldots w_{K-1}\}$$

using training data. The fitting is usually done by maximum likelihood. In practice the values are obtained for two-class case by using iterative Newton-Raphson type of algorithm (details can be found from Hastie et al., 2001).

3.4.5 Multilayer perceptron

Neural networks are nonlinear statistical models (see Bishop, 1995; Hastie et al., 2001). In this work we only use networks that have two layers of adaptive weights; more specifically we use multilayer perceptron models (see Figure 11). For K-class classification the network has K output units.

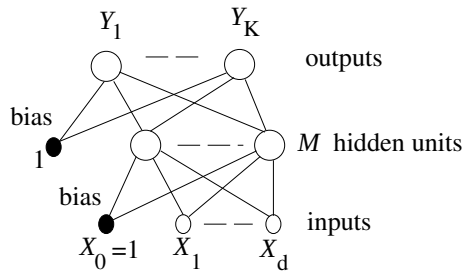

FIGURE 11 Multilayer perceptron with d inputs (and one bias), M hidden units (and one bias) and K outputs. All the links between nodes have weights.

Let M be the number of hidden units and d is the number of input units. The analytical function that corresponds to the two-layer multilayer perceptron in Figure 11 can be written in form

$$f_l(\mathbf{x}) = \tilde{g}(\sum_{j=0}^{M} w_{lj}^{(2)} g(\sum_{i=0}^{d} w_{ji}^{(1)} x_i)),$$

where $f_l(\mathbf{x})$ is the lth output of the network corresponding to the input vector $\mathbf{x}$, $\tilde{g}(\cdot)$ is the activation function for output unit, $g(\cdot)$ is the activation function for the hidden units, $w_{lj}^{(2)}$ are the weights for connections between hidden units and output unit and $w_{ji}^{(1)}$ are the weights for connections between input x_i and hidden units.

The hidden unit activation functions are often taken to be logistic sigmoids

$$g(a) = \frac{1}{1+e^{-a}}$$

or tanh activation functions

$$g(a) = \frac{e^a - e^{-a}}{e^a + e^{-a}}.$$

These activation functions are differentiable, which is a requirement in many learning algorithms. The common choice for output activation is a linear function $\tilde{g}(a) = a$, since it is simple but does not restrict the approximation capability of the network.

During network training the weight parameters are changed such that for each input vector the network produces output values close to the desired values (targets). To avoid the possibility of over-fitting it is desired that the network has a good generalization ability. Hence the prediction accuracy is in a central role when evaluating the network performance.

For regression (or classification) a typical choice for a measurement of fit is the sum-of-squared errors

$$\mathrm{SSE}(\Theta) = \sum_{l=1}^{K} \sum_{j=1}^{N} (Y_l(j) - f_l(\mathbf{X}(j), \Theta))^2,$$

where Θ contains the parameters (connection weights) of the network.

3.4.6 Nearest-neighbour methods

The k-nearest neighbour is usually used in classification problems. It determines the class for some point $\mathbf{x}$ by looking the k nearest neighbours and the point is associated to the class which has the largest portion of those k points. More generally, it can be applied to regression type problems by doing a fitting

$$\hat{f}(\mathbf{x}) = \frac{1}{k} \sum_{\mathbf{X}(j) \in N_k(\mathbf{x})} Y(j),$$

where $N_k(\mathbf{x})$ is the neighbourhood of $\mathbf{x}$, which consists of k closest points to $\mathbf{x}$. Hence, k-nearest-neighbour can be seen as an estimate of the regression function $\mathcal{E}(Y|\mathbf{X} = \mathbf{x})$.

3.4.7 Principal component analysis

Principal component analysis (PCA) is a statistical method that can be used for dimension reduction. Let us assume that data space is d-dimensional. The basic idea is to find a set of $p < d$ orthogonal vectors in a data space that accounts for as much data variance as possible. The dimensionality reduction may be carried out by projecting the data from the original d-dimensional space to the p-dimensional subspace (which is spanned by the new p orthogonal vectors). This may be given as an algorithm:

1. Compute the covariance matrix of the data.
2. Compute the eigenvectors and eigenvalues for the covariance matrix.
3. Sort the eigenvectors according to the absolute values of the corresponding eigenvalues.
4. Select the eigenvectors with largest eigenvalues and use them as a new coordinate system.
5. Project the data to the new coordinate axes.

The first principal component points into the direction where the data has the maximum amount of variance. The second principal component shows the next most important direction and it is perpendicular to the first one.

3.4.8 Sammon mapping

The Sammon mapping may be used to explore the data, to find clusters, correlations and inherent structures of the data. Sammon mapping preserves all inter-point distances.

Suppose that we have N observations, each of which is a point in d-dimensional space. The objective of Sammon mapping is to find m points in a p-dimensional space ($p < d$) such that the distances between the points match the distances between the original points well. The method minimizes the error function

$$E = \frac{1}{\sum_{j=1}^{N-1}\sum_{k=j+1}^{N}\delta_{jk}} \sum_{j=1}^{N-1}\sum_{k=j+1}^{N} \frac{(\delta_{jk} - d_{jk})^2}{\delta_{jk}},$$

where d_{jk} is the distance between two points in d-dimensional space and δ_{jk} is the distance between two points in p-dimensional space. For data visualization purpose $p = 2$ is a typical choice.

3.4.9 The self-organizing map

Although there are other interpretations, we may think of the SOM as a discretized implementation of principal curves and surfaces (Hastie and Stuetzle, 1989; LeBlanc and Tibshirani, 1994). By restricting us to two dimensional models the principal surface $\mathbf{v} \in \mathbb{R}^2$ is a nonlinear manifold that goes through the middle of d-dimensional data in $\mathbf{X} \in \mathbb{R}^d$. Formally this can be written as

$$\mathbf{x}'(\mathbf{v}) = \mathcal{E}[\mathbf{X}|\mathbf{v}'(\mathbf{X}) = \mathbf{v}] + \lambda R, \tag{4}$$

where $\mathbf{x}'(\mathbf{v})$ is the position of the principal surface in the data space, $\mathcal{E}$ denotes the expectation, $\mathbf{v}'(\mathbf{X})$ is a projection of $\mathbf{X}$ onto the surface $\mathbf{v}$, and R is an additional regulator to ensure the smoothness of the surface. The projection $\mathbf{v}'(\mathbf{x})$ is typically the closest Euclidean distance such that

$$\mathbf{v}'(\mathbf{x}) = \text{argmin}_{\mathbf{v}''}||\mathbf{x}'(\mathbf{v}'') - \mathbf{x}||. \tag{5}$$

This is illustrated in Figure 12.

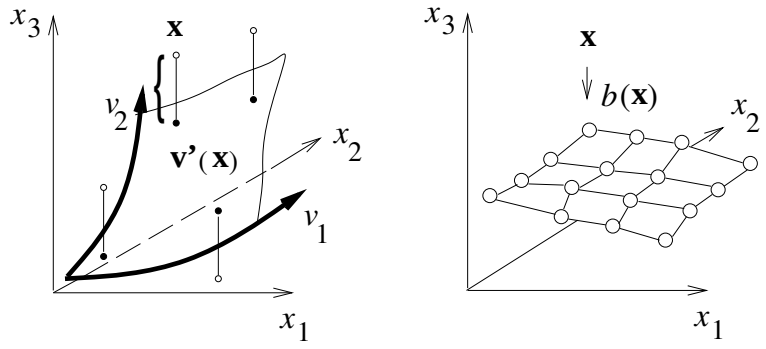

FIGURE 12 An illustration of the principal curve and its discrete implementation, the SOM.

Informally we can explain equation (5) such that each point $\mathbf{v}$ on the surface is in the expectation $\mathcal{E}[\mathbf{X}|\mathbf{v}]$ (middle) of all the points that are projected to this point via equation (4). Thus the principal surface $\mathbf{x}'(\mathbf{v})$ is a certain type of regression surface, the complexity (smoothness) of which is controlled by regulator R. With a small regulator (small λ) the surface tends to fold a lot, while a big λ makes smoother surfaces. Without any λ the surface would go through all data points $\mathbf{X}(l)$ of the training set.

Our interpretation of SOM (supported by others, see Hastie et al., 2001) is that SOM is a discrete implementation of principal curves such that the surface $\mathbf{v}$ is represented as a lattice of nodes v_i as depicted in Figure 12. Then equation (4) can be written as

$$\mathbf{w}_i = \mathbf{x}'(v_i) = \mathcal{E}[\mathbf{X}|b(\mathbf{X}) = i] + \lambda R, \tag{6}$$

where $b(\mathbf{x})$ is a projection of $\mathbf{x}$ to the closest node i on the SOM lattice

$$b(\mathbf{x}) = \text{argmin}_{i'}||\mathbf{w}_{i'} - \mathbf{x}||.$$

In SOM the regulator is usually implemented as a Nadaraya-Watson type of kernel estimator

$$\hat{y}_i = \frac{\sum_{j=i-a}^{i+a} K_j y_j}{\sum_{j=i-a}^{i+a} K_j} = \frac{\sum_{j=-a}^{+a} K_{i+j} y_{i+j}}{\sum_{j=-a}^{+a} K_j},$$

where K_z is a kernel smoother and $2a + 1$ is the width of the kernel. With a smoother the SOM equation can be written as

$$\mathbf{w}_i = \sum_{j=-a}^{+a} K_{i+j} \mathcal{E}[\mathbf{X}|b(\mathbf{X}) = i + j],$$

where K_j (with condition $\sum K_i = 1$) is, for example, a discrete implementation of a Gaussian shape or the Mexican hat function. In the simplest case $a = 1$ and $K_j = [0.3, 1.0, 0.3]/1.18$.

A typical usage of the SOM is data-analysis, where the relations of a multivariate data set are visualized on a two dimensional surface.

Consider a set of observations $\{\mathbf{x}\}$ with features $\mathbf{x} = \{x_1, x_2, \ldots, x_d\}$. After training the data can be projected onto the SOM surface such that each node i on the SOM lattice captures a set of samples

$$\Omega_i = \{\mathbf{X}(j)|b(\mathbf{X}(j)) = i\}.$$

The data can then be visualized on the SOM lattice by simple statistics. We can, for example, compute the probabilities of the nodes

$$\Pr(i|\mathbf{X}(j) \in \Omega_i)$$

and visualize them as bars. If the data contains several classes l then different colours of bars can represent the classes. This is illustrated in Figure 13.

3.5 Conclusions

Several possibilities were considered as a solution to the problem of feature selection. The problem was divided into three main parts: criterion, search strategy and the model. Each of these parts was then reviewed more thoroughly.

Comparative methodologies used in this thesis were listed and shortly explained. Later in this thesis these methods will be applied to the fiber image data

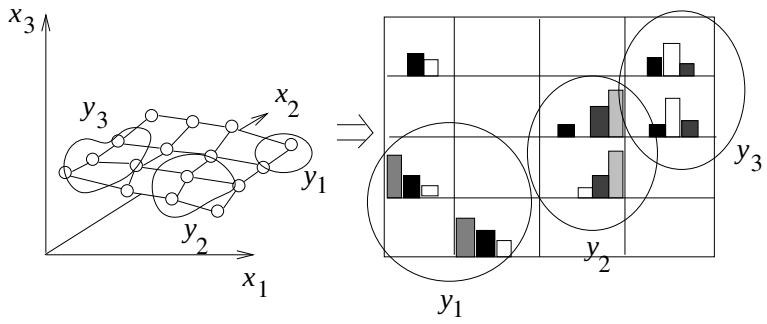

FIGURE 13 Visualization of classes.

and then the results will be compared to the results obtained by different fuzzy techniques. The main discrepancy between these and fuzzy methodologies is that the techniques explained in this chapter are not purposed for linguistic information extraction. However, they are methods which have become very popular and have been succesfully applied to different data analysis tasks.

4 THE PROPOSED FUZZY LOGIC SYSTEM

This chapter marks the beginning of the methodological part of the thesis. It starts with a short introduction to fuzzy computing and continues toward the fuzzy architecture, which has been used in the experiments.

4.1 Summary of fuzzy logic

In this summary the basic ideas of fuzzy logic are introduced. The presentation does not cover all aspects of the topic, but serves as an introduction to fuzzy concepts that are used in this thesis.

4.1.1 From crisp sets to fuzzy sets

Let Ω be a collection of objects, for example all points on a real axis. In the classical theory of sets the membership of element $x \in \Omega$ of a crisp (ordinary) set $A \subseteq \Omega$ is defined by a characteristic function or an indicator function $\mathbf{1}_A(x)$.

Definition 4.1

$$\mathbf{1}_A(x) = \begin{cases} 1, & \text{for } x \in A, \\ 0, & \text{for } x \notin A. \end{cases}$$

The element x either belongs to set A or not. A natural generalization of the indicator function is to also allow values between 0 and 1, or alternatively between 0 and infinity. The generalized indicator function could be defined as follows:

Definition 4.2

$$\mu_{\tilde{A}}(x) = \begin{cases} \xi : 0 < \xi \leq 1, & \text{for } x \in \tilde{A}, \\ 0, & \text{for } x \notin \tilde{A}. \end{cases}$$

The interpretation of the generalized indicator function value is a partial truth that an object can belong to set $\tilde{A}$. Since there is an infinite number of functions that $\mu : \Omega \mapsto [0, 1]$ or $\mu : \Omega \mapsto [0, \infty[$, the generalization is not unambiguous.

The generalization has some additional value with respect to the traditional approach; it simplifies our models of human inference and thus improves the

man-machine interaction. The objects of a set can be arranged with respect to their membership value. Furthermore, the sets may represent vague concepts in our language which we cannot define accurately. We may be able to define some fuzzy borders and "similarity" evaluations for the objects. Thus, the objects whose category is not certain will not be equipped with truth 1, and that information may be utilized later.

Definition 4.3 *A fuzzy set $\tilde{A} \subset \Omega$ is a set of ordered pairs*

$$\tilde{A} = \{(x, \mu_{\tilde{A}}(x)) | x \in \Omega\},$$

where the first part determines the element x and the membership function $\mu_{\tilde{A}}(x)$ associates with each element $x \in \Omega$ a value on interval $[0, 1]$.

We denote a fuzzy set as $\tilde{A}$ to distinguish it from crisp set A. In fuzzy literature definition 4.3 is commonly used for the fuzzy set because its interpretation is clear: set memberships can be seen as truth values of element containment in the set. From the definition we can see that the fuzzy set theory is a generalized set theory that includes the classical set theory as a special case. Since 0 and 1 are contained in $[0, 1]$, crisp sets are fuzzy sets.

4.1.2 Membership functions

The fuzzy generalization of the crisp set causes an immediate problem. It is difficult to find a theoretically justified ground for selecting the functional form (shape) of μ. There is not enough criteria to prefer one shape over another. We can find at least in the following two possible ways to constrain this problem, when elements can be interpreted in the real world context:

- Define a metric for a distance measure between elements and objects and use an appropriate function to transform these distances into fuzzy memberships.
- Use probability distribution function to define the membership.

According to the first argument the membership level is defined as a degree of proximity (based on normalized distance value) between an object and the closest ideal prototype. For example, membership value of 0.9 tells that the given value (e.g., a height of object) is at a distance equal to 0.1 from the prototype.

Typically, membership functions are described by some functions

$$\mu_{\tilde{A}}(x) = f(x),$$

where $f(x)$ may be, for instance, a triangular-shaped function

$$f(x) = \begin{cases} 1 - |x - a|/b & \text{for } a - b < x < a + b, \\ 0, & \text{otherwise}, \end{cases} \tag{7}$$

where a and b determine the centre point and the width of the membership function respectively. See (Mitaim and Kosko, 2001) for a list and comparison of different functional forms.

Membership functions can be used to fuzzify crisp values. In such cases we may use shorthand notation $\tilde{x}$ for a fuzzy value given by $\mu_{\tilde{A}}(x)$.

4.1.3 Fuzzy operations

Fuzzy set operations (Zadeh, 1965) extend the classical theory of sets in a straightforward way. Examples of operations are summarized in the following list:

empty set	$\emptyset$:	$\forall x$: $\mu_{\emptyset}(x) = 0$,
basic set, universe	Ω:	$\forall x$: $\mu_{\Omega}(x) = 1$,
identity	$\tilde{A} = \tilde{B}$:	$\forall x$: $\mu_{\tilde{A}}(x) = \mu_{\tilde{B}}(x)$,
subsethood	$\tilde{A} \subset \tilde{B}$:	$\forall, x$: $\mu_{\tilde{A}}(x) \leq \mu_{\tilde{B}}(x)$,
union	$\tilde{A} \cup \tilde{B}$:	$\forall x$: $\mu_{\tilde{A} \cup \tilde{B}} = \max(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x))$,
intersection	$\tilde{A} \cap \tilde{B}$:	$\forall x$: $\mu_{\tilde{A} \cap \tilde{B}} = \min(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x))$,
complement	$\tilde{A}^c$:	$\forall x$: $\mu_{\tilde{A}^c}(x) = 1 - \mu_{\tilde{A}}(x)$.

These fuzzy extensions do not extend all mathematical properties of sets. For example, the laws of contradiction, $A \cap A^c = \emptyset$ (basis for many mathematical proofs), and the excluded middle, $A \cup A^c = \Omega$, are not valid for fuzzy sets.

Minimum and maximum are not the only possible operators for the intersection and union of fuzzy sets. This is due to the fact that membership grades are no longer restricted to $\{0, 1\}$ and thus the operators can not be uniquely defined. To ensure the generalization of classical sets, the fuzzy operations should be subject to the properties of classical set operations.

Non-compensatory intersection operators are called *t-norms* and non-compensatory union operators are called *t-conorms* or *s-norms*. Compensatory ones can perform operations which are somewhere between intersection and union operations. For their exact definitions see Zimmermann (1985, 1993) or Driankov et al. (1993).

Zadeh's min-max formulas are usually suboptimal (Hisdal, 1994; Laviolette and Seaman, 1994). Hence, there is a tendency towards replacing these operations with more optimal ones. Quoting Kosko (1997):

Modern fuzzy systems dispense with min and max. They consist of only multiplies, additions, and divisions.

The choice between different operations will be considered and their usability evaluated later in this work, when we construct a fuzzy locig system.

4.1.4 Properties of fuzzy sets

Definition 4.4 *If the support of fuzzy set $\tilde{A}$ consists of only one point, i.e. $\tilde{A} = \{(x(1), \mu_{\tilde{A}}(x(1)))\}$, it is called a fuzzy singleton, where $x(1)$ denotes the first and only value of variable x.*

Later in this work, fuzzy singletons will be used as output membership functions, because in that way we can avoid some annoyances caused by defuzzification (defuzzification is a mapping from fuzzy sets back to a crisp value, see sections 4.2 and 4.3.4).

Definition 4.5 *Fuzzy set is convex if for all $x(i), x(j), x(k) \in \Omega$ and $x(i) \leq x(j) \leq x(k)$, it follows that*

$$\mu_{\tilde{A}}(x(j)) \geq \min(\mu_{\tilde{A}}(x(i)), \mu_{\tilde{A}}(x(k))).$$

Like probabilistic partitions, convexity increases the clarity of the fuzzy methodology and related interpretations.

Definition 4.6 *The cardinality, sometimes referred to as sigma count, of finite fuzzy set is*

$$|\tilde{A}| = \sum_{x \in \Omega} \mu_{\tilde{A}}(x).$$

Cardinality may be used to define the *degree of subsethood*, $|\tilde{A} \cap \tilde{B}|/|\tilde{A}|$ (Kosko, 1992b) and it is an important concept in rule extraction (see Chapter 6).

4.2 Fuzzy logic systems

Systems that are based on fuzzy reasoning have become almost standard techniques in some branches of engineering design. In our view the fundamental idea of a fuzzy logic system is to simplify things by using easily interpretable rules like

IF (temperature is low AND humidity is low)
THEN (power is high).

It is a characteristic of many of these rules that all terms of the logic are scalars (temperature, humidity, power) that occur only once per rule. To model a complex thing several rules must be used. More formally these kind of systems are built as follows. Consider fuzzification of the ith variable x_i with m_i fuzzy sets $\tilde{A}_i^{k_i}$, $k_i = 1, \ldots, m_i$ such that

$$x_i \in \tilde{A}_i^{k_i}, \qquad \text{if } \mu_{\tilde{A}_i^{k_i}}(x_i) > 0,\ k_i = 1, 2, \ldots, m_i,$$

which means that x_i belongs to $\tilde{A}_i^{k_i}$ with truth $\mu_{\tilde{A}_i^{k_i}}(x_i)$. Similarly there are fuzzy sets B^l that correspond to a fuzzified output of y,

$$y \in \tilde{B}^l, \qquad \text{if } \mu_{\tilde{B}^l}(y) > 0,\ l = 1, 2, \ldots, m_y,$$

where m_y denotes the number of sets on the output axis.

Let d be the dimension of the input space. A typical fuzzy logic system is composed of rules like

$$\begin{aligned} Rule_r : &\text{IF } (x_1 \text{ is } \tilde{A}_1^{k_1(r)} \text{ AND } x_2 \text{ is } \tilde{A}_2^{k_2(r)} \text{ AND} \dots \text{ AND } x_d \text{ is } \tilde{A}_d^{k_d(r)}) \\ &\text{THEN } (y \text{ is } \tilde{B}^{l(r)}), \end{aligned} \tag{8}$$

where $\{k_j(r)\}$ gives the indices for input fuzzy sets $\tilde{A}$ and $\{l(r)\}$ gives the index for output fuzzy set $\tilde{B}$ in rule r.

Rule (8) can be generalized to multivariate fuzzy memberships, if needed, which allows vector level partitions of the input space. Let $\mathbf{x}$ be a vector of inputs, $\tilde{A}^k$, $k = 1, \dots, m_{\mathbf{x}}$ be the multidimensional input fuzzy sets and $\tilde{B}^l$, $l = 1, \dots, m_y$ be the output fuzzy sets. The multivariate fuzzy logic system is made of rules

$$Rule_r : \quad \text{IF } (\mathbf{x} \text{ is } \tilde{A}^{k(r)}) \text{ THEN } y \text{ is } \tilde{B}^{l(r)}. \tag{9}$$

One can easily note that scalar rules (8) are special cases of (9).

The fuzzy system that is made of a set of rules is actually a function of type

$$y = f_{\text{fuzzy}}(\mathbf{x} | \{\mu_{\tilde{A}^k}\}, \{\mu_{\tilde{B}^l}\}, R), \tag{10}$$

where $\{\mu_{\tilde{A}^k}\}$ is the set of input membership functions, $\{\mu_{\tilde{B}^l}\}$ is a set of output membership functions, and R is a set of fuzzy rules that relate input values to output values. Equation (10) relates fuzzy systems to various function approximation and modeling methods, which is one of the subjects of this thesis.

4.2.1 Takagi-Sugeno-Kang type of systems

A wide variety of fuzzy systems can be found in literature, and may be grouped into two main categories: pure fuzzy logic systems and fuzzy logic systems with a fuzzifier and defuzzifier (see Figure 14).

A pure fuzzy logic system uses fuzzy sets as its inputs and outputs. It consists only of fuzzy rules (rule base) and a fuzzy inference engine. The applicability of a pure fuzzy system is somewhat limited since in real world applications the inputs are usually real-valued. The same holds for output values. For example, we might want to get some crisp-valued control action as an output instead of some vague fuzzy set.

Fuzzy systems with a fuzzifier and defuzzifier are interesting from a practical point of view. They are among the best-known and most successful in the application of fuzzy logic. A fuzzy logic system with a fuzzifier and defuzzifier may be understood as a mapping from crisp values to fuzzy values and then back to crisp values. Fuzzifier computes the firing degree of a rule with respect to the input. The fuzzification is done using the membership functions that correspond to the fuzzy sets in a rule antecedent. Thus, the firing degree is a matching degree between the input and the rule antecedent. If the input is a vector instead

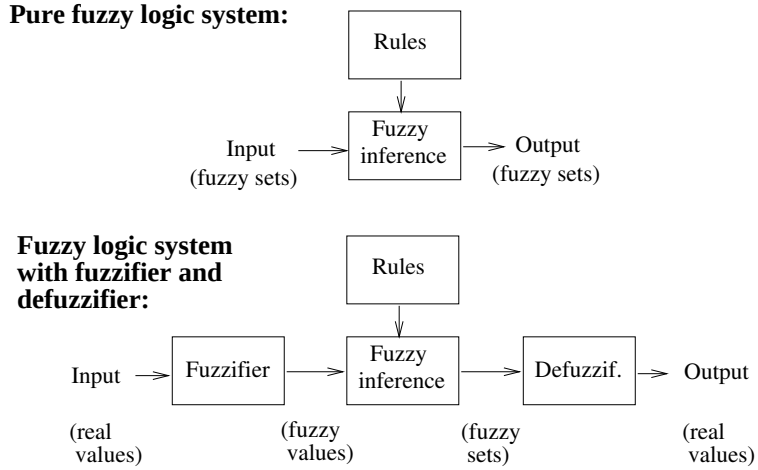

FIGURE 14 Fuzzy logic systems.

of a single variable, the firing degree is computed using a conjunctive operator (AND-operator), which in fuzzy logic literature is often referred to as a t-norm.

Takagi and Sugeno's fuzzy system (Takagi and Sugeno, 1985; Sugeno and Kang, 1988), which is also referred to as Takagi-Sugeno-Kang fuzzy system, is a practical implementation of a fuzzy logic system with fuzzifier and defuzzifier. The main characteristic of Takagi and Sugeno's fuzzy system (TSFS) is that it forms output as a weighted average of local functions (one function for each rule). See Figure 15 for the basic idea of TSFS.

TSFS uses the following type of fuzzy rules:

$$\begin{aligned} Rule_r : \text{IF } (x_1 \text{ is } \tilde{A}_1^{k_1(r)} \text{ AND } x_2 \text{ is } \tilde{A}_2^{k_2(r)} \text{ AND} \ldots \text{ AND } x_d \text{ is } \tilde{A}_d^{k_d(r)}) \\ \text{THEN } (y^r \text{ is } c_0^r + c_1^r x_1 + c_2^r x_2 + \ldots + c_d^r x_d), \end{aligned} \tag{11}$$

where c_i are real-valued parameters. Except the consequent part, the rules are similar to the rules (8). A system of rules is combined (aggregated) by using the weighted average of the y^r's

$$y(\mathbf{x}) = \frac{\sum_{r=1}^{q} w^r y^r}{\sum_{r=1}^{q} w^r},$$

where q is the number of rules and the weight w^r is the truth of the premise of the ith implication. The corresponding weight is calculated as a product of truths,

$$w^r = \prod_{i=1}^{d} \mu_{\tilde{A}_i^{k_i(r)}}(x_i).$$

The problem with this system is that the consequent part of the rule is not fuzzy, which somewhat decreases the interpretability of the system.

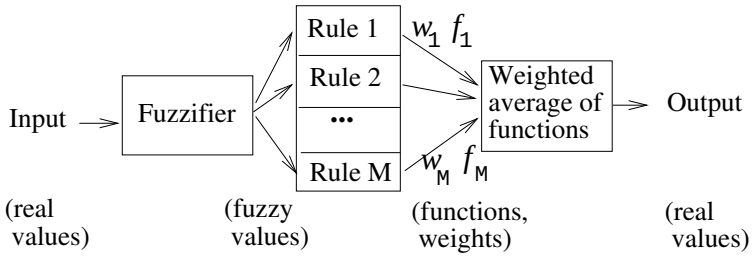

FIGURE 15 Takagi-Sugeno-Kang fuzzy logic system.

In fact, the TSFS interpolates between different local functions and smoothly joins them together. The functions may be global, like $y = c_0 + c_1 x_1$, but they are localized by fuzzy input sets. A special case of the TSFS is a system, which is easy to interpret, consists of rules with only one output parameter:

$$\begin{aligned} Rule_r : \text{IF } (x_1 \text{ is } \tilde{A}_1^{k_1(r)} \text{ AND } x_2 \text{ is } \tilde{A}_2^{k_2(r)} \text{ AND} \ldots \text{ AND } x_d \text{ is } \tilde{A}_d^{k_d(r)}) \\ \text{THEN } (y^r \text{ is } c^r). \end{aligned} \tag{12}$$

4.3 Variations in fuzzy systems

There is no single axiomatic theory for fuzzy logic. Rather there are many possibilities to formulate what the fuzzy operations are and how they are used. Therefore the user must make difficult decisions before designing a new type of fuzzy system.

In fuzzy systems the most important decisions are concerned with the implementation of

i) Fuzzification of input, including the selection of positions and shapes of fuzzy sets on input axes.

ii) Design of the rule base and positioning of the fuzzy sets on the output axes.

iii) Computation of truth of rule antecedent with t-norm.

iv) Computation of rule consequences with implication operation. The operation determines how the IF- and THEN-part of the rule is combined.

v) Rule aggregation that defines how individual consequences are combined.

vi) Defuzzification of output truths into crisp output values.

In this thesis the last four options are fixed in the beginning (see section 4.4 for more details), and all interest is in the automated design of the first two options. It is the role of the rest of this chapter to explain the choices behind the fixed options.

In fuzzy literature the roles of implication and defuzzification are often discussed separately. The role of implication is seen as a fuzzy generalization of a logical implication

$$A \Rightarrow B \equiv \neg A \vee B,$$

which is essential for *modus ponens* of reasoning

Premise	A is true
Implication	IF A then B
Conclusion	B is true.

In practice reasoning in fuzzy systems is quite different from reasoning in traditional logic. Logical inference is based on the chaining of rules that eliminates all conflicting rules. In fuzzy systems the reasoning is usually based on a simple single stage computation (Zimmermann, 1985), and it is the role of defuzzification to resolve the conflicts between the truth of different outcomes.

4.3.1 T-norm (fuzzy AND)

T-norm forms truth for the rule antecedent. For example, if we consider the rule system (8), we see that the rule antecedent may consist of several fuzzy sets $\tilde{A}_i^{k_i(r)}$, e.g.

$$\text{IF } (x_1 \text{ is } \tilde{A}_1^{k_1(r)} \text{ AND } x_2 \text{ is } \tilde{A}_2^{k_2(r)}).$$

To form a truth value, we need to "replace" the AND with some mathematical operation which combines 2 (or more) memberships into a single value. Examples from a large set of possible operations for the above kind of rule are:

- minimum:

$$t(\mu_{\tilde{A}_1^{k_1(r)}}(x_1), \mu_{\tilde{A}_2^{k_2(r)}}(x_2)) = \min(\mu_{\tilde{A}_1^{k_1(r)}}(x_1), \mu_{\tilde{A}_2^{k_2(r)}}(x_2)),$$

- algebraic product:

$$t(\mu_{\tilde{A}_1^{k_1(r)}}(x_1), \mu_{\tilde{A}_2^{k_2(r)}}(x_2)) = \mu_{\tilde{A}_1^{k_1(r)}}(x_1)\mu_{\tilde{A}_2^{k_2(r)}}(x_2),$$

- Hamacher product:

$$t(\mu_{\tilde{A}_1^{k_1(r)}}(x_1), \mu_{\tilde{A}_2^{k_2(r)}}(x_2)) = \frac{\mu_{\tilde{A}_1^{k_1(r)}}(x_1)\mu_{\tilde{A}_2^{k_2(r)}}(x_2)}{\mu_{\tilde{A}_1^{k_1(r)}}(x_1) + \mu_{\tilde{A}_2^{k_2(r)}}(x_2) - \mu_{\tilde{A}_1^{k_1(r)}}(x_1)\mu_{\tilde{A}_2^{k_2(r)}}(x_2)},$$

- Einstein product:

$$t(\mu_{\tilde{A}_1^{k_1(r)}}(x_1), \mu_{\tilde{A}_2^{k_2(r)}}(x_2)) = \frac{\mu_{\tilde{A}_1^{k_1(r)}}(x_1)\mu_{\tilde{A}_2^{k_2(r)}}(x_2)}{2 - (\mu_{\tilde{A}_1^{k_1(r)}}(x_1) + \mu_{\tilde{A}_2^{k_2(r)}}(x_2) - \mu_{\tilde{A}_1^{k_1(r)}}(x_1)\mu_{\tilde{A}_2^{k_2(r)}}(x_2))},$$

- Bounded product:

$$t(\mu_{\tilde{A}_1^{k_1(r)}}(x_1), \mu_{\tilde{A}_2^{k_2(r)}}(x_2)) = \max(0, \mu_{\tilde{A}_1^{k_1(r)}}(x_1) + \mu_{\tilde{A}_2^{k_2(r)}}(x_2) - 1).$$

Minimum and product are the most popular choices. Their generalization to d fuzzy sets is straightforward: for minimum it can be constructed simply by

$$t(\mu_{\tilde{A}_1^{k_1(r)}}(x_1), \mu_{\tilde{A}_2^{k_2(r)}}(x_2), \ldots, \mu_{\tilde{A}_d^{k_d(r)}}(x_d)) =$$
$$\min(\mu_{\tilde{A}_1^{k_1(r)}}(x_1), \mu_{\tilde{A}_2^{k_2(r)}}(x_2), \ldots, \mu_{\tilde{A}_d^{k_d(r)}}(x_d))$$

and for product by

$$t(\mu_{\tilde{A}_1^{k_1(r)}}(x_1), \mu_{\tilde{A}_2^{k_2(r)}}(x_2), \ldots, \mu_{\tilde{A}_d^{k_d(r)}}(x_d)) = \prod_{i=1}^{d} \mu_{\tilde{A}_i^{k_i(r)}}(x_i).$$

4.3.2 Implication

With implication we can compute the truth distribution of output y for a given rule like

$$\text{IF } \underbrace{(x \text{ is } \tilde{A}^{k(r)})}_{\mu_{\tilde{A}^{k(r)}}(x)} \text{ THEN } \underbrace{(y \text{ is } \tilde{B}^{l(r)})}_{\mu_{\tilde{B}^{l(r)}}(y)}$$

as a rule consequence

$$\mu_r^{\text{conseq}}(y) = \text{Implication}(\mu_{\tilde{A}^{k(r)}}(x), \mu_{\tilde{B}^{l(r)}}(y)) = I^r(x, y).$$

Several different implications can be found in the basic fuzzy logic textbooks (like Zimmermann, 1985; Driankov et al., 1993 and Niemi, 1996). Also the articles by Bandler and Kohout (1981) and Dubois and Prade (1999) list many such operations. In the following they are categorized as S-implications, R-implications, T-norm implications and other implications.

S-implications, which are based on the classical implication:

1. Kleene-Dienes $I_1(x, y) = \max(1 - x, y)$,
2. Dubois-Prade $I_2(x, y) = \begin{cases} 1 - x, & y = 0, \\ y, & x = 1, \\ 1, & \text{otherwise}, \end{cases}$
3. Reichenbach-Mizumoto $I_3(x, y) = 1 - x + x \cdot y$.

R-implications, which reflect partial ordering on propositions and they are obtained by residuation of a t-norm T

$$x \to y = \sup\{z \in [0, 1] | T(x, z) \leq y\} :$$

4. Gödel $I_4(x,y) = \begin{cases} 1, & x \leq y, \\ y, & \text{otherwise}, \end{cases}$
5. Goguen $I_5(x,y) = \begin{cases} \min(1, y/x), & x \neq 0, \\ 1, & \text{otherwise}, \end{cases}$
6. Lukasiewicz $I_6(x,y) = \min(1, 1-x+y)$.

T-norm implications, which do not generalize the classical implication, but are often used in fuzzy control:

7. minimum $I_7(x,y) = \min(x,y)$,
8. Hamacher product $I_8(x,y) = \frac{x \cdot y}{x+y-x \cdot y}$,
9. algebraic product $I_9(x,y) = x \cdot y$,
10. Einstein product $I_{10}(x,y) = \frac{x \cdot y}{1+(1-x) \cdot (1-y)}$,
11. bounded product $I_{11}(x,y) = \max(0, x+y-1)$,
12. drastic product $I_{12}(x,y) = \begin{cases} x, & y = 1, \\ y, & x = 1, \\ 0, & \text{otherwise}. \end{cases}$

Other implications:

13. QL-implication $I_{13}(x,y) = \max(1-x, \min(x,y))$.

The Kleene-Dienes implication is sometimes referred to as Dienes-Rescher, Gödel implication as Standard star and QL-implication as Zadeh implication. With respect to fuzzy control, Mamdani implication (minimum) is said to be the most important implication known in literature (Driankov, 1993).

4.3.3 Aggregation of fuzzy rules

The role of the aggregation operator is to combine individual rule outputs into one output truth $\mu^{\text{conseq}}(y)$. In practice, most systems do this either as a maximum

$$\mu^{\text{conseq}}(y) = \max_r \mu_r^{\text{conseq}}(y)$$

or as a sum

$$\mu^{\text{conseq}}(y) = \sum_r \mu_r^{\text{conseq}}(y)$$

of rule consequences.

4.3.4 Defuzzification of consequences

Given a distribution of truth $\mu^{\text{conseq}}(y)$ for output variable y, the role of defuzzification is to select one crisp value, $\hat{y}$, for it. Again there are several possibilities, the most common of which being

- Centre of gravity (CoG):

$$\hat{y} = \frac{\int y\mu^{\text{conseq}}(y)\,dy}{\int \mu^{\text{conseq}}(y)\,dy},$$

- Middle of Maxima (MoM):

$$\hat{y} = \frac{\min\{z \,|\, \mu^{\text{conseq}}(z) = \max_y \mu^{\text{conseq}}(y)\}}{2} + \frac{\max\{z \,|\, \mu^{\text{conseq}}(z) = \max_y \mu^{\text{conseq}}(y)\}}{2},$$

- Smallest of Maxima (SoM):

$$\hat{y} = \min\{z \,|\, \mu^{\text{conseq}}(z) = \max_y \mu^{\text{conseq}}(y)\},$$

- Largest of Maxima (LoM):

$$\hat{y} = \max\{z \,|\, \mu^{\text{conseq}}(z) = \max_y \mu^{\text{conseq}}(y)\},$$

- Bisection or Center of Area (CoA) denoted by y':

$$\hat{y} = y' : \int_{-\infty}^{y'} \mu^{\text{conseq}}(y)\,dy = \int_{y'}^{\infty} \mu^{\text{conseq}}(y)\,dy.$$

If the distribution of truth $\mu^{\text{conseq}}(y)$ consists of rule consequences $\mu_r^{\text{conseq}}(y)$ that are obtained from singleton output membership functions, the centre of gravity method is also referred to as height method (HM) (see Driankov et al., 1993). In that case CoG operation becomes a discrete sum of rules

$$\hat{y} = \left(\frac{\sum_r y\mu_r^{\text{conseq}}(y)}{\sum_r \mu_r^{\text{conseq}}(y)} \right)_0,$$

where

$$\left(\frac{a}{b}\right)_0 = \begin{cases} \frac{a}{b}, & \text{if } b > 0, \\ 0, & \text{otherwise.} \end{cases}$$

4.4 Empirical evaluation of choices

To evaluate the effect of different choices in implication, rule aggregation and defuzzification, two types of experiments were made. The first set of experiments tries to evaluate how well the fuzzification–rule aggregation–defuzzification chain can represent and identify relation $y = f(x) = x$. In this set of experiments a fixed algebraic product ($I_9(x, y)$) type of implication is used.

The second set of experiments tries to evaluate all three choices in the context of function approximation.

In addition to these two experiments a third set of experiments was made. Here the objective was to find out, which of the t-norms are the most suitable for the fuzzy AND-operation in the proposed fuzzy logic system.

The triangular functions were used as membership functions. The reasons for this choice are as follows:

- They are computationally simple.
- With trapezoidal membership functions they are the most commonly used membership functions.
- Trapezoidal membership functions are reported to perform poorly in some function approximation types of problems (Mitaim and Kosko, 2001).
- System with triangular membership functions can be made to perform a similar mapping as the system with trapezoidal functions.

It should be noted that there are also other good choices for membership functions. Usually their functional form is more complex than that of a triangular function.

4.4.1 Evaluation of defuzzification and rule aggregation

In this experiment two membership functions are placed on the input and output axes, as shown in Figure 16. So we have two fuzzy sets on each axis and two rules: one which relates $\mu_{\tilde{A}^1}(x)$ to $\mu_{\tilde{B}^1}(x)$ and the other rule which relates $\mu_{\tilde{A}^2}(x)$ to $\mu_{\tilde{B}^2}(x)$. In our test setting the implication used to relate input membership to output membership is fixed, since we only want to study the effect of defuzzification and aggregation on the result. The selected operation is the algebraic product

$$\mu_j^{\text{conseq}} = I_9(\mu_{\tilde{A}^j}(x), \mu_{\tilde{B}^j}(x)) = \mu_{\tilde{A}^j}(x)\mu_{\tilde{B}^j}(x),$$

where $\mu_{\tilde{A}^j}(x)$ is the jth input membership function and $\mu_{\tilde{B}^j}(x)$ is the jth output membership function. Since the membership functions and axes are identical for input and output, the system should be able to "reproduce the original input values" as well as possible.

The output

$$\hat{y} = f_{\text{fuzzy}}(x|\{\mu_{\tilde{A}^k}\}, \{\mu_{\tilde{B}^k}\})$$

for real axis interval $[5, 10]$ between the fuzzy sets (see Figure 16) is shown in Figure 17. The optimal output is a linear curve $y = x$ (shown in Figure 17 as a dashed line), because then the fuzzification–rule aggregation–defuzzification process would not have a distorting effect on the result. Numerical values of the representation error on the interval $[5, 10]$ for different rule aggregation and defuzzification methods are listed in Table 3. SoM and LoM defuzzification methods were not used since in this test setup (triangular membership functions and product implication) they form similar results with MoM. The values are the sum of squared

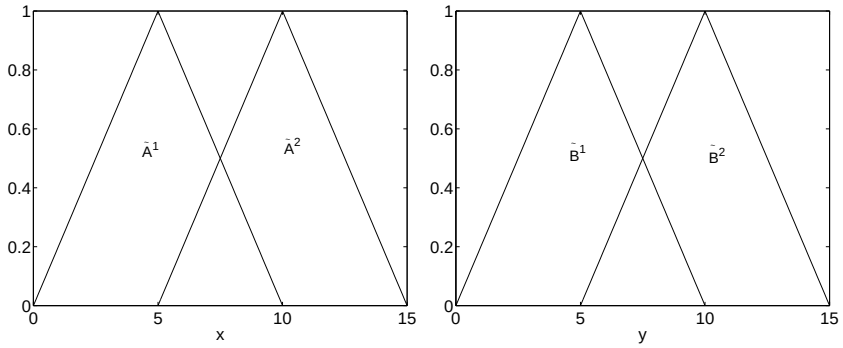

FIGURE 16 The left figure contains two input membership functions $\mu_{\tilde{A}^1}(x)$ and $\mu_{\tilde{A}^2}(x)$. Right figure has similar membership functions $\mu_{\tilde{B}^1}(y)$ and $\mu_{\tilde{B}^2}(y)$ for output y.

TABLE 3 The sum of squared errors for different defuzzifier–rule aggregation combinations.

Aggr./Defuzz.	CoG	CoA	MoM
Max-aggr.	7.94	148.77	1035.5
Sum-aggr.	0.05	34.05	1035.5

errors that were calculated over 501 discrete points on an interval [5,10]. From the results we can conclude that CoG defuzzification combined with the sum rule aggregation produces the best result. CoG combined with the maximum rule aggregation gives the second best representation, but the result is clearly worse than when using sum aggregation. Therefore we may have a good reason to prefer CoG defuzzification–sum aggregation combination to other options. The result also corresponds to the finding of Driankov et al. (1993) according to which this combination has many favorable characteristics.

4.4.2 Evaluating the combinations of defuzzification, rule aggregation and implication

The aim of this evaluation is to find a good combination of defuzzification method, rule aggregation and implication for function approximation types of problems. We seek for a combination that gives the best accuracy.

The test set is made with randomly generated functions. The fuzzy system is then fitted to each function of the set as well as possible.

Some adaptive method must be utilized to make the fuzzy systems to approximate the functions well. The discussion about such methods is postponed until Chapter 5. In the current context it is enough to know that a simple sum of

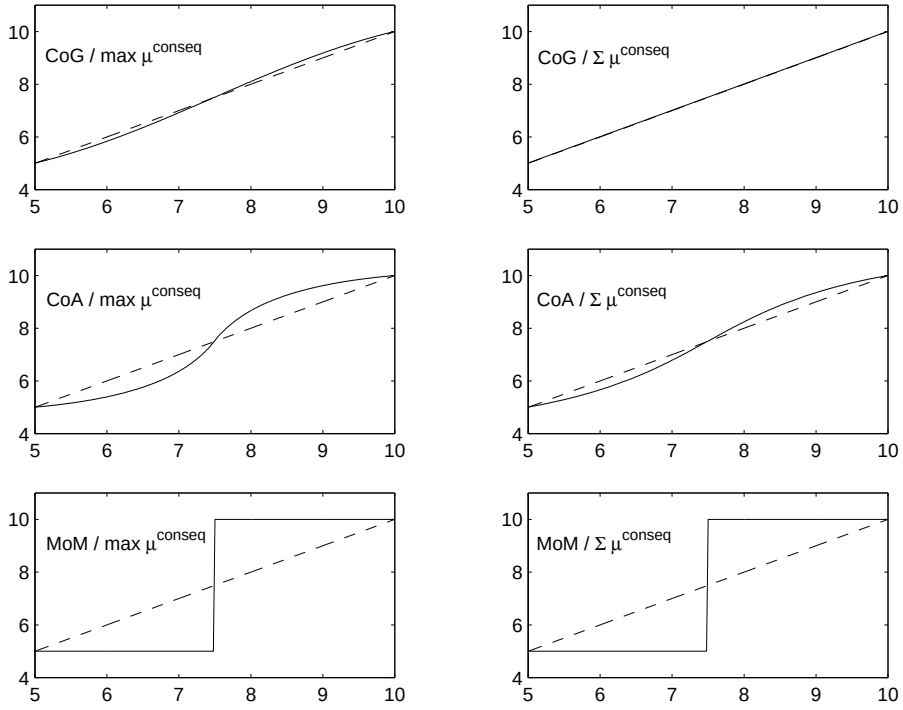

FIGURE 17 Left figures: various defuzzification methods with max-aggregation. Right figures: various defuzzification methods with sum-aggregation. Dashed line corresponds to the target: the closer the solid lines are to the target the better the defuzzification method is in reproducing the original input values.

squares error type of optimization has been used. In the optimization the fuzzy system parameters are "forced" towards values which minimize the error between a fuzzy system outputs and the desired ouputs (given by the target function). A summary of test setup is as follows:

1. **Problem**: Function approximation (1-dimensional functions). One should note that this is a more general problem than classification, since we can write any classification task as a regression over indicator functions. Furthermore, in the probabilistic setting we may use function approximation to model posterior probabilities of classes. Therefore we do expect that the results of this evaluation can be used for both regression and classification tasks.

2. **Objective**: To find out which combination of fuzzy operations (implication, rule aggregation and defuzzification) gives the best accuracy, when the locations of output membership functions are adaptive.

3. **Data**: The data was formed by generating 100 random functions

$$f(x) = c_1 \sin(c_2 x) + c_3 \cos(c_4 x),$$

where the coefficients c_i were varied randomly (i.i.d. from uniform distribution), such that coefficients c_1 and c_3 obtained values between -2.5 and 2.5, and coefficients c_2 and c_4 obtined values between -0.25 and 0.25. Each function consisted of a set of $N = 80$ data pairs $\{x, y\}$ and the range of x was set to $[1, 81]$. Some examples of the functions are shown in Figure 18.

4. **Model**: Fuzzy logic system with

- **Fixed input membership functions**: A fixed set of nine symmetric and triangular-shaped membership functions. They were positioned on the input axis to form a partition of unity (sum to one at each point x).
- **Adaptive output membership functions**: Nine symmetric triangular membership functions with fixed width. Their locations are determined using a training algorithm.
- **Rules**: In this simple example there are 9 rules, which correspond to 9 input membership functions, each with one unique output membership function.
- **Training procedure**: The weights for input membership functions are determined by a least-squares method

$$\theta = (\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{t},$$

where $\mathbf{A}$ is a $N \times M$ matrix of input membership function values, $\mathbf{t}$ is a vector of target values (function values). Letters N and M denote the number of data points and membership functions, respectively. The row j in matrix $\mathbf{A}$ corresponds to the fuzzified values of sample $X(j)$, and a matrix element is given by $a_{jk} = \mu_{\tilde{A}^k}(X(j))$, where $\mu_{\tilde{A}^k}$ is a symmetric triangular membership function (see equation 7).

- **Locations of output membership functions**: Computed weights θ for the input membership functions are interpreted directly as locations of singleton type of output membership functions, because we can write the approximation in points (input membership locations) as

$$\hat{y} = \sum_{k=1}^{9} \theta_k \mu_{\tilde{A}^k}(x) \approx \frac{\sum_{l=1}^{9} \int y \hat{\mu}_{\tilde{B}^l}(y|\theta_l) \mu_{\tilde{A}^l}(x) dy}{\sum_{l=1}^{9} \int \hat{\mu}_{\tilde{B}^l}(y|\theta_l) \mu_{\tilde{A}^l}(x) dy},$$

which becomes equality when $\mu_{\tilde{A}^l}(x) = 1$ and

$$\hat{\mu}_{\tilde{B}^l}(y|\theta_l) = \begin{cases} 1, & \text{if } y = \theta_l, \\ 0, & \text{otherwise.} \end{cases}$$

The equality was shown above for CoG defuzzification, but in our test setup similar relation can be formed for other defuzzification methods

used in the comparisons. This is then generalized with triangle output membership functions, where

$$\mu_{\tilde{B}^l}(y|\theta_l) = \begin{cases} \frac{a-|\theta_l - y|}{a}, & \text{if } |y-\theta_l| \leq a, \\ 0, & \text{otherwise.} \end{cases}$$

The role of the generalization is to (fuzzy) interpolate the obtained pointwise result over the y-axis. Note that after the output membership function positions have been computed, the rule system is ready, and the weights θ are discarded.

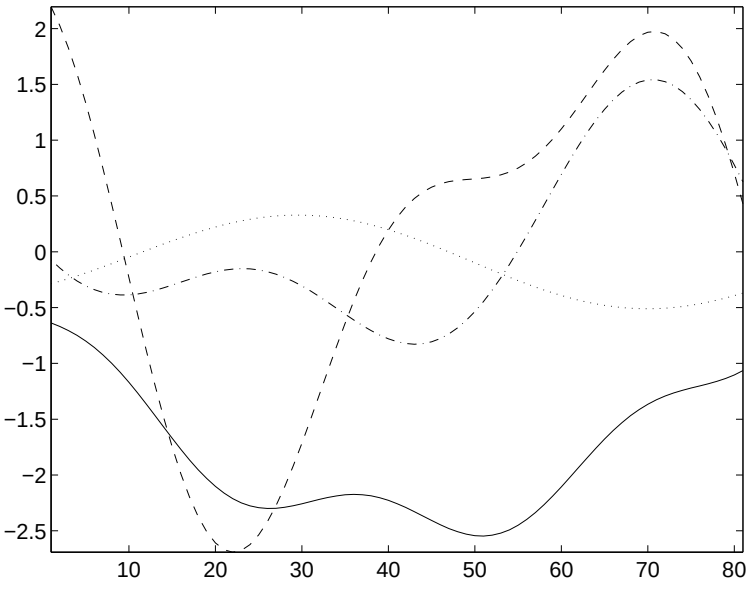

FIGURE 18 Examples of functions that are used for testing the accuracy of different implication-defuzzifier combinations.

An example of the results for one test example function is shown in Figure 19. Two top rows show the fixed input membership functions and the output membership functions that have been obtained by training, respectively. The next three rows in the figure plot the target function (dashed line) and fuzzy logic system output (solid line). These three separate figures correspond to three defuzzification methods (MoM, CoA and CoG). In this particular example fuzzy logic system utilized algebraic product implication and sum rule aggregation.

The results of approximating 100 functions are summarized in Tables 4 and 5. The results shown in the former table are obtained by using the max rule aggregation and in the latter table by the sum rule aggregation. The tables contain information about approximation accuracy of a fuzzy logic system when using

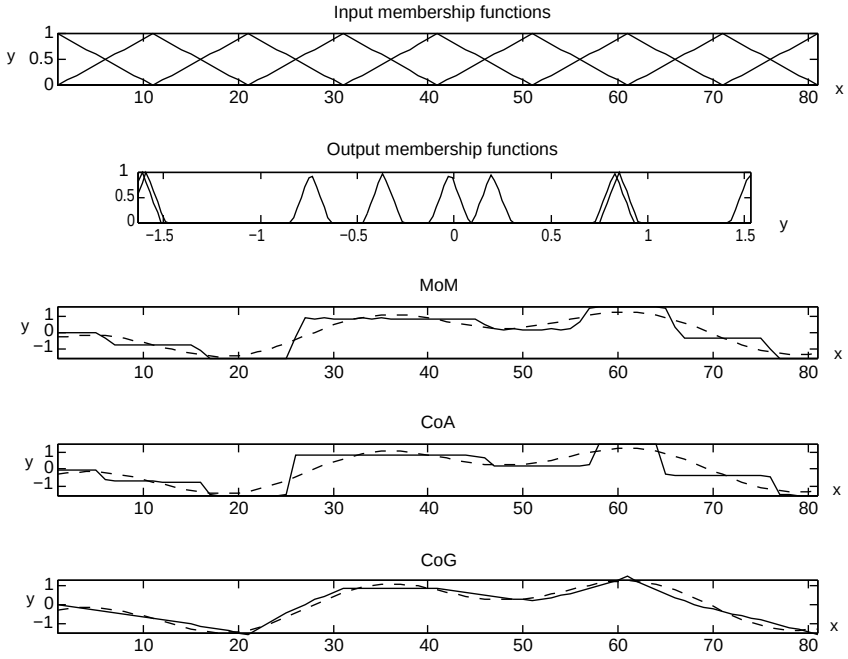

FIGURE 19 Membership functions and an example of approximation results with three different defuzzifiers. In the three lower figures the dashed line is the target function and the solid line is the system output.

different choices for implication, rule aggregation and defuzzification operations. The first column of the tables refer to the implications $I_i(x, y)$. The results are an averaged sum of the squared error values

$$\frac{1}{100}\sum_{j=1}^{100}\mathrm{SSE}_{f_j}, \tag{13}$$

where SSE_{f_j} is the error when approximating the jth function. Standard deviations of the accuracies are shown with $\pm$ -denotation.

The best results in Tables 4 and 5 correspond to combinations "algebraic product implication, max rule aggregation and CoG defuzzification" as well as "algebraic product implication, sum rule aggregation and CoG defuzzification". They seem equally good.

One thing which may grab our attention in Tables 4 and 5, is the weak performance of S-implications (I_1, I_2 and I_3), R-implications (I_4, I_5 and I_6) and I_{13}. We should notice, however, that the results do not imply that those implication operations are bad in general, they are such in this test setup and when using our

TABLE 4 Approximation accuracies of a fuzzy logic system with the max rule aggregation, different implication and defuzzifier combinations.

	Defuzzifier				
Implication	MoM	SoM	LoM	CoA	CoG
Kleene-Dienes	153.2 $\pm$123.2	166.4 $\pm$117.6	723.2 $\pm$644.4	153.2 $\pm$123.2	153.2 $\pm$123.2
Dubois-Prade	153.2 $\pm$123.2	166.4 $\pm$117.6	723.2 $\pm$644.4	153.2 $\pm$123.2	153.2 $\pm$123.2
Reich.-Miz.	153.2 $\pm$123.2	166.4 $\pm$117.6	723.2 $\pm$644.4	153.2 $\pm$123.2	153.2 $\pm$123.2
Gödel	153.2 $\pm$123.2	166.4 $\pm$117.6	723.2 $\pm$644.4	153.2 $\pm$123.2	153.2 $\pm$123.2
Goguen	153.2 $\pm$123.2	166.4 $\pm$117.6	723.2 $\pm$644.4	153.2 $\pm$123.2	153.2 $\pm$123.2
Lukasiewicz	153.2 $\pm$123.2	166.4 $\pm$117.6	723.2 $\pm$644.4	153.2 $\pm$123.2	153.2 $\pm$123.2
Minimum	34.4 $\pm$44.1	46.7 $\pm$58.6	42.1 $\pm$51.6	43.9 $\pm$59.6	5.2 $\pm$9.1
Hamacher pr.	43.5 $\pm$52.3	43.8 $\pm$53.3	43.8 $\pm$53.2	42.9 $\pm$55.8	4.7 $\pm$7.5
Algebraic pr.	43.5 $\pm$51.9	43.8 $\pm$52.9	43.8 $\pm$52.8	42.6 $\pm$54.5	**4.6** $\pm$**7.2**
Einstein pr.	43.8 $\pm$52.8	44.1 $\pm$53.8	44.1 $\pm$53.7	42.0 $\pm$52.4	5.1 $\pm$7.6
Bounded pr.	43.9 $\pm$52.9	44.2 $\pm$53.9	44.2 $\pm$53.8	43.3 $\pm$53.3	14.8 $\pm$21.5
Drastic pr.	136.7 $\pm$110.5	148.7 $\pm$105.5	642.5 $\pm$573.0	- -	- -
QL	153.2 $\pm$123.2	166.4 $\pm$117.6	723.2 $\pm$644.4	153.2 $\pm$123.2	153.2 $\pm$123.2

fixed set of rule aggregation and defuzzification operations. According to Jager (1995) the aggregation of fuzzy rules should be performed by a conjunction operation when applying implication that complies with the classical implication. This was briefly tested: using the minimum as a rule aggregation with implications I_1-I_6 and I_{13} produced comparable results to those with sum rule aggregation. In addition to this, it was found that the goodness of the results when using S- and R-implications is highly dependent on the output membership function widths. An increase in widths improved the performance considerably. However, this sensitivity to widths is not a desired property for our fuzzy logic system.

TABLE 5 Approximation accuracies of a fuzzy logic system with the sum rule aggregation, different implication and defuzzifier combinations.

	Defuzzifier				
Implication	MoM	SoM	LoM	CoA	CoG
Kleene-Dienes	44.2 $\pm$53.3	44.6 $\pm$54.4	45.0 $\pm$55.9	153.2 $\pm$123.2	152.7 $\pm$123.0
Dubois-Prade	36.4 $\pm$46.3	45.1 $\pm$58.4	44.5 $\pm$54.2	151.9 $\pm$123.5	151.4 $\pm$122.4
Reich.-Miz.	43.5 $\pm$51.9	43.8 $\pm$52.9	43.8 $\pm$52.8	153.2 $\pm$123.2	152.5 $\pm$122.9
Gödel	149.4 $\pm$120.7	164.1 $\pm$118.9	624.8 $\pm$591.2	153.0 $\pm$123.3	152.9 $\pm$123.1
Goguen	105.7 $\pm$132.2	165.6 $\pm$213.0	141.8 $\pm$164.8	152.5 $\pm$123.8	151.6 $\pm$122.6
Lukasiewicz	34.4 $\pm$44.1	46.7 $\pm$58.6	42.1 $\pm$51.6	153.2 $\pm$123.3	152.4 $\pm$122.9
Minimum	34.4 $\pm$44.1	46.7 $\pm$58.6	42.1 $\pm$51.6	43.9 $\pm$59.6	5.2 $\pm$9.1
Hamacher pr.	43.5 $\pm$52.3	43.8 $\pm$53.3	43.8 $\pm$53.2	42.9 $\pm$55.8	4.7 $\pm$7.5
Algebraic pr.	43.5 $\pm$51.9	43.8 $\pm$52.9	43.8 $\pm$52.8	42.6 $\pm$54.5	**4.6** $\pm$**7.2**
Einstein pr.	43.8 $\pm$52.8	44.1 $\pm$53.8	44.1 $\pm$53.7	42.0 $\pm$52.4	5.1 $\pm$7.6
Bounded pr.	43.9 $\pm$52.9	44.2 $\pm$53.9	44.2 $\pm$53.8	43.3 $\pm$53.3	14.7 $\pm$21.5
Drastic pr.	136.7 $\pm$110.5	148.7 $\pm$105.5	642.5 $\pm$573.0	- -	- -
QL	48.2 $\pm$51.8	49.4 $\pm$50.8	104.9 $\pm$102.4	153.2 $\pm$123.2	152.8 $\pm$123.0

4.4.3 Selection of t-norm

In this subsection we try to estimate how much the choice of AND-operation affects the approximation accuracy. The following test setup was used:

1. **The problem**: Function approximation (2-dimensional functions: $f : \mathbb{R}^2 \mapsto \mathbb{R}$).

2. **The objective**: To find a t-norm operation which outperforms the others in the function approximation type of problems.

3. **The data**: As earlier, we had 100 different randomly selected function shapes for the evaluation. Now, instead of 1-D functions a set of 2-D functions

$$f(x_1, x_2) = c_1 \sin(c_2 x_1) \cos(c_3 x_2)$$

were used, where all the coefficients c_i were in the range $[-\pi, \pi]$ and $x_1, x_2 \in [-2, 2]$. See Figure 20 for examples of such functions.

4. **The model**: Fuzzy logic system with

   - **Input membership functions**: A fixed set of 5×5 symmetric and triangular-shaped membership functions (5 membership function for each axis). They were positioned on the input axis to form a partition of unity.
   - **Output membership functions**: 25 singleton membership functions. Their location was determined by least squares fitting.
   - **Rules**: There are 25 rules, which correspond to 5×5 input membership functions, each with one output singleton membership function.
   - **Implication**: Algebraic product.
   - **Defuzzification**: HM.

5. **Training procedure**: See the description in section 4.4.2.

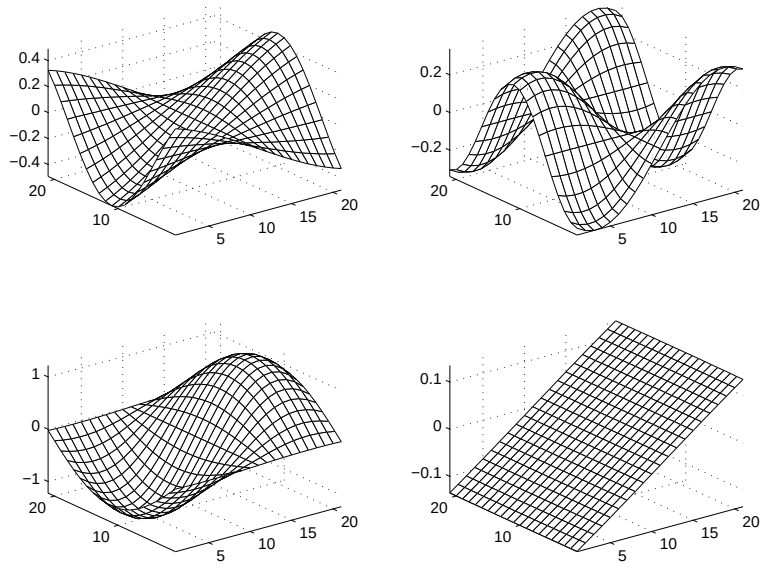

FIGURE 20 Four examples of functions that were used for testing the accuracy of different t-norms.

The results of experiments for each t-norm are given in Table 6. The values shown are the smallest, the largest and the average sum of the squared error (SSE). They were calculated from the 100 squared error values in the same way as earlier (see equation (13)). Also the standard deviation of the averages are shown in table (with sign $\pm$).

TABLE 6 The evaluation of t-norms.

T-norm	smallest SSE	largest SSE	average SSE $\pm$ std.dev.
Minimum	0.01	5.16	0.85 ± 1.26
Hamacher product	0.01	4.39	0.70 ± 1.08
Algebraic product	0.00	4.05	0.54 ± 0.90
Einstein product	0.00	4.52	0.60 ± 0.97
Bounded product	0.15	25.24	5.02 ± 6.39

4.4.4 A note about rules

In the previous examples no attempts were made to make the rules readable. There were, implicitely, as many rules of the type

$$\text{IF } (x_i \text{ is } \tilde{A}_i^k) \text{ THEN } (y \text{ is } \tilde{B}^l)$$

as there were input-output membership combinations. In practice the set of rules are reorganized and selected such that the linguistic interpretation becomes easy. This will require an understanding about what is readable and how it is combined with the building of the rule system. These issues are discussed in the next chapter of this thesis.

4.5 The fuzzy system used in this thesis

The evaluation of fuzzy systems with different variations lead naturally to the following fuzzy architecture. In this work we use conjunctive rules with algebraic t-norm (AND), algebraic implication I_9, sum aggregation, and centre of gravity defuzzification. These choices are general and accurate enough for our purpose. They also support other types of rules, like conventional production rules, as special cases (see, e.g., Wang, 1994). For example, a rule

$$\begin{aligned}&\text{IF } (x_{i_1(r)} \text{ is } \tilde{A}_{i_1(r)}^{k_{i_1}(r)} \text{ AND } x_{i_2(r)} \text{ is } \tilde{A}_{i_2(r)}^{k_{i_2}(r)} \text{ OR } x_{i_3(r)} \text{ is } \tilde{A}_{i_3(r)}^{k_{i_3}(r)})\\ &\text{THEN } (y \text{ is } \tilde{B}^{l(r)}),\end{aligned}$$

which contains the operator "OR" can be split without a loss of generality into two separate rules

$$\text{IF } (x_{i_1(r)} \text{ is } \tilde{A}_{i_1(r)}^{k_{i_1}(r)} \text{ AND } x_{i_2(r)} \text{ is } \tilde{A}_{i_2(r)}^{k_{i_2}(r)}) \text{ THEN } (y \text{ is } \tilde{B}^{l(r)})$$

and

$$\text{IF } (x_{i_3(r)} \text{ is } \tilde{A}_{i_3(r)}^{k_{i_3}(r)}) \text{ THEN } (y \text{ is } \tilde{B}^{l(r)}).$$

Rules containing the "NOT" operation can also be considered as special cases of (8), since we can always form a new fuzzy set to replace $\text{NOT}(\tilde{A}_i^{k_i(r)})$.

We prefer the use of connective rules instead of disjunctive rules because they are easier to understand. Support for this claim can be found in the cognitive psychology. Eysenck and Keane (1999) write:

Finally, one of the earliest findings in concept formation was that conjunctive concepts were easier to learn than disjunctive concepts (see Bruner et al., 1956). So, for example, it is easier for people to learn a concept called DRAF consisting of the conjoined features—black and round and furry—than when its features are disjunctive—"black OR round OR furry".

However, in the presence of background information disjunctive rules may become easier to understand (see Eysenck and Keane, 1999). In our fuzzy modeling problems, especially when extracting the rules from the data, we do not suppose such prior information to be available.

4.5.1 Detailed presentation of the fuzzy system

As a conclusion we can sketch the basic form for the fuzzy logic system that is used throughout this thesis. Our definition of the fuzzy function

$$y = f_{\text{fuzzy}}(\mathbf{x}|\{\mu_{\tilde{A}^k}\}, \{\mu_{\tilde{B}^l}\}, R)$$

can be explained in four steps

1. Fuzzification of inputs, which replaces all variables $x_i \in \{x_1, \ldots, x_d\}$ with fuzzy truth values, i.e.

   $$x_i \mapsto \tilde{x}_i^{i_1}, \tilde{x}_i^{i_2}, \ldots, \tilde{x}_i^{i_{m_i}},$$

   where $\tilde{x}_i^{i_k} = \mu_{\tilde{A}_i^{k_i}}(x_i)$, $k = 1, \ldots, m_i$, $i = 1, \ldots, d$.

2. Evaluation of fuzzy rules, which gives the truth ω_r for each rule r. An explicit definition requires definitions for fuzzy operations, for example

   $$\begin{cases} \tilde{x}_a \text{ AND } \tilde{x}_b = \min\{\tilde{x}_a, \tilde{x}_b\}, \\ \tilde{x}_a \text{ OR } \tilde{x}_b = \max\{\tilde{x}_a, \tilde{x}_b\}, \end{cases}$$

   where $\tilde{x}_a$ and $\tilde{x}_b$ are fuzzy values. In this thesis the rules are usually in the conjunctive form

   $$\tilde{x}_{a_1} \text{ AND } \tilde{x}_{a_2} \text{ AND } \tilde{x}_{a_3}, \ldots, \text{AND } \tilde{x}_{a_d} = \tilde{\bigwedge}_i \tilde{x}_{a_i},$$

   and the corresponding fuzzy AND-operation is implemented with a product:

   $$\tilde{x}_{a_1} \cdot \tilde{x}_{a_2} \cdot \tilde{x}_{a_3} \cdots = \prod_i \tilde{x}_{a_i}.$$

3. Rule-aggregation begins with the computation of output "truths" for fuzzified output variables $\tilde{y}^l$. A typical usage of the idea is that the truth of a fuzzy output variable $\tilde{y}^l$ is a disjunction (fuzzy OR-operation) of all rules r that assign truths to output fuzzy set $\tilde{B}^l$ (fuzzy value $\tilde{y}^l$). For example,

$$\tilde{y}^l = \omega_a \text{ OR } \omega_b \text{ OR} \ldots = \tilde{\bigvee}_{r|l} \omega_r,$$

where notation $r|l$ indicates that only those matching rules are used that deduce the values $\tilde{y}^l$ of set $\tilde{B}^l$. In this thesis the operation is usually implemented as a sum, **strength** v^l

$$v^l = \sum_{r|l} \omega_r,$$

where the condition (y is $\tilde{B}^l$) exists in the rule r. The problem is that the sum v^l can be bigger than one if several rules for the output set $\tilde{B}^l$ are used. Keeping this in mind, we still like to use v^l instead of $\tilde{y}^l$. If needed, we can always introduce a constant c such that $\tilde{y}^l = cv^l$ (c is constant over all l).

4. Defuzzification of output truths $\tilde{y}^l$ (or strengths v^l) to a crisp output y can be implemented in several ways. Here we prefer to use a weighted average, which combines the sum-aggregation of rules and centre of gravity defuzzification as follows

$$y = \frac{\sum_l \tilde{y}^l \int y \mu_{B^l}(y)\, dy}{\sum_l \tilde{y}^l \int \mu_{B^l}(y)\, dy} = \frac{\sum_l v^l \int y \mu_{B^l}(y)\, dy}{\sum_l v^l \int \mu_{B^l}(y)\, dy},$$

which also explains why strengths v^l can be used instead of $\tilde{y}^l$.

In this thesis the functional form

$$f_{\text{fuzzy}}(\mathbf{x}|\{\mu_{\tilde{A}^k}\}, \{\mu_{\tilde{B}^l}\}, R)$$

of a fuzzy logic system has a major role. Thus it is important to write it explicitly by combining the previously introduced steps 1,2,3 and 4. Using the previously introduced choices for fuzzy operations the fuzzy logic system can be written in the form

$$\hat{y} = \frac{\sum_l \overbrace{\left[\sum_{r|l} \overbrace{\prod_j \mu_{\tilde{A}_{i_j(r)}^{k_{i_j(r)}}}(x_{i_j}(r))}^{\text{fuzzy rule } r|l}\right]}^{\text{strength } v^l} \overbrace{\int y \mu_{\tilde{B}^l}(y)\, dy}^{\text{output member weight } y^l}}{\underbrace{\sum_l \left[\sum_{r|l} \prod_j \mu_{\tilde{A}_{i_j(r)}^{k_{i_j(r)}}}(x_{i_j}(r))\right] \underbrace{\int \mu_{\tilde{B}^l}(y)\, dy}_{\text{normalization constant } z^l}}_{\text{normalization constant}}}. \tag{14}$$

4.6 An illustrative example

These steps are illustrated in the following example (see Figure 21). Let the inputs x_1, x_2 and x_3 correspond to temperature, relative humidity and change in temperature, and let the output variable y be the change in control action (increase or decrease heat).

1. Fuzzification of crisp value 15 of variable x_1 produces two fuzzy values $\mu_{\mathrm{Cold}}(15) = 0.3$ and $\mu_{\mathrm{Hot}}(15) = 0.7$. Fuzzification of crisp value 30 of variable x_2 produces fuzzy values $\mu_{\mathrm{Low}}(30) = 0.5$ and $\mu_{\mathrm{High}}(30) = 0$ The third measurement value -5 produces $\mu_{\mathrm{Negative\,high}}(-5) = 1.0$ and $\mu_{\mathrm{Positive\,high}}(-5) = 0$.

2. Evaluation of rules (in this example, only three rules are used) is done next. Fuzzy AND-operation is implemented here as a product operation: $\omega_1 = 0.3 \cdot 0.5 = 0.15$, $\omega_2 = 0.7 \cdot 0.5 = 0.35$, and $\omega_3 = 1$.

3. Calculation of variables v^l: $v^{Increase} = \omega_1 + \omega_3 = 0.15 + 1 = 1.15$, $v^{Decrease} = \omega_2 = 0.35$.

4. Defuzzification:

$$
\begin{aligned}
y &= \frac{\sum_l v^l \int y \mu_{B^l}(y)\, dy}{\sum_l v^l \int \mu_{B^l}(y)\, dy} \\
&= \frac{v^{Decrease} \int y \mu_{\mathrm{Decrease}}(y)\, dy + v^{Increase} \int y \mu_{\mathrm{Increase}}(y)\, dy}{v^{Decrease} \int \mu_{\mathrm{Decrease}}(y)\, dy + v^{Increase} \int \mu_{\mathrm{Increase}}(y)\, dy} \\
&= \frac{0.35 \cdot 75 + 1.15 \cdot 30.55}{0.35 \cdot 15 + 1.15 \cdot 15} = 2.7.
\end{aligned}
$$

4.7 Conclusions

We defined the objective of this work, which is to be able to extract meaningful linguistic information from the numerical data. A fuzzy set theory is chosen to accomplish this objective. We reviewed the basic definitions of fuzzy sets and outlined a fuzzy logic system that will work as a default system in this thesis.

As a result of experimental testing it was found that defuzzifiers that rely on the centre of gravity principle, give the best accuracy. Hence CoG and HM seem as good choices. The performance of MoM, SoM, LoM and CoA was quite poor.

Different defuzzifiers were tested with different implication operations. A combination of CoG with t-norm implications proved to outperform the other combinations. Especially the algebraic product, Einstein product and Hamacher product implications gave distinctly better accuracy than the other candidates. The inconvenience with t-norm implications is the fact that they do not generalize the Boolean rule of implication. Nevertheless, they seem to be more practical than the operations which have this property.

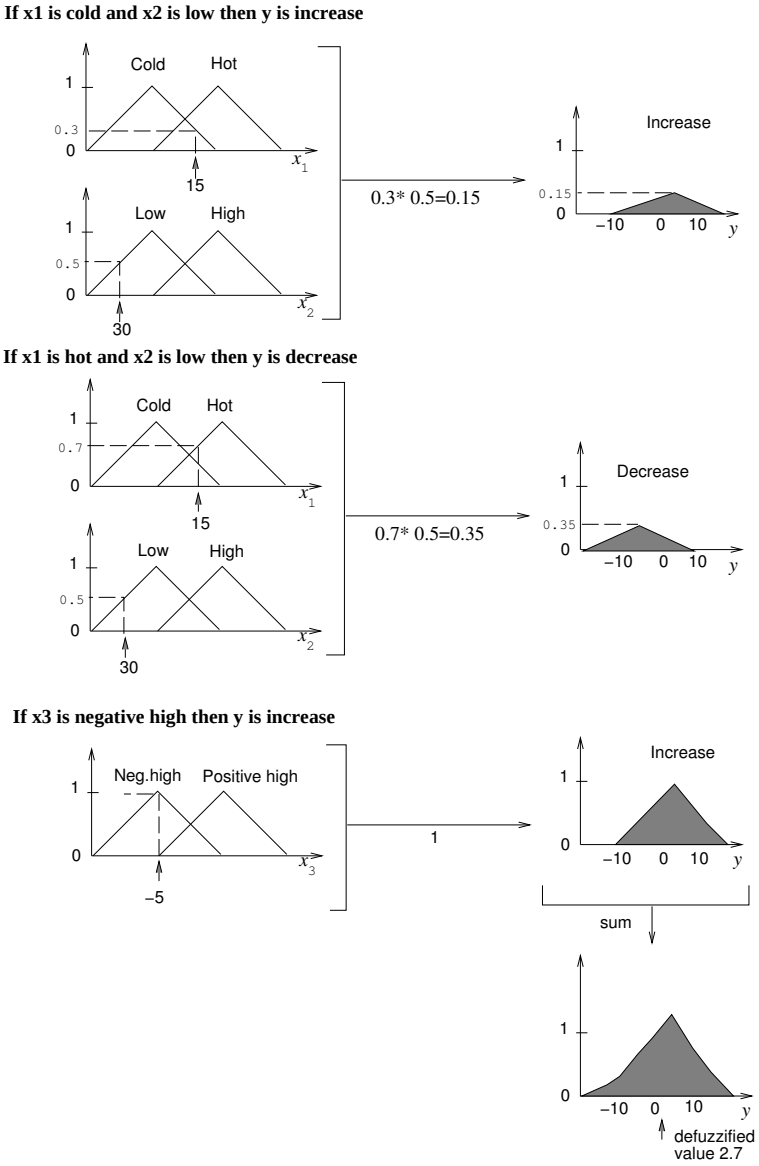

FIGURE 21 An example of fuzzy inference with product implication. The input membership functions are shown on the left, while the scaled output membership functions are shown on the right. The "distribution" in the lower right corner has been formed by summing the scaled output membership functions.

Sum rule aggregation was found to perform well. Our preference for sum aggregation is not unique. For example, Kosko (1992a, 1992b) has criticized the systems which combine the output fuzzy sets with pairwise maxima. His critique

is founded on the fact that maximum aggregation tends to produce a uniform output as the number of combined fuzzy sets increases. A uniform distribution always has the same mode and centroid.

The algebraic product performed well as a conjunctive AND-operation. It gave a distinctly smaller error value than the minimum operation in the function approximation type of problem setup. Furthermore since it is simple and utilizes all information, unlike minimum which takes only the smaller component into consideration, it is easy to adopt the product for AND-operation.

Finally, we could construct and formalize a fuzzy logic system, which is both accurate and simple. A consequence of the selected representation is that the algorithmic complexity of the model is reduced. The same system can be used with nonfixed or prespecified membership functions, which is a good property when considering the use of fuzzy systems from either an accuracy or interpretability point of view. The representation makes it easy to draw comparisons between fuzzy systems and other more conventional techniques.

5 LEARNING FUZZY SYSTEMS

This chapter is about learning fuzzy systems. In fuzzy literature this is a difficult topic, since adaptive algorithms are often hard to express with linguistic rules. Therefore many learning fuzzy systems seem to be closer to statistical learning, rather than linguistic fuzzy reasoning. In this work an attempt is made to give support for both of these directions. As a first concrete step, only the most fundamental learning algorithms, based on linear methods, are considered. Yet, the generic overall objectives can be presented for any learning fuzzy systems as follows:

1. To define the learning fuzzy system:
   - What is the structure and what is fixed in that system?
   - What is allowed to change during learning?
2. To describe parameter estimation techniques that are suitable for the selected system.
3. To be able to extract meaningful rules despite learning.

Technically the role of learning is to tune the parameters of the fuzzy function $\hat{y} = f_{\text{fuzzy}}(\mathbf{x}|\{\mu_{\tilde{A}}(\mathbf{x})\}, \{\mu_{\tilde{B}}(y)\}, R)$ using a set of training data $\mathcal{D} = (\mathbf{X}(1), Y(1)), (\mathbf{X}(2), Y(2)), \ldots, (\mathbf{X}(N), Y(N))$. With a suitable loss function $Q(y, f_{\text{fuzzy}})$, and a parameterization of fuzzy sets and rules, the objective of fuzzy learning can be expressed as a statistical parameter estimation problem. In the simplest case, rules and output fuzzy sets are fixed, and the objective is to find the best positions and shapes for input fuzzy sets $\tilde{A}_i^{k_i}$, defined by $\mu_{\tilde{A}_i^{k_i}}(x_i|\Theta_{k_i})$, by optimizing the parameters Θ_{k_i}.

5.1 A review of learning methods for fuzzy systems

In many application tasks, fuzzy rules are derived from human experts as linguistic rules. To simplify the rule generation several automatic methods have been

proposed. The number of methods that have been developed to construct fuzzy systems directly from data is vast. The majority of such learning methods can be categorized into one of the following:

- **Gradient descent based learning**: Guely and Siarry (1993) use gradient descent to find the membership locations and their widths (left width and right width) in Takagi-Sugeno fuzzy systems. Shi and Mizumoto (2000) develop a gradient descent approach for symmetric triangular membership functions. Driankov et al. (1993) describe the use of iterative gradient descent algorithm for fuzzy control systems, where the optimized parameters are input membership function widths, locations and the real-valued control outputs. Chi et al. (1996) present an approach for optimizing the widths and locations of input membership functions, which are then used in classification problems. Other references where the gradient based learning has been studied are (Jang, 1993), (Wang, 1994), (Pedrycz, 1995) and (Dadone and VanLandingham, 2000).

- **Evolutionary learning**: These are based on the optimization strategies that use randomness in their search. In fuzzy applications genetic algorithms have become one of the most popular techniques to achieve learning. For example, genetic algorithms have been used in optimizing fuzzy clustering criteria (Bezdek and Hathaway, 1994), feature selection (Casillas et al., 2001), finding redundant rules (Lekova et al., 1998), integrating fuzzy knowledge (Wang et al., 1998).

- **Clustering**: Clustering can perform an unsupervised detection of patterns in data. Fuzzy clustering has been studied extensively. Automatic target recognition (Lim and Bezdek, 1998) has been constructed by utilizing fuzzy c-means (FCM) clustering algorithm. The relationship of FCM to competitive learning algorithms and learning vector quantization is investigated in (Karayiannis and Bezdek, 1997). Chi et al. (1996) give an algorithm for constructing membership functions from clusters. The clusters are formed by fuzzy c-means, adaptive vector quantization or a self-organizing map. Their algorithm gives positions and widths for the input membership functions. Subclust-algorithm finds cluster centres with subtractive clustering (Chiu, 1994). Niskanen (2001) examines the ways to describe data and infer from data with words. He also shows ways to produce linguistic rules from clusters that have been produced either by FCM or Subclust-algorithm. A survey of fuzzy clustering algoritms for pattern recognition can be found in (Baraldi and Blonda, 1999).

- **Neurofuzzy techniques**: In these techniques a fuzzy system is considered as a neural network structure. The parameters of the model are tuned by backpropagation or other neural network optimization techniques. Jang and Sun (1993) have shown the functional equivalence between radial basis function

networks (RBFN) and a certain class of fuzzy systems. Therefore, learning used for RBFN may be applied to find the parameters (centres and widths of input membership functions and their weights) of fuzzy systems, as noted in (Cherkassky and Mulier, 1998). Neurofuzzy approaches are considered in (Simpson, 1992), (Jang, 1993), (Brown and Harris, 1994), (Wang, 1994), (Bossley, 1997), (Pal and Mitra, 1999) and (Mitra and Hayashi, 2000).

There are also other approaches that cannot clearly be categorized into any of the above groups. For example, Cherkassky and Mulier (1998) consider using support vector machines to learn the centre locations of membership functions. Ho et al. (2001) use fuzzy wavelet networks for function learning. Thuillard (2001) employs wavelets to form input membership functions. This multiresolution analysis type of idea determines the locations and widths of membership functions and their weights (scalings).

One of the simplest ways to incorporate learning into fuzzy systems is presented in (Wang and Mendel, 1991), where the idea is to learn rules from examples by collecting data samples into fuzzy hyperboxes. If there are enough samples in some box, then a rule is formed. Also Nozaki et al. (1996) propose a simple grid-based fuzzy classification system. In their approach the grades of certainty of fuzzy rules are adjusted automatically.

5.2 The proposed architecture for learning fuzzy systems

We already saw an example of a learning fuzzy system with adaptive output membership functions (fuzzy sets) in the previous chapter (section 4.4). In a more general case, rules R and the positions of the input sets are also adaptive. However, we assume that the structure (model class) of the fuzzy system $f_{\text{fuzzy}}(\mathbf{x}|\{\mu_{\tilde{A}}\}, \{\mu_{\tilde{B}}\}, R)$ is fixed. That is, we shall assume that

i) There is a fixed number of given inputs $\{x_i\}$, $i = 1, 2, \ldots, d$. (If not, feature selection, which is the topic of Chapter 6, would be required.)

ii) For each input x_i, there is a number of membership functions $\mu_{\tilde{A}_i^{k_i}}(x_i|\Theta_{k_i}^x)$, $k_i = 1, 2, \ldots, m_i$ (fuzzy sets $\tilde{A}_i^{k_i}$, which partition the axis x_i). All input membership functions are single modal triangles, where the scale, height, of a membership function may be adaptive. Triangles may also be adaptive in their width and locations, but in this chapter they are assumed to be fixed.

iii) For output variable y there is a number of output membership functions $\mu_{\tilde{B}^l}(y|\Theta_l^y)$, $l = 1, 2, \ldots, m_y$. Like with input membership functions the number of output membership functions, m_y, is usually fixed. Output membership functions can be adaptive in their locations, but we also consider the case when they are fixed.

iv) There is only one output fuzzy set $\tilde{B}^l$ for any rule r (but many rules can associate to the same output).

v) All the rules use algebraic product $\mu_{\tilde{A}_{i_1}} \cdot \mu_{\tilde{A}_{i_2}}$ for AND operation, product for implication, CoG defuzzification and sum for rule aggregation.

vi) To support easy interpretation each rule is made of

- Conjunction of input literals (fuzzified variables) only:

$$\text{IF } (\tilde{x}_1 \wedge \tilde{x}_2 \wedge \ldots) \text{ THEN } y$$

- There is only one literal (fuzzy set) for each variable x_i in one rule.

The outcome will be a system that is defined with fixed input membership functions and adaptive output membership functions and rules. In the first stage the roles of output membership functions and rules are expressed jointly with a set of weights $\{w_r\}$, which contain nonzero weights for each rule. Given the weights it is then possible to determine the positions (and shapes if needed) of the output memberships. The general implementation of these steps is described in the next sections.

5.3 A relation between linear systems and fuzzy learning

Learning in a fuzzy system of type $\hat{y} = f_{\text{fuzzy}}(\mathbf{x}|\{\mu_{\tilde{A}}(\mathbf{x})\}, \{\mu_{\tilde{B}}(y)\}, R)$ means modification of the parameters of the input membership functions $\mu_{\tilde{A}_i^{k_i}}(x_i)$, $k = 1, 2, \ldots, m_i$ or/and output membership functions $\mu_{\tilde{B}^l}(y)$, $l = 1, 2, \ldots, m_y$. Assuming that the parameters define the shapes and locations of membership functions, there are several options to achieve learning. For example, the objective may be

a) Learn all the parameters of input membership functions, while the output membership functions are fixed.

b) Learn all the parameters of input and output membership functions.

c) Only the role of rules and output membership functions is adaptive (see also the previous chapter for examples).

Options a) and b) allow large modifications to the system, which means that the interpretation of such a system may become difficult. Option c) is not ideal for interpretation either, but the approach can be modified to achieve it together with adequate accuracy. Hence, approach c) is considered more thoroughly in the following.

5.3.1 Adaptive rules as weighted dictionarys

In Chapter 4 a fuzzy system of type

$$\hat{y} = \frac{\sum_l v_l \int y \mu_{\tilde{B}^l}(y)\, dy}{\sum_l v_l \int \mu_{\tilde{B}^l}(y)\, dy}. \tag{15}$$

was introduced. We can interpret this, under certain assumptions (Friedman, 1995; Hastie et al., 2001; Cherkassky and Mulier, 1998) either as a kernel estimator

$$\hat{y} = \sum_j K_j(\mathbf{x}, \bar{\mathbf{x}}^j) Y(j), \qquad (\mathbf{X}(j), Y(j)) \in \mathcal{D},$$

where $\hat{y}$ is expressed as a weighted combination of output values $Y(j)$ (weighted with kernel functions $K_j(\cdot)$), or as a dictionary method

$$\hat{y} = \sum_j w_j b_j(\mathbf{x}|\Theta_j),$$

where output is represented as a weighted combination of basis functions (dictionary). The most striking difference between kernel methods and dictionary methods is that kernel methods use data points $(\mathbf{X}(j), Y(j))$ more directly, while dictionary methods are more typical in function approximation of data. The methods are, however, equivalent under some assumptions (we can define kernels as basis functions and use prototypes as data points).

Dictionary methods use a predefined library of basis functions $b_j(\mathbf{x}|\Theta_j)$ on the domain of $f(\mathbf{x})$. Although we could interpret this as a certain type of kernel method, it is more realistic to think of memberships $\mu_{\tilde{A}^j}(\mathbf{x})$ as basis functions $b_j(\mathbf{x}|\Theta_j)$. Since the dictionary is a weighted sum of these functions, multivariate memberships must be used to combine rule conjunctions to basis functions

$$b_r(\mathbf{x}|\Theta_r) = \prod_i \mu_{\tilde{A}_i^{k_i(r)}}(\mathbf{x}).$$

Unlike with kernel methods, there are no strict restrictions about the shape of $\mu_{\tilde{A}_i^k}(x_i)$. The shape could be a function that does not satisfy the symmetry requirements of kernel functions. All the other terms of the fuzzy logic system must be expressed as a weight w_r for rule r such that

$$\hat{y} = \sum_r w_r b_r(\mathbf{x}|\Theta_r).$$

Due to the fact that there is only one output set $\tilde{B}^l$ per rule ($\{\tilde{B}^l|\text{rule } r\}$ are disjoint) we can write

$$\sum_l \sum_{r|l} b_r(\mathbf{x}|\Theta_r) y^l = \sum_r b_r(\mathbf{x}|\Theta_r) y^{l|r}.$$

Notation $y^{l|r}$ denotes the output weight $y^l = \int y \mu_{\tilde{B}^l}(y)\, dy$ for output set $\tilde{B}^l$, which is deduced from the rule $r:\ (y \text{ is } \tilde{B}^l)$. The dictionary interpretation of the fuzzy

logic system can now be written as

$$\hat{y} = \sum_r \underbrace{\left[\underbrace{\frac{1}{\sum_{r'} b_{r'}(\mathbf{x}|\Theta_{r'}) \int \mu_{\tilde{B}^{l|r'}}(y)\, dy}}_{\text{normalization constant}} \cdot \int y \mu_{\tilde{B}^{l|r}}(y)\, dy \right]}_{w_r} \underbrace{\left[\prod_j \mu_{\tilde{A}_{i_j(r)}^{k_{i_j(r)}}}(x_{i_j(r)}) \right]}_{b_r(\mathbf{x}|\Theta_r)}. \tag{16}$$

As a conclusion we can say that our generic fuzzy logic system (10) is equivalent to the dictionary method when each rule corresponds to one item in the dictionary of basis functions. There are no limitations to the shape of membership functions, except the possible requirements due to linguistic interpretation. One additional requirement for the system (16), due to interpretation, is that membership values must be positive, which means that for any $\mathbf{x}$ and Θ_r, $b_r(\mathbf{x}|\Theta_r) > 0$.

5.3.2 Linear weights and linguistic rules

Note that in equation (16) weight w_r replaces the output set. **This implies that many systems that consist of a weighted sum of the basis function could be converted to fuzzy systems and, furthermore, to linguistic rules.**

Recall from 5.2 that each rule has one input membership function for each input variable x_i and one output membership function for its output variable y. The construction of the actual rule is done by computing their truths after the training phase from the set of rule candidates. Thus the number of rule candidates is $(m_i m_y)^d$, where m_i is the number of input membership functions, m_y is the number of output membership functions, and d is the dimension. The rule for one candidate is illustrated in Figure 22.

The problems that come with this construction are mostly practical in nature. Since there will be one rule for every nonzero w_r, we want as many zero weights as possible. In addition, the conversion from the weights to output membership functions may require sophisticated optimization if the shapes and locations of output memberships are adaptive. It may also result that after optimization the configuration of output sets does not give the weights w_r exactly, which degrades the accuracy of the final rules. Often these difficulties do not cause any major problems, as demonstrated in this thesis.

In practice three different approaches will be explained in this chapter. The first approach is linear learning, where weights are interpreted as adaptive output membership functions. The second is a crude approximation of linear adaptive weights with fixed output membership functions. The final approach is regularized linear learning that is expected to eliminate most of the weights automatically.

In the final rules the role of weights might be seen in three ways:

i) Rule picking, where each nonzero or thresholded weight adds one fuzzy rule to the rule system.

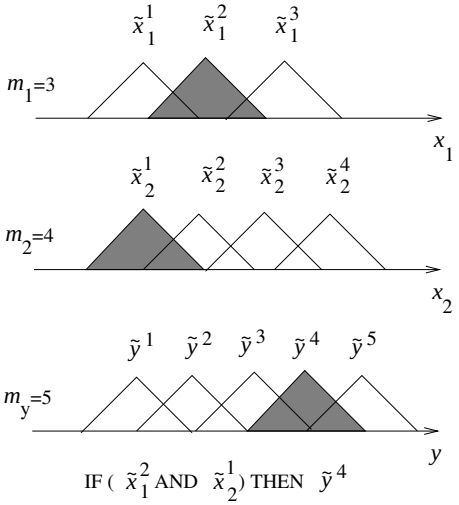

FIGURE 22 The structure of one rule.

ii) Giving the positions of output membership functions, or

iii) giving the scales of input membership functions.

From a linguistic point of view only the first way is safe, but either way ii) or way iii) might be needed in addition to i) to guarantee satisfactory precision of the final rules.

5.4 Fuzzy learning with linear regression

The linear regression was already used in section 4.4.2. Here it is explained more carefully and generalized for multivariate inputs.

In a parameter estimation a suitable loss function $Q(y, f_{\text{fuzzy}})$ is used to seek parameters for the model f_{fuzzy} such that the model output matches the target output well. Loss function penalizes the errors between the output and target. One common choice is the squared error loss $Q(y, f_{\text{fuzzy}}) = (y - f_{\text{fuzzy}})^2$. In this case the solution is known to be

$$f(\mathbf{x}) = \mathcal{E}(Y|\mathbf{X} = \mathbf{x}), \tag{17}$$

where $\mathbf{X}$ and Y are random vector and variable corresponding to the input and output, and $\mathcal{E}(\cdot|\cdot)$ is the conditional expectation.

How can model (16) fit into this framework? Model (16) is linear in its parameters, if only the values w_r are estimated. Therefore, it can be considered as a linear regression problem, for which (17) is the solution. Model (16) may be

written as

$$f(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{b}(\mathbf{x}),$$

where $\mathbf{w}$ is the vector of weights w_r and $\mathbf{b}$ is a vector of basis functions $b_r(\mathbf{x}|\Theta_r)$. The well-known result for linear models is that $\mathbf{w}$ can be estimated using

$$\hat{\mathbf{w}} = (\mathbf{B}^{\mathrm{T}}\mathbf{B})^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{y}, \tag{18}$$

where the $N \times M_{\mathbf{x}}$ matrix $\mathbf{B}$ consists of values $b_{jr} = b_r(\mathbf{X}(j)|\Theta_r)$, N is the number of data points, and $M_{\mathbf{x}} \ll N$ is the number of functions b_r.

Of course, in our case functions b_r are expressed with multivariate input membership functions, which relate the optimization to the fuzzy architecture. There is a total of $M_{\mathbf{x}} = m_1 \cdot m_2 \cdots m_d$ such functions, where m_i is the number of membership functions on x_i axis. Thus b_r is a function in d-dimensional input space

$$b_r(\mathbf{x}) = \prod_{i=1}^{d} \mu_{\tilde{A}_i^{k_{i(r)}}}(x_i). \tag{19}$$

As it was done in section 4.4.2, we can interpret the resulting weight vector $\hat{\mathbf{w}}$ as positions for singleton membership functions such that

$$\hat{\mu}_{\tilde{B}^r}(y|\hat{w}_r) = \begin{cases} 1, & \text{if } y = \hat{w}_r, \\ 0, & \text{otherwise.} \end{cases}$$

Also as before (see 4.4.2), we may assume some predefined width for the membership functions such that the final membership functions are of the type

$$\mu_{\tilde{B}^r}(y|\hat{w}_r) = \begin{cases} \frac{a-|\hat{w}_r-y|}{a}, & \text{if } |y-\hat{w}_r| \leq a, \\ 0, & \text{otherwise.} \end{cases}$$

The problem with this approach is that the number of rules, $M_{\mathbf{x}}$, is huge in high dimensional input spaces, and each rule has a different interpretation on the output axis. The dimension reduction problem is the main topic of Chapter 6, and the readability is discussed in the next section.

5.4.1 A note about the weight values

Without any restrictions the weights $\hat{\mathbf{w}}$ of regression (18) could take any finite value between $-\infty$ and ∞. In our case, however, the weight values are restricted to be on the same range as the output variable y. This restriction is easily satisfied with an additional box constraint in optimization, which leads to a need to use of quadratic solvers. In practice, however, the weights are usually in the desired range automatically, which is due to the piecewise linear structure of input membership functions.

Later, when the weights are used as scaling factors for input membership functions, the output range will be between 0 and 1. This is due to the fact that weights are scaled for output membership functions that take truth values between 0 and 1.

5.5 Selecting output membership functions from a fixed set of candidates

An obvious consequence of the previous linear model is that there will be as many rules, each with one adaptive output membership function, as there are input membership function combinations. Many of these outputs will be heavily overlapping on the y-axis, which brings only a little extra information to the system. Therefore we may quantize (or fuzzify) the real valued positions $\hat{w}_r$ of the y-axis with predefined output fuzzy sets. This also improves the linguistic readability: the user knows the exact positions and widths of the membership functions and the rules formed by such a system are easy to interpret.

The effect of weights can now be utilized in two ways. We may use them to scale the input membership functions (but this might lower the level of interpretation), and we can use them to select correlating combinations of input–output membership functions for rules. Both approaches will be evaluated in the following.

The construction for rule selection is based on $M_{\mathbf{x}} \times M_y$ selector matrix $\mathbf{C}$, which fuzzifies the output positions (input weights w_r) with a discrete set of M_y output membership functions $\mu_{\tilde{B}^l}$. An element of the matrix is therefore

$$c_{rl} = \mu_{\tilde{B}^l}(\hat{w}_r).$$

Most of the elements (depending on the support of the output membership functions) will be zero. Now the actual rule selection can be done in several ways. First, it is possible to eliminate all combinations of multivariate input membership functions b_r and output membership functions $\mu_{\tilde{B}^l}$, where c_{rl} is zero. If only the nearest output membership functions overlap, this would give at most $2^d \cdot M_{\mathbf{x}}$ rules. In many cases, however, it is sufficient to select only one membership $\mu_{\tilde{B}^r}$ with the largest truth:

$$\mu_{\tilde{B}^r} = \max_l \mu_{\tilde{B}^l}(\hat{w}_r).$$

We may also note that further reduction in the number of rules might be possible by using the simplest logically equivalent combination of univariate input membership functions $\mu_{\tilde{A}_i^{k_i(r)}}(x_i)$ instead of multivariate b_r's, but in the current context such an approach is not necessary.

5.5.1 A brief evaluation of the approach

The conversion from linear weights to fixed output membership functions is tested by extending the random function experiment of section 4.4. As before 100 random sin-cos functions were generated and linear weights for nine input membership functions were computed with linear regression.

The tested fuzzy system is now based on the sum-aggregation of rules with CoG (centre of gravity) defuzzification, which was found as the best architecture for the task in section 4.4.

The construction is shown in Figure 23. As an illustration, we study an observation $X(j)$ that enters the fuzzy system. Via fuzzy inference, the system outputs a crisp y-value. Observation is first fuzzified using the input membership functions. Then the "rule matrix" $\mathbf{C}$ is checked to see which output fuzzy sets should be connected to the two activated input fuzzy sets. By inspecting rows 1 and 2 (these correspond to the 1st and the 2nd input fuzzy sets that have been activated) it can be seen that columns 5 and 3 contain nonzero values. So the "pseudorules" are: "IF x is in the 1st input fuzzy set THEN y is in the 5th output fuzzy set" and "IF x is in the 2nd input fuzzy set THEN y is in the 3rd output fuzzy set".

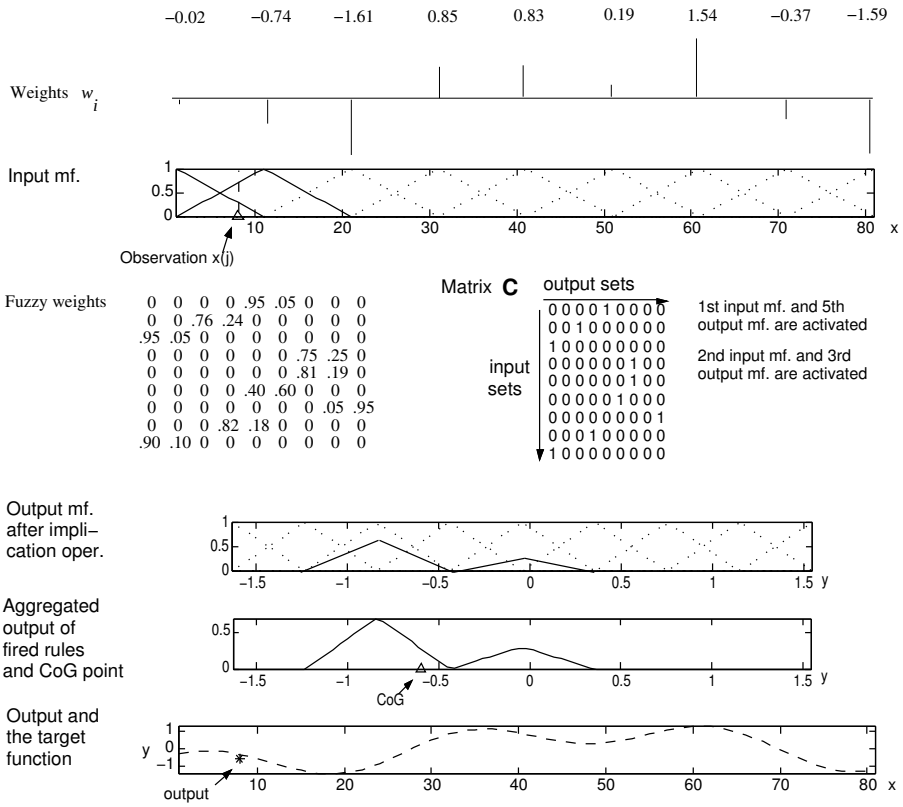

FIGURE 23 Illustration of fuzzy inference with fixed membership functions. Observation "activates" two membership functions, and matrix $\mathbf{C}$ shows which output sets should be used with these two input sets. After an implication operation the consequences are aggregated and CoG point is computed. For a given input value $x(i)$, the crisp output of the system is shown with '*', when the target function was as shown with a dashed line.

The accuracy of rules (compared to adaptive output set experiments in Chapter 4) is shown in Table 7. When the weights are totally omitted and only

one rule for a given input value x, with single fixed output membership function that corresponds to the largest weight is used, the mean prediction error increases from 4.6 to 12.2. However, if the system includes computed weights (scales) for all nonzero input membership functions, then the rules are able to achieve 4.4 mean prediction error.

TABLE 7 Results of an experiment. Mean and deviation prediction errors of linear learning for fuzzy rules.

	Positions of output membership functions are adaptive (by weights)	Output membership functions are fixed
Input membership functions are fixed	4.6 $\pm$7.2 (from Chapter 4)	12.2 $\pm$16.6 (only the best rules used)
Input membership functions are scaled with weights		4.4 $\pm$7.3 Weights scale input membership functions. All rules with nonzero weights are used.

Although the selection of best fixed rule whose accuracy 12.2 is clearly worse than the results of "weighted" rules, it is still better than most of the results in section 4.4. This is quite expected, and we may note that in regression problems it is often better that some form of weighted adaptation is included in the system. It is application dependent whether the scaled input membership functions or the adaptive membership function positions are easier to read linguistically.

5.6 Rule selection via regularization

The statististical learning theory offers some interesting methodologies, which should not be overlooked. Here we shall use the least absolute shrinkage and selection operator, lasso (see Hastie et al., 2001), which is a regularization procedure that can be used to decrease the variance of an estimator. In most applications regularizers like lasso decrease the effective number of parameters by smoothing. In the current context, we apply the idea to decrease the number of rules in a fuzzy system.

The general form of regularized (least squares) estimators can be written in the form (see e.g. Friedman, 1995)

$$\hat{f}(\mathbf{x}) = \arg\min_{g(\mathbf{x})} \sum_{j=1}^{N} [Y(j) - g(\mathbf{X}(j))]^2 + \lambda\varphi[g(\mathbf{x})], \tag{20}$$

where $g(\mathbf{x})$ is the estimator, λ regulates the strength (smoothness) of the solution, and $0 \leq \varphi[g(\mathbf{x})] \leq \infty$ is the penalty functional. To be able to use (20) in the fuzzy context, the fuzzy function $f_{\text{fuzzy}}(\mathbf{x}|\{\mu_{\tilde{A}}\}, \{\mu_{\tilde{B}}\}, R)$ must be written in a form that allows regularization with respect to the number of rules in set R. For simplicity we only do this for the pattern recognition case, where targets y_l are binary and output membership functions $\mu_{\tilde{B}^l}$ are crisp indicator functions for classes $l = 1, 2, \ldots, K$. By constraining the weights into a range between zero and one, we may again use the linear regression for $f(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{b}(\mathbf{x})$ as our model, which allows relatively simple conversion to fuzzy rules. We shall do this in such a way that the number of rules is relative to the number of nonzero weight values.

To minimize the effective number of weights, the lasso method uses L_1 constraint, which can be written in the form

$$\min_{\mathbf{w}_l} \sum_{j=1}^{N} [Y_l(j) - \mathbf{w}_l^{\mathrm{T}}\mathbf{b}(\mathbf{X}(j))]^2 + \lambda \sum_i |w_{li}|, \tag{21}$$

subject to $0 \leq w_{li} \leq 1$. The effect of regularization constant λ is that with increased λ the weights move towards zero. More importantly, when the weight becomes zero, it remains there (this is not the fact with L_2 constraints).

Since minimization (21) is defined for each class l separately, there will be different weights for each class, which prevents us from using "weight scaled" input membership functions. Rather, a set of rules with fixed input membership functions is simply selected by nonzero weights. These rules are of type

Class is l:

IF ($\mathbf{x}$ is $b_{l_1}(\mathbf{x})$)
OR IF ($\mathbf{x}$ is $b_{l_2}(\mathbf{x})$)
$\vdots$
OR IF ($\mathbf{x}$ is $b_{l_{m_l}}(\mathbf{x})$),

where $b_{l_i}(\mathbf{x})$ is a fixed multivariate input membership function of type (19).

The main problem with this approach is that the rule system becomes hard to understand when the number of rules m_l per class increases. This happens easily when the data dimension increases, because the minimization is forced to pick all the needed input combinations separately. A better approach would be to rearrange the rules such that b_{l_i}'s could have a varying number of important input variables. This is the topic of Chapter 6.

5.7 Evaluation of regularized fuzzy learning in pattern recognition

The objective of this simple evaluation is to test how well fuzzy rules, selected with linear model and lasso regularizer, perform with respect to some well-known classifiers. Since the rules are picked from a set of fixed candidates, we may ex-

pect that performance is lower than with fully adaptive learning algorithms. Another problem with the proposed fuzzy learning is that the number of multivariate fuzzy rules is high, although there might be a way to express the same fuzzy logic with a small number of "optimized" combination of univariate membership functions.

The evaluation is done with the fiber data set (see Chapter 2) using three previously selected features from a blob distribution of segmented and thresholded images. The three features, which are known to be expressive for the task, are

- vol_ent (entropia of blob volume distribution),
- ste_mea (mean of blob steepness distribution),
- fou_8 (the amount of high frequencies in image).

This is a small data set with 52 observations from four classes, where there is an uneven number of observations per class: 8 in class 1, 4 in class 2, 20 in class 3, and 20 in class 4. From Figure 24 we may see that the decision regions for classes seem to require nonlinear models. The small number of samples in some classes (4 in class 2) is usually not enough for a classifier building, which will be seen in the results.

Due to the size and type of the data set, the classifiers were built for each class separately. There are implicit priors for the classes (no sample weights were used during the classifier building). Thus the classifier for class 2 has an implicitely high prior to ignore observations, since most of the data points (48/52) do not belong to the class. In evaluation the best class decision is done with majority voting (largest probability of a class, or largest truth when fuzzy method was used). Thus, the evaluation expects that the class priors for test data are the same as they are for the training data. This might not be the case in the real world, but it is acceptable for the current evaluation.

The fuzzy architecture is made using five fixed triangle input membership functions for each of the feature variables, as shown in Figure 25. This gives a total of 125 ($5 \times 5 \times 5$) candidate rules (multivariate membership functions $b(\mathbf{x})$) for the fuzzy system. The input membership functions were positioned on the input axis to form a partition of unity (sum to one at each point x), and fuzzy sets were labeled as "very low", "low", "medium", "high" and "very high". The output membership functions are singletons that express truths of the classes.

From the candidate membership functions a regularized linear (lasso) predictor was built, and rules were constructed from those multivariate input membership functions that corresponded to nonzero weights.

The classification accuracy was tested with leave-one-out cross-validation. The classifier was built with all data, except one observation, and the missing one was predicted by the classifier. This was then repeated over all observations. As a

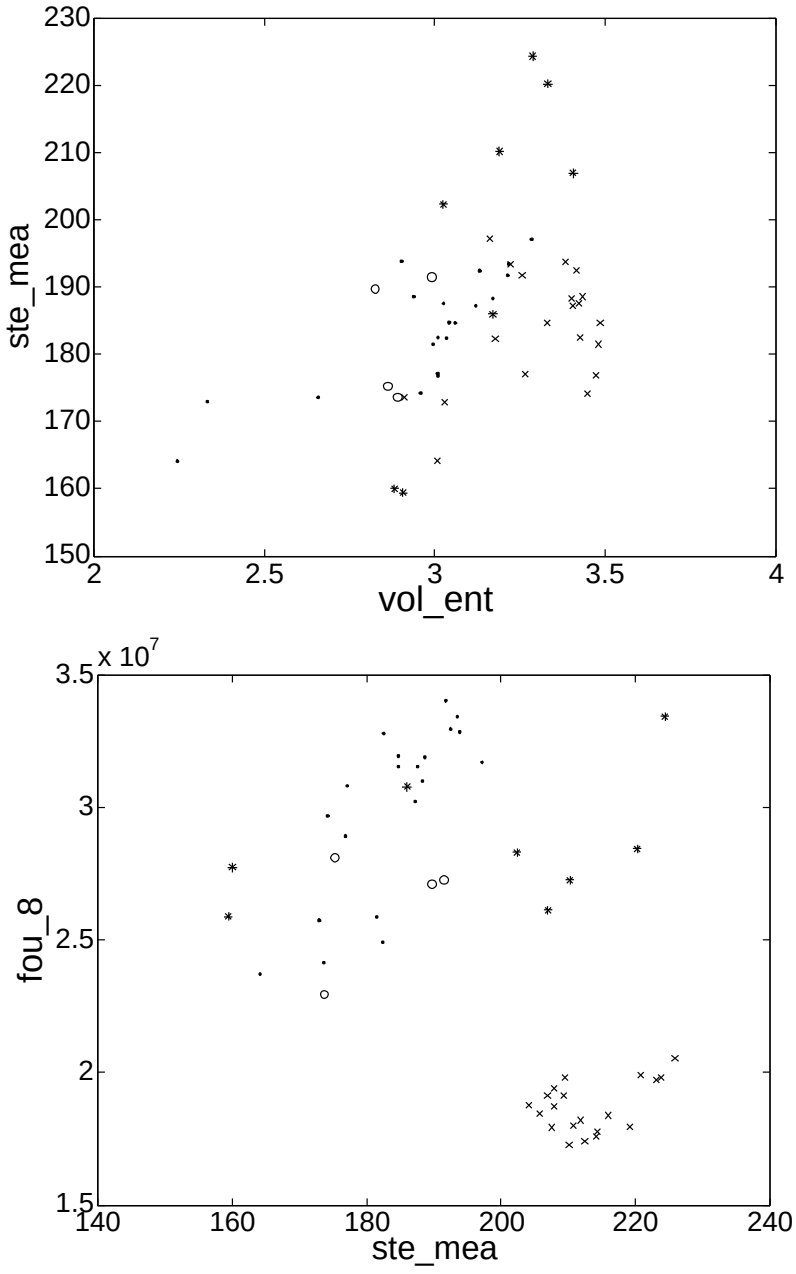

FIGURE 24 Data shown with two two-dimensional dependencies. Classes are marked as follows; class 1: '*', class 2: 'o', class 3: '.' and class 4: 'x'.

result the mean and deviation of predictor was computed with

$$E_{\text{mean}} \approx \frac{1}{4N} \sum_{l=1}^{4} \sum_{j=1}^{N} (Y_l(j) - f_l^{(-j)}(\mathbf{x}))^2,$$

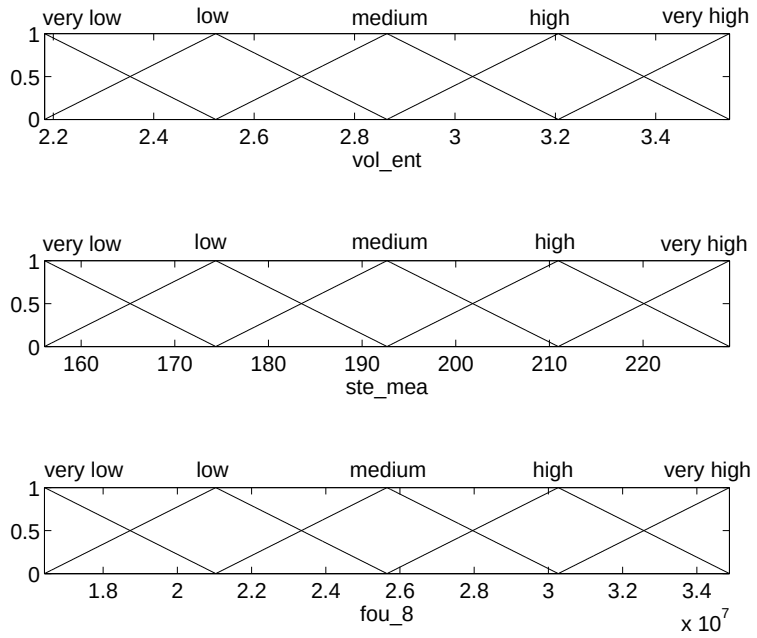

FIGURE 25 The membership functions for features "volume entropy", "steepness mean" and "Fourier 8".

$$E_{\mathrm{dev}} \approx \frac{1}{4(N-1)} \sqrt{\sum_{l=1}^{4} (\sum_{j=1}^{N} Y_l(j) - f_l^{(-j)}(\mathbf{x}))^2},$$

where $f_l^{(-j)}$ denotes the classifier for class l that was built using all data except observation j.

The classification accuracy of the fuzzy classifier and the corresponding number of rules were computed for different values of regularization constant λ. The results are shown as a function of λ in Figure 26. The trade-off between accuracy and interpretability can be seen in the curves; a larger set of rules tends to generate a more accurate result. However, a very large rule set (more than 22 rules) reduces the prediction accuracy. The best mean accuracy 82.7% is achieved when $\lambda = 0.1$ and the number of rules is 22. By relaxing the accuracy requirement (by seting $\lambda = 0.3$) to 80.8%, the number of rules can be reduced to 14 (3 rules for each class 1, 2 and 4, and 5 rules for class 3). The maximum value for λ shown in figures is 1.5. Increasing this value did not cause large deviations from the already observed values; $\lambda = 10$ ended in 8 rules which could classify 73.1% of the samples correctly.

As noted before, the proposed technique is not optimal for readable linguistic rules, because a large number of rules (22 in the best case) is quite difficult to comprehend. One should note, however, that most of the rules could be com-

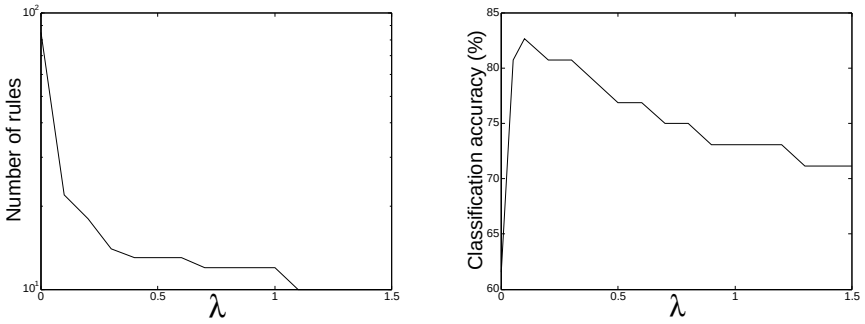

FIGURE 26 The effect of lasso penalty on the number of rules (left) and accuracy (right) in fiber data example (note the logarithmic scale on left figure y-axis).

bined, if multivariate membership functions (each with three variables) were replaced with an equvalent single variable multivariate logic. This is obvious, for example, from the rule system for 14 rules ($\lambda = 0.3$). The rules are given in Table 8. For example, the first row in the table for class 1 corresponds to fuzzy rule "IF vol_ent is medium AND ste_mea is very low AND fou_8 is medium THEN class is 1".

Another observation of the rule base is that it does not generalize the result (extrapolate it) beyond known data range. Opinions about this property (extrapolate or not) differ in the literature. The opinion of the current author is that models that give large posterior probabilities over the real axis in places where there is no data, should be based on human intuition rather than inductive learning. In fuzzy systems such an inclusion of human intelligence is easier than in most other models of learning.

5.7.1 Comparative evaluation

Here the previous results are compared with other classification techniques. The comparative methods are k-nearest neighbours (k-nn), quadratic discriminant analysis (QDA), linear discriminant analysis (LDA), linear regression and multilayer perceptron (MLP), which were briefly explained in Chapter 3.

For each method several experiments were made to optimize the performance of the method. Then the best construction was used for evaluation, which was done with the same leave-one-out cross-validation as before.

The best mean prediction accuracy (86.5%) was obtained by using a k-nearest neighbour classifier (with $k = 1$). Increasing k reduced the result considerably. The performance was 82.7% for $k = 2$, 76.9% for $k = 3$ and 65.4% for $k = 4$. The rather surprising result with $k = 1$ indicates that the best predictor is done by taking the class from the nearest observed data sample. Thus the decision boundaries seem to be highly nonlinear and the classes seem to separate quite

TABLE 8 Summary of fuzzy rules.

Class 1			
Rule	vol_ent	ste_mea	fou_8
1	medium	very low	medium
2	high	high	medium
3	high	medium	high
Class 2			
Rule	vol_ent	ste_mea	fou_8
1	medium	low	low
2	medium	low	medium
3	medium	medium	medium
Class 3			
Rule	vol_ent	ste_mea	fou_8
1	low	low	medium
2	medium	low	high
3	medium	medium	high
4	high	medium	high
3	high	medium	very high
Class 4			
Rule	vol_ent	ste_mea	fou_8
1	high	high	very low
2	very high	high	very low
3	high	high	low

well. Therefore linear methods are expected to work rather badly with the problem, while nonlinear methods might work well, even with complex structures that normally might overlearn.

A summary of the classification results for all methods is given in Table 9. Linear discriminant analysis (LDA) produced the most unsatisfactory result: a prediction accuracy of 73.1%. Quadratic discriminant analysis (QDA) could give the second best prediction accuracy, 84.6%, but it omits class 1 completely. Linear regression achieved 76.9%, which is a bit low value when compared to the best predictors. The application of these techniques was quite straightforward.

However, application of multilayer perceptron (MLP) network was more difficult. The network has several parameters and possibilities which the user must decide:

- the number of weight layers,
- the number of hidden units,
- the type of activation,

- the optimization strategy.

A two layer network (two weight layers) which has three inputs, a predefined number of hidden units and four outputs were selected for the testing. Hyperbolic tangent sigmoid transfer function was used as an activation function in hidden units and linear function as activation function in output nodes. Levenberg-Marquardt algorithm was employed to minimize the SSE function. The network was trained with different number of hidden layer nodes. The networks with four, five and six hidden nodes gave prediction accuracies of 67.3%, 80.8% and 78.9%, respectively. Table 9 shows the results corresponding to the "3-5-4" -network (the bias nodes are not counted). Possibly, better results could have been achieved with more thorough testing of different alternatives. The use of moment terms and regularization might have improved the generalization ability to some degree.

5.8 Conclusions

The basic idea of fuzzy learning with linear methods was presented in this chapter. The main contribution was to show the feasibility of using simple linear learning as a basis for fuzzy rule construction.

Fuzzy rule extraction in classification was also tested by comparing it to the other techniques. Despite the simplicity, fuzzy approach compared quite well in respect to others. The best method in our evaluation, the k-nearest neighbours is not considered readable since when a sample is given to the system for classification it analyzes the data studying the nearest points only. Hence, the classification "rule set" which this system uses is enormous, although not infinite. Other methods, like the multilayer perceptron, hide knowledge in their parameters, which does not allow one to present the knowledge in a linguistic form.

Still, the problem is not completely solved with the proposed technique. When the data dimension increases, the number of multivariate rules grows exponentially. Thus, what was presented in this chapter, lay foundations for the next one.

TABLE 9 Classification results for fuzzy rules, k-NN, LDA, QDA, linear regression and MLP. Note that there were (8,4,20,20) samples in classes (1,2,3,4) respectively.

Actual classes	Predicted classes 1	2	3	4
Fuzzy (made with lasso), 82.7% ±38.2, avg.				
1	**87.5%** ±35.4		12.5%	
2	25.0%	**25.0%** ±50.0	50%	
3	10.0%	15.0%	**75.0%** ±44.4	
4				**100.0%** ±0.0
k -nn, 86.5% ±34.5, avg.				
1	**75.0%** ±46.6		25.0%	
2		**25.0%** ±50.0	75.0%	
3	5.0%	5.0%	**90.0%** ±30.8	
4				**100.0%** ±0.0
LDA, 73.1% ±44.8, avg.				
1	**37.5%** ±51.8	37.5%	25.0%	
2	75.0%	**0.0%** ±0.0		25.0%
3		25.0%	**75.0%** ±44.4	
4				**100.0%** ±0.0
QDA, 84.6% ±36.4, avg.				
1	**0.0%** ±0.0		62.5%	37.5%
2		**100.0%** ±0.0		
3			**100.0%** ±0.0	
4				**100.0%** ±0.0
linear regression, 76.9% ±42.5, avg.				
1	**0.0%** ±0.0		75.0%	25.0%
2		**0.0%** ±0.0	100.0%	
3			**100.0%** ±0.0	
4				**100.0%** ±0.0
MLP, 80.8% ±39.8, avg.				
1	**62.5%** ±51.8	12.5%	25.0%	
2		**25.0%** ±50.0	75.0%	
3	10.0%	10.0%	**80.0%** ±41.0	
4				**100.0%** ±0.0

6 FUZZY RULE EXTRACTION

In the previous chapter fuzzy learning was done using fixed multivariate membership functions, where each rule had exactly the same input variables. This leads to very large rule sets with high-dimensional input vector. The purpose of this chapter is to introduce a new fuzzy rule extraction methodology that is designed to solve highly multidimensional problems with tens of features. The methodology is related to some feature selection techniques, where the objective is to find a suitable minimal set of features that discriminates the observations reasonably well. In our case it is also desired that the role of the features can be expressed in the form of linguistic, human readable rules.

The methodology will be tested in Chapter 7 with real world data and evaluated against other commonly used variable selection methods and classifiers. In this chapter we seek answers, in the context of fuzzy rule construction, to the following questions:

1. How to reduce the dimensionality via variable selection?

2. How to guarantee good accuracy and interpretability simultaneously?

This means that we might, for example, try to eliminate those variables that do not contribute to the target variable, either directly or in the presence of other variables. The task could be to build a predictor with as few variables as possible without reducing the prediction accuracy.

We will assume a supervised feature selection, which means that the target variable is always available. If this is not the case, which is common in many data mining applications, the user should construct the labels with some clustering method. One option is an interactive use of a self-organizing map, where the idea is to map the large-dimensional data onto a small-dimensional latent space (e.g. 2-dimensional lattice). The user may then select the interesting data groups from the computer screen and thus provide the groupings with class labels (e.g. indices). Then the role of the variable selection is to say which variables are behind the user-made classification.

6.1 A short review of fuzzy feature selection and rule extraction

A review of feature selection was already given in Chapter 3. In fuzzy literature the following three approaches are commonly used in the same context:

a) conventional search techniques,

b) neural networks,

c) genetic algorithms.

As an example of case a), Ramze Rezaee et al. (1999) give results of applying exhaustive and sequential backward selection searches to the "fuzzy feature selection" problem.

Case b), the neural networks approach, has been the most popular among fuzzy researchers and the most popular method has been the multilayer perceptron (MLP). Often features are selected by evaluating the weights of the links between features and the hidden layer. If the absolute values of the weights for some feature is near zero, then it can be removed from the feature set. Examples of this are given in Mitra and Hayashi (2000) and Nauck and Kruse (1999). Many of these methods involve some kind of regularization. A three-layer neural network is often used for feature selection, where by adding a penalty term to the error function, redundant connections (with small weights) can be distinguished from the relevant ones. This kind of idea has been considered in Pal and Mitra (1999) and in Pal et al. (2000).

Genetic algorithms have also gained some popularity among fuzzy applications for feature selection and rule extraction (Karr, 1991; Wang et al., 1998; Lekova et al., 1998; Raymer et al. 2000; Casillas et al., 2001; Cordón et al., 2001). González and Pérez (2001) utilize a mutual information criterion and GA for fuzzy rule induction and feature selection. Their learning algorithm (called SLAVE) selects features for each particular rule. As a result each rule may have a different subset of variables to identify the class.

Thawonmas and Abe (1997) suggest the analysis of class regions which are generated by a fuzzy classifier. The degree of overlap in the class regions is defined as the "exception ratio" and is used as a measure for feature evaluation. This measure consists of a ratio between an overlapping hyperbox (hyperbox that is formed as an intersection of two class-specific hyperboxes) and a class hyperbox that surrounds the points which belong to some specific class (corresponds to the fuzzy rule). A hierarchical structure of hyperboxes is proposed. The algorithm begins with all the features and eliminates the most irrelevant feature, which means that it utilizes the backward selection search strategy. Abe et al. (1998) propose a similar approach but they replace the hyperboxes by ellipsoids.

In Tsang et al. (2003) the feature subset is sought such that the overlapping degree of fuzzy membership functions and the size of feature subset are minimized. The search is carried out by a greedy search algorithm. Their method

(OFFSS) works on two classes, and problems with more classes must be transfered to two-class problems.

In the following sections a new method, which utilizes entropy measures to guide feature selection and rule extraction, will be introduced. The idea of using entropy is not new. It has been proposed in Pal and Chakraborty (1986), Ichihashi et al. (1996), Rudas and Kaynak (1998) and Lee et al. (2001). Also the use of entropy for fuzzy rule extraction has been proposed earlier, and some common aspects with the proposed and the ideas expressed in Lee et al. (2001) can be found. However, the method here is considered to be novel. The most visible differences when comparing it to the other methods are:

- the use of a restricted full search in feature selection,
- fuzzy two-part architecture,
- structure of rules,
- the use of class-specific features.

6.2 Description of the problem

For simplicity, the fuzzy feature selection methodology is developed for the pattern recognition (classification) problem. This makes it easier to understand the choices behind the methodology and gives clear objectives for the evaluation. Yet one should note that the underlying fuzzy logic system is made for real valued functions, which works as well for indicator function targets (classification) as for real valued targets (regression).

Now, consider a set of m classes represented by indicator functions $y_1, y_2, \ldots, y_m$, where each class occupies a convex (connected) region in a d-dimensional feature space $x_1, x_2, \ldots, x_d$. Let the region for class l on axis i be expressed as a fuzzy set $\tilde{A}_i^{k(l)}$ and let $\tilde{B}^l$ be the output set for class l. Index function $k_i(l)$ gives the k_ith fuzzy set for class l on input axis i. The rule for class l can now be written as

$$\begin{aligned}&\text{IF } (x_1 \text{ is } \tilde{A}_1^{k_1(l)} \text{ AND } x_2 \text{ is } \tilde{A}_2^{k_2(l)} \text{ AND} \ldots \text{ AND } x_d \text{ is } \tilde{A}_d^{k_d(l)})\\ &\text{THEN } (y \text{ is } \tilde{B}^l).\end{aligned}$$

This corresponds to the computation of strength (see section 4.2) for output class l,

$$v^l = \prod_{i=1}^{d} \mu_{\tilde{A}_i^{k_i(l)}}(x_i). \tag{22}$$

Although the number of rules is small by definition (one for each class), we cannot avoid the curse of the dimensionality (COD) problem. If the data dimension is large we have at least the following two problems:

a) Rules become unreadable. Consider, for example, a rule with one hundred binary terms

$$\begin{aligned}&\text{IF}\,(x_1 = 1,\ x_2 = 0,\ x_3 = 1, \ldots)\ \text{THEN}()\\ &\hat{=}\ \text{IF}\,(\underbrace{1011011\ldots1}_{\#100})\ \text{THEN}().\end{aligned}$$

One can easily argue that it is unreadable.

b) Identification of accurate fuzzy sets $\tilde{A}_i^k$ in high dimensional spaces from real world data is practically impossible. The data is not dense enough to estimate the positions of $\tilde{A}_i^k$ reliably on all axes i.

To solve the problem we propose to use the additive fuzzy partitioning, which allows us to write (22) in the form

$$v^l = \sum_{r=1}^{S^l} \overbrace{\prod_{j=1}^{d_r} \mu_{\tilde{A}_{i_j(r)}^{k_{i_j(r)}(l)}}(x_{i_j(r)})}^{\text{term set } a_r^l}, \tag{23}$$

where $i_j(r)$ is an indexing function for the i_jth variable of rule r for class l. Equation (23) means that one big rule with d terms is divided into S^l smaller rules for class l, with **varying number of input membership functions**:

$$\begin{aligned}r = 1 :\ &\text{IF}\,(x_{i_1(1)}\ \text{is}\ \tilde{A}_{i_1(1)}^{k_{i_1(1)}(l)}\ \text{AND}\ x_{i_2(1)}\ \text{is}\ \tilde{A}_{i_2(1)}^{k_{i_2(1)}(l)})\ \text{THEN}\ (y\ \text{is}\ \tilde{B}^l)\\ &\text{OR}\\ r = 2 :\ &\text{IF}\,(x_{i_1(2)}\ \text{is}\ \tilde{A}_{i_1(2)}^{k_{i_1(2)}(l)}\ \text{AND}\ x_{i_2(2)}\ \text{is}\ \tilde{A}_{i_2(2)}^{k_{i_2(2)}(l)})\ \text{THEN}\ (y\ \text{is}\ \tilde{B}^l)\\ &\text{OR}\\ &\ldots\end{aligned}$$

The assumptions behind this type of partitioning are:

i) The total number of terms in all additive rules is much smaller than the number of original variables. For example, we assume that about a hundred variables could be reduced to less than ten, where only two or three terms would be needed for each subrule r.

ii) We expect to be able to identify those sets $\tilde{A}_i^{k(l)}$ from the example data that are best (most representative) for the class l. This leads to the variable selection that we are seeking.

iii) The identification of positions and the shape of sets $\tilde{A}_i^k$ can be considered as a separate, independent problem. (Actually it is not, but in our methodology we assume it is.)

6.3 A new method for fuzzy rule extraction (FRE)

The idea of the method is to build an additive set of fuzzy rules under feature selection constraints. This starts from simple and reliable rules that are additively joined with more complex and possibly more unreliable rules until a certain stopping criterion is met. To keep the interpretation simple and to seek reliability for the rules some assumptions must be made.

The methodology is based on two assumptions:

i) There is only one fuzzy set $\tilde{A}^l$ for each class l and variable (feature) combination. How these sets could be obtained or interpreted linguistically is explained later. Currently these practicalities are ignored.

ii) There is a set of observations $\{\mathbf{X}(j), \mathbf{Y}(j)\}_{j=1}^{N}$, where $\mathbf{X}(j)$ is the variable (feature) vector and $\mathbf{Y}(j)$ is class identifier vector such that elements

$$y_l = \begin{cases} 1, & \text{if class is l,} \\ 0, & \text{otherwise.} \end{cases}$$

In the following we may use the notations

$$\mathcal{D} = \{\mathbf{X}(j)\}_{j=1}^{N}$$

for all observations and

$$\mathcal{D}^l = \{\mathbf{X}(j) | Y_l(j) = 1\}$$

for the observations of class l.

Furthermore, we may assume that features are in a d-dimensional metric space $\mathbf{x} = \{x_1, x_2, \ldots, x_d\} \in \mathbb{R}^d$ and that observations for class l form a convex region in the data space. These assumptions are not as restrictive as they may seem. The assumptions can be relaxed, if there is a need for expressive methods. In most high-dimensional problems, the assumptions are quite reasonable because in a high-dimensional data case there is very little information for complex decision surfaces.

6.3.1 The objective

The objective is to build a minimal set of rules with a minimal set of terms (fuzzified variables) to classify the observations $\mathcal{D}$ with additive conjuctive (fuzzy AND) type of rules, with reasonable computational effort.

The assumption of the reasonable effort states that the problem must be solvable in polynomial time. In practice this leads to heuristic search methods, including greedy algorithms and time constrained enumeration. An exhaustive search of all possible rules is not solvable in polynomial time.

6.3.2 The rule space

The classifier for class l consists of one or more subrules $r = 1, 2, \ldots$ of the type

$$Rule_r^l: \text{IF } (x_{i_1(r)} \text{ is } \tilde{A}_{i_1(r)}^{k_{i_1(r)}(l)} \text{ AND } x_{i_2(r)} \text{ is } \tilde{A}_{i_2(r)}^{k_{i_2(r)}(l)} \ \ldots) \text{ THEN } (y \text{ is } \tilde{B}^l)$$

The terms of the rule for class l can be of any combination of fuzzified variables

$$\{\tilde{x}_1^l, \tilde{x}_2^l, \ldots, \tilde{x}_d^l\}.$$

Therefore each subrule is an element of a cardinality ordered power set of fuzzified features for class l,

$$\mathcal{A}^l = \{\underbrace{\emptyset}_{a_0^l}, \underbrace{\{\tilde{x}_1^l\}}_{a_1^l}, \underbrace{\{\tilde{x}_2^l\}}_{a_2^l}, ..., \underbrace{\{\tilde{x}_1^l, \tilde{x}_2^l, ..., \tilde{x}_d^l\}}_{a_{M^{\mathrm{TOT}}}^l}\},$$

which is a set of features such that the first elements after the empty set are single variable sets and the last element is the full set of features. The purpose of the ordering is to arrange the rules from the simplest to the more complex ones in respect to the number of terms. This information is used in the rule extraction heuristics.

Using the ordered power set $\mathcal{A}^l$, each candidate subrule can be named as an element a_r^l of the set. The total number of possible candidates (sub-rules) is now

$$M^{\mathrm{TOT}} = \sum_{i=1}^{d} \left(\begin{array}{c} d \\ i \end{array} \right) = 2^d - 1,$$

which grows exponentially fast when the number of variables d increases. It is therefore unrealistic to search through all possible rules. Rather it makes sense to use only the $r^{\max} \ll M^{\mathrm{TOT}}$ simplest ones.

Using the notation a_r^l for candidate variables of $Rule_r^l$ for class l, the rule can now be expressed as

$$Rule_r^l: \text{IF } (\mathbf{x} \text{ is } \tilde{\mathbf{A}}_r^l) \text{ THEN } (y \text{ is } \tilde{B}^l),$$

where the term $(\mathbf{x} \text{ is } \tilde{\mathbf{A}}_r^l)$ is an abbreviation for multidimensional membership

$$\mu_{\tilde{\mathbf{A}}_r^l}(\mathbf{x}) = \prod_{i|x_i \in a_r^l} \mu_{\tilde{A}_i^{k_i(l)}}(x_i).$$

Thus the reader should note that by selecting some term set a_r^l for class l, one also selects the corresponding set of fuzzy sets $\tilde{\mathbf{A}}_r^l$ (for the same class l).

6.3.3 Candidate selection via a fuzzy evaluation index (FEI)

To select subrules from data a criterion for the rule performance is needed. Such a criterion, called the fuzzy evaluation index (FEI), is introduced in this section.

The FEI is an entropy based measure to test the classification performance of the term set a_r^l (fuzzy sets for class l). It is defined via fuzzy classification probabilities $\Pr(l^*|a_r^l, \mathcal{D}^{l^*})$, where l^* denotes any class, including the currently tested class l, and $\mathcal{D}^{l^*} = \{\mathbf{X}(j)|y_{l^*} = 1\}_{j=1}^N$ is data for class l^*.

The definition of FEI is

$$\mathrm{FEI}(a_r^l|\mathcal{D}) = -\sum_{l^*} \Pr(l^*|a_r^l, \mathcal{D}^{l^*}) \log \Pr(l^*|a_r^l, \mathcal{D}^{l^*}). \tag{24}$$

Thus FEI tries to classify all data classes l^* with a term set a_r^l for class l. The optimal result would be that only the correct class is classified well

$$\Pr(l^*|a_r^l, \mathcal{D}^{l^*}) = \begin{cases} 1, & \text{if } l^* = l, \\ 0, & \text{otherwise,} \end{cases}$$

and then FEI will be zero.

The classification probabilities $\Pr(l^*|a_r^l, \mathcal{D}^{l^*})$ of fuzzy rules using a term set a_r^l can be computed via fuzzy membership values. Let us start with a univariate case (a_r^l contains one element $\tilde{x}_i^l$).

For each fuzzified variable $\tilde{x}_i^l$ and class l we wish to express a discriminative performance of the corresponding fuzzy set $\tilde{A}_i^l$ with respect to data samples $\mathcal{D}^{l^*}$ of all classes l^*. This can be done by computing an empirical "fuzzy" classification probability for classes l^*

$$\tilde{\Pr}(l^*|\tilde{x}_i^l, \mathcal{D}^{l^*}) = \frac{1}{N^{l^*}} \sum_{i|X_i(j) \in \mathcal{D}^{l^*}} \mu_{\tilde{A}_i^{k_i(l)}}(X_i(j)),$$

where N^{l^*} is the number of elements in class l^*. Due to the fact that membership valuess for different classes are not normalized and they are not disjoint, it is possible that probabilities **do not sum** to one

$$\tilde{\Pr}(l^*|\tilde{x}_i^l, \mathcal{D}^{l^*}) \neq 1.$$

Therefore the actual probabilities must be normalized

$$\Pr(l^*|\tilde{x}_i^l, \mathcal{D}^{l^*}) = \frac{\Pr(l^*|a_i^l, \mathcal{D}^{l^*})}{\sum_{l'} \Pr(l'|a_i^l, \mathcal{D}^{l'})} = \frac{\sum_{i|X_i(j) \in \mathcal{D}^{l^*}} \mu_{\tilde{A}_i^{k_i(l)}}(X_i(j))}{\sum_{i|X_i(j) \in \mathcal{D}} \mu_{\tilde{A}_i^{k_i(l)}}(X_i(j))}.$$

This is generalized for multivariate cases with a term set $a_r^l = \{\tilde{x}_{r_1}^l, \tilde{x}_{r_2}^l, \ldots, \tilde{x}_{r_d}^l\}$ as follows

$$\Pr(l^*|a_r^l, \mathcal{D}^{l^*}) = \frac{\sum_{j|\mathbf{X}(j) \in \mathcal{D}^{l^*}} \prod_{i|x_i \in a_r^l} \mu_{\tilde{A}_i^{k_i(l)}}(X_i(j))}{\sum_{j'|\mathbf{X}(j') \in \mathcal{D}} \prod_{i|x_i \in a_r^l} \mu_{\tilde{A}_i^{k_i(l)}}(X_i(j'))},$$

where a_r^l is the set of features (variables) as described in section 6.3.2.

An equation of type $-\sum_l p_l \log p_l$ is also known as Shannon entropy (1948) and it describes the uncertainty of the classifier as depicted in Figure 27. Thus, the smaller value of FEI is better. In fuzzy literature an equation of type (24) is sometimes described as a fuzzy entropy when the probabilities come from fuzzy membership values, under the condition that the fuzzy entropy must satisfy the Luca-Termini axioms (see e.g. Zimmermann, 1985). The entropy considered in this work will be based on Shannon's entropy.

Due to the information theory many properties of an entropy measure are known (see Cover and Thomas, 1991). For example, it describes the expected number of misclassifications in our set of examples.

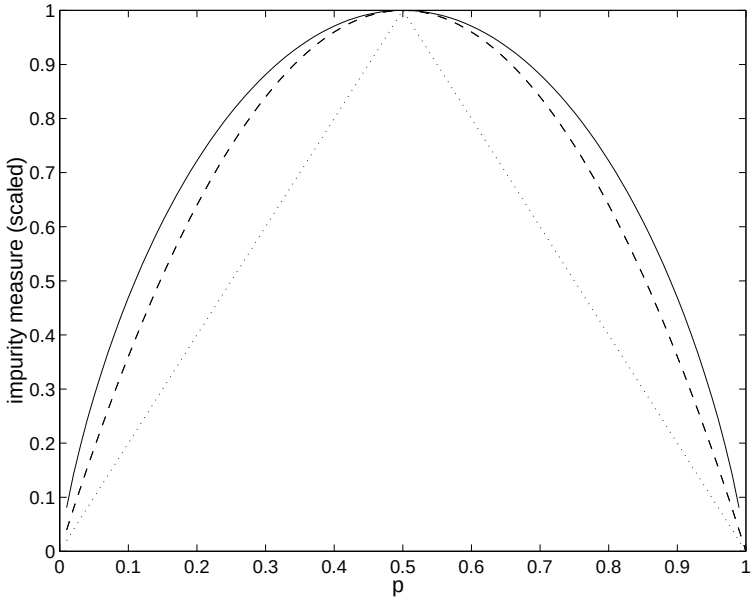

FIGURE 27 Impurity measures for two-class case, $l = (0, 1)$ (see Duda et al., 2001): Shannon's entropy $\sum_l p_l \log p_l$ (solid line), variance impurity $p_0 p_1$ (dashed line) and misclassification impurity $1 - \max_l p_l$ (dotted line).

A potential problem with an entropy based measure is that it also reports good performance when the classifier is always wrong. This is, however, an academic problem only. In practice such classifiers are quite easy to remove from the set of possible solutions.

To illustrate the idea of FEI, consider a two class problem with features (variables) x_1 and x_2, classes 1 and 2 and the corresponding (fixed) fuzzy sets $\tilde{A}_1^1$, $\tilde{A}_1^2$, $\tilde{A}_2^1$ and $\tilde{A}_2^2$, as depicted in Figure 28. We may also assume that fuzzy sets $\tilde{A}^1$ and $\tilde{A}^2$ approximate the distribution of samples in classes 1 and 2, where classes can

be overlapping. Using data $\{\mathbf{x}\} = \{(x_1, x_2)\}$ that consist of samples

$$\mathcal{D}^1 = \{(1,1), (1,2), (1,3), (2,2)\}$$

for class 1, and

$$\mathcal{D}^2 = \{(2,2), (2.5,3), (3,2)\}$$

for class 2, we notice that the discriminative performance of axis x_1 is better than x_2 for both classes 1 and 2 because

$$\text{FEI}(a_1^1|\mathcal{D}) \approx 0.32 < \text{FEI}(a_2^1|\mathcal{D}) \approx 0.67$$

and

$$\text{FEI}(a_1^2|\mathcal{D}) \approx 0.38 < \text{FEI}(a_2^2|\mathcal{D}) \approx 0.68,$$

where $a_1^1 = \{x_1\}$, $a_1^2 = \{x_1\}$, $a_2^1 = \{x_2\}$, $a_2^2 = \{x_2\}$ and $\mathcal{D} = \{X(j)\}_{j=1}^7$.

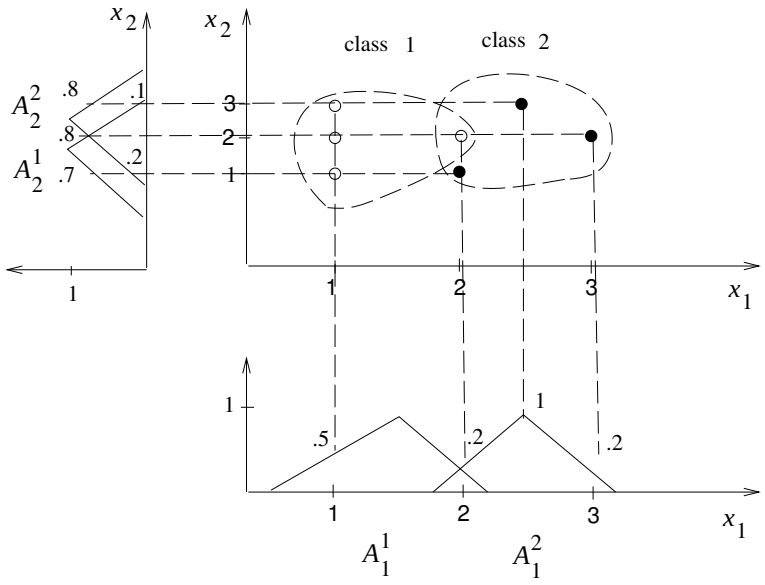

FIGURE 28 The idea of using a fuzzy entropy for feature evaluation. There are two overlapping classes defined on two-dimensional feature space. The problem is to determine the FEI-value for class 1 and 2 with respect to the features x_1 and x_2.

6.3.4 The construction of additive rules

With the help of an ordered power set (term set) of fuzzified features $\mathcal{A}^l = \{a_0^l, a_1^l, a_2^l, \ldots\}$ for class l and the evaluation index $\text{FEI}(a_r^l, \mathcal{D})$ we finally have all the necessary tools for the rule extraction algorithm. The basic idea is to go through the term sets and accept all "good" terms to an additive collection of

conjunctive subrules. The rules for different classes can be constructed independently, but for computational reasons it is often better to compute the term sets for all rules at the same time.

There are several possibilities for practical rule construction algorithms:

i) The fuzzy evaluation index $\mathrm{FEI}(a_r^l, \mathcal{D})$ can be used to prune all term sets a_r^l that are considered too unreliable. This is introduced via threshold θ^{FEI} such that all accepted term sets of rules satisfy

$$\mathrm{FEI}(a_r^l, \mathcal{D}) < \theta^{\mathrm{FEI}}.$$

A typical value for θ^{FEI} is 0.6 for the case of two classes. For more classes this value must be multiplied by mc, where m is the number of classes and $c = -0.5\log 0.5$ is a constant.

ii) Due to computational and linguistic reasons the final number of accepted terms must be limited to some maximum $r^{\max}$.

iii) We may (or may not) test the actual increase of classification performance before accepting a new rule (term set a_r^l) to the active collection of rules. This could be done via the score function that is introduced in section 6.4.1.

iv) The candidates can be tested using the order of

   a) complexity (number of terms in the set a_r^l),

   b) reliability (small FEI's first),

   c) actual classification performance (or increase of it).

In this work the most commonly used (and perhaps the simplest) algorithm is based on possibilities i), ii), and ivb). Thus the actual rule performance in classification is not tested explicitly during the construction of the classifier, only the reliability of the rule is. There are, however, many implicit factors, like the specification of fuzzy sets $\tilde{A}_i^l$ that contribute to the performance.

6.3.5 The FRE algorithm

The fuzzy rule extraction algorithm can now be given as follows. The classifier for class l can be expressed as a collection of term sets

$$\Omega^l = \{a_{r_1}^l, a_{r_2}^l, \ldots, a_{r^{S^l}}^l\}.$$

The idea is to evaluate $r^{\max}$ terms from the power set $\mathcal{A}^l$ of all term sets and select the best for the rules of the classifier as follows.

1. Select $r^{\max}$ simplest term sets $a_1^l, a_2^l, \ldots, a_{r^{\max}}^l$ (smallest numbers of terms $\tilde{x}_i^l$) from the power set $\mathcal{A}^l$.

2. The term sets are ordered according to FEI using data $\mathcal{D}$ such that

$$\mathrm{FEI}(a^l_{r_1}) < \mathrm{FEI}(a^l_{r_2}) < \ldots < \mathrm{FEI}(a^l_{r^{S^l}}) < \ldots < \mathrm{FEI}(a^l_{r^{r^{\max}}}).$$

3. The number of S^l first term sets are used as subrules. In practice S^l can be given, or it can be determined as the biggest element that is smaller than some given threshold θ^{FEI}

$$S^l = \mathrm{argmax}_i\{\mathrm{FEI}(a^l_{r^i}) < \theta^{\mathrm{FEI}}\}.$$

4. Construct the linguistic form of the additive fuzzy classifier (see section 6.2) from the collection of term sets. This is explained in section 6.5.1.

6.4 Using FRE as a classifier

Sometimes, in addition to the linguistic explanation of data, we like to use FRE as a classifier. Although we could use the rule set as it is, we also have an additional support for the decision, the reliability of the rules FEI. This can be combined to the classification result in order to select the most reliable output set $\tilde{B}^l$ according to the matching rules.

The classifier for sample $\mathbf{x}$ can be defined via the score function such that

$$\text{best class } l = \mathrm{argmax}_{l'}\{\mathrm{Score}(l'|\mathbf{x}, \Omega^r)\},$$

where Ω^l is the set of rules (set of term sets $a^l_{r_i}$) for class l.

6.4.1 The Score function

One should recall that each rule r evaluates the truth that a given sample $\mathbf{x}$ belongs to class l:

$$\text{IF } (\mathbf{x} \text{ is } \tilde{\mathbf{A}}^l_r) \text{ THEN } (y \text{ is } \tilde{B}^l).$$

The outcome of the rule is a fuzzy truth (or strength) v^l which indicates that l is the correct class for $\mathbf{x}$, but the other measure $\mathrm{FEI}(a^l_r|l)$ explains how reliable the rule r is. Thus our combined "fuzzy" belief should use both information simultaneously. To do this we can combine the fuzzy outcome and FEI as a combined measure Score as follows

$$\mathrm{Score}(l|\mathbf{x}, \Omega^r) = \sum_{r=1}^{S^l} \frac{v^l_r(\mathbf{x})}{\mathrm{FEI}(a^l_r|\mathcal{D})} = \sum_{r=1}^{S^l} \frac{\mu_{\tilde{\mathbf{A}}^l_r}(\mathbf{x})}{\mathrm{FEI}(a^l_r|\mathcal{D})}$$

$$= \sum_{r=1}^{S^l} \frac{\prod_{i|x_i \in a^l_r} \mu_{\tilde{A}^{k_i(l)}_i}(x_i)}{\sum_{l^*} \Pr(l^*|a^l_r, \mathcal{D}^{l^*}) \log \Pr(l^*|a^l_r, \mathcal{D}^{l^*})}.$$

Thus the score measures how well $\mathbf{x}$ is classified by the set of rules for class l. The bigger the score is, the more certain we are that l is the correct class for $\mathbf{x}$. Optimally $v^l \geq 1$ and $\mathrm{FEI}(a_r^l|\mathcal{D}^l) = 0$, which leads to infinity value for score.

Due to the nature of the score function, the classifier tends to select class l according to one very reliable subrule a_r^l that deduces l as the most "truthful" output.

If necessary, the score function could be made more comparable via a logistic activation function, that can be interpreted as the output truth y^l for class l:

$$y^l = \frac{g(\mathrm{Score}(l|\mathbf{x}, \Omega^l))}{\sum_{l^*} g(\mathrm{Score}(l|\mathbf{x}, \Omega^l))},$$

where $g(a) = \frac{1}{1+e^{-a}}$.

6.5 The identification of input fuzzy sets $\tilde{A}_i^l$ from data

The FRE method assumes that the fuzzy sets $\tilde{A}_i^{k_i(l)}$ are given in advance. In some applications it is necessary to construct adaptive input fuzzy sets $\tilde{A}_i^{k_i(l)}$ from data (the other possibility is to use human expertise). The objective is then to find such sets $\tilde{A}_i^{k_i(l)}$ that reasonably approximate the true behavior of data. To achieve this consider a class of samples $\mathcal{D}^l = \{\mathbf{X}(j)|y_l = 1\}_{j=1}^N$, which we wish to represent with fuzzy memberships $\prod_{i=1}^d \mu_{\tilde{A}_i^{k_i(l)}}(x_i)$. Assuming that $\mathcal{D}^l$ is approximately convex, the data can be represented with scalar sets $\tilde{A}_i^{k_i(l)}$ that cover the range of data on axes $i,\ i = 1, 2, \ldots, d$. It is then apparent that we may take the marginal density distribution $f_{X_i}(x_i|l)$ of class l as a guideline for building membership functions. This can be stated as follows: The more points that exist in some subregion $[\alpha_i, \gamma_i]$ on the real axis x_i of class l, the more confident we can be that the point in that subregion is a member of that particular class. This strategy combines statistical and fuzzy logic viewpoints to achieve our objectives, i.e. to model $f_{X_i}(x_i|l)$ by a fuzzy membership function $\mu_{\tilde{A}_i^{k_i(l)}}(x_i)$.

As usual, the shape of the membership function affects both the linguistic interpretation and the representation accuracy of the approximation. For simplicity, in the following we consider only unsymmetrical triangles (see Figure 29). The symmetrical triangle is naturally a special case of these.

These sets are controlled by three control points $[\alpha, \beta, \gamma]$ that define the left, centre and right positions of the membership function.

The centre of the fuzzy set is selected to be the mean of those of data points that belong to the class l:

$$\beta_{i,l} = \frac{1}{N_l} \sum_{X_i(j) \in \mathcal{D}^l} X_i(j), \tag{25}$$

where N_l is the number of points in $\mathcal{D}^l$. The membership function for a triangular

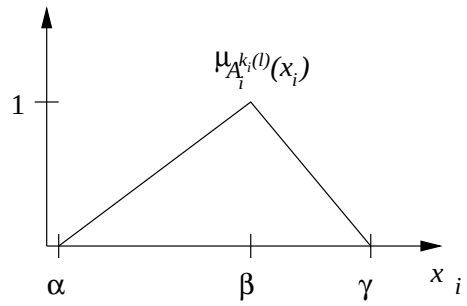

FIGURE 29 Unsymmetrical fuzzy membership function $\mu_{\tilde{A}_i^{k_i(l)}}(x_i)$.

shape, centered on the class mean can be expressed as

$$\mu_{\tilde{A}_i^{k_i(l)}}(x_i) = \begin{cases} (x_i - \alpha_{i,l})/(\beta_{i,l} - \alpha_{i,l}) & , \alpha_{i,l} < x_i \leq \beta_{i,l}, \\ (\gamma_{i,l} - x_i)/(\gamma_{i,l} - \beta_{i,l}) & , \beta_{i,l} < x_i < \gamma_{i,l}, \\ 0 & , \text{otherwise}, \end{cases} \tag{26}$$

where $\beta_{i,l}$ is the class mean, $\alpha_{i,l}$ and $\gamma_{i,l}$ define the support for the fuzzy set as shown in Figure 29. Note that if $|\beta_{i,l} - \alpha_{i,l}| \neq |\gamma_{i,l} - \beta_{i,l}|$ the membership function is nonsymmetrical.

In the usual implementation the control points $\alpha_{i,l}$, $\beta_{i,l}$ and $\gamma_{i,l}$ are selected such that

$$\alpha_{i,l} = \min_j \{X_i(j) | X_i(j) \in \mathcal{D}^l\}$$

and

$$\gamma_{i,l} = \max_j \{X_i(j) | X_i(j) \in \mathcal{D}^l\}.$$

For unreliable data (with outliers) min and max should be replaced with, say 5% and 95% fractiles. For reliable data the support could be widened to increase the generalization ability of the method.

6.5.1 Making adaptive fuzzy sets readable via fixed labels

The problem with adaptive fuzzy input sets is that their readability might be lost. Therefore we shall consider a case with an additional set of preselected linguistic input labels as shown in Figure 30.

The usage of fixed labels can be done by representing the adaptive fuzzy variable via fixed labels such that there is a minimal loss of accuracy. This can be written, e.g. as a simple optimization of the type

$$\min_{w_k, w_k > 0} \int (\mu_{\tilde{A}_i^{k_i(l)}}(x_i) - \sum_k w_k \mu_{\tilde{A}_{L_k}}(x_i))^2 dx_i,$$

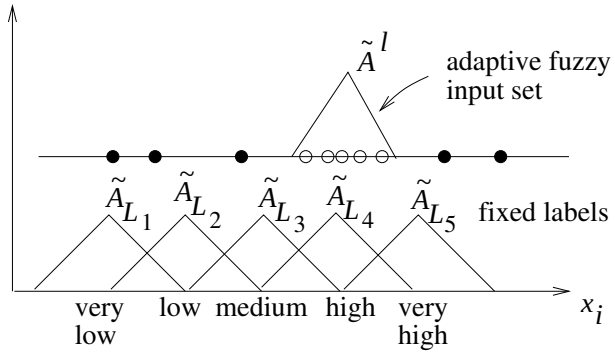

FIGURE 30 Adaptive fuzzy input sets and fixed labels.

which leads to a selection of those fixed labels $\tilde{A}_{L_k}$ for $\tilde{A}_i^{k_i(l)}$ that have a large weight w_k. All output classes $l = 1, 2, \ldots, m$ can share the same fixed input labels $\tilde{A}_{L_k}$, while adaptive inputs $\tilde{A}_i^{k_i(l)}$ are different for each class l.

6.6 Computational complexity and other implementation issues

Although the proposed methodology is based on the full enumeration of the term sets, it is relatively efficient because the computations are limited to the first $r^{\max}$ term sets only. Also the evaluation of a term set can be made rather fast by keeping those term sets in the memory that will be used in other computations. One notable difference to other learning algorithms is that the data is not used iteratively. When the probabilities $\Pr(l^*|a_r^l, \mathcal{D}^{l^*})$ are computed for all tested term sets a_r^l and classes l^* data is not needed any longer. One could also find clever ways to compute the responses $\prod_{i|x_i \in a_r^l} \mu_{\tilde{A}_i^{k_i(l)}}(X_i(j))$ when the computation is done from the simplest to more complex term sets, but this technique is not used in this thesis.

If one or more of the classes at some time get a zero entropy (an exact rule for the class is found), they could be marked as "not to be included" in further calculations. In practice it is difficult to give an exact generic formula for the computational cost, but we can study it experimentally. The results indicate that the computational complexity increases linearly with respect to the number of data points, while the increase as a function of the dimension is more of an exponential type (Figure 31). The artificial data was generated by forming two normally distributed classes, which were made to overlap each other. By changing the number of data points, the dependency on data points could be estimated. The dependency on dimension was found by generating two $5, 10, 15, \ldots, 100$ dimensional normally distributed classes, each of which had an equal and fixed number of data points.

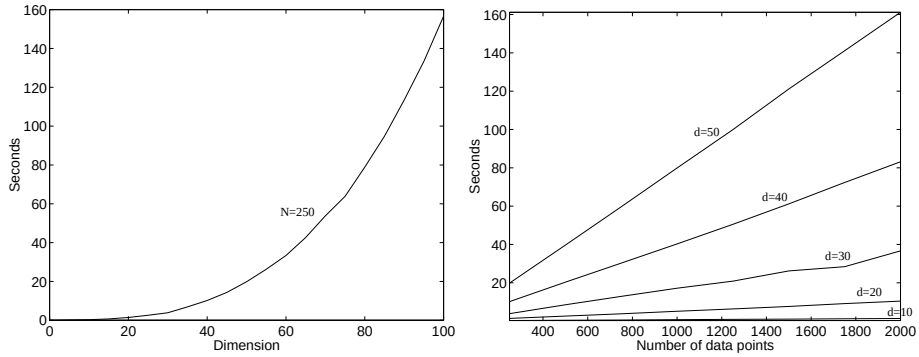

FIGURE 31 Left: Time increases exponentially with respect to dimension (N is number of data points). Right: Computation time is a linear function of number of data points (d denotes dimension).

6.7 An illustrative example with Iris data

The purpose of this simple example is to show how FRE can be used in practice, rather than evaluate the algorithm. Iris data is used because it is well known and easy to experiment with. However there is a problem with this data. In the literature there are at least two versions of the iris data set available: one of which is allegedly the original set and a slightly simplified version of it. The problem is that in literature test results have been reported for one or the other, but it is hard to find out which of them has been used. Also, it is difficult to verify if the data set at hand is the original (See Bezdek et al., 1999). For this reason we used both versions to test the rule extraction. The data and corresponding linguistic labels are illustrated in Figure 32.

The simplest basic version of FRE was used in the tests such that only 3 rules for each class were allowed. For all classes triangular memberships were used as explained in section 6.5.

The following set of the best rules for the **original iris data** was extracted

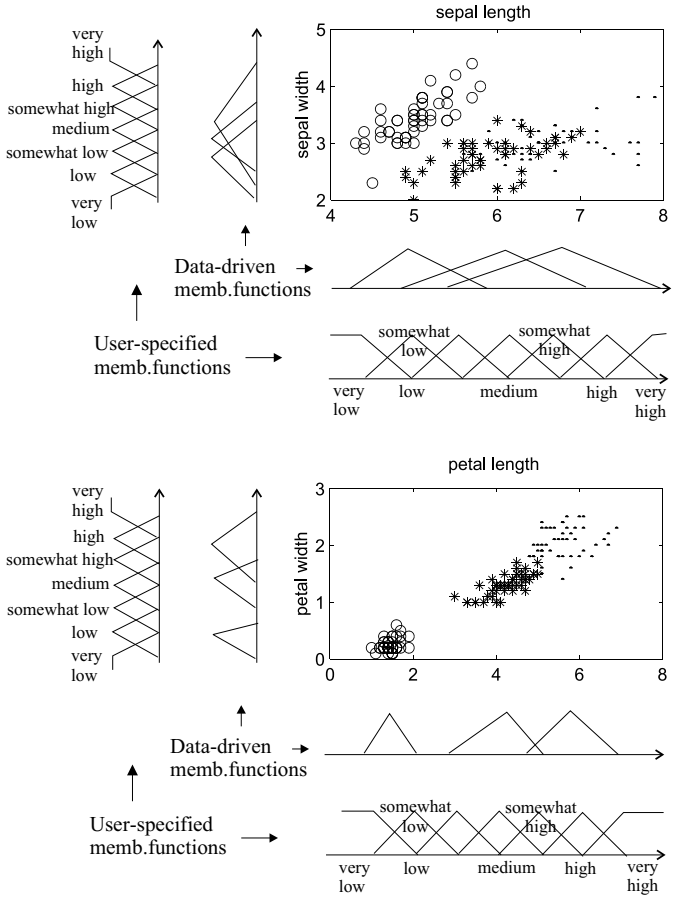

FIGURE 32 Iris data plotted with respect to two input variables.

(more rules were formed but here only the best are shown):

IF		petal_length is in	$[1.0, 1.9]$	(VERY_LOW)
THEN		class is 1		(EXACT rule)
IF		sepal_length is in	$[4.9, 7.0]$	(MEDIUM)
	AND	petal_length is in	$[3.0, 5.1]$	(MEDIUM)
	AND	sepal_width is in	$[2.0, 3.4]$	(LOW)
	AND	petal_width is in	$[1.0, 1.8]$	(MEDIUM)
THEN		class is 2		(RELIABLE rule)
IF		sepal_width is in	$[2.2, 3.8]$	(SOMEWHAT_LOW)
	AND	petal_length is in	$[4.5, 6.9]$	(HIGH)
	AND	petal_width is in	$[1.4, 2.5]$	(HIGH)
THEN		class is 3		(RELIABLE rule)

When the simplified Iris-data was used the rules changed a little. The following rules were formed (the best rule for each class is shown):

IF		petal_length is in	$[1.0, 1.9]$	(VERY_LOW)
THEN		class is 1		(EXACT rule)
IF		petal_length is in	$[3.0, 5.0]$	(MEDIUM)
	AND	petal_width is in	$[1.0, 1.7]$	(MEDIUM)
THEN		class is 2		(EXACT rule)
IF		sepal_width is in	$[2.5, 3.8]$	(SOMEWHAT_LOW)
	AND	petal_length is in	$[4.8, 6.9]$	(HIGH)
	AND	petal_width is in	$[1.4, 2.5]$	(HIGH)
THEN		class is 3		(EXACT rule)

From both iris rule sets it is clearly visible that class 1 can be distinguished by small petal lengths, while petal lengths and widths seem to be largest for class 3. In the case of simple data sets like iris this is easy to verify visually (see Figure 32). Thus the real benefit of FRE is with truly high-dimensional data sets, where good visual displays are hard to make.

In the case of iris data, the classification accuracy is less important than linguistic interpretation. We may still note that FRE is comparative to other methods. The classification accuracy of FRE was estimated by a cross-validation. On each run 90% of the data (randomly chosen) was used for training, while the remaining portion of the data was left for testing. The final estimate of the classification accuracy was calculated by taking the mean of the separate accuracies. In the case of the original iris data a classification rate of 96.7% could be achieved. In the case of simplified data the classification rate was 98.7%. FRE was also tested against some other classifiers (without explicit feature selection). In summary (see Table 10), the classification rate of FRE was better than that of C5.0 (successor for C4.5, see Quinlan, 1993b) and CART (Breiman et al., 1984) but distinctly worse than the result obtained by support vector machines (Vapnik, 1995; Chapelle et al., 1999), which could easily achieve, depending on the parameter setup, a 1–3% classification error, and at best less than 1% error (more details can be found from Niittylä, 2000b). More careful evaluation and comparison of FRE and other methods, including feature selection will be given in Chapter 7.

6.8 Conclusions

In general terms the purpose was to build a system that finds the best possible explanation for a class of observations. It is understood that "the best possible explanation" is the simplest or shortest way of expressing an acceptable solution to a classification problem using some predefined classifier. The explanation was

TABLE 10 Mean classification results for iris data.

Method	Classification accuracy
FRE	96.7%
CART	95.7%
C5.0	94.6%
SVM	99.3%

composed of a set of fuzzy rules that indicate what type of feature combination can be used as a hypothesis of a class.

As a separate problem the construction of fuzzified input variables was also discussed. In practice, the membership functions that are used to fuzzify input valuess must be adaptive to achieve good classification performance.

To increase readability of the rules a way to express them by using additional linguistic labels was also proposed. The main motivation for this two-part (adaptive and fixed) rule set was that it supports simultaneously precise but a low interpretability solution and the suboptimal but readable solution.

7 EXPERIMENTING WITH DATA SETS

This chapter presents an empirical evaluation of Fuzzy Rule Extraction (FRE) methodology, which was described in Chapter 6. The evaluation aims to show that FRE methodology is a feasible choice for the analysis of high-dimensional data sets. This implies that the methodology supports both qualitative aspects like data descriptions, as well as quantitative aspects like good prediction accuracy.

The two aspects are studied simultaneously using the fiber data set, where the practical objectives are to

- Explain data, i.e. find out what the most important features are in the data set for different classes of observations.
- Build a classifier and evaluate its performance.

The first objective requires data analysis with human readable descriptions. It is not enough to only give the names of variables that are selected using some quantitative criterion. Rather the importance of variables must be based on interpretation of the role of variables also. The second objective is easier. We may quantify the mean prediction error and standard deviation of prediction empirically using cross-validation methods and compare the result directly with the results of other methods and with several publicly available data sets.

7.1 Methodological comparison

The construction of FRE combines three methodological approaches: variable selection, descriptive rules, and a classifier. This makes the comparative evaluation of methodology challenging. It is practically impossible to evaluate FRE against all combinations of feature selection methods and classifiers. Instead, a very basic feature selection method, sequential forward selection (SFS), is used with several classifiers. These classifiers are

- linear discriminant analysis (LDA),

- quadratic discriminant analysis (QDA),
- linear regression,
- logistic regression,
- multilayer perceptron (MLP),
- k-nearest neighbors (k-nn).

The classifiers are representatives from three different types of methods: discriminants, parametric posterior predictors, and nonparametric posterior predictors. It is characteristical for discriminants that classifier training focuses only to finding the optimal class boundaries. The posterior predictors, on the other hand, try to give the posterior probability $\Pr(l|\mathbf{X}(j))$ for an observation $\mathbf{X}(j)$ over the whole sample space. The difference between parametric and nonparametric predictors is that parametric ones are identifiable, while there is no unique solution for nonparametric ones. Yet nonparametric methods are more flexible and they are based on less assumptions about the data set. Thus the comparative evaluation may inherently reveal some properties of data if some type of classifiers are clearly better than others.

7.1.1 Single class or Multi-class?

Two types of performance evaluations are used in this thesis: one with single class decisions, and another with multi-class decisions. In both evaluations classifiers are built via variable selection (sequential forward selection, SFS) but the former uses separate classifiers for each of the classes, while in the latter method all classes share the structure of the classifier. The use of single class decisions is more complicated and it is reserved for the analysis of fiber data only.

The performance of multi-class decisions is evaluated with all methods and all data sets, including fiber data. This evaluation is quite straightforward and it is limited to quantitative evaluation only. Thus, it does not answer all objectives of the developed fuzzy rule extraction methodology.

7.1.2 Classification accuracy with single-class decisions

FRE algorithm builds separate rule sets for each of the classes $l = 1, 2, \ldots, m$. Therefore a fair evaluation uses similar approaches for comparative classifiers also. With each method, single-class classifiers $g_l(\mathbf{x})$ are built for each class separately. This makes it possible to use different variables for different classes, which supports the qualitative interpretation of the data set.

The drawback is that the construction of combined, best class detector for a given observation $\mathbf{X}(j)$ becomes difficult. Especially, there is no the best way to combine the results of several single class discriminant methods, which use

different variables in the feature space. The problem is not as severe with posterior predictors. One could, in principle, look the highest posterior probabilities to select the class.

In this work the combination is done according to the following approach, which allows the computation of prediction accuracy and confusion matrix.

- In discriminant methods the problem is treated as a two-class case which means that two discriminant functions are built (one for the class and one for the observations outside the class). If the class-specific discriminant function gives a value that is larger than the value produced by the other discriminant function, the observation is considered to belong to the class (with full membership 1). All the memberships used consist of either 0's or 1's.

- The output of regression methods is interpreted as a posterior probability of the class. If it is larger than 0.5, the observation is taken to belong to the class with full probability. Otherwise the observation is assumed not to belong to the class. As with discriminant methods this information is coded only with either 0 or 1.

The construction of multi-class decisions, based on single class predictions is done as follows.

Let $\{\mathbf{x}, \mathbf{y}\}$ be the data set, where element $y_l = 1$ is the indicator for the true class of $\mathbf{X}(j)$.

1. First, all class-specific decisions, made by classifiers $g_l(\mathbf{X}(j))$, are coded with class indicators $\mathbf{1}_l(j)$, which states for class l and observation $\mathbf{X}(j)$

$$\mathbf{1}_l(j) = \begin{cases} 1, & \text{if } g_l(\mathbf{X}(j)) \text{ decides class } l, \\ 0, & \text{otherwise.} \end{cases}$$

   Note that there can be more than one nonzero indicators (classes) for observation j. It is possible that all indicators are zeros: $\forall l' \ \mathbf{1}_{l'}(j) = 0$.

2. The prediction (truth) of a class l_p is now

$$\rho_{l_p}(j) = \begin{cases} \frac{1}{\sum_{l'} \mathbf{1}_{l'}(j)}, & \text{if } \mathbf{1}_{l_p}(j) = 1, \\ \frac{1}{m}, & \text{if } \forall l' : \mathbf{1}_{l'}(j) = 0, \\ 0, & \text{if } \exists l' : \mathbf{1}_{l'}(j) = 1 \text{ and } \mathbf{1}_{l_p}(j) = 0, \end{cases}$$

   where m is the number of classes. Thus it is assumed that for a non-unique decision, the truth of indicators is divided between candidate classes, where $\mathbf{1}_{l'}(j) = 1$. If the decision is zero for all the classes, then truth is divided between all the classes.

3. The prediction probabilities for classes with respect to true class l_T are now

$$\Pr(l_p|l_T) = \frac{1}{N_{l_T}} \sum_{j:Y_{l_T}(j)=1} \rho_{l_p}(j),$$

where N_{l_T} is the number of observations in true class l_T and $y_{l_T}(j)$ is the indicator of the true class l_T.

4. The total prediction probability is now

$$\mathrm{P_{TOT}} = \frac{1}{N} \sum_{l'} N_{l'} \mathrm{Pr}(l'|l'),$$

where N is the total number of observations and $N_{l'}$ is the number of observations in class l'.

The probabilistic interpretation of prediction truth ρ_{l_p} is that a non-unique prediction is solved by random selection between the classes. The values $\mathrm{Pr}(l_p|l_T)$ can be interpreted as elements of a confusion matrix. In this matrix each row shows how the observations from a true class are distributed by the classifiers between all classes. The diagonal of the matrix corresponds to the correctly classified observations.

Note that class priors are automatically included in the performance computations because all classes are evaluated over the whole data set. There can be unequal number of observations per class, and these proportions must be the same for both classifier training and performance evaluation.

7.2 Data sets and preprocessing

A total of six data sets were used in the evaluation. The deepest evaluation was done with fiber data, including both qualitative data analysis, as well as a test for prediction accuracy. This was done using a total of 48 features from fiber images (blob shapes and Fourier spectrum bands). The other data sets, Fisher's iris plants data, Sonar data, Wine recognition data, Vowel recognition data and Glass identification data, which were explained in Chapter 2, were used in classification experiments only.

The FRE methodology uses raw data without any preprocessing. For all other methods preprocessing is essential for good performance. After some experimentation data whitening (standardization) was used to preprocess all observations:

$$\mathbf{X}^{\mathrm{pre}}(j) = \mathbb{A}^{\frac{1}{2}} \mathbb{U}^{\mathrm{T}} (\mathbf{X}(j) - \bar{\mathbf{x}}),$$

where $\mathbb{A}$ is a diagonal matrix of data eigenvalues, $\mathbb{U}$ is a matrix of corresponding eigenvectors and $\bar{\mathbf{x}}$ is the mean vector of observations.

The results of all data sets were validated using leave-one-out cross-validation, except sonar and vowel data, which came with separate data sets for classifier building and validation.

7.2.1 Validation of results

All evaluations, except with vowel and sonar data, were done with leave-one-out cross-validation. This is a computationally expensive way to estimate the mean prediction probability

$$\mathrm{P}_{\mathrm{mean}} = \frac{1}{N}\sum_{j=1}^{N} \Pr(l|\mathcal{D}^{(-j)}, \mathbf{X}(j))$$

and its deviation

$$\mathrm{P}_{\mathrm{std}} = \frac{1}{N-1}\sqrt{\sum_{j=1}^{N} (\Pr(l|\mathcal{D}^{(-j)}, \mathbf{X}(j)) - \mathrm{P}_{\mathrm{mean}})^2},$$

where $\Pr(l|\mathcal{D}^{(-j)}, \mathbf{X}(j))$ is some estimated prediction probability of true class l with observation $\mathbf{X}(j)$, and the classifier was built with observation j excluded from the training set.

The same validation was also used in the SFS algorithm to find the best variables for the class. This might be dangerous, because there is a dependency between the selected "best" variables and the evaluation statistics. Therefore an additional test was done with one of the most flexible methods (k-nn) to estimate if variable selection could increase optimism in the classification results. This was done by two-level cross-validation, where the cross-validation procedure inside a variable selection was itself cross-validated with independent leave-one-out-samples. Since the comparison between prediction results (see Table 16) with one-level cross-validation is not significantly different (there is 0.9% difference between the two prediction accuracies) we do expect that the estimated prediction accuracies are reliable enough.

7.3 Analysis of fiber data

There are 48 features in this data set: 8 features from radial power spectra, and 40 features from distributions of blob shapes, taken from thresholded images (see Chapter 2 for details). It is expected that a small subset of these features is enough to classify data reasonably well into four quality classes with different prior probabilities (see Table 11). A simple computation with priors show that we may expect mean prediction probabilities and deviations for classes with random selection over the whole data to be as shown in Table 12. Thus, the classification algorithms should do better than this.

7.3.1 Principal component analysis

To understand the quality of the results, using FRE and other algorithms, a simple analysis of fiber data was done. This begins with principal component analysis

TABLE 11 Total of 52 observations, 48 features.

Class	Observations	Prior
C_1	8	$\Pr(C_1) = \frac{2}{13}$
C_2	4	$\Pr(C_2) = \frac{1}{13}$
C_3	20	$\Pr(C_3) = \frac{5}{13}$
C_4	20	$\Pr(C_4) = \frac{5}{13}$

TABLE 12 Mean prediction probabilities and deviations for random selection.

Mean prediction probability	deviation
$p_1 = \Pr(l = C_1) = (\frac{2}{13})^2 + (\frac{11}{13})^2 = 0.74$	$\mathrm{std}(l_1) = \sqrt{p_1(1-p_1)} = \pm 0.44$
$p_2 = \Pr(l = C_2) = (\frac{1}{13})^2 + (\frac{12}{13})^2 = 0.86$	$\mathrm{std}(l_2) = \sqrt{p_2(1-p_2)} = \pm 0.35$
$p_3 = \Pr(l = C_3) = (\frac{5}{13})^2 + (\frac{8}{13})^2 = 0.53$	$\mathrm{std}(l_3) = \sqrt{p_3(1-p_3)} = \pm 0.50$
$p_4 = \Pr(l = C_4) = (\frac{5}{13})^2 + (\frac{8}{13})^2 = 0.53$	$\mathrm{std}(l_4) = \sqrt{p_4(1-p_4)} = \pm 0.50$

(PCA) using all input variables, plus an extra threshold variable. The threshold is the gray level, upper 25% quantile of the gray level histogram, which was used in the blob segmentation. Threshold should not hold important information for the analysis since image intensity is related to image capture, rather than material itself.

Before computing the eigenvalues and loadings, the data was preprocessed by equalizing all variables into an equal range and then normalizing the input vectors $\mathbf{x}$ to unit length $||\mathbf{x}'|| = 1$. The resulting covariance matrix is the so-called sign-covariance matrix (see Ollila, 2002), which makes the PCA robust for data outliers.

The results of the PCA are shown in Table 13. Six eigenvectors, corresponding to the largest eigenvalues, explain about 95% of the data. Thus we may expect that classifiers may not need more variables either. Another observation is that variables with strong loadings on the same eigenvectors are correlating strongly. Thus one of these variables should be used in the same classifier. For example, the distribution of the perimeters of the blobs (measured with entropy, kurtosis and skewness) is correlating with the mean volume of the blobs. This is expected since large blobs are more "fractal shaped" than the small ones. Also the distribution of blob steepness seems to correlate with Fourier features, which is expected since both of them measure gray-level intensity changes. Also, as was expected, the threshold does not explain other variables very well. It is not loaded in the largest eigenvectors well.

It should be noted that the largest eigenvalues correspond to the largest variation, which may or may not be essential for good classification. Therefore variables need to be analyzed with respect to classes as well. This is done in Figure 33

TABLE 13 Six largest eigenvalues of fiber data and the main directions of the corresponding eigenvectors. The numbers are ordered loadings (probabilities) of the fifteen best correlating variables.

λ_1^2 0.498		λ_2^2 0.322		λ_3^2 0.081	
vec_1	name	vec_2	name	vec_3	name
-0.482	vol_var	-0.336	ste_ske	0.382	ori_var
0.433	per_kur	-0.252	rou_ske	0.316	ecc_mea
0.400	vol_mea	-0.231	fou_3	-0.305	fou_4
0.288	vol_ske	0.231	fou_2	0.302	ste_mea
0.265	are_ent	0.230	vol_ske	0.278	ste_ent
-0.191	per_ske	0.229	rou_ent	-0.250	fou_7
0.169	per_var	-0.223	ori_ent	-0.232	vol_kur
-0.154	rou_ske	0.218	rou_kur	0.199	ori_ent
-0.150	thres	0.199	are_mea	-0.177	fou_6
-0.148	rad_var	0.195	ste_kur	-0.167	are_ske
0.135	are_ske	0.184	are_kur	0.161	ste_kur
-0.126	per_ent	-0.181	are_ske	-0.159	ori_mea
0.124	ecc_var	-0.178	fou_8	0.151	fou_8
-0.114	rad_kur	-0.177	are_var	-0.137	ecc_ent
-0.107	are_kur	-0.174	fou_4	0.132	rou_ent

λ_4^2 0.057		λ_5^2 0.016		λ_6^2 0.010	
vec_4	name	vec_5	name	vec_6	name
0.399	fou_3	-0.4702	fou_2	0.386	ori_ent
0.294	ecc_ske	-0.303	ste_kur	-0.368	ori_var
-0.262	ste_kur	-0.289	fou_8	0.315	ste_ske
-0.243	fou_5	-0.256	fou_6	-0.270	ecc_var
0.236	ecc_mea	-0.226	per_var	0.223	ste_mea
0.218	fou_7	-0.221	ste_ske	0.223	are_mea
-0.211	thres	0.216	fou_5	0.217	ecc_ent
0.211	ecc_ent	-0.197	thres	0.192	vol_mea
-0.209	rou_ent	0.175	ecc_kur	-0.179	fou_1
-0.198	fou_1	0.166	ecc_ske	-0.161	vol_ent
-0.195	per_var	-0.165	are_mea	-0.158	fou_5
0.161	fou_8	0.160	rou_kur	-0.154	rad_mea
0.161	rad_ent	-0.153	fou_4	-0.149	per_mea
-0.156	are_mea	-0.151	ori_kur	0.143	rad_var
-0.155	ori_ske	-0.137	are_ske	0.140	per_ent

with the principal component axes that correspond to the three largest eigenvectors. It can be seen that only the first principal component, PC1, seems to separate class 4 to some extent from the rest of data. For comparison, a Sammon mapping for same data is shown in the lower right subimage, which does not seem to separate classes any better (class regions are quite complex).

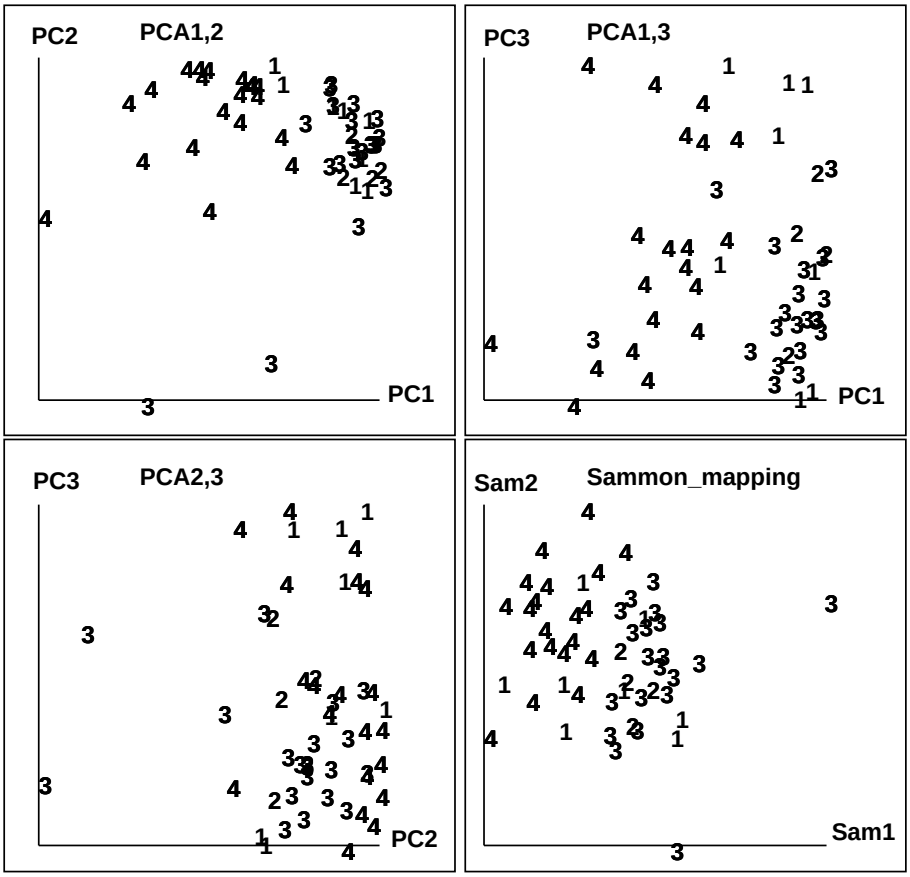

FIGURE 33 Principal component analysis and Sammon mapping with all variables. Projections are shown onto the first three principal components, and onto 2-D latent space with Sammon mapping.

7.3.2 Self-organizing map

The analysis continues with a self-organizing map (SOM), which was trained with the same preprocessed data as the PCA and Sammon mapping. The SOM was built using the TS-SOM algorithm (Koikkalainen, 1994), and after training the class indicators were projected onto the 8×8 nodes, SOM lattice, which separates almost all observations on node level, as shown in Figure 34. With some user

imagination one can draw the class boundaries onto the SOM lattice. This is illustrated with different shades for classes C_1, C_2 and C_4. Note that classes are not used in training. To see which variables might correspond to which classes the

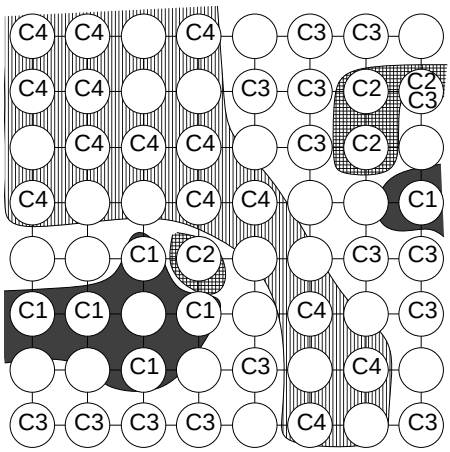

FIGURE 34 The self-organizing map, trained with all input variables and visualized with projected class boundaries.

means of original variables (in the original scales, without preprocessing) were displayed on the SOM lattice as well, as shown in Figure 35. Each variable is shown as a component plane, separately. We may see, for example, that C_4 can be explained well by Fou_8. More exactly, high frequency components, represented by Fourier component 8, seem to be missing from C_4, but exists in all other classes C_1, C_2 and C_3. Yet it is quite hard to make any conclusive decisions based on this visual analysis only.

7.3.3 Variable–Class dependencies

Another viewpoint to the role of different variables can be obtained by looking at their dependencies between classes. Since one can expect these dependencies to be nonlinear, the dependencies were computed with different algorithms, using univariate models of the type $y_l = g_l^{\mathrm{model}}(x_i)$. Thus, the better variable x_i is able to predict class l using the model g_l^{model}, the bigger the dependency is. These prediction probabilities must be normalized with respect to random class prediction probabilities (see Table 12) in order to visualize the dependencies.

For the sake of an explorative data analysis, only a rough scale of dependencies is used according to the following scale of "+" and "-" characters.

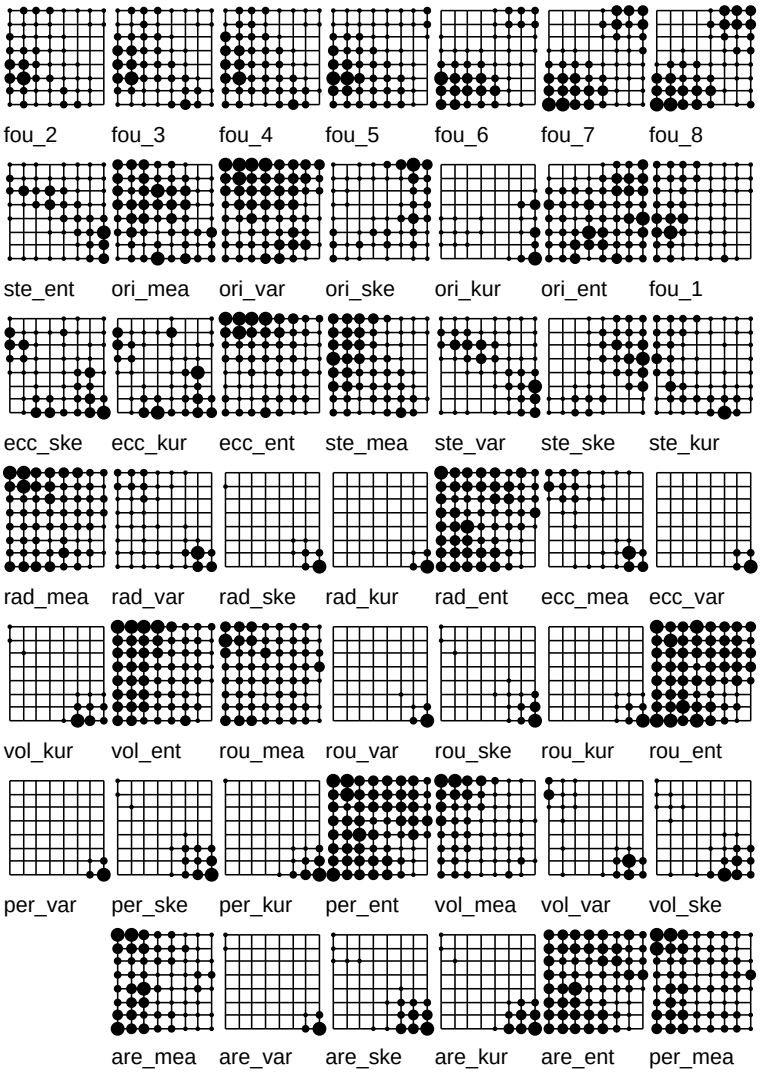

FIGURE 35 SOM lattices. Each variable is shown as a component plane that corresponds to Figure 34.

-	No dependency	$\delta \leq 1$
+	Small dependency	$1 < \delta \leq 1.3$
++	Dependency	$1.3 < \delta \leq 1.6$
+++	Strong dependency	$\delta > 1.6$

where

$$\delta = \frac{\Pr(l|\text{model}(x_i))}{\Pr(l|\text{prior prob.})}.$$

For the computation of dependencies all evaluation models, LDA, QDA, linear regression, logistic regression, MLP, and k-nn, were used. The cross-validated results are shown in Figures 36 and 37.

	Class 1						Class2					
	LDA	QDA	linreg	logreg	MLP	knn	LDA	QDA	linreg	logreg	MLP	knn
are_mea	-	-	+	+	+	-	-	-	+	+	+	-
are_var	-	-	+	+	+	-	-	-	+	+	+	-
are_ske	-	-	+	+	+	-	-	-	+	+	+	-
are_kur	-	-	+	+	+	-	-	-	+	+	+	-
are_ent	-	-	+	+	+	-	-	-	+	+	+	-
per_mea	-	-	+	+	+	-	-	-	+	+	+	-
per_var	-	-	+	+	+	-	-	-	+	+	+	-
per_ske	-	-	+	+	+	-	-	-	+	+	+	-
per_kur	-	-	+	+	+	-	-	-	+	+	+	-
per_ent	-	-	+	+	+	-	-	-	+	+	+	-
vol_mea	-	-	+	+	+	-	-	-	+	+	+	-
vol_var	-	-	+	+	+	-	-	-	+	+	+	+
vol_ske	-	-	+	+	+	-	-	-	+	+	+	-
vol_kur	-	-	+	+	+	-	-	-	+	+	+	-
vol_ent	-	-	+	+	+	-	-	-	+	+	+	-
rou_mea	-	-	+	+	+	-	-	-	+	+	+	-
rou_var	-	-	+	+	+	-	-	-	+	+	+	-
rou_ske	-	-	+	+	+	-	-	-	+	+	+	-
rou_kur	-	-	+	+	+	-	-	-	+	+	+	-
rou_ent	-	-	+	+	+	-	-	-	+	+	+	+
rad_mea	-	-	+	+	+	-	-	-	+	+	+	-
rad_var	-	-	+	+	+	-	-	-	+	+	+	-
rad_ske	-	-	+	+	+	-	-	-	+	+	+	-
rad_kur	-	-	+	+	+	-	-	-	+	+	+	-
rad_ent	-	-	+	+	+	-	-	-	+	+	+	+
ecc_mea	-	-	+	+	+	+	-	-	+	+	+	+
ecc_var	-	-	+	+	+	-	-	-	+	+	+	+
ecc_ske	-	-	+	+	+	-	-	-	+	+	+	-
ecc_kur	-	-	+	+	+	-	-	-	+	+	+	-
ecc_ent	-	-	+	+	+	+	-	-	+	+	+	-
ste_mea	-	-	+	+	+	-	-	-	+	+	+	-
ste_var	-	-	+	+	+	-	-	-	+	+	+	-
ste_ske	-	-	+	+	+	-	-	-	+	+	+	-
ste_kur	-	-	+	+	+	-	-	-	+	+	+	-
ste_ent	-	-	+	+	+	-	-	-	+	+	+	-
ori_mea	-	-	+	+	+	-	-	-	+	+	+	-
ori_var	+	+	+	+	+	+	-	-	+	+	+	-
ori_ske	-	-	+	+	+	-	-	-	+	+	+	+
ori_kur	+	+	+	+	+	+	-	-	+	+	+	-
ori_ent	-	-	+	+	+	-	-	-	+	+	+	-
Fou_1	-	+	+	+	+	+	-	-	+	+	+	+
Fou_2	-	+	+	+	+	+	-	-	+	+	+	-
Fou_3	-	+	+	+	+	-	-	-	+	+	+	-
Fou_4	-	+	+	+	+	-	-	-	+	+	+	-
Fou_5	-	+	+	+	+	-	-	-	+	+	+	+
Fou_6	-	-	+	+	+	-	-	-	+	+	-	+
Fou_7	-	-	+	+	+	-	-	-	+	+	+	+
Fou_8	-	-	+	+	+	-	-	-	+	+	+	+

FIGURE 36 Univariate dependencies for classes 1 and 2.

The variables that have the strongest dependencies for a class with a given method are those that will be selected first by the forward selection algorithm, but it is more difficult to say if several good candidates are dependent on each other,

	Class 3						Class 4					
	LDA	QDA	linreg	logreg	MLP	knn	LDA	QDA	linreg	logreg	MLP	knn
are_mea	+	+	+	+	+	-	+	-	+	+	+	+
are_var	-	+	+	+	+	-	+	-	+	+	+	+
are_ske	-	-	+	++	+	-	++	+	++	+	+	++
are_kur	-	+	+	+	+	+	++	++	+	+	+	+
are_ent	-	+	+	+	-	+	-	-	+	+	-	+
per_mea	+	+	+	-	-	-	-	+	+	+	-	+
per_var	-	+	+	++	++	+	++	-	+	++	++	+
per_ske	-	+	+	+	+	+	++	++	+	+	+	+
per_kur	-	+	+	+	+	-	++	+	+	+	+	+
per_ent	-	+	+	+	-	+	-	-	+	+	-	+
vol_mea	++	++	++	++	+	+	++	++	++	++	++	++
vol_var	+	+	-	++	+	+	++	++	++	+	+	+
vol_ske	-	-	+	+	+	-	+	++	++	+	+	+
vol_kur	-	-	+	+	+	-	+	++	+	+	+	+
vol_ent	++	+	+	++	++	+	++	++	++	++	++	++
rou_mea	-	-	+	-	-	-	+	+	+	-	-	+
rou_var	-	+	+	++	+	-	++	-	+	+	++	+
rou_ske	-	+	+	++	+	-	++	+	+	+	+	+
rou_kur	-	+	+	+	+	-	+	-	+	+	+	+
rou_ent	-	+	+	+	-	+	-	-	+	+	+	+
rad_mea	-	-	+	++	+	+	+	+	++	++	+	+
rad_var	-	-	+	+	++	+	+	+	++	+	++	++
rad_ske	-	+	+	+	+	+	++	-	+	+	+	+
rad_kur	+	+	+	+	+	-	+	-	+	+	+	+
rad_ent	-	+	+	+	-	-	-	-	+	+	-	+
ecc_mea	+	-	+	++	+	+	++	++	++	++	+	+
ecc_var	-	+	+	++	++	-	++	-	+	+	++	+
ecc_ske	+	-	+	+	-	+	+	-	+	+	-	+
ecc_kur	+	+	+	+	-	+	-	+	+	+	+	+
ecc_ent	-	-	+	++	+	+	+	+	+	+	+	+
ste_mea	++	++	++	++	+++	++	+++	+++	+++	+++	+++	+
ste_var	-	-	+	+	+	-	+	+	+	++	+	+
ste_ske	++	++	++	++	++	+	++	++	++	++	++	++
ste_kur	+	+	+	++	+	++	++	++	++	++	++	+
ste_ent	+	+	+	+	+	-	-	-	+	+	+	+
ori_mea	+	+	+	++	++	-	++	++	+	++	++	+
ori_var	+	-	+	++	++	+	+	+	++	++	++	+
ori_ske	+	+	+	+	+	-	++	+	++	+	+	+
ori_kur	-	+	+	+	++	-	+	+	++	+	++	+
ori_ent	+	+	+	++	++	+	++	++	++	++	++	+
Fou_1	+	+	++	++	++	+	++	+	-	+	++	++
Fou_2	++	++	++	+	++	+	++	++	+	+	++	+
Fou_3	++	++	++	+	++	++	++	++	+	+	++	+
Fou_4	++	++	++	+	+	++	++	++	+	+	+	+
Fou_5	+	-	+	+	+	-	-	+	+	+	+	+
Fou_6	+	+	+	++	++	-	++	++	++	+	++	+
Fou_7	++	++	++	++	+++	++	+++	+++	+++	+++	+++	+++
Fou_8	++	++	++	++	+++	++	+++	+++	+++	+++	+++	+++

FIGURE 37 Univariate dependencies for classes 3 and 4.

and will therefore be omitted when the next variables are selected.

Visual inspection of Figures 36 and 37 reveals that

- LDA and QDA have problems with classes 1 and 2. Over all 48 variables only a few variables bear some information for these models. In class 2 there is no single variable that would be a good selection. However, what cannot

be infered from the figures is that these models may work well if more than one feature is selected for each class.

- For classes 1 and 2 the best univariate methods are based on regression.
- The results for classes 2 and 3 are more interesting. There seem to be many good variables (with high rating) for both classes. Especially the variables ste_mea, Fou_7 and Fou_8 are rated high.
- The figures give an overall impression that the variables are rated quite similarly by the different models.

7.4 Rule extraction with FRE algorithm

This section explains an experiment that was made using fuzzy rule extraction (FRE) for fiber data. The obtained linguistic rule set for classes 1, 2, 3 and 4 are shown in Table 14. The rule set is based on automatically selected fuzzy input sets, which are shown in Figures 38 ans 39 for those rules that were picked by the FRE.

TABLE 14 Fuzzy rules that were proposed by FRE algorithm from the set of 48 features.

Class is 1 (Exact rule with entropy 0.0)		
IF	are_mea is in [87, 127]	(SOMEWHAT_HIGH)
AND	rad_kur is in [17.2, 24.4]	(VERY LOW)
AND	ecc_ent is in [1.22, 1.4]	(somewhat_low or MEDIUM)

Class is 2 (Exact rule with entropy 0.0)		
IF	are_ent is in [4.7, 4.88]	(MEDIUM OR somewhat_high)
AND	ecc_mea is in [1.87, 2.15]	(LOW)

Class is 3 (Quite reliable rule with entropy 0.22)		
IF	ste_mea is in [164, 197]	(SOMEWHAT_LOW)
AND	ste_kur is in [-0.328, -0.139]	(SOMEWHAT_LOW)

OR (Quite reliable rule with entropy 0.29)

IF	ste_kur is in [-0.328, -0.139]	(SOMEWHAT_LOW)
AND	fou_3 is in [4.78e+7, 6.11e+7]	(SOMEWHAT_LOW)

Class is 4 (Exact rule with entropy 0.0)		
IF	fou_8 is in [1.73e+7, 2.05e+7]	(VERY_LOW)

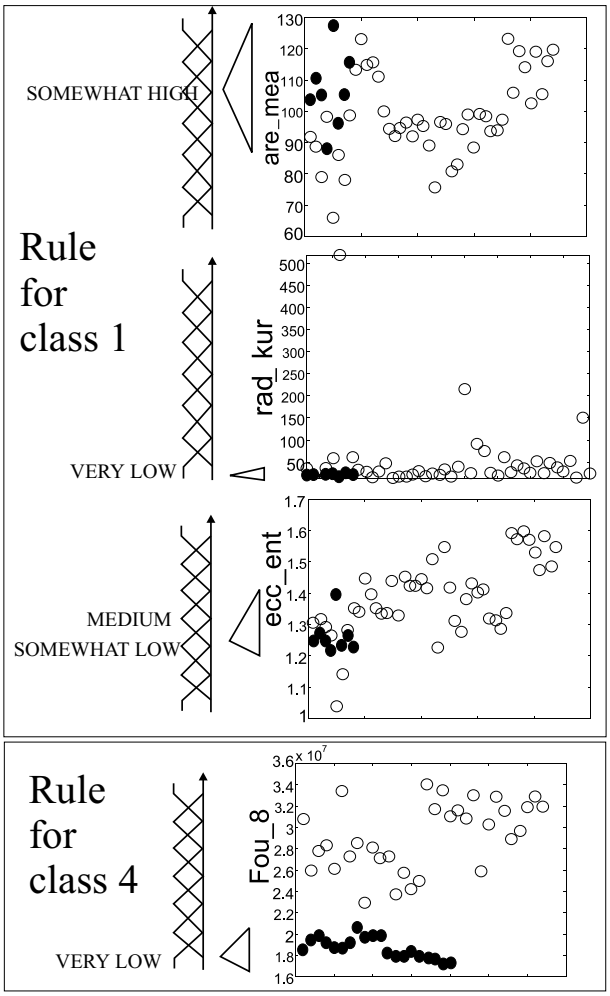

FIGURE 38 Fuzzy sets for variables that were selected by the FRE algorithm. Rules for classes 1 and 4 are shown.

It can be seen that FRE proposes reliable rules for classes 1, 2, and 4, but there is some confusion about class 3. This will be confirmed when the cross-validated classification accuracies are computed. With some certainty we may say that these rules can be used to predict the quality class of paper, and this is verified via cross-validation in section 7.5.2. The linguistic interpretation also reveals how the classification is done, which can be used in the development of new feature extraction methods. This work is in progress.

Current rules are based on features that are not easy for physical interpretation, they are made mainly for prediction purposes. In addition the physical interpretation requires paper process engineering expertise, that the current

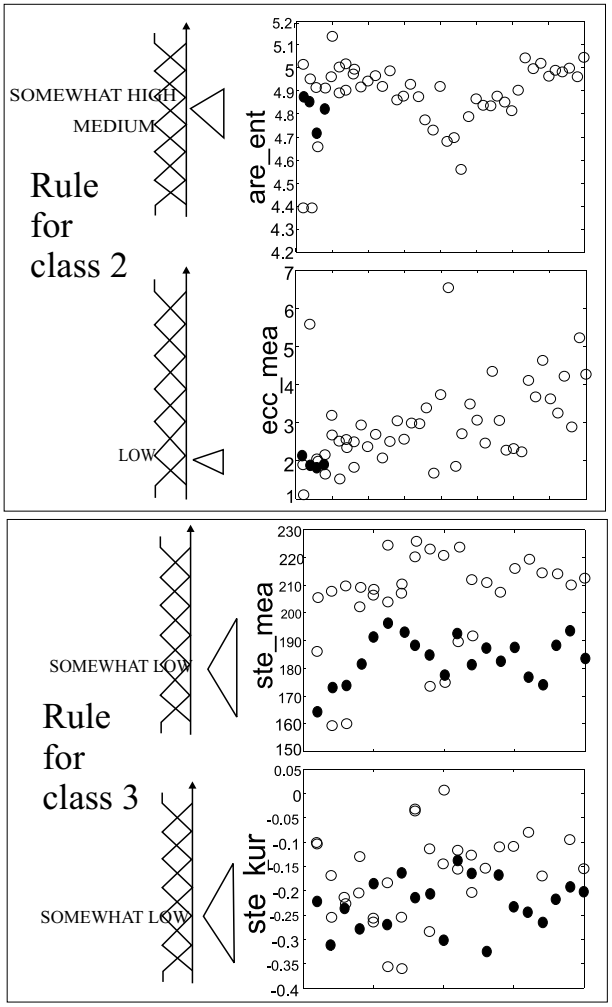

FIGURE 39 Fuzzy sets for variables that were selected by the FRE algorithm. Rules for classes 2 and 3 are shown.

author does not have. Yet for an illustration the following speculation is given without any guarantee that this meets with physical reality.

As it was expected after the PCA analysis, only a couple of variables are needed for each class-specific rule set. Most notably, class 4 can be explained with one feature only, a low amount of high frequency (Fou_8), which separates this class from the others. Usually this indicates that extracted blobs are larger, and inside the larger blobs, there can be a bigger variation of gray-level values.

In other classes more variables are needed. In class 1 the blobs seem to be medium large (are_mea is somewhat high), the distribution of blob radiuses is symmetric (rad_cur is low), and there is “normal” variation in the symmetry of

blobs (ecc_ent is low medium). This might be a sympton of "typical" large variation Gaussian random process, where it is quite difficult to say any distinct blob shape when looking at the images directly.

The distribution of blobs in the images of class 2 seem to be characterized by symmetrical blobs (ecc_mea is low) and typical variation of blob sizes (are_ent is medium). Thus, at least on the blob level, the process seems to be more isotropic than others.

Class 3 seems to be the most difficult to discriminate from the other classes. It seems that there are many gray-level related features, indicating a low variation of gray levels inside the blobs. This may mean many small blobs, and large high frequency components. Therefore class 3 might be most different from class 4.

Some of these conclusions can be confirmed by using the algorithm with a smaller set of variables. When the Fourier features are omitted, FRE replaces the last two rules with the rules of Table 15. Clearly rules for class 3 seem to be almost in reverse to those of class 4.

TABLE 15 Rules produced by FRE when only features from blob distributions were used.

Class is 3 (Quite reliable rule with entropy 0.22)		
IF	ste_mea is in [164, 197]	(SOMEWHAT_LOW)
AND	ste_kur is in [-0.328, -0.139]	(SOMEWHAT_LOW)
OR (Reduced reliability rule with entropy 0.45)		
IF	ste_mea is in [164, 197]	(SOMEWHAT_LOW)
AND	ste_var is in [1.87e+3, 2.27e+3]	(SOMEWHAT_LOW)
Class is 4 (Exact rule with entropy 0.0)		
IF	ste_mea is in [204, 226]	(somewhat_high or HIGH)
AND	ste_ent is in [5.14, 5.22]	(somewhat_low or MEDIUM)

We may also investigate the "goodness" of variables, selected by FRE, visually using principal component analysis, Sammon mapping, and the self-organizing map. Using the PCA algorithm as it was done in section 7.3.1 for all variables, the results of ten features, used by both FRE rule sets, are visualized in Figure 40. The features seem to separate classes better than before, at least in some cases.

Now class 4 is clearly separable from the rest of the data. It might also be reasonable to believe that most observations of class 1 can be separated from other classes. It is difficult to say anything about class 2, except that all four examples seem to be close to each other.

The analysis with SOM gives the results that are shown in Figures 41 and 42. The component planes explain some of the selections of FRE algorithm quite well.

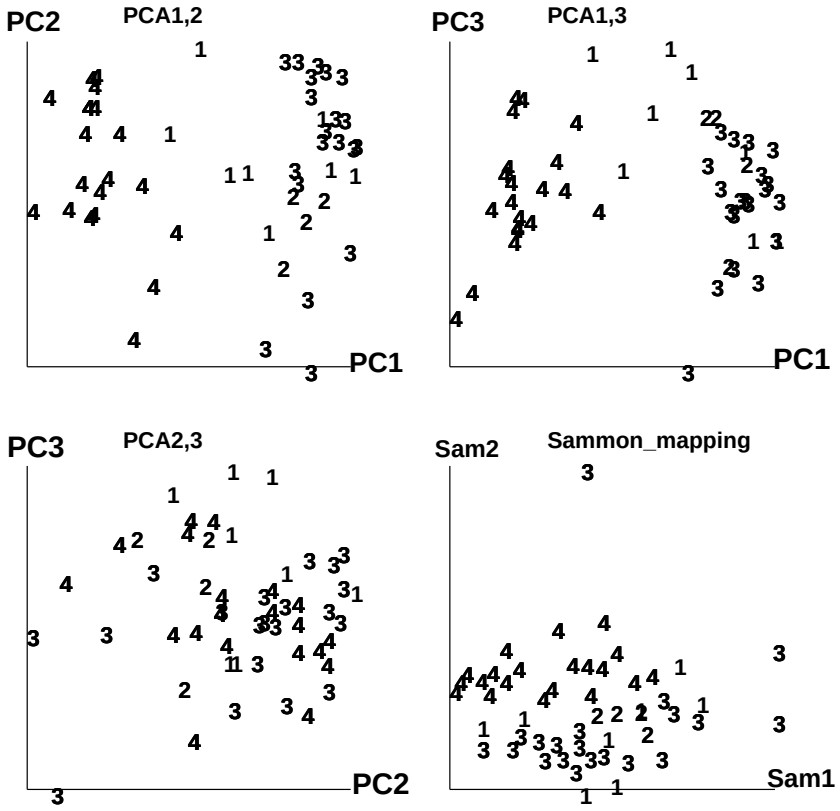

FIGURE 40 Principal component analysis and Sammon mapping with variables that were selected by FRE. Projections are shown onto three first principal components, and onto 2-D latent space with Sammon mapping.

This is especially true for the Fourier component 8 (Fou_8), where there is very little activity in the nodes of class 4. Similarly Fourier components Fou_3 seems to be low in nodes that are labeled by class 3. One interesting observation is that feature rad_kur might be good for picking a subset of class 3. This fact is totally ignored by FRE since it would require XOR type of rules instead of conjunctive (AND) types.

7.5 A comparison between FRE and other classifiers using fiber data

The classification accuracy of FRE rules was computed and compared against other classifiers (as mentioned in section 7.1). This is quite problematic, because the mechanism of FRE is quite unique: it does not require preprocessing of data,

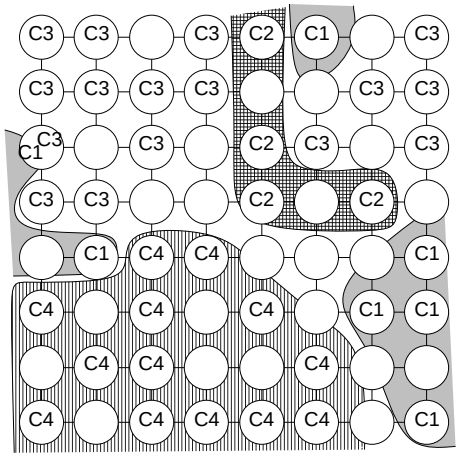

FIGURE 41 The SOM trained with features that were selected by FRE algorithm.

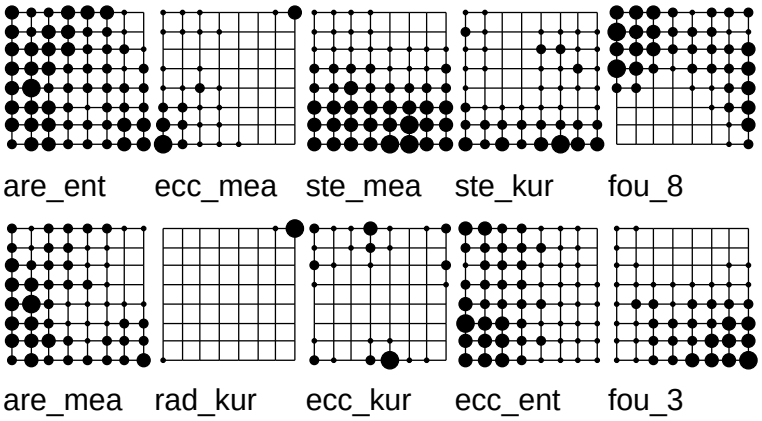

FIGURE 42 The SOM component planes for FRE features, that correspond to Figure 41.

it includes a variable selection, and it builds rules independently for each of the classes. Therefore two kinds of comparative experiments were made with other methods, one with single-class decisions, and the other with multi-class ones. Both evaluations use sequential forward selection to find the best predictive variables for the problem. In addition the performance of shrinkage methods without explicit feature selection was briefly tested with MLP combined with weight decay and linear regression combined with lasso.

The overall results are summarized in Table 16. In general, standard classifiers work best in multiclass (vectors) mode, except linear regression, which is not flexible enough to express multi-class decisions with one set of regression coefficients. Also k-nn gave better results with combined single-class classifiers, which

is because of very good single-class models that it was able to make. For comparison purposes a sequential search version of FRE was also made, which is significantly faster than the proposed algorithm. The time values shown in Table 16 measure only the modelling times without sequential forward selection. Thus the value is time used for one step in selection process (one training). The value for FRE (original) contains time needed to do a restricted full search.

As expected, the results of the shrinkage methods were not as good as the methods that rely on explicit feature selection. MLP with weight decay performed significantly worse than with SFS. Linear regression with lasso, however, was able to achieve a similar performance as with feature selection. The total computing times of shrinkage methods are lower since the features are "selected" in one run.

TABLE 16 Cross-validated mean prediction probabilities and their standard deviations for fiber data.

Method	Single-class accuracy % (combined)	Multi-class accuracy %	Computing time
Lin.reg.(SFS)	84.1 ± 16.1	82.7 ± 38.2	0.031
Log.reg.(SFS)	79.9 ± 25.1	86.5 ± 34.5	20.18
MLP (SFS)	90.4 ± 23.6	92.3 ± 26.9	18.21
k-nn (SFS)	96.6 ± 14.1	94.2 ± 23.5	0.042
k-nn (SFS, with 2-level cross-valid., see 7.2.1)	95.7 ± 14.3		
LDA (SFS)	75.3 ± 25.3	80.8 ± 39.8	0.097
QDA (SFS)	91.8 ± 19.2	92.3 ± 26.9	0.105
FRE (original)	96.2 ± 19.7		22.7
FRE (SFS)	94.2 ± 23.5		0.04
MLP (weight decay)		67.3 ± 47.4	19.7
Lin.regr. (lasso)		82.7 ± 38.2	0.07

The benefit of single class methods is that they allow a direct comparison of the selected variables for each class separately. This makes it possible to compare the classifier with the rules that were presented in the previous section 7.4. The selected variables, obtained via forward selection, are shown in Table 17. The numbers below the features show the total classification accuracy (over all 52 observations) for a given class with each of the single-class models. New variables were added only if the accuracy was improving, and therefore the value of the last included variable shows the final (mean) prediction accuracy.

7.5.1 Qualitative data analysis of QDA variables

Unlike with FRE, there is no direct explanation about the reasoning logic behind different models. This task is a separate process that must be done using the qual-

TABLE 17 The variables (in the order of selection) that were selected by single-class models using sequential forward selection and the progress of classification accuracy.

Class 1

	Lin.regr.	Log.regr.	MLP	k-nn	LDA	QDA
1.	ori_ent	ori_var	ecc_ent	ori_kur	ori_var	ori_var
	86.5	88.5	90.3	90.4	92.3	92.3
2.	per_var	per_kur	rad_ent	are_ent		
	88.5	92.3	92.3	96.2		
3.			per_var	are_var		
			94.2	98.1		
4.				ecc_ent		
				100.0		

Class 2

	Lin.regr.	Log.regr.	MLP	k-nn	LDA	QDA
1.	are_mea	are_mea	vol_ent	ecc_mea	vol_ent	vol_ent
	92.3	92.3	94.2	90.4	73.1	78.9
2.				are_kur		ori_kur
				92.3		98.1
3.				ori_kur		
				98.1		

Class 3

	Lin.regr.	Log.regr.	MLP	k-nn	LDA	QDA
1.	Fou_4	Fou_8	Fou_8	ste_mea	ste_mea	ste_mea
	80.8	78.9	88.5	82.7	80.8	82.7
2.	Fou_5	are_mea	Fou_7	ste_var	ori_mea	ste_var
	86.5	80.8	92.3	96.2	82.7	92.3
3.	ste_ent		are_ent			ori_ske
	94.2		94.2			94.2
4.						ste_kur
						96.2

Class 4

	Lin.regr.	Log.regr.	MLP	k-nn	LDA	QDA
1.	ste_mea	Fou_8	Fou_8	Fou_8	Fou_8	Fou_8
	96.2	94.2	100.0	100.0	96.2	100.0
2.	ste_var	per_mea				
	100.0	96.2				

itative data analysis. This is done as an illustration for the eight variables of QDA algorithm.

The qualitative data analysis reveals that these variables are rather similar to those that were selected by the FRE algorithm: class 4 is almost always selected

via Fou_8, and class 3 is often described by steepness distribution (or Fou_8). As before we may compare the classification capability of these features using principal components, Sammon mapping and self-organizing maps. From principal components and Sammon mapping, Figure 43, it can be seen that class 4 separates very well by PC1, and PC3 makes a reasonably good separation for class 1 with the help of PC1.

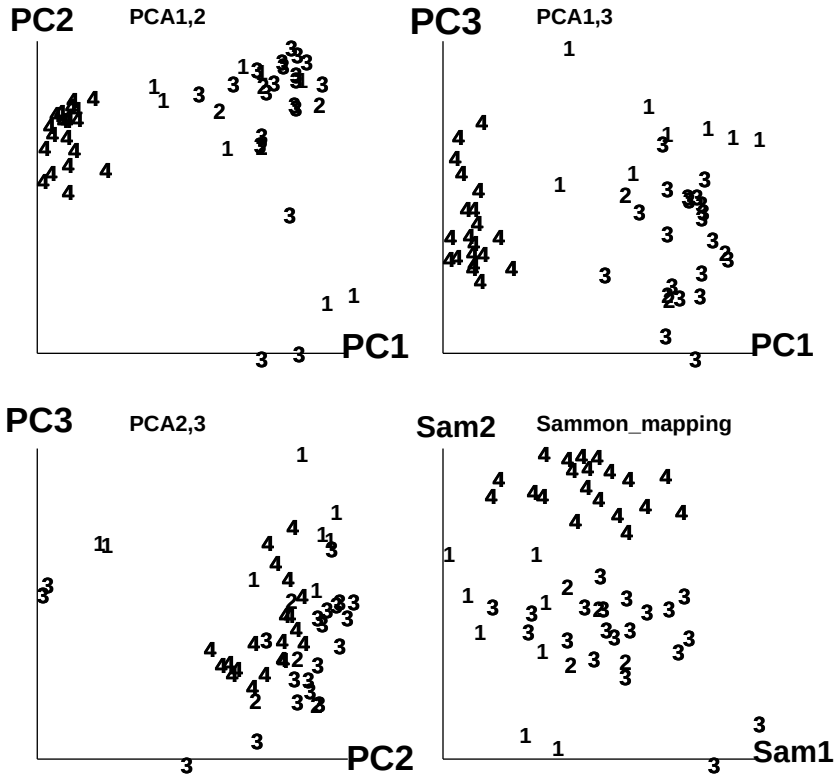

FIGURE 43 Principal component analysis and Sammon mapping with variables that were selected by QDA. Projections are shown onto three first principal components, and onto 2-D latent space with Sammon mapping.

SOM based analysis is done with the variables of the QDA algorithm. One may try to explain the reasons behind classification, as it was done by the FRE rules: feature Fou_8 seems to have high values in class 3 (C3) and low in class 4 (C4). The logic behind variable ori_var for class 1 must be something like "between small and large". For class 2 (C2) it is very difficult to see any "explanation" when using the variables selected by QDA.

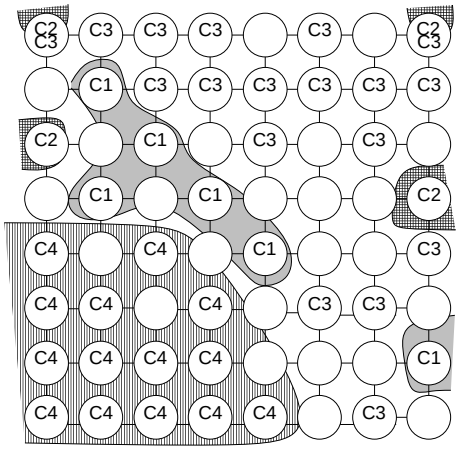

FIGURE 44 Using SOM to analyze the role of variables picked by the QDA+SFS algorithms.

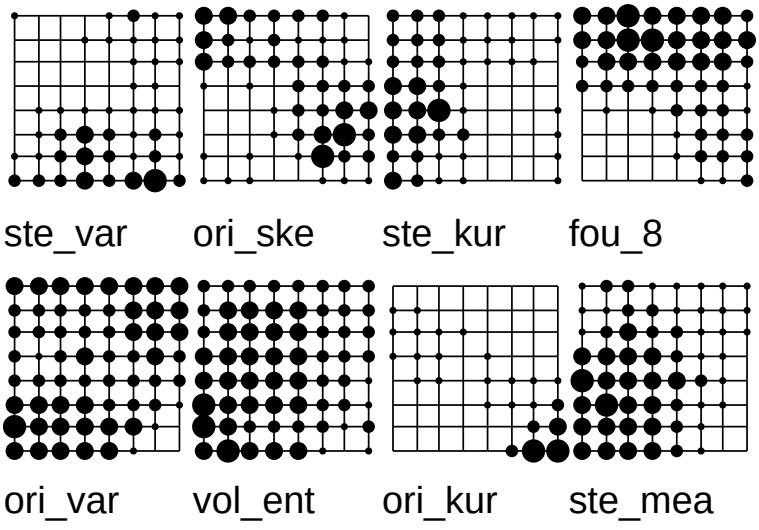

FIGURE 45 The SOM component planes for QDA features.

7.5.2 Full quantitative evaluation of fiber data classifiers

The evaluations of classification performances of all algorithms has been done on the level of confusion matrices. The matrix that shows the prediction errors of the FRE algorithm is shown in Table 18.

Rather surprisingly, the cross-validated mean prediction errors after applying FRE 52 times for data of 51+1 (training + validation) observations are 100% for classes 1, 3 and 4. The only confusion is between class 1 and 3.

TABLE 18 Classification results for FRE.

FRE

Actual classes	Predicted classes 1	2	3	4
1	**100.0%** ± 0.0			
2		**100.0%** ± 0.0		
3	10%		**90.0%** ± 30.78	
4				**100.0%** ± 0.0

The evaluation of single class models with all algorithms gives performances as shown in Table 19 (to discriminate one class from the rest). Although this is not directly comparable with FRE results, one can make a conclusion that this problem requires nonlinear models like QDA, MLP or k-nn in order to be competive with FRE.

TABLE 19 Class-specific results of single-class classifiers.

	Class 1	Class 2	Class 3	Class 4
lin. regr.	88.5 ± 32.3	92.3 ± 26.9	94.2 ± 23.5	100.0 ± 0.0
log. regr.	92.3 ± 26.9	92.3 ± 26.9	80.8 ± 39.8	96.2 ± 19.4
MLP	94.2 ± 23.5	94.2 ± 23.5	94.2 ± 23.5	100.0 ± 0.0
k-nn	100.0 ± 0.0	98.1 ± 13.8	96.2 ± 19.4	100.0 ± 0.0
LDA	92.3 ± 26.9	73.1 ± 44.8	82.7 ± 38.2	96.2 ± 19.4
QDA	92.3 ± 26.9	98.1 ± 13.8	96.2 ± 19.4	100.0 ± 0.0

Unfortunately there is no optimal way to compute matrices for combined single-class classifiers. Therefore two types of matrices were built. One that shows the number of truths for predicting a certain class with respect to the given class, and another that computes a confusion matrix of probabilities and deviations (see section 7.1.2). The second type of matrix can be computed from the first, and it is made to help the comparison between FRE and multi-class models. These matrices are shown in Table 20.

When classifiers were used in multi-class mode, variables were shared by all the classes. The variables, after using sequential forward selection were as shown in Table 21. Now the computation of confusion matrices are straightforward and the results are directly comparable with those of FRE. The cross-validated mean predictions and their deviations are shown in Table 22. The conclusion of these results is that the performances of FRE, QDA and k-nn are roughly on the same level. The main differences seem to be related to the class priors. QDA ignores

TABLE 20 Estimates of combined confusions, based on single class models. Results on the left show the mean number of predicted and confused samples. Probabilities and deviations are given on the right (%).

Linear regression

	Predicted classes				Predicted classes			
Cl.	1	2	3	4	1	2	3	4
1	3	1	3	1	**37.5**±40.1	12.5	37.5	12.5
2	0.75	0.75	1.75	0.75	18.8	**18.8**±12.7	43.8	18.8
3			20				**100.0**±0.0	
4				20				**100.0**±0.0
Logistic regression								
1	4.25	0.25	3.25	0.25	**53.1**±33.9	3.1	40.6	3.1
2	0.75	0.25	2.75	10.25	18.8	**6.2**±12.7	68.8	6.3
3	1	0.5	18	0.5	5.0	2.5	**90.0**±24.9	2.5
4	0.5	0.5	0.5	18.5	2.5	2.5	2.5	**92.5**±23.0
MLP								
1	6.25	0.75	0.75	0.25	**78.1**±31.1	9.4	9.4	3.1
2	1	2	1		25.0	**50.0**±57.7	25.0	
3	0.75	0.25	18.75	0.25	3.8	1.3	**93.8**±19.8	1.3
4				20				**100.0**±0.0
k-**nn**								
1	7.5		0.5		**93.8**±17.6		6.3	
2		4				**100.0**±0.0		
3	0.25	0.75	18.75	0.25	1.3	3.8	**93.8**±19.8	1.3
4				20				**100.0**±0.0
LDA								
1	5.42	0.92	1.42	0.25	**67.7**±35.2	11.5	17.7	3.1
2	0.33	1.67	1.67	0.33	8.3	**41.7**±9.5	41.7	8.3
3	1.08	5.08	13.08	0.75	5.4	25.4	**65.4**±30.3	3.8
4		1		19		5.0		**95.0**±15.5
QDA								
1	7.25	0.25	0.25	0.25	**90.6**±26.5	3.1	3.1	3.1
2	0.75	2.25	0.75	0.25	18.8	**56.3**±31.5	18.8	6.3
3	1.25	0.25	18.25	0.25	6.3	1.3	**91.3**±21.9	1.3
4				20				**100.0**±0.0

class 2 with only four observations completely, while FRE predicts that very well but makes errors with class 3.

TABLE 21 The selected features (in selection order) in multi-class case.

	Lin.regr.	Log.regr.	MLP	k-nn	LDA	QDA
1.	ste_mea	ste_mea	Fou_8	Fou_7	ste_ske	ste_mea
2.	Fou_3	Fou_7	Fou_3	Fou_2	ori_kur	Fou_1
3.	ste_var	Fou_2	Fou_6	Fou_8	Fou_8	ori_ske
4.		vol_var	ecc_var	ste_var		

7.6 Using FRE to assist fiber data analysis

The analysis of new data is never straightforward. In the beginning there are many open questions about the best practices, objectives and assumptions that further studies are based on. In this section we study two issues of this nature:

a) The selection of best preprocessing method for fiber images before blob features are computed. This mainly concerns the threshold level that is used before image segmentation.

b) The selection of suitable quality classes when no predefined information about classes is available. In this case the role of FRE is to tell rules that discriminate two or more handpicked data clusters from each other. This should help the user to decide what the quality classes are.

7.6.1 Using FRE to decide image thresholds

In paper research the usage of thresholded blob images has been reported in (Kajanto, 1989), where the threshold level was set to median of the gray level histogram. The quantitative explanation for the choice is that median seems to be the percolation point of a random set process that generates these binary images. However, the percolation point is not necessarily the best choice for data analysis and classification.

To investigate the selection of the best threshold value for fiber image data (set 1) features were extracted from images using different thresholds and four different levels of smoothing were applied to these images. In a range from 10% quantile to 50% (median). Then the FRE algorithm was applied to rule generation and classification. The hypothesis is that some threshold levels are more informative than others.

The results of using FRE are given in terms of (miss)classification entropy in Figure 46. Although the results are not fully conclusive about the best threshold level, one can easily say that a 20% quantile level is better than median (50%) from a classification point of view. The reason for this is obvious when one compares two thresholded images in Figure 47. In a median thresholded image blobs are joined together and they form large complex shapes, while blobs on a 20%

TABLE 22 Confusion matrices for multi-class models for fiber data(%).

Linear regression

	Predicted classes				Predicted classes			
Class	1	2	3	4	1	2	3	4
1	3		3	2	**37.5** ± 51.8		37.5	25.0
2			4			**0.0** ± 0.0	100.0	
3			20				**100.0** ± 0.0	
4				20				**100.0** ± 0.0
Logistic regression								
1	5		3		**62.5** ± 51.8		37.5	
2		2		2		**50.0** ± 57.7		50.0
3		2	18			10.0	**90.0** ± 30.8	
4				20				**100.0** ± 0.0
MLP								
1	7	1			**87.5** ± 35.4	12.5		
2		3		1		**75.0** ± 50.0		25.0
3		1	18	1		5.0	**90.0** ± 30.8	5.0
4				20				**100.0** ± 0.0
k**-nn**								
1	7		1		**87.5** ± 35.4		12.5	
2		2	2			**50.0** ± 57.7	50.0	
3			20				**100.0** ± 0.0	
4				20				**100.0** ± 0.0
LDA								
1	5		1	2	**62.5** ± 51.8		12.5	25.0
2		2	1	1		**50.0** ± 57.7	25	25.0
3	2	3	15		10.0	15.0	**75.0** ± 44.4	
4				20				**100.0** ± 0.0
QDA								
1	8				**100.0** ± 0.0			
2	2		2		50.0	**0.0** ± 0.0	50.0	
3			20				**100.0** ± 0.0	
4				20				**100.0** ± 0.0

level better indicate the variability of material on a blob distribution level. As a conclusion 20% threshold level was used in the experiments of this thesis.

7.6.2 Picking classes from data with help of FRE

In this example a typical data-analysis problem is studied with the help of a linguistic rule extraction. The problem is as follows:

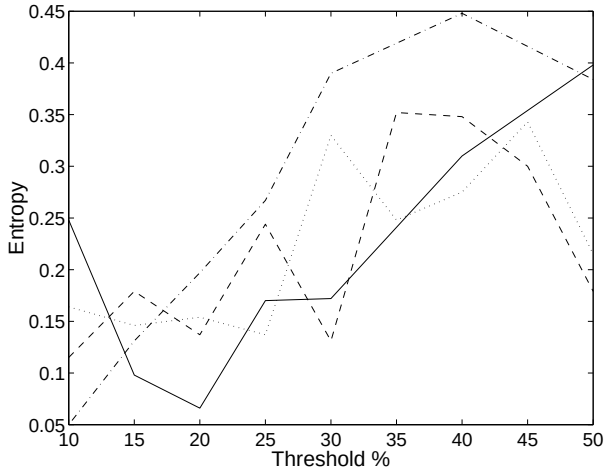

FIGURE 46 The goodness of rules extracted from data (shape descriptors as features). The lower the value is, the better the rules are. 20 percent seems to give a good result. The curves correspond to the original (solid line), 3×3 -median filtered (dashed line), 5×5 -median filtered (dotted line) and 7×7 -median filtered data (dash-dot line).

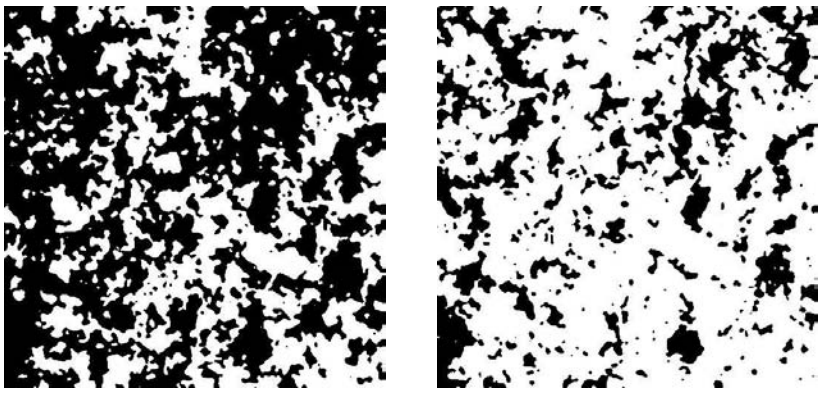

FIGURE 47 The result of thresholding the fiber image with the gray-level histogram quantiles of 50% (left) and 20% (right). Flocs are shown as black areas.

- Given data, the user must decide what kind of groups (clusters) exists.
- Instead of automatic clustering, the user wants to select and verify the groups using his domain expertise.

This kind of problem has been studied, for example, in recent Ph.D thesis (Niskanen, 2003) at the University of Oulu. The proposed solution was to classify all

observations using the self-organizing map, and visually investigate which of the observations belong to the same groups. In the Oulu example the test material was wood knots, which are relatively easy to visualize and interpret in image format. In the case of fiber image data, direct groupings of images is not as easy.

In the following experiments a second type of fiber image data set was used. There were 40 images without any class information. The training data was generated by sampling 12 subimages (256×256) from original (1024×656) images, thresholding and segmenting the images, and then computing 40 blob distribution features for them.

Using the self-organizing map, the most frequent examples of the original 40 images were shown in each of the SOM nodes, as depicted in the upper left part of Figure 48. For the human eye, it is almost impossible to say anything conclusive about the difference between the two images. Resizing the images does not improve the readability significantly. Therefore the user must have some kind of feature information available for decision making. This provides a good application possibility for automated rule extraction algorithm, like FRE.

By handpicking three groups G1, G2 and G3 of nodes, and all the observations that are assigned to them, the following rules are obtained:

```
IF   are_mea is in [256, 332]   (SOMEWHAT_HIGH OR high)
THEN class IS 1                 (EXACT)  with entropy 0.00

IF   are_ske is in [11.9, 15.1] (HIGH)
THEN class IS 2                 (EXACT)  with entropy 0.00

IF   are_mea is in [190, 246]   (LOW OR somewhat_low)
 AND rou_ent is in [4.94, 5.13] (SOMEWHAT_HIGH OR high)
THEN class IS 3                 (EXACT)  with entropy 0.00
```

Clearly these groups are very distinct because the difference between them can be described easily and reliably. In class G1 we have large blobs in average (although it is not very apparent visually), blobs in G2 come from unsymmetric distribution (due high variability of blobs), and class G3 seems to have small blobs. One can also verify the rules by representing the average loadings of features as bars on the SOM display, as shown in upper right, and lower parts of Figure 48.

The situation changes when more groups are introduced. When a user handpicks groups G1, G2, G3, and G4 the corresponding rules that discriminate these groups from each other are now

```
IF are_mea is in [256, 332]        (SOMEWHAT_HIGH OR high)
THEN class IS 1                    (EXACT) with entropy 0.00

IF are_ske is in [11.9, 15.1]      (HIGH)
THEN class IS 2                    (EXACT) with entropy 0.00

IF   vol_mea is in [30, 54.1]      (LOW OR somewhat_low)
```

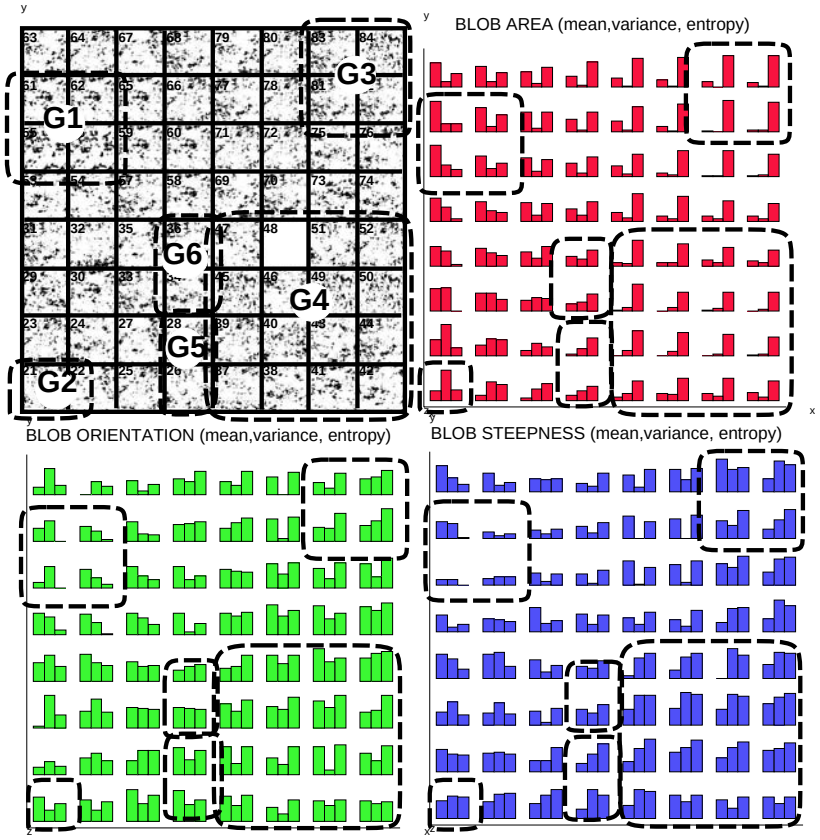

FIGURE 48 Visual display of thresholded fiber images on SOM display, and three displays of representative feature loadings for the same nodes.

```
 AND rad_ske is in [2.24, 6.62]     (LOW OR somewhat_low)
 AND ori_ske is in [-0.259, 0.215] (somewhat_low OR MEDIUM
                                    OR somewhat_high)
THEN class IS 3                    (REDUCED REL.)
                                    with entropy 0.60

IF   are_mea is in [183, 254]      (LOW OR somewhat_low)
 AND are_kur is in [21.6,142]      (low OR SOMEWHAT_LOW)
 AND ecc_var is in [37.1, 2.5e+05] (LOW OR somewhat_low
                                    OR medium)
THEN  class IS 4                   (QUITE RELIABLE)
                                    with entropy 0.29
```

Although the rules are not able to discriminate groups perfectly, the separation is still quite good. There is still some overlapping between groups G3 and G4, which causes an increase of variables in the rules.

As a final demonstration, the user handpicks only two groups (G5 and G6). These two groups are not good candidates as classes because the FRE algorithm is not able to find reliable rules for them. The groups are too similar to each other. This possibly means that their samples should be considered to belong to the same class.

```
IF   per_kur is in [72.2, 116]     (SOMEWHAT_LOW)
 AND ecc_ske is in [4.8, 14.8]     (somewhat_low OR MEDIUM
                                    OR somewhat_high)
 AND  ecc_kur is in [24.8, 222]    (somewhat_low OR MEDIUM
                                    OR somewhat_high)
THEN class IS 5                    (LOW REL.) with entropy 1.49

IF   are_ske is in [7.99, 10.3]     (MEDIUM)
 AND ecc_var is in [85.2,4.69e+04] (VERY LOW)
 AND ecc_ske is in [4.53, 14.8]    (somewhat_low OR MEDIUM
                                    OR somewhat_high)
THEN class IS 6 (REDUCED REL.)      with entropy 1.06
```

7.7 Empirical evaluation of classification performance

Although FRE is mainly targeted for linguistic data analysis, it is important to know how good its classification performance is in general. Although the classification performance of FRE seems to be competitive for the fiber image data, it does not have to be as good for the other data sets. Therefore FRE was tested against some publicly available data sets that were desctibed in Chapter 2. This evaluation was done for classification purposes only, and therefore no linguistic rules or data analysis is presented in addition to the quantitative results.

Five data sets were evaluated with all the methods in a multi-class mode, and sequential feature selection was applied with all the models except FRE, which used its own restricted full search. As before, the results are cross-validated mean prediction probabilities and their deviations.

The results are summarized in Table 23. As before FRE seems to give comparative results with other nonlinear methods like MLP, k-nn and QDA. A notable exception is the Vowel data. This is badly classified with all methods, and FRE is the second worst.

All reported runs were performed on a 1.67 GHz AMD Athlon MP 2000+ dual processor system, which has 2 GB memory. Only one processor was employed; no parallelizations of algorithms of any kind were attempted. Linear regression, logistic regression, MLP, k-nn, LDA and QDA were coded using Matlab (Version 6.5). FRE was implemented in ANSI-C. According to some test runs Matlab 6.5 implementations use twice as much time as the corresponding C-coded versions.

The computational complexity in respect to the number of data points was also tested with an artificial 10-dimensional data set. The data set had 2 normal-

TABLE 23 Summary of results on benchmark data sets.

	Iris	Vowel	Glass	Sonar	wine
lin.regr.(SFS)	83.3 ± 37.4	42.0	54.2 ± 49.9	59.6	99.4 ± 7.5
log.regr.(SFS)	94.0 ± 23.8	52.4	58.4 ± 49.4	57.5	98.3 ± 12.9
MLP(SFS)	96.7 ± 18.0	55.0	64.0 ± 48.1	76.7	97.2 ± 16.6
FRE (orig.)	96.7 ± 18.0	42.6	57.5 ± 49.6	71.2	96.1 ± 19.5
FRE (SFS)	96.0 ± 19.7	42.6	51.9 ± 50.1	71.2	96.1 ± 19.5
k-nn (SFS)	96.0 ± 19.7	52.6	72.0 ± 45.0	73.3	97.2 ± 16.6
LDA (SFS)	75.3 ± 43.3	48.5	52.3 ± 50.0	42.5	89.3 ± 31.0
QDA (SFS)	97.3 ± 16.2	56.1	58.9 ± 49.3	71.2	100.0 ± 0.0

distributed classes which overlapped. The center point of the first data cluster was positioned into origin and the center of the second cluster was set around point $(0.5, 0.5, \ldots, 0.5)$. Both classes had a unit variance for each variable. The time needed for training and classification of points was recorded for each method (see Table 24). Times corresponding to FRE consisted mostly of restricted full search times. The time required to classify 5000, 10000, 15000 and 20000 points were 0.01s, 0.03s, 0.05s and 0.07s, respectively. Hence, it can be seen that the time requirement is linear for FRE with respect to the number of data points.

TABLE 24 Time in seconds required to learn and classify 5000–20000 points. Average classification accuracy is given in the right column.

	5000	10000	15000	20000
linear regression	4 (77.9)	17 (78.3)	41 (78.6)	76 (78.5)
logistic regression	46 (78.7)	91 (79.2)	136 (79.5)	181 (79.5)
MLP	50 (82.3)	107 (82.4)	172 (82.9)	249 (82.7)
FRE	38 (79.3)	38 (79.5)	38 (79.6)	38 (79.6)
k-nn	65 (84.8)	268 (84.3)	592 (84.0)	1064 (83.6)
LDA	5 (81.1)	18 (81.4)	43 (81.4)	80 (81.3)
QDA	6 (86.2)	22 (86.3)	50 (86.3)	90 (86.3)

7.8 Concluding remarks

A full evaluation of FRE against other methods is not easy, because FRE presents a unique combination of different techniques. Most notably, FRE was not tested against other fuzzy methods, some of which do support feature selection. The reason for the lack of fuzzy comparisons was that such a task would have required

too much adaptation for the methods. Then the methods would no longer have the characteristics that they were designed for. Therefore testing was done with "standard" classifiers and a sequential forward selection.

Tests on real world data examples and well-known benchmark data sets showed that the proposed technique has potential when the main purpose is not only to produce a good classification but a clear and interpretable result. If the method cannot produce good rules, this is shown to the user as high-valued rule entropies. If the rule entropy is close to zero, the user can immediately see that the rule is reliable. The same cannot be said about the other techniques used in comparison; a lot of testing must be done to make sure that the method has actually learnt the data set well enough. Even then it is difficult to find out what the method has learnt.

8 DISCUSSIONS AND CONCLUSION

The main objective of this work is to find techniques for extracting meaningful linguistic information from numerical data. The purpose was to be able to produce fuzzy rules that are understandable and as precise as possible. Although the requirements for interpretability and precision are often contradictory, it was demonstrated in this work that they can be satisfied by using fuzzy rule-based systems.

Fuzzy systems were considered in this work, because they have something interesting to offer for data mining applications. From the users point of view this is simplicity and interpretability of automatically generated rules. If the user prefers accuracy, very accurate results may be produced (for the proof of universal approximation capability, see for example, Wang, 1994), but in this case there is no guarantee that the solution is easy to read. The same arguments do not apply to some other techniques used in data mining. Although the results of these techniques may be very accurate (e.g. classify new observations well) they usually appear as "black-box outputs" for the user. That is, it is difficult to find out why the technique produced some particular output. Another benefit of fuzzy rule-based systems is that rules can be made to possess both qualitative and quantitative information.

The formulation of fuzzy methods presented in this work was done such that the direct comparison to other methods (statistical classifiers, neural networks, interpolation and approximation techniques etc.) would be easy. This was a guideline when outlining a default fuzzy logic system for the work (Chapter 4) and when considering different approaches to achieve learning in such systems (Chapter 5). Also in Chapter 5 some parallels to so-called dictionary and kernel methods were drawn.

When constructing a fuzzy logic system, many decisions must be made. The user must decide the shape of membership functions, operations used for rule conjunction, disjunction and implication, and defuzzification method that converts the fuzzy output to a crisp value. The diversity of possible choices may confuse the user. Therefore an attempt was made to experimentally justify the choices made in this work (see Chapter 4). For example, as a result of experi-

mental testing it was found that the defuzzifiers that rely on the centre of gravity principle, the implications that are based on t-norms and the conjunctive operations that use algebraic product give good accuracy in certain type of function approximation problems. We could construct and formalize a fuzzy logic system that is both accurate and simple. Another benefit of the approach is that the algorithmic complexity of the model is linear in respect to the number of observations, which makes the method feasible to very large data sets as well. However, this was not explicitely tested in the applications.

The basic idea of fuzzy learning with linear methods was presented in Chapter 5. The main contribution was to show the feasibility of using simple linear learning as a basis for fuzzy rule construction. Also a rule selection via regularization was proposed. It was found that the least absolute shrinkage and selection operator (lasso) was well-suited for this task since it could effectively restrict the number of rules. By lasso regularization the weights of those rules could be set to zero which did not have much contribution to the result. The method was applied to classification task and it was then compared to some well-known classifiers.

Most fuzzy methods work best when the number of variables is quite small. Therefore a new fuzzy rule extraction (FRE) methodology was designed to solve highly multidimensional problems with tens of features. In this approach the curse of dimensionality is attacked by a technique that is related to some supervised feature selection techniques. Special for this methodology is the use of a restricted full search in feature selection, two-part architecture for the rules and the use of class-specific features. To increase readability of the rules a way to express them by using additional linguistic labels was proposed. The main motivation for this two-part (adaptive and fixed) rule set was that it simultaneously supports a precise but low interpretability solution and the suboptimal but readable solution. The usefulness of the readable rules was enhanced by incorporating both qualitative and quantitative information into them.

The evaluation of the methods proposed in this work was carried out by using empirical testings. Several data sets were used, but in these tests the role of fiber image data was emphasized. Empirical testing was preferred since theoretical justification of some method does not necessarily ensure that it works the best or even adequately in some real world problems. Empirical testings also requires full implementation of the methods.

Several non-fuzzy methods were applied to the fiber image data and then the results were compared to the results obtained by the fuzzy rule extraction technique. When comparing these methods to FRE we can see that the main difference is that they are not purposed for linguistic information extraction. Therefore the comparisons were based mostly on inspecting the classification accuracies and visual displays.

From the experiments it became clear that same information can be found using both traditional visual and new linguistic fuzzy data analysis methods. One such experiment was done that the user first selected the interesting data groups

from the computer screen and thus provided the groupings with class labels (e.g. indices). Then the role of variable selection was to say which variables were behind the user-made classification.

Tests with fiber image data and some well-known benchmark data sets showed that the proposed technique has potential when the main purpose is not only to produce good classification but a clear and interpretable result. As an example of this, it was shown that fuzzy linguistic rule extraction is feasible for the analysis of fiber image data.

REFERENCES

Abe, S., Thawonmas, R., and Kobayashi, Y. (1998). Feature selection by analyzing class regions approximated by ellipsoids. *IEEE Transactions on Systems, Man, and Cybernetics—Part C: Applications and Reviews*, 28(2):282–287.

Ackley, D.H., Hinton, G.E., and Sejnowski, T.J. (1985). A learning algorithm for Boltzmann machines. *Cignitive Science*, 9(1):147–169.

Aeberhard, S., Coomans, D., and de Vel, O. (1992). *Comparison of Classifiers in High Dimensional Settings*. Tech. Rep. no. 92-02, , Dept. of Computer Science and Dept. of Mathematics and Statistics, James Cook University of North Queensland.

Bandler, W., and Kohout, L.J. (1981). Semantics of implication operators and fuzzy relational products. In Mamdani, E.H. and Gaines, B.R., editors,*Fuzzy Reasoning and Its Applications*, Academic Press, New York.

Baraldi, A., and Blonda, P. (1999). A survey of fuzzy clustering algorithms for pattern recognition. *IEEE Transactions on Systems, Man, and Cybernetics, – Part B: Cybernetics*, 29(6):778–801.

Bekkermann, R., El-Yaniv, R., Tishby, N., and Winter, Y. (2003). Distributional Word Clusters vs. Words for Text Categorization. *JMLR Special Issue on Variable and Feature Selection*, 3:1183–1208.

Bellman, R. (1961). *Adaptive Control Processes: A Guided Tour*. Princeton University Press, New Jersey.

Bezdek, J.C. (1981). *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum, New York.

Bezdek, J.C., Keller, J.M., Khrisnapuram, R., Kuncheva, L.I., and Pal, N.R. (1999). Will the real iris data please stand up? *IEEE Transactions on Fuzzy Systems*, 7(3):368–369.

Bishop, C. (1995). *Neural Networks for Pattern Recognition*. Oxford University Press, Oxford.

Bossley, K.M. (1997). *Neurofuzzy Modelling Approaches in System Identification*. PhD thesis, University of Southampton, Faculty of Engineering and Applied Sciences.

Breiman, L., Friedman, J.H., Olshen, R.A., and Stone, C.J. (1984). *Classification and Regression Trees*. Chapman & Hall.

Brown, M., and Harris, C.J. (1994). *Neurofuzzy Adaptive Modelling and Control*. Prentice Hall, New York.

Casillas, J., Cordon, O., Del Jesus, M.J., and Herrera, F. (2001). Genetic feature selection in fuzzy rule-based classification system learning process for high-dimensional problems. *Information Sciences*, 136:135–157.

Chapelle, O., Haffner, P., and Vapnik, V.N. (1999). Support vector machines

for histogram-based image classification. *IEEE Transactions on Neural Networks*, 10(5):1055-1064.

Cherkassky, V., and Mulier, F. (1998). *Learning from Data: Concepts, Theory, and Methods.* John Wiley & Sons, New York.

Chi, Z., Yan, H., and Pham, T. (1996). *Fuzzy Algorithms: with Applications to Image Processing and Pattern Recognition*. World Scientific, Singapore.

Chiu, S. (1994). Fuzzy Model Identification Based on Cluster Estimation. *Journal of Intelligent & Fuzzy Systems*, 2(3):267–278.

Combs, W., and Andrews, J. (1998). Combinatorial rule explosion eliminated by a fuzzy rule configuration. *IEEE Transactions on Fuzzy Systems*, 6(1):1–11.

Cordón, O., Herrera, F., and Villar, P. (2001). Generating the knowledge base of a fuzzy rule-based system by the genetic learning of the data base. *IEEE Transactions on Fuzzy Systems*, 9(4):667–674.

Cover, T.M., and Thomas, J.A. (1991). *Elements of Information Theory*. Wiley, New York.

Dadone, P., and VanLandingham, H.F. (2000). On the non-differentiability of fuzzy logic systems. In *Proc. 2000 IEEE Int. Conf. Systems, Man & Cybernetics*, pages 2703–2708.

Devijver, P.A., and Kittler, J. (1982). *Pattern Recognition: A Statistical Approach*. Prentice Hall, London.

Driankov, D., Hellendoorn, H., and Reinfrank, M. (1993). *An Introduction to Fuzzy Control*. Springer-Verlag, Berlin Heidelberg.

Dubois, D., and Prade, H. (1999). Fuzzy sets in approximate reasoning, Part 1: Inference with possibility distributions. *Fuzzy Sets and Systems*, 100:73–132.

Duda, R.O., Hart, P.E., and Stork, D.G. (2001) *Pattern Classification*. Second edition, John Wiley & Sons.

Eysenck, M.W., and Keane, M.T. (1999). *Cognitive Psychology*. Third edition, Psychology Press Ltd.

Ferri, F.J., Pudil, P., and Kittler, J. (1994). Comparative Study of Techniques for Large-Scale Feature Selection. In Gelseman, E.S., and Kanal, L.N., editors, *Pattern Recognition in Practice IV: Multiple Paradigms, Comparative Studies, and Hybrid Systems*, pages 403–413, North Holland, Amsterdam.

Fisher, R. A. (1936). The Use of Multiple Measurements in Taxonomic Problems. *Annals of Eugenics* 7:179–188.

Friedman, J.H. (1995). *Introduction to Computational Learning and Statistical Prediction*. Tutorial, Stanford University, July 5.

Fukunaga, K. (1990). *Introduction to statistical Pattern Recognition*, 2nd edition, Academic Press, New York.

González, A., and Pérez,R. (2001). Selection of relevant features in a fuzzy genetic learning algorithm. *IEEE Transactions on System, Man, and Cybernetics–*

Part B: Cybernetics, 31(3):417–425.

Gonzalez, R.C., and Woods, R.E. (1992). *Digital Image Processing*. Addison-Wesley, 1992.

Gorman, R.P., and Sejnowski, T.J. (1988). Analysis of hidden units in a layered network trained to classify sonar targets. *Neural Networks*, 1:75–89.

Greenspan, H., Goodman, R., Chellappa, R., and Anderson, C.H. (1994). Learning texture discrimination rules in a multiresolution system. *IEEE Transactions on pattern analysis and machine intelligence*, 16(9):894–901.

Guely, F., and Siarry, P. (1993). Gradient descent method for optimizing various fuzzy rule bases. In *Second IEEE International Conference on Fuzzy Systems*, volume 2, pages 1241–1246.

Guyon, I., and Elisseeff, A. (2003). An Introduction to Variable and Feature Selection. *JMLR Special Issue on Variable and Feature Selection*, 3:1157–1182.

Häkkinen, E. (2001). *Design, Implementation and Evaluation of Neural Data Analysis Environment.* PhD Thesis, Jyväskylä Studies in Computing 12, University of Jyväskylä.

Hastie, T., Stuetzle, W. (1989). Principal curves. *Journal of the American Statistical Association*, 84(406):502–516.

Hastie, T., Tibshirani, R., and Friedman, J. (2001). *The Elements of Statistical Learning*. Springer-Verlag, New York.

Hisdal, E. (1994). Interpretative versus prescriptive fuzzy set theory. *IEEE Transactions on Fuzzy Systems*, 2(1):22–26.

Ho, D.W., Zhang, P.-A., Xu, J. (2001). Fuzzy wavelet networks for function learning. *IEEE Transactions on Fuzzy Systems*, 9(1):200–211.

Holland, J. H. (1995). *Adaptation in natural and artificial systems : an introductory analysis with applications to biology, control, and artificial intelligence*. 4th edition, Cambridge (MA): MIT Press.

Ichihashi, H., Shirai, T., Nagasaka, K., and Miyoshi, T. (1996). Neurofuzzy ID3: a method of inducing fuzzy decision trees with linear programming for maximizing entropy and an algebraic method for incremental learning. *Fuzzy Sets and Systems*, 81(1):157–167.

Jager, R. (1995). *Fuzzy Logic in Control*. Diss. Delft Universrity of Technology, Delft, the Netherlands.

Jain, A.K. (1989). *Fundamentals of Digital Image Processing*. Prentice Hall, New Jersey.

Jain, A.K., and Zongker, D. (1997). Feature selection: evaluation, application, and small sample performance. *IEEE Transactions on pattern analysis and machine intelligence*, 19(2):153–158.

Jain, A.K., Duin, R.P.W., and Mao, J. (1999). Statistical pattern recognition: a review. http://www.cfas.umd.edu/ kanunga/cmsc828K/resources/MSU-

CSE-00-5.ps.gz (20 October 2002).

Jang, J.-S.R. (1993). ANFIS: adaptive-network-based fuzzy inference systems. *IEEE Transactions on Systems, Man, and Cybernetics*, 23(3):665–685.

Jang J.-S.R., and Sun, C.-T. (1993). Functional equivalence between radial basis function networks and fuzzy inference systems. *IEEE Transactions on Neural Networks*, 4(1):156–159.

Kajanto, I.M., Komppa, A., and Ritala, R.K. (1989). How formation should be measured and characterized. *Nordic Pulp and Paper Research Journal*, 3.

Karayiannis, N.B., and Bezdek, J.C. (1997). An integrated approach to fuzzy learning vector quantization and fuzzy c-means clustering. *Transactions on Fuzzy Systems*, 5(4):622–628.

Karr, C. (1991). Applying genetics to fuzzy logic. *AI expert*, 6(3):26–33.

Kohonen, T. (1997). *Self Organizing Maps*. Springer Verlag.

Koikkalainen, P. (1994). Progress with the tree-structured self-organizing map. In Cohn, A., editor, *Proc. ECAI'94*, pages 211–215, John Wiley & Sons.

Kosko, B. (1997). *Fuzzy Engineering*. Prentice-Hall, Upper Saddle River, New Jersey.

Kosko, B. (1992a). Fuzzy systems as universal approximators. In *Proc. the First IEEE Conference on Fuzzy Systems*, pages 1153–1162, San Diego.

Kosko, B. (1992b). *Neural Networks and Fuzzy Systems: a Dynamical Systems Approach to Machine Intelligence*. Prentice-Hall, Englewood Cliffs, New Jersey.

Kudo, M., and Sklansky, J. (2000). Comparison of algorithms that select features for pattern classifiers. *Pattern Recognition*, 33:25–41.

Lantuéjoul, C. (2002). *Geostatistical simulation : models and algorithms*. Berlin: Springer, New York.

Laviolette, M., and Seaman Jr., J.W. (1994). The efficacy of fuzzy representations of uncertainty. *IEEE Transactions on Fuzzy Systems*, 2(1):4–15.

LeBlanc, M., and Tibshirani, R. (1994). Adaptive principal surfaces. *Journal of the American Statistical Association*, 89(425):53–64.

Lee, H.-M., Chen, C.-M., Chen, J.-M., and Jou, Y.-L. (2001). An efficient fuzzy classifier with feature selection based on fuzzy entropy. *IEEE Transactions on Systems, Man, and Cybernetics–Part B: Cybernetics*, 31(3):426–432.

Lekova, A., Mikhailov, L., Boyadjiev, D., and Nabout, A. (1998). Redundant fuzzy rules exclusion by genetic algorithms. *Fuzzy Sets and Systems*, 100(1-3): 235–243.

Lensu, A. (2002). *Computationally Intelligent Methods for Qualitative Data Analysis.* PhD Thesis, Jyväskylä Studies in Computing 23, University of Jyväskylä.

Lim, G., and Bezdek, J.C. (1998). Small targets in LADAR images using fuzzy clustering. In *Proc. IEEE World Congress on Computational Intelligence*, volume

1, pages 61–66.

Mark, R.E., Habeger Jr., C.C., Borch, J., and Lyne, M.B. (2002). *Handbook of Physical Testing of Paper*. Volume 1, Marcel Dekker, New York.

Mitaim, S., Kosko, B. (2001). The shape of fuzzy sets in adaptive function approximation. *IEEE Transactions on Fuzzy Systems*, 9(4):637–656.

Mitra, S., and Hayashi, Y. (2000). Neuro-fuzzy rule generation: survey in soft computing framework. *IEEE Transactions on Neural Networks*, 11(3):748–768.

Narendra, P.M., and Fukunaga, K. (1977). A branch and bound algorithm for feature subset selection. *IEEE Transactions on Computers*, 26(9):917–922.

Nauck D., and Kruse, R. (1999). Obtaining interpretable fuzzy classification rules from medical data. *Artificial Intelligence in Medicine*, 16:149–169.

Niemi, A. (1996). *Johdatus sumeisiin joukkoihin ja sumeaan logiikkaan*. Opetushallitus, Helsinki.

Niittylä, J. (2000a). *Kuva-analyysin menetelmiä kuitujakaumien karakterisointiin* (in Finnish). Master's thesis. Department of Mathematical Information Technology, University of Jyväskylä, Finland.

Niittylä, J. (2000b). *Tukivektorikoneet* (in Finnish). Report. Department of Mathematics and Statistics, University of Jyväskylä, Finland.

Niskanen, M. (2003). *A visual training based approach to surface inspection*. PhD Thesis, Department of Electrical and Information Engineering, University of Oulu.

Niskanen, V. (2001). Prospects for soft statistical computing: describing data and inferring from data with words in the Human Sciences. *Information Sciences*, 132:83–131.

Nozaki, K., Ishibuchi, H., and Tanaka, H. (1996). Adaptive fuzzy rule-based classification systems. *IEEE Transactions on Fuzzy Systems*, 4(3):238–250.

Ohser, J., and Mücklich, F. (2000). *Statistical Analysis of Microstructures in Material Science*. John Wiley & Sons.

Ollila, E. (2002). *Sign and rank covariance matrices with applications to multivariate analysis.* PhD Thesis, Department of Mathematics and Statistics, University of Jyväskylä.

Pal, S.K., and Chakraborty, B. (1986). Fuzzy set theoretic measure for automatic feature evaluation. *IEEE Transactions on Systems, Man, and Cybernetics*, 16:754–760.

Pal, S.K., and Mitra, S. (1999). *Neuro-Fuzzy Pattern Recognition*. John Wiley & Sons.

Pal, S.K., Rajat, K. De, and Basak, J. (2000). Unsupervised feature evaluation: a neuro-fuzzy approach. *IEEE Transactions on Neural Networks*, 11(2):366–376.

Pedrycz, W. (1995). *Fuzzy Sets Engineering*. CRC Press, Boca Raton.

Pudil, P., Novovicová, J., and Kittler, J. (1994). Floating search methods in feature selection. *Pattern Recognition Letters*, 15(11):1119–1125.

Quinlan, J.R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann.

Ramze Rezaee, M., Goedhart, B., Lelieveldt, B.P.F., and Reiber, J.H.C. (1999). Fuzzy feature selection. *Pattern Recognition*, 32:2011–2019.

Raymer, M.L., Punch, W.F., Goodman, E.D., Kuhn, L.A., and Jain, A.K. (2000). Dimensionality reduction using genetic algorithms. *IEEE Transactions on Evolutionary Computation*, 4(2):164–171.

Ripley, B. (1996). *Pattern Recognition and Neural Networks*. Cambridge University Press, Cambridge.

Rudas, I.J., and Kaynak, M.O. (1998). Entropy based operations on fuzzy sets. *IEEE Transactions on Fuzzy Systems*, 6(1):33–40.

Shannon, C.E. (1948). A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423,623–656.

Schürmann, J. (1996). *Pattern classification: a unified view of statistical and neural approaches.* John Wiley & Sons, New York.

Shi, Y., and Mizumoto, M. (2000). A new approach of neuro-fuzzy learning algorithm for tuning fuzzy rules. *Fuzzy Sets and Systems*, 112(1):99–116.

Simpson, P.K. (1992). Fuzzy min-max neural networks. I. Classification IEEE Transactions on Neural Networks, 3(5):776–786.

Sonka, M., Hlavac, V., and Boyle, R. (1996). *Image Processing, Analysis and Machine Vision*. Int. Thomson Computer Press.

Stoyan, D., Kendall, W.S., and Mecke, J. (1995). *Stochastic Geometry and its Applications*. John Wiley & Sons.

Sugeno, M., and Kang, G.T. (1988). Structure identification of fuzzy model. *Fuzzy Sets and Systems*, 28:15–33.

Takagi, T., and Sugeno, M. (1985). Fuzzy identification of systems and its applications to modelling and control. *IEEE Transactions on Systems, Man, and Cybernetics*, 15:116–132.

Theodoridis, S., and Koutroumbas, K. (2003). *Pattern Recognition, Second Edition*. Academic Press, Elsevier Science (USA).

Thawonmas, R., and Abe, S. (1997). A novel approach to feature selection based on analysis of class regions. *IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics*, 27(2):196–207.

Thuillard, M. (2001). *Wavelets in Soft Computing*. World Scientific, Singapore.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *IEEE Journal of Royal Statistical Society B.*, 58:267–288.

Tishby, N., Pereira, F., and Bialek, W. (1999). The information bottleneck method. In *Proc. of the 37th Annual Allerton Conference on Communication, Control, and Computing*.

Trepanier, R.J., Jordan, B.D., and Nguyen, N.G. (1998). Specific perimeter: statistic for assessing formation and print quality by image analysis. *Tappi Journal*,

81(10).

Tsang, E.C.C., Yeung, D.S., and Wang, X.Z. (2003). OFFSS: Optimal fuzzy-valued feature subset selection. *IEEE Transactions on Fuzzy Systems*, 11(2):202–213.

Vapnik, V.N. (1995). *The Nature of Statistical Learning Theory*. Springer-Verlag, New York.

Wang, C.-H., Hong, T.-P., and Tseng, S.-S. (1998). Integrating fuzzy knowledge by genetic algorithms. *IEEE Transactions on Evolutionary Computation*, 2(4):138–149.

Wang, L., and Mendel, J.M. (1991). Generating fuzzy rules by learning from examples. In *Proc. 1991 IEEE International Symposium on Intelligent Control*, pages 263–268, Arlington, Virginia, U.S.A.

Wang, L.-X. (1994). *Adaptive Fuzzy Systems and Control Design and Stability Analysis*. Prentice Hall, New Jersey.

Webb, A. (1999). *Statistical Pattern Recognition*. Butterworth-Heinemann.

Zadeh, L. (1965). Fuzzy sets. *Information and Control*, June, 338–354.

Zadeh, L. (1999). From computing with numbers to computing with words–from manipulation of measurements to manipulation of perceptions. *IEEE Transactions on Circuits and Systems–1: Fundamental Theory and Applications*, 45:105–119.

Zimmermann, H.J. (1985). *Fuzzy Set Theory—and Its Applications*. Kluwer, Boston.

Zimmermann, H.J. (1993). *Fuzzy Sets, Decision Making, and Expert Systems*. Kluwer, Boston.

APPENDIX : IMPLEMENTATION DETAILS

For the sake of clarity some implementation details were omitted from the text. These concern mainly about parameter settings of comparative methods in Chapter 7.5. These settings are summarized in the following.

Multilayer perceptron

MLP network had 4 output nodes (one for each class) and either 5 or 10 hidden layer nodes. The number of input nodes varied during the feature selection process. MLP with 5 hidden layer nodes performed slightly better than the network with 10 nodes. Therefore the results shown in this thesis were produced by a network with 5 hidden units. MLP was used with weight decay (ridge shrinkage) and it was applied to the problem with all the 48 fiber image features (shape and Fourier features):

- Levenberg-Marquardt (LM) optimization algorithm. The number of training epochs was limited to 50.
- Tanh activation functions in hidden units.
- Linear activation functions in output nodes.
- 5 hidden units, where the tanh activation functions of the form

$$g(a) = \frac{e^a - e^{-a}}{e^a + e^{-a}}$$

  were used.

Linear regression

Linear regression with lasso and all the 48 fiber image features:

- Quadratic programming as optimization.
- The parameter $\lambda = 0.2$ seemed to produce the best classification accuracy (estimated by leave-one-out validation). Therefore the result corresponding to it was used in the comparison.

The response categories were coded via an indicator variable. There were 4 indicators y_k, $k = 1, \ldots, 4$, with $y_k = 1$ if the class corresponding to the sample was k, otherwise $y_k = 0$. The values of y_k were collected to a vector $\mathbf{y} = (y_1, \ldots, y_4)$ and there was one vector for each sample. These vectors formed a

52×4 indicator response matrix $\mathbf{Y}$, which consisted of only 0's and 1's. The linear regression model was fit by

$$\hat{\mathbf{y}} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y},$$

where $\mathbf{X}$ is the model matrix with $p+1$ columns corresponding to the $p \leq d$ inputs. The leading column consists of 1's for the intercept. A new observation with input $\mathbf{x}$ was classified computing

$$\hat{\mathbf{f}}(\mathbf{x}) = [(1, \mathbf{x}^T)(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}]^T$$

and then finding the best class by $\mathrm{argmax}_k \hat{f}_k(\mathbf{x})$.

Logistic regression

Logistic regression requires a nonlinear optimization routine for training. A Matlab nlinfit-function which performs a nonlinear least-squares data fitting by the Gauss-Newton method was used.

K-nearest neighbors

With all tested $k = 1, 2, 3$ values k-nearest neighbors produced rates over 90%. The best result was obtained with $k = 2$.

Linear and quadratic discriminant analysis

The application of these techniques was straightforward. There were no parameters that had to be fixed beforehand.

FRE

Only one rule for each class was allowed. When using the restricted full search 5 features for each rule were allowed at maximum.

YHTEENVETO (FINNISH SUMMARY)

Väitöskirjan tarkoituksena on löytää menetelmiä sumeiden kielellisten sääntöjen tuottamiseen numeerisesta datasta. Menetelmien antamien tulosten tulee olla sekä ymmärrettäviä että tarkkoja. Kuitenkin tämä vaatimus on yleensä ristiriitainen, sillä riittävän tarkan ratkaisun saavuttaminen vaatii lähes poikkeuksetta mallin parametrien sovittamista. Jotta mallista saataisiin tarpeeksi joustava, joudutaan mallin yksinkertaisuudesta usein luopumaan (esim. luopumaan etukäteen kiinnitettyjen jäsenyysfunktioiden käytöstä). Tämä ymmärrettävyyden ja tarkkuuden välinen tasapaino on keskeisellä sijalla väitöskirjan tarkasteluissa.

Väitöskirjan yhtenä tavoitteena on löytää sumea järjestelmä, joka on riittävän yleinen siten että sitä voi soveltaa pienin modifikaatioin sekä regressio- että luokitteluongelmiin. Lähtökohtana on se että järjestelmä on oppiva ja sen avulla aikaansaatu datan representaatio (sumeat jos–niin säännöt) on helposti tulkittavissa.

Työssä esitettyjen menetelmien arviointi perustuu pääosin empiirisiin testauksiin. Arvioinnin pohjana käytetään erilaisia datajoukoja, joista kuitukuvista saadun aineiston roolia on korostettu. Perusteluna tähän on aineiston läheinen yhteys konkreettiseen reaalimaailman ongelmaan, joka voidaan esittää kysymyksenä: "kuinka karakterisoida kuituisesta materiaalista koostuva paperinäyte β-radiogrammikuvista laskettuja piirteitä käyttäen?"

Edellisten ohella työssä on arvioitu tärkeäksi myös piirreavaruuden dimensiosta aiheutuvien ongelmien minimointi. Hilarakenteeseen perustuvat järjestelmät, jollaisia useimmat sumean logiikan järjestelmistä ovat, kärsivät dimension kasvusta monia muita menetelmiä enemmän. Sumeissa järjestelmissä tämä tarkoittaa ensisijaisesti jäsenyysfunktioiden ja sääntöjen lukumäärän eksponentiaalista kasvua. Tästä syystä työhön on sisällytetty myös lyhyehkö katsaus erilaisista piirteiden valintaan soveltuvista menetelmistä.

Sääntojenirroitus on sääntöjen karsimisessa käytettävän lasso-menetelmän ohella työn ehkä kehityskelpoisin ja parhaiten käytännön sovelluksiin siirrettävissä oleva menetelmä. Sumean sääntöjenirroittimen ideana on löytää sopivat piirteet isosta piirrejoukosta ja käyttää löydettyjä piirteitä datan, yleensä erilaisten luokkien, selittämiseen. Sumean logiikan ansiosta selitykset saadaan helposti ymmärrettävinä jos–niin sääntöinä. Menetelmää testataan ja verrataan muihin, lähinnä tilastollisiin, menetelmiin pääasiassa kuituaineistoon pohjautuen.

JYVÄSKYLÄ STUDIES IN COMPUTING

1 ROPPONEN, JANNE, Software risk management - foundations, principles and empirical findings. 273 p. Yhteenveto 1 p. 1999.
2 KUZMIN, DMITRI, Numerical simulation of reactive bubbly flows. 110 p. Yhteenveto 1 p. 1999.
3 KARSTEN, HELENA, Weaving tapestry: collaborative information technology and organisational change. 266 p. Yhteenveto 3 p. 2000.
4 KOSKINEN, JUSSI, Automated transient hypertext support for software maintenance. 98 p. (250 p.) Yhteenveto 1 p. 2000.
5 RISTANIEMI, TAPANI, Synchronization and blind signal processing in CDMA systems. - Synkronointi ja sokea signaalinkäsittely CDMA järjestelmässä. 112 p. Yhteenveto 1 p. 2000.
6 LAITINEN, MIKA, Mathematical modelling of conductive-radiative heat transfer. 20 p. (108 p.) Yhteenveto 1 p. 2000.
7 KOSKINEN, MINNA, Process metamodelling. Conceptual foundations and application. 213 p. Yhteenveto 1 p. 2000.
8 SMOLIANSKI, ANTON, Numerical modeling of two-fluid interfacial flows. 109 p. Yhteenveto 1 p. 2001.
9 NAHAR, NAZMUN, Information technology supported technology transfer process. A multi-site case study of high-tech enterprises. 377 p. Yhteenveto 3 p. 2001.
10 FOMIN, VLADISLAV V., The process of standard making. The case of cellular mobile telephony. - Standardin kehittämisen prosessi. Tapaustutkimus solukkoverkkoon perustuvasta matkapuhelintekniikasta. 107 p. (208 p.) Yhteenveto 1 p. 2001.
11 PÄIVÄRINTA, TERO, A genre-based approach to developing electronic document management in the organization. 190 p. Yhteenveto 1 p. 2001.
12 HÄKKINEN, ERKKI, Design, implementation and evaluation of neural data analysis environment. 229 p. Yhteenveto 1 p. 2001.
13 HIRVONEN, KULLERVO, Towards Better Employment Using Adaptive Control of Labour Costs of an Enterprise. 118 p. Yhteenveto 4 p. 2001.
14 MAJAVA, KIRSI, Optimization-based techniques for image restoration. 27 p. (142 p.) Yhteenveto 1 p. 2001.
15 SAARINEN, KARI, Near infra-red measurement based control system for thermo-mechanical refiners. 84 p. (186 p.) Yhteenveto 1 p. 2001.
16 FORSELL, MARKO, Improving Component Reuse in Software Development. 169 p. Yhteenveto 1 p. 2002.
17 VIRTANEN, PAULI, Neuro-fuzzy expert systems in financial and control engineering. 245 p. Yhteenveto 1 p. 2002.
18 KOVALAINEN, MIKKO, Computer mediated organizational memory for process control. Moving CSCW research from an idea to a product. 57 p. (146 p.) Yhteenveto 4 p. 2002.
19 HÄMÄLÄINEN, TIMO, Broadband network quality of service and pricing. 140 p. Yhteenveto 1 p. 2002.
20 MARTIKAINEN, JANNE, Efficient solvers for discretized elliptic vector-valued problems. 25 p. (109 p.) Yhteenveto 1 p. 2002.
21 MURSU, ANJA, Information systems development in developing countries. Risk management and sustainability analysis in Nigerian software companies. 296 p. Yhteenveto 3 p. 2002.
22 SELEZNYOV, ALEXANDR, An anomaly intrusion detection system based on intelligent user recognition. 186 p. Yhteenveto 3 p. 2002.
23 LENSU, ANSSI, Computationally intelligent methods for qualitative data analysis. 57 p. (180 p.) Yhteenveto 1 p. 2002.
24 RYABOV, VLADIMIR, Handling imperfect temporal relations. 75 p. (145 p.) Yhteenveto 2 p. 2002.
25 TSYMBAL, ALEXEY, Dynamic integration of data mining methods in knowledge discovery systems. 69 p. (170 p.) Yhteenveto 2 p. 2002.
26 AKIMOV, VLADIMIR, Domain Decomposition Methods for the Problems with Boundary Layers. 30 p. (84 p.). Yhteenveto 1 p. 2002.
27 SEYUKOVA-RIVKIND, LUDMILA, Mathematical and Numerical Analysis of Boundary Value Problems for Fluid Flow. 30 p. (126 p.) Yhteenveto 1 p. 2002.
28 HÄMÄLÄINEN, SEPPO, WCDMA Radio Network Performance. 235 p. Yhteenveto 2 p. 2003.
29 PEKKOLA, SAMULI, Multiple media in group work. Emphasising individual users in distributed and real-time CSCW systems. 210 p. Yhteenveto 2 p. 2003.
30 MARKKULA, JOUNI, Geographic personal data, its privacy protection and prospects in a location-based service environment. 109 p. Yhteenveto 2 p. 2003.
31 HONKARANTA, ANNE, From genres to content analysis. Experiences from four case organizations. 90 p. (154 p.) Yhteenveto 1 p. 2003.
32 RAITAMÄKI, JOUNI, An approach to linguistic pattern recognition using fuzzy systems. 165 p. Yhteenveto 1 p. 2003.