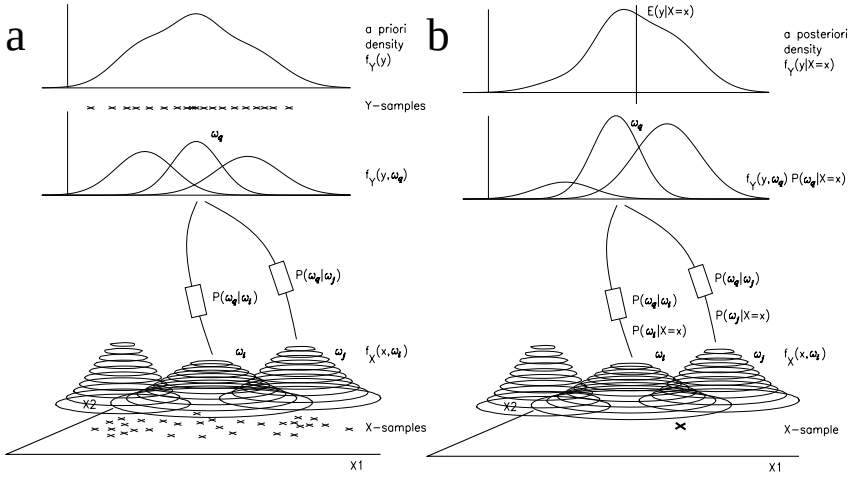
KUNGL TEKNISKA HÖGSKOLAN

STOCKHOLMS UNIVERSITET

On Data Mining and Classification Using a Bayesian Confidence Propagation Neural Network

Roland Orre



Stockholm 2003

Doctoral Dissertation

Royal Institute of Technology

Department of Numerical Analysis and Computer Science

Cover: Illustration of function approximation with output as a posterior distribution.

a: Training, estimate density in data set.

b: Input of specific x, which gives posterior y distribution as output.

Akademisk avhandling som med tillstånd av Kungl Tekniska Högskolan framlägges till offentlig granskning för avläggande av teknologie doktorsexamen måndagen den 22 September 2003 kl 10.00 i Kollegiesalen, Administrationsbyggnaden, Kungl Tekniska Högskolan, Vallhallavägen 79, Stockholm.

ISBN 91-7283-508-7
TRITA-NA-0308
ISSN 0348-2952
ISRN KTH/NA/R--03/08—SE

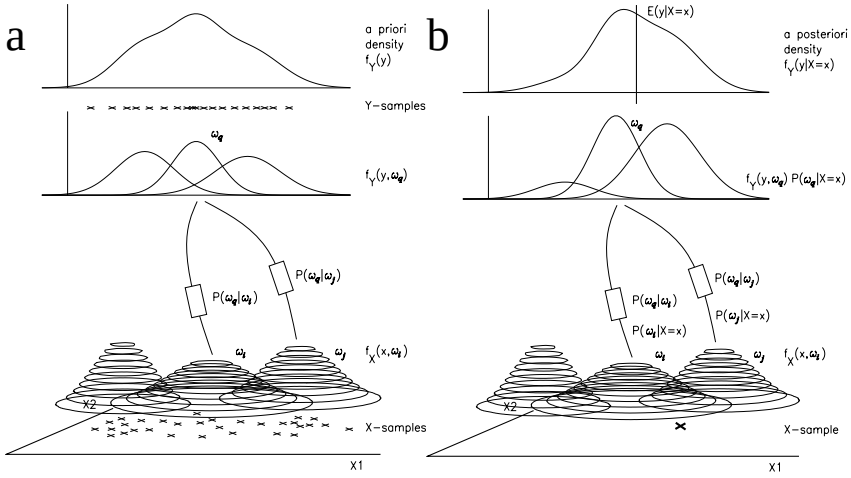
Universitetsservice US-AB, Stockholm 2003

KUNGL TEKNISKA HÖGSKOLAN

STOCKHOLMS UNIVERSITET

On Data Mining and Classification Using a Bayesian Confidence Propagation Neural Network

Roland Orre



Stockholm 2003

Doctoral Dissertation

Royal Institute of Technology

Department of Numerical Analysis and Computer Science

To Ottilia and Sebastian,
my beloved children

$\mathcal{T}$hat which is truly beautiful and worthwhile in life
is generally hidden away in mines of secrecy.
– *Paramahansa Yogananda*

Abstract

The aim of this thesis is to describe how a statistically based neural network technology, here named BCPNN (Bayesian Confidence Propagation Neural Network), which may be identified by rewriting Bayes' rule, can be used within a few applications, data mining and classification with credibility intervals as well as unsupervised pattern recognition.

BCPNN is a neural network model somewhat reminding about Bayesian decision trees which are often used within artificial intelligence systems. It has previously been successfully applied to classification tasks such as fault diagnosis, supervised pattern recognition, hiearchical clustering and also used as a model for cortical memory. The learning paradigm used in BCPNN is rather different from many other neural network architectures. The learning in, *e.g.* the popular backpropagation (BP) network, is a gradient method on an error surface, but learning in BCPNN is based upon calculations of marginal and joint probabilities between attributes. This is a quite time efficient process compared to, for instance, gradient learning. The interpretation of the weight values in BCPNN is also easy compared to many other network architechtures. The values of these weights and their uncertainty is also what we are focusing on in our data mining application. The most important results and findings in this thesis can be summarised in the following points:

- We demonstrate how BCPNN (Bayesian Confidence Propagation Neural Network) can be extended to model the uncertainties in collected statistics to produce outcomes as distributions from two different aspects: uncertainties induced by sparse sampling, which is useful for data mining; uncertainties due to input data distributions, which is useful for process modelling.
- We indicate how classification with BCPNN gives higher certainty than an optimal Bayes classifier and better precision than a naïve Bayes classifier for limited data sets.
- We show how these techniques have been turned into a useful tool for real world applications within the drug safety area in particular.
- We present a simple but working method for doing automatic temporal segmentation of data sequences as well as indicate some aspects of temporal tasks for which a Bayesian neural network may be useful.
- We present a method, based on recurrent BCPNN, which performs a similar task as an unsupervised clustering method, on a large database with noisy incomplete data, but much quicker, with an efficiency in finding patterns comparable with a well known (Autoclass) Bayesian clustering method, when we compare their performane on artificial data sets. Apart from BCPNN being able to deal with really large data sets, because it is a global method working on collective statistics, we also get good indications that the outcome from BCPNN seems to have higher clinical relevance than Autoclass in our application on the WHO database of adverse drug reactions and therefore is a relevant data mining tool to use on the WHO database.

Key words: Artificial neural network, Bayesian neural network, data mining, adverse drug reaction signalling, classification, learning.

Acknowledgements

Before I make any acknowledgements I want to say that it is hard to decide who to thank explicitly in projects with many people involved, there will always be someone missed.
Anyway, I have decided upon a minimal collection of people to thank personally.
Yes, to start with I would of course thank my supervisor Stefan Arnborg who has patiently discussed issues and given me many useful hints on my way from lic to phd.
Enchanting thanks to Juni Palmgren for suggesting that I become a guest researcher at the mathematical statistical department at Stockholm University, a great place to be at.
Supreme thanks to the people at Uppsala Monitoring Centre Andrew Bate, Ralph Edwards, Marie Lindquist, Sten Olsson and many many others who all make our joint collaborative research and development effort an inspiring and exciting journey.
Innermost thanks to my co-workers Jonathan Edwards, Niklas Norén and Erik Swahn at NeuroLogic. A well disciplined and very clever team, it is a real joy to work with you all.
Also great thanks to Mikael Djurfeldt for being a good friend and for insightful discussions about *e.g.* GNU/Guile, the Scheme language being our high level development language.
Next I would like to thank all the people at the mathematical statistics department Mikael Andersson, Tommi Asikainen, Anders Björkström, Jelena Bojarova, Gudrun Brattström, Tom Britton, Maria Deijfen, Annica Dominicus, Maria Grünewald, Ola Hammarlid, Thomas Höglund, Ola Hössjer, Gudrun Jonasdottir, Louise af Klintberg, Anders Martin-Löf, Christina Nordgren, Andreas Nordvall Lagerås, Esbjörn Ohlsson, Jan-Olov Persson, Örjan Stenflo, Rolf Sundberg, Åke Svensson, Joanna Tyrcha, who are part of the group at the time of writing and Bogdan Godymirski, Mia Hinnerich, Johan Irbäck, Johan Lindbäck, Marie Linder, Marianne Mähle Schmidt, Tina Orusild, Samuli Ripatti, Niklas Sjögren, who also have been members of the group during my time here.
–
Lustrous thanks to Anders Lansner, Örjan Ekeberg, Anders Holst, Anders Sandberg and all the people in the SANS group where I did my lic thesis. Anders L. and Örjan E. were the pioneers behind BCPNN. Anders H. and Anders S. have developed the BCPNN further.
Obeisant thanks to Rogelio de Freitas who took the initiative to bring people from UMC and SANS together, thus joining the WHO database and the BCPNN methodology.
Great thanks to Richard M. Stallman founder of GNU and FSF (Free Software Foundation).
If Linus Torvalds hadn't started the Linux kernel as a community effort it would probably have taken much more time for free software to become so widely accepted. Thanks!
Crash proof, reliable, programmer friendly and almost resistant against all forms of virus and worm attack, GNU/Linux is a great and productive environment to work within.
–
To end this with a guarantee that I haven't missed anyone I just want to say:
Overwhelming thanks to all people I have not mentioned here, like supporting friends, family, relatives as well as all other people that certainly know that you are a part of this.
–
At last I want to say it was fun doing this. It's a lot of work and it puts pressure on oneself and people around you but it's stimulating, inspiring and you feel very good when it's done.
If I got the question, "Would you have started with this if you had known how much work it would imply?", I would unconditionally answer, Yes! Thanks again to all involved.

Contributions

This thesis consists of two parts. The first part is a summary and overview of the included papers with introductions to the problem areas including discussions about some of the results obtained. The second part consists of eight supplements embodying the following papers appearing in non-chronological order. They will be referred to in the text by their Roman numerals:

I Orre R., Lansner A.,Bate A. and Lindquist M. (2000). Bayesian neural networks with confidence estimations applied to data mining. *Computational Statistics and Data Analysis*, **34**(4):,473–493

II Bate A., Lindquist M., Edwards I. R., Olsson S., Orre R., Lansner A., and Freitas R. M. D. (1998). A Bayesian neural network method for adverse drug reaction signal generation. *European Journal of Clinical Pharmacology* **54**: 315–321.

III Koski T. and Orre R. (1998). Statistics of the information component in Bayesian neural networks. Tech. Rep. TRITA-NA-P9806, Dept. of Numerical Analysis and Computing Science, Royal Institute of Technology, Stockholm, Sweden.

IV Orre R. and Lansner A. (1996). Pulp quality modelling using Bayesian mixture density neural networks. *Journal of Systems Engineering* **6**: 128–136.

V Norén N. and Orre R. (2003). The Bayesian bootstrap as a means to analyze the imprecision in probabilistic classification. *Manuscript*

VI Orre R. and Lansner A. (1990). A method for temporal association in Bayesian networks. Tech. Rep. TRITA-NA-P9018, Dept. of Numerical Analysis and Computing Science, Royal Institute of Technology, Stockholm, Sweden.

VII Orre R. and Lansner A. (1992). A Bayesian network for temporal segmentation. In *Artificial Neural Networks, Proceedings ICANN-92*, pp. 1081–1084. Royal Institute of Technology, Stockholm, Sweden, 1992.

VIII Orre R., Bate A., Norén N. G., Swahn E., Arnborg S. and Edwards I. R. (2003). A Bayesian recurrent neural network for unsupervised pattern recognition in large incomplete data sets. *Submitted*

Contents

Papers

Thesis Summary

1 Introduction

What we are trying to do here, can from some perspective appear to be quite simple. By observing some events we want to find out if they have some relation, and we also want to be able to tell something about the future of similar types of events when we know only some of them. Ultimately we also want to gain some understanding and knowledge about the world in which these events were created.

Even though this may at first sound simple the process includes many obstacles of which we are be able to deal only with a few of them here. Scientific discovery, for instance, which is our main method to gain knowledge about this world, is based upon sampling a subspace of events on which hypotheses can be tested and theories be built. These events can be measured in probabilities and rules can be deduced about relations between different events. To measure probabilities, however, is nothing that can be done directly. A probability is a property that can only be estimated by indirect measures. We can only estimate a probability to a certain accuracy as long as we don't have infinite amounts of data available, and even if we had infinite amounts of data we could not really be sure that the probability we have estimated is not generated by a large set of different processes. To really understand what is going on we would need to understand the process which generated the probabilities we are estimating. This generating process is in itself, however, most often not possible to explore. So, we have to satisfy ourselves by the fact that nothing can be known for sure. As we will deal a lot with probabilities in this text, let us define a few *axioms*. The first three axioms, although here somewhat simplified, were defined by Kolmogorov [Kolmogorov, 1933]. These implies other rules, where one very important one is no 4, about *conditional probability* [Gut, 1995]. Let A and B be two events:

1. $P(A)$ is a number between 0 and 1.
2. If A represents a certain event, then $P(A) = 1$.
3. If A and B are mutually exclusive events, then $P(A \text{ or } B) = P(A) + P(B)$.
4. If A, B is the joint event where both A and B are true, then the *conditional probability* or *belief* for event A given that we know B is

$$P(A \mid B) = \frac{P(A,B)}{P(B)} \left[= \frac{P(A\&B)}{P(B)} = \frac{P(A \cap B)}{P(B)} \right] \tag{0}$$

As the above axioms deal with probabilities as numbers, which we cannot measure directly, only by estimations, we will to a large extent consider the probabilities themselves to be parameters with continuous probability density functions. Here we are especially interested in conditional density functions. It is not trivial to show that these axioms above are also applicable to continuous density functions. In the general case it requires elaborate measure theory to prove [Moran, 1968], which is studied in [Doob, 1953] [Kolmogorov, 1933] and [Loève, 1963]. Let us therefore summarise that for two random variables X and Y which have a joint distribution where $f_X(x) > 0$ the conditional density function of Y given $X = x$ is [Gut, 1995]:

$$f_Y(y \mid X = x) = \frac{f_{X,Y}(x,y)}{f_X(x)}, \text{ [or for discrete events:] } p_Y(y \mid X = x) = \frac{p_{X,Y}(x,y)}{p_X(x)}$$

The last axiom gives us the possibility to perform *inferences* about some events when we know something about other events. Often this axiom is written in the following way:

$$P(A \mid B)\, P(B) = [P(A, B)] = P(B \mid A)\, P(A)$$
$$\implies P(A \mid B) = P(A)\, \frac{P(B \mid A)}{P(B)}, \tag{1}$$

which use to be referred to as *Bayes rule* or *Bayes theorem* to honour Thomas Bayes, whose publication; *An essay towards solving a problem in the doctrine of chances* which was published posthumously 1763, two years after Bayes death, by Richard Price in *Philosophical Transactions of the Royal Society* [Bayes, 1763]; contained this theorem as the key result.

This theorem (1) can be interpreted such as we have a hypothesis $P(A)$ (*prior belief*) about the probability for event A, which is updated to obtain $P(A \mid B)$ (*posterior belief*) when we get some new evidence B.

1 Simple example using Bayes rule

We can illustrate Bayes rule (eq 1) with a simple diagnosing example from an oncology clinic: Let A represent the event "person has cancer" and let B represent the event "person is a smoker". Assume that we know that $P(A) = 0.1$ as 10% of the patients coming to the clinic turned out to have cancer. From our studies we also know that half of the patients coming to the clinic are smokers, *i.e.* $P(B) = 0.5$ and among the patients which were diagnosed to have cancer 80% were smokers. That is, the conditioned probability $P(B|A) = 0.8$, which can also be seen as the *likelihood* for data, expressed by the event B, given the *model* A. Now we can use Bayes rule (eq 1) to calculate the *posterior probability* for response A given the explanatory event B as

$$\text{posterior probability} = \text{prior probability} \cdot \frac{\text{likelihood}}{\text{probability for data}}, \textit{ i.e.} \tag{2}$$

$$P(\text{cancer} \mid \text{smoker}) = P(\text{cancer}) \cdot \frac{P(\text{smoker} \mid \text{cancer})}{P(\text{smoker})} = 0.1 \cdot \frac{0.8}{0.5} = 0.16 \tag{3}$$

When seeing examples like this we should be aware about that they are always conditioned on some background event space H. In this case $H = \{\text{people being diagnosed at the clinique}\}$. So the more correct writing of the above example should be

$$P(\text{cancer} \mid \text{smoker}, H) = P(\text{cancer}|H) \cdot \frac{P(\text{smoker} \mid \text{cancer}, H)}{P(\text{smoker} \mid H)}$$

Most often it is OK to write it like in (eq 3) though, as the background H, which the whole expression is conditioned on, can be construed as implicit as long as it is not forgotten.

Although the example above is simple it expresses the *fundamental* principle behind Bayesian logic [1] or *logical inference*, that prior information is updated with evidence from data to obtain posterior information. However, in general, it is not trivial to find the likelihood function, neither to estimate the prior, nor to actually calculate the complete posterior distribution.

[1] Probability theory, as originated by Laplace, is a generalisation of Aristotelian logic that reduces to deductive logic in the special case of a true/false hypothesis [G. L. Bretthorst]

2 Another example using Bayes rule "The three door problem"

As the previous simple example probably was quite intuitive let us have a look at an example which is sometimes considered controversial and people may argue about it as its outcome is based upon *subjective* information. The famous "Monty hall problem" which is sometimes referred to as the "Three door problem". It got its name from an American TV show called "Let's Make a Deal" hosted by Monty Hall [2]

Assume you are a guest in this game show. You have the choice to select between three doors. Behind two of these doors there is a goat and behind the third one there is a prize, a big fortune. We assume that you prefer the fortune before a goat. The game leader (Monty Hall) asks you to select one of the doors. After you have chosen a door he will unconditionally open one of the other doors, but not the door where the fortune is hidden. Assume you choose door A. He then opens one of the other doors and reveals a goat. You are then offered the option to switch to the other closed door. The question is: should you switch door because this would increase the probability to win the prize or should you stay with the already chosen one? The *a priori* probability to find the fortune behind any of the doors is $1/3$ as we have no reason to suspect that any of the doors would be to prefer. The probability that Monty opens door B if the prize is behind door A is:

$$P(M\,opens\,B\,|\,A) = \frac{1}{2}$$

The probability that Monty opens door B if the prize is behind door B is:

$$P(M\,opens\,B\,|\,B) = 0$$

The probability that Monty opens door B if the prize is behind door C is:

$$P(M\,opens\,B\,|\,C) = 1$$

And finally, the probability that Monty opens door B is:

$$\begin{aligned} P(M\,opens\,B) &= P(A)\cdot P(M\,opens\,B\,|\,A) \\ &+ P(B)\cdot P(M\,opens\,B\,|\,B) \\ &+ P(C)\cdot P(M\,opens\,B\,|\,C) \\ &= \frac{1}{3}\cdot\frac{1}{2} + \frac{1}{3}\cdot 0 + \frac{1}{3}\cdot 1 = \frac{1}{2} \end{aligned}$$

Now Bayes theorem (1) gives:

$$P(A\,|\,M\,opens\,B) = P(A)\cdot\frac{P(M\,opens\,B\,|\,A)}{P(M\,opens\,B)} = \frac{1}{3}\cdot\frac{1/2}{1/2} = \frac{1}{3} \tag{4}$$

$$P(C\,|\,M\,opens\,B) = p(C)\cdot\frac{P(M\,opens\,B\,|\,C)}{P(M\,opens\,B)} = \frac{1}{3}\cdot\frac{1}{1/2} = \frac{2}{3} \tag{5}$$

That is, the probability for the prize to be behind door C when Monty opens door B is $2/3$, *i.e.* you should switch. It can of course be considered overkill to use Bayesian analysis for a simple problem like this. This may, however, be necessary because people who have not analysed this problem often intuitively suggest that there would be a $50/50$ chance for the prize to be behind any of the remaining doors. Alternatively one could do a simpler analysis saying that as $P(A) = 1/3$ then $P(B\,or\,C) = 1 - 1/3 = 2/3$ as the total probability must sum to one, which is also a correct solution.

[2] The Monty Hall problem is not at all truly reflecting the show, it is only *inspired* by it.

3 Intention with the thesis

As the previous examples may indicate this thesis will deal with Bayesian inference, but not from a general perspective, we will only deal with a specific implementation of Bayesian inference embedded in a computational structure named *artificial neural networks*.

The specific neural network architecture we are dealing with here is called *Bayesian Confidence Propagation Neural Network* (BCPNN). As a Bayesian classifier this type of network can be considered to be a *semi-optimal* Bayesian classifier. It is less general than the popular *Bayesian networks* [Pearl, 1988], [Heckerman, 1997], [Jensen, 2001] but powerful enough to deal with most classification and prediction problems which can be defined in *explanatory* and *response variables*, that is *input layer* and *output layer* using the terminology within the neural network area. The BCPNN used as a Bayesian classifier is basically a so called *feed forward* neural network, but it can also be used as a *recurrent* neural network [Holst and Lansner, 1993b], an attractor network which can be used as *e.g.* an associative memory [Kohonen, 1988] or, as we will describe later, a method for *unsupervised* pattern recognition.

As a Bayesian classifier BCPNN is semi-optimal in that sense that it is more powerful than a *naive* or *simple* Bayesian classifier but it is less powerful than an *optimal* Bayesian classifier. In the optimal Bayesian classifier we do not assume anything about the variable's dependencies and in the naive Bayesian classifier we assume that the different input events are independent. Consider for instance the following example of a prediction of $P(y|x_1, x_2, x_3, x_4)$, for optimal Bayes (eq 6) vs naive Bayes (eq 7):

$$P(y|x_1, x_2, x_3, x_4) = P(y) \cdot \frac{P(x_1, x_2, x_3, x_4 \,|\, y)}{P(x_1, x_2, x_3, x_4)} \tag{6}$$

[assume x_1, x_2, x_3, x_4 are independent conditioned on y]

$$= P(y) \cdot \frac{P(x_1 \,|\, y)}{P(x_1)} \cdot \frac{P(x_2 \,|\, y)}{P(x_2)} \cdot \frac{P(x_3 \,|\, y)}{P(x_3)} \cdot \frac{P(x_4 \,|\, y)}{P(x_4)} \tag{7}$$

For practical reasons the optimal Bayes classification can most often not be implemented. For this to work we would need data from all possible combinations of the input and output variable states. Therefore, BCPNN uses an approach between these two extremes, *i.e.* to consider the joint distributions for those variables which are much dependent and to consider the rest of the variables to be independent. As for instance, assume we found that x_1 and x_2 are dependent but x_3 and x_4 are independent, then the *semi-optimal* classification is:

$$P(y|x_1, x_2, x_3, x_4) = P(y) \cdot \frac{P(x_1, x_2 \,|\, y)}{P(x_1, x_2)} \cdot \frac{P(x_3 \,|\, y)}{P(x_3)} \cdot \frac{P(x_4 \,|\, y)}{P(x_4)} \tag{8}$$

The intention with this thesis is to show how this type of classification performed by the BCPNN can be used for *data mining* and classification with uncertainty. Data mining, is part of the process to turn data into knowledge. The data mining part we are dealing with here are of two kinds, one is to find dependencies between variables. The other kind of data mining we deal with is unsupervised pattern recognition which is related to clustering. That is, to find interesting new patterns in data. Concerning classification we deal with some aspects of predictions with uncertainty estimates of the prediction.

4 Content of the thesis

This thesis is a compilation of a few papers presented as supplements. These papers are presented in a logical order, not in chronological order. The connecting thought in the first part is the fundamentals about finding dependencies between variables or, to be more specific, dependencies between variable value states. These dependency derivations are then applied to classification problems, where we also discuss classification with uncertainty. The rest of the thesis deals with some aspects of pattern recognition; how to automatically do segmentation and how to find new patterns in data.

In **Paper I** we start with the fundamental principles with BCPNN and describe the weights in this network which are defined by the measure $W_{ij} = \frac{P(x_i|y_j)}{P(x_i)}$ or as we often use them in their logarithmic form, $IC_{ij} = \log \frac{P(x_i|y_j)}{P(x_i)}$. The logarithmic IC_{ij} we denote *information component* between state i of variable x and state j of variable y because its relation to *mutual information*. We describe how the shape, or more simplified the variance, of the IC_{ij} function determines the uncertainty when we use this measure for data mining. In Paper I we also discuss how to select suitable priors for IC_{ij} and how to propagate the uncertainties in IC_{ij} through a BCPNN.

Paper II describes a successful data mining application where the technique defined in Paper I is used for data mining of a large database, the WHO database of suspected adverse drug reactions, to look for early signals of adverse drug reactions.

In **Paper III** is a presentation of how the logarithmic information component and its variance can be calculated with arbitrary precision, but this does not imply that the implementation is straight forward though.

Paper IV tells how the BCPNN network is used for modelling pulp quality in an industrial paper manufacturing process. This can be seen as a simple type of process modelling. In this case the BCPNN network is set up to use mixture distributions in input and output layers. As it is presented here this network implements an *optimal Bayes classifier* for real values as we use a mixture distribution covering all dimensions of the real valued input space using an *expectation maximisation* kernel regression method to model the input distribution with Gaussian *radial basis functions*. The outcome here are mixture distributions from which we can estimate *confidence intervals* for the predicted values. It may be noted that these confidence intervals are not so called Bayesian *credible intervals* or *probability intervals* which is the relevant denotation when we estimate the uncertainty of a probability. The estimated intervals here reflects a model of the data generation and therefore it is more relevant to denote them confidence intervals. It should be stressed that what this network does is in principle an association of an input density function with an output density, which makes this method more powerful than any pure regression technique like *e.g.* multilayer perceptrons using *backprop* [Hertz *et al.*, 1991]. The uncertainties which are dealt with here are only the uncertainties of the data samples, not the uncertainties of the weights within the network itself, which is addressed in papers Paper I and Paper V .

In **Paper V** we deal with the problem of accurately propagating the uncertainties due to the distribution functions of the weights in the BCPNN as fully posterior distributions for the outcome classes. This is unfortunately something that is not possible to do analytically, even though better approximation methods can be considered. In Paper V we deal with the problem using *Bayesian bootstrap* [Rubin, 1981] methods to sample from distributions modelling the underlying data. We compare predictions of posterior distributions for naive Bayes, optimal Bayes and BCPNN, where the choice of distributions for the variables fulfils

the dependency criterion for the BCPNN *predictor* (or *explanatory*) variables.

Now we leave the feed forward only network and look upon some applications using *recurrent* networks. **Paper VI** presents an idea of one way a neural network can be used to handle spatio-temporal patterns and possibly also do spatio-temporal function approximation which is an important issue for any kind of general process modelling involving both prediction and control as well as perceptive temporal tasks like speech recognition or the more speculative ones like prediction of future interest rates. In Paper VI we are using a *recurrent* BCPNN as an associative memory linking future events to past events. This is applied to a toy problem of recognising and generating character sequences.

In the next paper, **Paper VII** , we make an attempt to cope with the subject of self organising temporal segmentation, which is an important issue when dealing with temporal problems, such as *e.g.* speech recognition. A temporal segment can be seen as a piece of sequential information which is often seen together and thus may have a certain meaning. The temporal segmentation problem has some resemblance with the data mining problem as presented in Paper I because it is based upon pair-wise variable dependencies, but here an inference is also performed to recall the most likely pattern.

In **Paper VIII** we present an approach to do data mining of many interacting variables, automatically forming patterns from noisy, incomplete data sets. This reminds much about *cluster analysis* and we compare BCPNN with a "standard" clustering/classification method named Autoclass [Cheeseman and Stutz, 1995a]. The performance of the two methods is measured on an artificial data set as their resistance against two types of noise and their ability to cope with incompleteness in the data. The paper in Paper VIII deals with a somewhat simpler issue than Autoclass, because from the Autoclass output you can also assign a probability for each attribute in each class. Even though such attribute probabilities can easily be obtained also with BCPNN by using the obtained patterns as "classifiers" with the help of a a distance/similarity measure. This way each sample can be classified, also with partial class assignments, in a similar way as in Autoclass, and thus attribute probabilities can be assigned. We are, however, not discussing such methods here as they are not directly related to BCPNN. We find in Paper VIII that BCPNN can find patterns in huge data sets tremendously much quicker and often use much less memory than Autoclass, we also get indications that the patterns obtained by BCPNN have higher clinical relevance in our application on the WHO database.

5 Some clarification comments

The publications in this thesis has been produced during quite a long period of time. Over this time the author's understanding about the problems has developed significantly, therefore it may be adequate to make a few clarifications.

The term "Bayesian" was in the early publications only related to the usage of Bayes rule (eq 1). Bayes rule has in this sense nothing to do with "Bayesian statistics" a term we have avoided in the thesis. Bayes rule, which is essentially a "truism", is used a lot in probability theory and is as such only a way to reason with probabilities. For this reason we prefer to see probability theory as an extension to deductive logic [Jaynes, 1990].

An important insight was the difference in view between so called *Bayesian statistics* and *classical statistics*. In Bayesian statistics all unknown parameters may have a distribution. In classical statistics, parameters as probabilities are considered to be *constants* which may be unknown but are usually not seen as a having a density function [O'Hagan, 1999]. If a

reasoning involves the term *Bayesian* it is usually referred to by statisticians as any inference involving a *prior distribution* giving a *posterior distribution* when we have taken actual data into account.

The term *confidence interval* has been used, especially in Paper I and Paper II , as a general term expressing uncertainty in a distribution. The term *confidence interval*, however, refers to the distribution of data from a frequentist's perspective. When the uncertainty of a parameter is referenced the correct term to use is *credible interval* or *credibility interval* [Lee, 1997].

The notation used is not consistent over the publications. In, for instance, Paper I we have used $P(p)$ to refer to the density function for the parameter p. A capital "$P(X)$" is, however, in usual statistical language used to express a *probability* and "$p(X)$" a *probability function*.

2 On finding dependencies in data

The concept of *data mining*, one of these rather new *buzz words*, is more or less about finding dependencies in data. Jerry Friedman, a statistician and data mining expert was asked after a data mining talk "So, *is* data mining different from statistics?" Jerry paused and asked if the questioner wanted the long or the short answer. Since the short answer was requested, Jerry's answer was, "No" [Veaux, 2001]. The long version of the answer would probably have been a little different. If we look up the term in a statistical dictionary; **data mining:** "A term generally used in a pejorative sense for the process of considering a large number of models including many which are 'data-driven' in order to obtain a good fit. See also **data dredging**. And about **data dredging:** "A term used to describe comparisons made within a data set not specifically prescribed prior to the start of the study." [Everitt, 1998].

Clearly, the term "data dredging" is not a very positive one. It is said that if you search long enough you will always find correlations. Of course, what we need to consider when we have found a correlation is the reason for this specific correlation and not necessary take it as a casual relationship. A correlation may appear just by random, or it may appear because both variables are influenced by some third known or unknown confounding factor.

Data mining in one sense can be seen as the whole process of turning data into information and knowledge, thus involving searching for patterns, investigating possible representational forms, performing classifications, clustering, regression, function approximation, building inductive trees, finding logical rules describing the dependencies within the data set etc. [Fayyad *et al.*, 1996]. We do, however, live in a world where probability theory is necessary to understand all possible phenomenon in nature. Quantum physics, chemistry, economy, medicine as well as information processing in the brain. In a way there are no hard rules for anything, every possible process can be modelled from a probabilistic perspective.

Marquis de Laplace said "Strictly speaking it may even be said that nearly all our knowledge is problematical; and in the small number of things which we are able to know with certainty, even in the mathematical sciences themselves, the principal means for ascertaining truth - induction and analogy - are based on probabilities" [Laplace, 1814]. He indicated that the entire system of human knowledge is connected with probability theory. Therefore, we have here restricted the meaning of *data mining* merely to be the search for conditional dependencies and patterns within the data set.

There are several methods available in statistics to measure dependencies as well as there are many different methods to find patterns. After we have applied some method and have found dependencies or associations, alternatively more complex patterns in our data sets, then we can ask, *why* is there a dependency between these two variables or specifically between these two variable states, or alternatively when we have found a pattern, what is the possible meaning or interpretation of this pattern? A reasonable first thing to ask for an association found, is this really an association or is this a *coincidence by random* explainable by the sampling process? However, first we have to define what we mean by a *dependency*, *association* or *correlation* when speaking about, usually, discrete events. In Paper I [same as [Orre *et al.*, 2000]] we present and discuss the fundamentals of the dependency relations we are using.

1 Data mining with Information Components

Although there are many ways to measure dependencies we will in this text use the term *dependency* merely as a synonym for association or correlation, even if all of them have a rather

precise meanings in mathematical and statistical literature. About *linear dependencies*, for functional transformations we can consider the functions $f_1(x), \cdots, f_i(x), \cdots, f_n(x)$ to be linearly dependent if for some $c_1, \cdots, c_i, \cdots, c_n$ not all zero, we have $\sum_i c_i \cdot f_i = 0$. For stochastic variables, *e.g.* X and Y we may use the *covariance* which is the *expected value* [3] of the product of their deviations from their respective means μ_X and μ_Y, *i.e.* $cov(X,Y) = E(X - \mu_X)E(Y - \mu_Y) = E(XY) - E(X)E(Y)$. The covariance for two independent variables is then zero. We often hear about the *correlation coefficient* which is the covariance *normalised* by dividing with the product of the *standard deviations* and is often denoted r. The correlation coefficient between X and Y is then

$$r(X,Y) = \frac{cov(X,Y)}{\sigma(X)\,\sigma(Y)}$$

where $\sigma(X) = \sqrt{V(X)}$ *i.e.* $\sigma(X) = \sqrt{E[(X - \mu_X)^2]}$. The correlation coefficient is practical because it is a value in the range $[-1, 1]$ where *e.g.* $r(X,Y) = 1 \Rightarrow y = k \cdot x$, *i.e.* y is completely dependent on x. The higher value of r the higher the *predictability* of one variable from the other. For prediction or classification analysis we may denote X an *explanatory* or *predictor* variable and Y a *response* or *dependent* variable.

For discrete variables, which are often *binomial* or *multinomial* distributed as are the variables we mostly deal with here, the correlation coefficient is, however, not very practical. Instead we prefer discussing about *conditional dependency*, defined as [Pearl, 1988]:

DEFINITION: *Let $U = \{\alpha, \beta, \ldots\}$ be a finite set of variables with discrete values. Let $P(.)$ be a joint probability function over the variables in U. X and Y are said to be* **conditionally independent given** *Z if*

$$P(x \,|\, y, z) = P(x \,|\, z) \text{ whenever } P(y, z) > 0. \tag{9}$$

As we mentioned earlier, the *background* , here Z, can be omitted from the expression and construed implicit as long as it is not forgotten. A natural way to define something similar to the correlation coefficient would then be to say that when X and Y are *independent* then

$$\frac{P(x \,|\, y)}{P(x)} = 1 \text{ whenever } P(y) > 0. \tag{10}$$

When $P(x \,|\, y)$ is larger than $P(x)$ we have a positive correlation and when X and Y are *anti correlated* the expression (10) comes closer to zero. To make this expression to more resemble the correlation coefficient, which is symmetrical around zero for independence, we can take its logarithm and at the same time rewrite it using the axiom about conditioned probability (0), this measure is what we in this text denote *information component* (IC)

$$IC_{xy} = \log \frac{P(x \,|\, y)}{P(x)} = \log \frac{P(x,y)}{P(x)P(y)} = 0 \text{ when } X \text{ and } Y \text{ are independent.} \tag{11}$$

The name "information component" we chose because it is related to *mutual information* or *cross entropy*. Originally from information theory where Shannon [Shannon, 1948] defines information as how much "choice" is given from a set of events with probabilities $p_1, p_2, \ldots, p_n$ as the (Shannon) *entropy* (here the discrete version) function

$$H(p_1, p_2, \ldots, p_n) = -K \sum_{i=1}^{n} p_i \log p_i \tag{12}$$

[3] expected value for X: $E(X) = \int_{-\infty}^{\infty} x f_X(x) dx$ and for XY: $E(XY) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} xy f_{XY}(x,y) dx\, dy$

The positive constant K is often set to $1/\log(2)$ to measure the information in *bits* to relate it to *representation* of information in computers. The term *bits* (binary digits) was suggested by John W. Tukey 1946 [T. Soc. f. Math. Biol., 2000]. Information theory often deals with information being sent over some communication channel, but as a simple example we can look upon the information storable in a binary variable with two states with equal probability $p_i = 0.5$

$$H_2(0.5, 0.5) = \frac{-1}{\log 2}(0.5 \cdot \log 0.5 + 0.5 \cdot \log 0.5) = 1. \tag{13}$$

That is, the information content of a *noise free* binary variable is one bit. If we now have some system which *predicts* variable Y from variable X there is some information shared between these two variables, this is the *mutual information* or *cross entropy* which is defined as [Cover and Thomas, 1991]

$$\begin{aligned} I(X;Y) &= H(Y) - H(Y \mid X) && (14)\\ &= H(X) - H(X \mid Y) && (15)\\ &= H(X) + H(Y) - H(X,Y) && (16) \end{aligned}$$

That is, when the *conditional entropy* on Y given X $[H(Y \mid X)]$, is the same as $H(Y)$ then the mutual information is zero. It should also be clear that $I(X;X) = H(X)$, *i.e.* information is a special case of mutual information. Shannon's original syntax for the conditional entropy $H(Y \mid X)$ was $H_y(X)$. The conditional entropy of Y given that we know X is defined as [Shannon, 1948] (but not using Shannon's syntax):

$$\begin{aligned} H(Y \mid X) &= -\sum_x P(x) \cdot H(Y \mid X = x) && (17)\\ &= -\sum_x P(x) \sum_y P(y|x) \log P(y|x) && (18)\\ &= -\sum_x P(x) \sum_y \frac{P(y,x)}{P(x)} \log \frac{P(y,x)}{P(x)} && (19)\\ &= -\sum_x \sum_y P(x,y) \log \frac{P(x,y)}{P(x)} && (20)\\ &= \sum_x \sum_y P(x,y) \log P(x) - \sum_x \sum_y P(x,y) \log P(x,y) && (21) \end{aligned}$$

Then, by writing $H(Y)$ as

$$H(Y) = -\sum_y P(y) \log P(y) = -\sum_x \sum_y P(x,y) \log P(y) \tag{22}$$

we end up with the familiar

$$I(X;Y) = H(Y) - H(Y|X) = \sum_x \sum_y P(x,y) \log \frac{P(x,y)}{P(x)P(y)} \tag{23}$$

expression for the mutual information between X and Y. The mutual information can also be seen as a special case of *Kullback-Leibler distance* [Neelakanta, 1999] between pdf p with

respect to the pdf q, defined as:

$$K(p||q) = \sum_x p(x) \log \frac{p(x)}{q(x)} \tag{24}$$

where $p(x) = P(x, y)$ and $q(x) = P(x) \cdot P(y)$. According to (23) the mutual information can thus be seen as a weighted, by $P(x, y)$, sum of the information shared between each combination of states of the variables X and Y, *i.e.* (here using $\log 2$ as we most often do to get the IC measure in bits)

$$I(X;Y) = \sum_x \sum_y P(x,y) \log_2 \frac{P(x,y)}{P(x)P(y)} = \sum_x \sum_y P(x,y)\, IC_{xy} \tag{25}$$

This also gives an indication why the mutual information as such may not be a good measure for data mining. Those variable states which may be considered interesting from a data mining perspective may not add much to the general predictability of the response variable as the joint state may be rare.

Let us take a few examples of $p(x), p(y)$ and $p(x, y)$ values between two variables X, Y and how they affect the IC_{xy}, $I(X;Y)$ and the Bayes factor for the outcome $Y = 1$ given the assumptions $X = 0$ vs $X = 1$. Bayes factor is the ratio between the posterior and prior odds [O'Hagan, 1999, Lee, 1997, Bernardo and Smith, 1994], *i.e.* in this case

$$\frac{P(X=0|Y=1)}{P(X=1|Y=1)} \Big/ \frac{P(X=0)}{P(X=1)} = \frac{P(Y=1|X=0)}{P(Y=1|X=1)} \tag{26}$$

We present the joint and the marginal probabilities as tables:

	p(Y=0)	p(Y=1)
p(X=0)	p(X=0,Y=0)	p(X=0,Y=1)
p(X=1)	p(X=1,Y=0)	p(X=1,Y=1)

In the first example below, X and Y are independent of each other. Their joint probabilities are the same ($= 1/4$) for all state combinations. Their joint distribution P_{XY} is the same as the product $P_X P_Y$ of their marginal distributions. The mutual information is, as we could expect, zero. For two independent variables no information is conveyed from one variable to another. and the Bayes factor for $X = 0$ vs. $X = 1$ when $Y = 1$ is 1. This example would correspond to a *Bernoulli trial* [Jeffreys, 1939] between two binary random variables with equal probability $1/2$ for both states, like two perfect coins thrown an infinite number of times. With this we may indicate that, even though we have two "perfectly" independent random sources, it is still very unlikely to get the exact marginal probability as here $1/2$ and thus for the joint probability $1/4$ in a finite number of trials. The Bernoulli series does also not converge in the usual meaning of convergence, only with probability 1.

	1/2	1/2
1/2	1/4	1/4
1/2	1/4	1/4

$p_x p_y$

1/4	1/4
1/4	1/4

$p_{xy}/p_x p_y$

1	1
1	1

IC_{xy}

0	0
0	0

$p_{xy} IC_{xy}$

0	0
0	0

$$\mathbf{I(X;Y) = 0}$$

$$\frac{p(Y=1|X=0)}{p(Y=1|X=1)} = 1$$

The next example below is merely the opposite of the previous with independent variables. The two variables X and Y are almost as dependent as they can be. The probability to be in one of the states for one variable correlates to almost 100 % to one of the states of the other variable. Their joint probabilities P_{XY} are much higher than the product distribution $P_X P_Y$ for the correlated states and much lower than the product for the anti correlated states. The mutual information is here close to 1, that is almost *one bit* of information is conveyed between the variables and Bayes factor for $X = 0$ vs. $X = 1$ when $Y = 1$ is quite high (≈ 498). This example is quite relevant for Shannon's original intentions and for what information theory has been mostly used for, that is to model the transferred information on some type of communication channel. Due to noise and other imperfections of the channel a transmitted state may not be the same as the received state. These imperfection may, as in the example, also cause slight asymmetries in the transfer probabilities.

	0.499	0.501
0.499	0.498	0.001
0.501	0.001	0.500

$p_x p_y$

0.2490	0.2500
0.2500	0.2510

$p_{xy}/p_x p_y$

2.000	0.004
0.004	1.992

IC_{xy}

1.0000	-7.9658
-7.9658	0.9942

$p_{xy} IC_{xy}$

0.4980	-0.0080
-0.0080	0.4971

$$\mathbf{I(X;Y)} \approx \mathbf{0.9792}$$

$$\frac{p(Y=1|X=0)}{p(Y=1|X=1)} \approx 498.0$$

The following example below is a more relevant one for a typical data mining application as described in Paper I and Paper II . The interesting states, here $X = 1$, *e.g.* corresponding to a specific drug appearing in an adverse drug reaction report, and $Y = 1$, *e.g.* corresponding to a specific adverse reaction, may have a low value on both the marginal probability, which is here 0.0050 and the joint probability, which is here 0.0025.

For prediction of $P(Y|\mathbf{x})$ we could be correct in as much as 99.5% of all cases, just based upon the *prior* probability $p(Y = 1) = 0.995$, even though this prediction would be quite useless for most purposes. The mutual information is low, $I(X;Y) \approx 0.0152$, *i.e.* very little information is transferred between the variables but the ratio $[p_{11}/(p_{1\cdot}p_{\cdot 1}) = 100]$ [4] causing IC for the interesting combination to be as high as $IC_{11} \approx 6.6439$ here.

	0.9950	0.0050
0.9950	0.9925	0.0025
0.0050	0.0025	0.0025

$p_x p_y$

0.9900	0.0050
0.0050	0.0000

$p_{xy}/p_x p_y$

1.0025	0.5025
0.5025	100.0000

IC_{xy}

0.0036	-0.9928
-0.9928	6.6439

$p_{xy} IC_{xy}$

0.0036	-0.0025
-0.0025	0.0166

$$\mathbf{I(X;Y)} \approx \mathbf{0.0152}$$

$$\frac{p(Y=1|X=0)}{p(Y=1|X=1)} \approx 199.0$$

Further on, the predictability for the different state combinations apart $p(Y = 0|X = 0)$ are not so high:

$p(Y|X)$

$p(Y{=}0\|X{=}0) \approx 0.9975$	$p(Y{=}1\|X{=}0) \approx 0.0025$
$p(Y{=}0\|X{=}1) = 0.5$	$p(Y{=}1\|X{=}1) = 0.5$

[4] here we use $p_{1\cdot}$ to say $p(X = 1)$ and $p_{\cdot 1}$ to say $p(Y = 1)$

However, knowing that on the reports where DRUG=X the serious adverse drug reaction Y appears on as many as 50% of the reports may be quite essential information. The Bayes factor between $p(Y=1|X=0)$ and $p(Y=1|X=1)$ which is here ≈ 199 clearly indicates a high dependency on X for Y. Bayes factor is used a lot in model selection [Bernardo and Smith, 1994] but for the purpose of data mining as well as classification with BCPNN we are in this thesis using estimates of either the IC value or the simple $[p_{xy}/(p_x p_y)]$ expression, which follows directly from the definition of dependency. For data mining or signalling purposes there are several other measures used, like The Reporting Odds Ratio, Proportional Reporting Ratio, Yule's Q, the Poisson probability and Chi-square test. These are all compared to the IC measure for 17330 different Drug-ADR combinations in [van Puijenbroek *et al.*, 2002]. There it was found that concordance between the different measures was quite similar when there were many reports available for each combination.

We should, however, note that as IC is a function of probability distributions, so is also IC a random variable for which a credibility interval [Lee, 1997] can be estimated from data. To make the IC measure even more accurate and sensitive at the same time for very few combinations is one important goal for data mining purposes but also for classification as we present in Paper V . In Paper I we define the basic calculations on how to calculate IC with good performance and reasonable accuracy for both a point value estimate of its expectation value as well as its variance. In Paper III we present one way to make the expectation value and also the variance more accurate. Alternative methods to generate accurate credibility intervals as well as expectation values are also discussed in [Norén, 2002].

We will take a closer look at how to calculate the expectation value and credibility interval for IC, but as IC is a function of probability distributions we will first continue in some detail with the fundamental subject on how to measure a probability,

2 Finding the probability from a finite series of Bernoulli trials

There are two types of events we want to find the probability of, one is the *discrete event* where the most simple one is binary as true/false, 1/0 or head/tail on a coin, the other type of event is expressed as a *belief value* being part of a *mixture*. In this section we will only deal with the discrete type of event, although belief values in mixtures will be discussed later on. To make things as simple as possible we start with a binary event such as flipping a coin to achieve head or tail as outcome.

If we have a *perfect coin* and throw this perfectly the probability for the coin to get either head or tail is 1/2. Even if the coin is not perfect, we would at least expect a fixed ratio for the probability. The probabilities for the two outcomes can be expressed as $P(H)$ and $P(T)$. There are only two possible *mutually exclusive* outcomes, so they sum to one, *i.e.* $[P(H)+P(T)=1]$. If we now toss the coin twice there are four possible outcomes HH,HT,TT and TH, whose probabilities, assuming that the outcome of the two tosses are independent of each other, are:

$$(P(H)+P(T))^2 = P(H)^2 + 2P(H)P(T) + P(T)^2.$$

The probabilities to get two head, one head, zero heads are then $P(HH) = P(H)^2$, $P(HT) = 2P(H)P(T)$ and $P(TT) = P(T)^2$ respectively, which is extendable to n tosses as

$$(P(H)+P(T))^n = \sum_{m=0}^{n} \frac{n!}{m!(n-m)!} P(H)^m P(T)^{n-m} = \sum_{m=0}^{m} \binom{n}{m} P(H)^m P(T)^{n-m} \quad (27)$$

which is known as the *binomial distribution* [Stuart and Ord, 1994]. The probability to get exactly m heads in n tosses is then

$$P(H = m|n) = \binom{n}{m} P(H)^m P(T)^{n-m} \tag{28}$$

For instance, the probability to get, *e.g.* 5 heads when tossing a perfect coin, say 10 times $P(H = 5|n = 10) \approx 0.25$ and to get 500 heads when tossing the same coin 1000 times $P(H = 500|n = 1000) \approx 0.025$.

In the binomial distribution we have fixed probabilities and the counters being the parameters. To be able to find the probability distribution of the belief values we need a distribution where these are the parameters. From the so called *principle of inverse probability* [Jeffreys, 1939] [First discussed by Jacob Bernoulli in *Ars Conjectandi* 1713, defined by Thomas Bayes 1763 and reinvented and developed by Marquis Laplace] which we have already given examples of (eq 3) as it followed from the axiom about conditioned probability (eq 0) and Bayes rule (eq 1) it implies that

$$\text{posterior probability} \propto \text{prior probability} \cdot \text{likelihood} \tag{29}$$

and we are interested in finding expressions for the prior and posterior distributions as functions of the probabilities. These functions can in principle have any shape, but to be mathematically tractable we here limit ourselves to a restricted type of prior called a *conjugate prior*. A conjugate prior of a likelihood function has the property that both the prior and posterior distribution belong to the same class of functions, *i.e.* they both have the same form [Stuart and Ord, 1994].

Let us look upon the probability to obtain m heads from a series of tosses (*Bernoulli trials*) with a coin thrown n times with the unknown probability p_h to get a head. This is the same as (eq 28) but now instead write this probability as a constant multiplied with the *likelihood function*, *i.e.* $P(H = m|p) = k * f(m|p) = k\, p_h^m (1-p)^{n-m}$. The likelihood function $f(m|p)$ is sometimes written with the condition reversed as $L(p|m)$ [Edwards, 1992]. The posterior probability function $f(p|m)$ can thus be written as

$$f(p|m) = k \cdot f(p) \cdot f(m|p) \tag{30}$$

where $f(p)$ is the prior function we are looking for. To find this prior is not a trivial task and it can in general only be done for the exponential family of distributions [Lindsey, 1996, Bernardo and Smith, 1994] but we will sketch the principle here. If $x = (x_1, \ldots, x_n)$ is a random sample from a regular exponential family distribution such that

$$p(\boldsymbol{x} \,|\, \boldsymbol{\theta}) = \prod_{j=1}^{n} f(x_j)[g(\boldsymbol{\theta})]^n \exp\left[\sum_{i=1}^{k} c_i \phi_i(\boldsymbol{\theta}) \left(\sum_{j=1}^{n} h_i(x_j)\right)\right] \tag{31}$$

then the conjugate family for $\boldsymbol{\theta}$ has the form

$$p(\boldsymbol{\theta} \,|\, \boldsymbol{\tau}) = [\kappa(\boldsymbol{\tau})]^{-1}[g(\boldsymbol{\theta})]^{\tau_0} \exp\left[\sum_{i=1}^{k} c_i \phi_i(\boldsymbol{\theta}) \tau_i\right], \boldsymbol{\theta} \in \Theta, \tag{32}$$

where Θ is the *domain*, *i.e.* set of parameters ($\boldsymbol{\theta}$) for which normalising constant is strictly positive [Lindsey, 1996], and $\boldsymbol{\tau}$ such that the normalising constant

$$\kappa(\boldsymbol{\tau}) = \int_{\Theta} [g(\boldsymbol{\theta})]^{\tau_0} \exp\left[\sum_{i=1}^{k} c_i \phi_i(\boldsymbol{\theta}) \tau_i\right] d\boldsymbol{\theta} < \infty. \tag{33}$$

Proof of (eq 32) from [Bernardo and Smith, 1994] [following from the Neyman factorisation, proposition 4.10]. The sufficient statistics for ϕ have the form

$$\boldsymbol{t}_n(x_1,\ldots,x_n) = \left[n, \sum_{j=1}^{n} h_1(x_j), \ldots, \sum_{j=1}^{n} h_k(x_j)\right] = [n, \boldsymbol{s}(x)] \tag{34}$$

so that, for any $\boldsymbol{\tau} = (\tau_0, \tau_1, \ldots, \tau_n)$ such that $\int_\Theta p(\boldsymbol{\theta} \,|\, \boldsymbol{\tau}) d\boldsymbol{\theta} < \infty$, a conjugate prior density has the form:

$$p(\boldsymbol{\theta} \,|\, \boldsymbol{\tau}) \propto p(s_1(\boldsymbol{x}) = \tau_1, \ldots, s_k(\boldsymbol{x}) = \tau_k \,|\, \boldsymbol{\theta}, n = \tau_0) \tag{35}$$

$$\propto [g(\boldsymbol{\theta})]^{\tau_0} \exp\left[\sum_{i=1}^{k} c_i \phi_i(\boldsymbol{\theta}) \tau_i\right] \tag{36}$$

In the applications we describe here we are so far only interested in Bernoulli trials, so let us go back to the original problem in (eq 30). We write the Bernoulli likelihood $f(m|p)$ on its explicit exponential form, and set $\tau_0 = n$ and $\tau_1 = m$:

$$f(x_1,\ldots,x_n \,|\, p) = \prod_{i=1}^{\tau_0} p^{x_i}(1-p)^{1-x_i} \tag{37}$$

$$= (1-p)^{\tau_0} \exp\left[\log\left(\frac{p}{1-p}\right) \sum_{i=1}^{\tau_0} x_i\right] \tag{38}$$

Watch out, because the product in (eq 37) may be a little confusing. After the product symbol there is actually a selection of which of the probabilities to multiply with dependent on the value of x_i. According to (eq 32) the conjugate prior $f(p)$ is then given by

$$f(p \,|\, \tau_o, \tau_1) \propto (1-p)^{\tau_0} \exp\left[\log\left(\frac{p}{1-p}\right) \tau_1\right] \tag{39}$$

$$= \frac{1}{\kappa(\tau_0, \tau_1)} p^{\tau_1}(1-p)^{\tau_0 - \tau_1} \tag{40}$$

where normalising constant $\kappa(\tau_0, \tau_1)$ is

$$\kappa(\tau_0, \tau_1) = \int_0^1 p^{\tau_1}(1-p)^{\tau_0-\tau_1} dp = \frac{\Gamma(\tau_1+1)\Gamma(\tau_0-\tau_1+1)}{\Gamma(\tau_0-\tau_1+2)}. \tag{41}$$

When writing $\alpha = \tau_1 + 1$ and $\beta = \tau_0 - \tau_1 + 1$ we recognise the Beta density function $Be(\alpha, \beta)$, which is the conjugate prior for the binomial distribution or as we see it here a series of Bernoulli trials, *i.e.* the prior $f(p)$ is

$$f(p) = Be(p \,|\, \alpha, \beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} p^{\alpha-1}(1-p)^{\beta-1} \tag{42}$$

Now the steps to come from (eq 12) to (eq 14) in Paper I should be somewhat more clear. What we now have is a prior density function for the probability p defined in the hyperparameters α and β. In Paper I the posterior probability for parameter p_1, which is the estimated probability for true counts, is defined as

$$f(p_1 \,|\, c_1, c_0) = \frac{\Gamma(C+\alpha_1+\alpha_0)}{\Gamma(c_1+\alpha_1)\Gamma(c_0+\alpha_0)} p_1^{c_1+\alpha_1-1}(1-p_1)^{c_0+\alpha_0-1} \tag{43}$$

where c_1 is the number of true counts, c_0 the number of false counts and $C = c_1 + c_0$. The parameters α_1 and α_0 can here be seen as *pseudocounts* expressing our *a priori* belief or *prior* information about the data set. **Observe**, in Paper I we have used the notation $P(p_1 \mid c_1, c_0)$ to express the density function for the parameter p_1 but here we conform to the standard within statistics to use a lower case letter to designate a continuous density function.

3 How to assert the prior

One main difference between *classical inference* and *Bayesian inference* is that in the latter we start by asserting a prior expressing our prior information. This prior can be asserted either by a qualified guess or we may use data in some way to assert it as is done in *empirical Bayes* analysis. When we make a guess the prior is *subjective* because it is based on prior beliefs, alternatively we can assert a prior which contains as little information as possible, to let our belief affect the posterior result as little as possible.

If we start with the empirical Bayes method, assuming we have made an assumption about a prior distribution for the parameters θ, but only to a certain class of prior distributions as in (eq 42). The prior is a distribution $f(\theta \mid \phi)$ where ϕ denotes the *hyperparameters* describing the family of priors. A likelihood can then be constructed as

$$f(x \mid \phi) = \int f(x \mid \theta) f(\theta \mid \phi) d\theta \tag{44}$$

which relates the data to the hyperparameters. In the basic empirical Bayes approach ϕ is estimated from (eq 44) by classical methods like an unbiased estimator giving $\hat{\phi}$. The prior used then is $f(\theta \mid \hat{\phi})$. The use of apparently Bayesian method to estimate θ hides the non-Bayesian treatment of the fundamental parameters ϕ and empirical Bayes methods are therefore not really Bayesian as they do not allow a distribution for ϕ [O'Hagan, 1999].

Let's now take a look at (eq 42) again. The *a priori* information is defined by the pseudocounts α_1 and α_0. Principally you can set these to anything that is consistent with your prior belief about the parameters, but the most common approach is that you don't have any prior information and you want this to be reflected by giving a *non informative* or *ignorant* prior. There are several approaches to non informative priors, both Bayes and Laplace suggested to use a uniform prior [Lee, 1997] which in this case corresponds to $\alpha_1 = 1$ and $\alpha_0 = 1$, *i.e.* $f(p) = Be(p \mid 1, 1)$. Another approach is *Jeffreys prior* which affects the *information* provided by the prior as little as possible and we have *Haldane's prior* which affects the posterior distribution as little as possible. Haldane suggested in 1931 [Lee, 1997] that $\alpha_0 = 0$ and $\alpha_1 = 0$, which has the density

$$f(p) \propto p^{-1}(1-p)^{-1} \tag{45}$$

which, however, is an improper density function as the integral

$$\int_0^1 p^{-1}(1-p)^{-1} dp \tag{46}$$

does not exist. For an improper prior, there are usually data such that the posterior is also improper. It has been argued that improper priors are acceptable as long as one checks that the posterior is proper, but this is still a question debated among Bayesian statisticians.

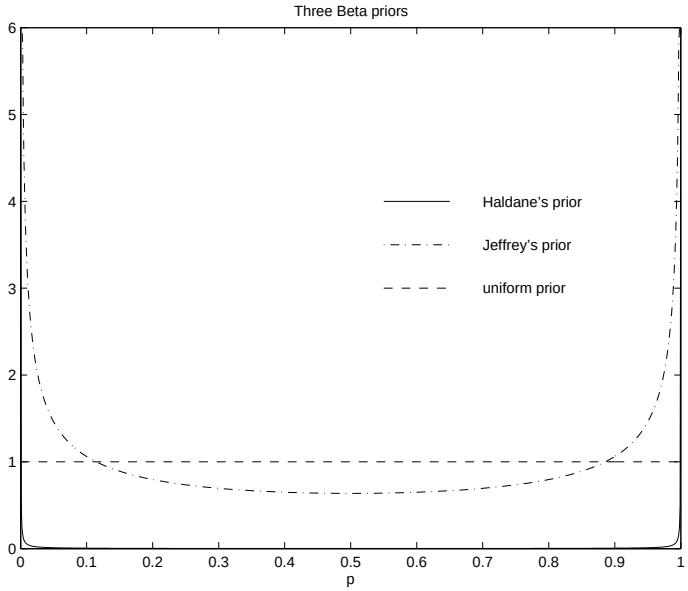


Figure 1: This figure shows three different priors for the Beta density, Haldane's, Jeffrey's ($\alpha_0 = \alpha_1 = 1/2$) and the uniform prior. Observe that the Haldane's prior shown is not exactly the Haldane's prior because that would coincide with the axes. Here we have set α_1 and α_0 to 0.001.

One possible advantage with this prior is that it is *unbiased* in the sense that the mean of the posterior distribution, which for p_1 is

$$E(p_1|c_1, c_0) = \frac{\alpha_1 + c_1}{\alpha_1 + \alpha_0 + c_1 + c_0} \tag{47}$$

coincides with the maximum likelihood estimate (max mode) $c_1/(c_1 + c_0)$. For practical purposes it is, however, only usable in the calculation if you have some data and don't want to perform some special handling in the case of no data. From this perspective it is therefore not necessarily well suited for automatic reasoning and, especially data mining with computers where we want to be able to make calculations also when no specific data is available. Even though we may sound negative against using this prior we have, however in this text, used the maximum likelihood estimate for calculating probabilities in the following publications presented here; Paper IV , Paper VI , Paper VII and Paper VIII . In none of these publications, we have, however used the Bayesian approach to estimate a posterior distribution based upon a prior distribution, in these publications we only calculate a point estimate of the probabilities.

Now, let us go back to the question about non informative priors which try to minimise the prior's impact on the information. First the question about what is information is not unambiguous because the term *information* in statistics and information theory does not refer to the same measure. Earlier (eq 12) we defined Shannon's information (entropy) in the information theoretical meaning but there is also a measure in statistics denoted information or alternatively *Fisher's information*. Fisher defined in 1925 [Lee, 1997] the

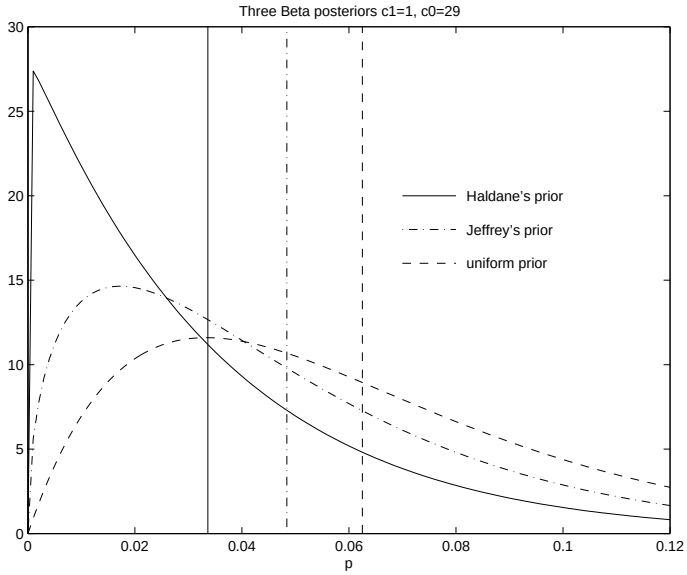


Figure 2: This figure shows how posteriors look for the Beta density for three different priors, Haldane's, Jeffrey's and the uniform prior, when we have obtained 1 count for c_1 and 29 counts for c_0. The vertical lines shows the different expectation values for these three posterior densities. The maximum likelihood value of the density with uniform prior coincides with the expectation value of the Haldane prior.

information provided by an experiment as

$$I(\theta \,|\, x) = -E\left[\frac{\partial^2(\log p(x|\theta)}{\partial\theta^2}\right] \tag{48}$$

We will not get into details here how to arrive to this as the derivation going from the Hessian matrix $H(\theta)$, measuring the local curvature of the log-likelihood function at its maximum $\hat{\theta}$, often named *observed information matrix* to the *Fisher (or expected) information matrix*, can be looked up in many textbooks *e.g.* [Bernardo and Smith, 1994, Lindsey, 1996]. Instead we will just mention the conclusion when the Fisher information is minimised for the Beta distribution giving that $f(p \,|\, \alpha_0, \alpha_1) = Be(1/2, 1/2)$ [Lee, 1997], *i.e.* $\alpha_0 = 1/2$ and $\alpha_1 = 1/2$. This is sometimes called the *arc-sine distribution* [Lee, 1997] which *Jeffrey's rule* suggests as a guideline, not to be followed blindly, to be the correct reference prior. It can be shown [Lee, 1997] that this is a prior uniform for

$$z = \arcsin\sqrt{\pi} \tag{49}$$

when $x \sim B(n, \pi)$ and z is defined as $z = \arcsin\sqrt{x/n}$. To illustrate how Jeffrey's prior looks compared to Haldane's and the uniform *flat* prior we have plotted them in figure 1. The plot of Haldane's prior is not exactly a Haldane prior because that would coincide with the axes, here we set the pseudocounts $\alpha_1 = \alpha_0 = 0.001$.

Although the theoretical support for using the uniform prior may not be so high, even though it may be intuitive in that way that you don't put preference on any specific probability and as such was also suggested by both Bayes and Laplace, the uniform prior is,

probably, the most commonly used in applications with Bayesian inference today. After we have counted a high number of events the priors will differ very little of course, but for applications like data mining and inference it is important to be able to find coincidents and make conclusions also with very little data available as accurate as possible. To illustrate how the posterior density $f(p_1|c_1, c_0)$ looks when using the three different priors we have drawn these in figure 2 for a case when we have collected one count for the true event and 29 counts for the false event, *i.e.* $c_1 = 1$ and $c_0 = 29$. One possible advantage with the uniform prior, from a theoretical point of view is that the maximum likelihood estimate, as we mentioned in section 2 in Paper I , corresponding to the maximum of the density function (solving $d/dp_1 f(p_1|c_1, c_0) = 0$ giving $\hat{p}_1 = \frac{c_1}{c_1+c_0}$) coincides with the expectation value of the *bias free* Haldane prior. This is also indicated by the diagram in figure 2.

The priors we have actually used for probabilities of single events p_i in Paper I and Paper II are, however, the uniform prior suggested by Bayes and Laplace.

In this section we discussed about finding the probability and a prior density function for a single binomial event. The events we are dealing with do not need to be simple *Bernoulli* events, though, each event could have several outcomes, like when you throw a dice you have six different possible outcomes. This would give a *multinomial* likelihood function, and in a similar way as we ended up with the Beta density as a conjugate prior for the binomial likelihood we would find that the *Dirichlet* density is the conjugate prior for the multinomial likelihood. The Dirichlet distribution is thus a very powerful tool for analysing experiments with any discrete probability distribution.

4 Asserting a joint prior

Now, let's look upon the joint outcome of two binomial events as two discrete variables with two states. This can be as simple as throwing two coins which will have four possible outcomes HH,HT,TT,TH or it can be as in the data mining application we are discussing in Paper I and Paper II where one variable X represents a certain drug occurring on a report and Y represents a certain adverse drug reaction appearing on the same report. In these cases the joint distribution will be a *multinomial* distribution and the conjugate prior a *Dirichlet* distribution which is a probability density function over a discrete probability distribution. It has thus four different real valued states but only three dimensions as the fourth state is bounded by the others as the probability for the four states should sum to one. The expectation value of each p_{ij} for a three dimensional Dirichlet distribution becomes (as is shown in section 2.1 in Paper I for p_{11})

$$E(p_{ij}) = \frac{c_{ij} + \gamma_{ij}}{c_{11} + \gamma_{11} + c_{10} + \gamma_{10} + c_{01} + \gamma_{01} + c_{00} + \gamma_{00}} = \frac{c_{ij} + \gamma_{ij}}{C + \gamma} \tag{50}$$

If we would use Jeffrey's prior here, then we would set $\gamma_{ij} = \frac{1}{2}$ and $\gamma = 2$ because the Jeffrey's prior for a multinomial distribution generalises in this way [Fan and Tsai, 1999], as was earlier pointed out for the binomial distribution. On the other hand, if we would use the Laplace and Bayes suggestion to assert a uniform prior here then we would set $\gamma_{ij} = 1$ and $\gamma = 4$. There are, however, disadvantages with this approach, especially for data mining which has as outcome a warning signal as is discussed in Paper I section 4 and in Paper II where we use the IC_{ij}-value and its credibility interval as a measure of the dependency and its significance between the states of two different variables. The reason is that the IC may fluctuate a lot when we have small amounts of data. When we have no data samples at all we consider the variables to be independent, *i.e.* we want the prior to

fulfil

$$\lim_{c_i,c_j,c_{ij}\to 0} IC_{ij} = \log \frac{\hat{p}_{ij}}{\hat{p}_i\hat{p}_j} \approx 0 \tag{51}$$

and also

$$\lim_{c_{ij},C\to 0} IC_{ij} = \log \frac{\hat{p}_{ij}}{\hat{p}_i\hat{p}_j} = \log \frac{\frac{c_{ij}+\gamma_{ij}}{C+\gamma}}{\hat{p}_i\hat{p}_j} \approx 0. \tag{52}$$

By setting γ_{ij} to a constant *e.g.* $\gamma_{ij} = 1$ and then $\gamma = \frac{\gamma_{ij}}{\hat{p}_i\hat{p}_j}$ both (eq 51) and (eq 52) are fulfilled. As is mentioned in Paper I section 2.4 this would give a prior which is not coherent *i.e.* $\sum_{ij} p_{ij} = 1$ which is true, although empirical studies have shown that the sum $\sum_{ij} p_{ij} \approx 1$ even though this may not be good enough. Further on we have suggested in Paper I section 2.4 that "To get a coherent prior for p_{ij} we would then set $\gamma_{ij} = 1$, $\gamma = \alpha\beta$." This would correspond to the Laplace suggestion about a uniform prior. Observe that α here represents the number of states for variable X, which in the example given above means $\alpha = 2$, and β represents the number of states for variable Y which here gives $\beta = 2$ as well. To fulfil both (eq 51), (eq 52) and the coherence criterion is, however, not that hard if we remember that the dimensionality of the Dirichlet distribution is actually one less than the product of the number of states, for the case with two binary variables the prior density for p_{ij} is

$$\begin{aligned} f(p_{ij}) &= Di(p_{ij}|\gamma_{11},\gamma_{10},\gamma_{01},\gamma_{00}) \qquad (53)\\ &= \frac{\Gamma(\gamma_{11}+\gamma_{10}+\gamma_{01}+\gamma_{00})}{\Gamma(\gamma_{11})\Gamma(\gamma_{10})\Gamma(\gamma_{01})\Gamma(\gamma_{00})} {p_{11}}^{\gamma_{11}-1}{p_{10}}^{\gamma_{10}-1}{p_{01}}^{\gamma_{01}-1}(1-p_{11}-p_{10}-p_{01})^{\gamma_{00}-1}. \end{aligned}$$

that is, one of the parameters, here p_{00}, is *bound* due to the coherence condition. Let us therefore set the prior as suggested below, when $\hat{p}_i = E(p_i)$ and $\hat{p}_j = E(p_j)$. For both p_i and p_j we are using a flat prior in Paper I and Paper II why $E(p_i) = \frac{c_i+1}{C+\alpha}$ and $E(p_j) = \frac{c_j+1}{C+\beta}$ respectively. Our suggestion for a coherent *product prior* then is:

$$\begin{aligned} a_{ij} &= \hat{p}_i \cdot \hat{p}_j \text{ when } i \neq 0 \text{ or } j \neq 0 \qquad (54)\\ a_{00} &= 1 - \sum_{i=1}^{\alpha}\sum_{j=1}^{\beta} a_{ij} \qquad (55)\\ \gamma_{ij} &= \frac{\rho \cdot a_{ij}}{a_{\alpha\beta}} \qquad (56)\\ \gamma &= \sum_{ij} \gamma_{ij} \qquad (57) \end{aligned}$$

This gives a coherent prior for p_{ij} and the expectation value for p_{ij} is then

$$\hat{p}_{ij} = E(p_{ij}) = \frac{c_{ij}+\gamma_{ij}}{C+\gamma} \tag{58}$$

We can see the constant ρ in (eq 56) as a *regulating factor* between the criterion to obtain a *smooth "filtering"* of $E(IC_{ij})$ suitable for signalling purposes and the criterion to obtain maximum sensitivity by a *Haldane* like prior which is obtained when $\rho \to 0$. In the data mining application which is described in Paper I and Paper II we have set $\rho = 1$ which

empirically [Lindquist *et al.*, 2000] seems to be a good choice for the signalling criterion. The factor ρ can also be seen as regulating between an empirical Bayes like informative prior and a non informative Haldane like prior. The expression for the IC_{ij} then becomes (here expressed for two binary variables)

$$IC_{ij} = \log \frac{p_{ij}}{(p_{i0} + p_{i1}) \cdot (p_{0j} + p_{1j})} \tag{59}$$

or more general for the multinomial case:

$$IC_{ij} = \log \frac{p_{ij}}{(\sum_j p_{ij}) \cdot (\sum_i p_{ij})}. \tag{60}$$

5 Actual priors used in the thesis

The priors we have used in the thesis will be summed up here. Using a similar notation which have been used in the papers.

5.1 Priors in Paper I and Paper II

In Paper I and Paper II are the flat, so called *ignorant* prior suggested by Bayes and Laplace used for the single variable probabilities that is, for the marginal probability p_1:

$$f(p_1) = Be(p_1 \,|\, \alpha_1, \alpha_0) = \frac{\Gamma(\alpha_1 + \alpha_0)}{\Gamma(\alpha_1)\Gamma(\alpha_0)} p_1^{\alpha_1 - 1}(1 - p_1)^{\alpha_0 - 1} \tag{61}$$

giving the posterior density for p_1 given the counters c_1 and c_0 as

$$f(p_1 \,|\, c_1, c_0) = \frac{\Gamma(c_1 + c_0 + \alpha_1 + \alpha_0)}{\Gamma(c_1 + \alpha_1)\Gamma(c_0 + \alpha_0)} p_1^{c_1 + \alpha_1 - 1}(1 - p_1)^{c_0 + \alpha_0 - 1}, \tag{62}$$

with the expectation value

$$\hat{p}_1 = E(p_1 \,|\, c_1, c_0) = \frac{c_1 + \alpha_1}{C + \alpha} \tag{63}$$

where $\alpha_1 = 1$, $\alpha_0 = 1.$ and $C = c_1 + c_0$. For the joint probability p_{11} we have a prior which is Dirichlet distributed as

$$\begin{aligned} f(p_{11}) &= Di(p_{11}|\gamma_{11}, \gamma_{10}, \gamma_{01}, \gamma_{00}) \qquad (64) \\ &= \frac{\Gamma(\gamma_{11} + \gamma_{10} + \gamma_{01} + \gamma_{00})}{\Gamma(\gamma_{11})\Gamma(\gamma_{10})\Gamma(\gamma_{01})\Gamma(\gamma_{00})} {p_{11}}^{\gamma_{11}-1} {p_{10}}^{\gamma_{10}-1} {p_{01}}^{\gamma_{01}-1} (1-p_{11}-p_{10}-p_{01})^{\gamma_{00}-1}. \end{aligned}$$

with a posterior density, given the counters $c_{11}, c_{10}, c_{01}, c_{00}$ as

$$P(p_{11}|c_{11}, c_{10}, c_{01}, c_{00}) = Di(p_{11}|c_{11} + \gamma_{11}, c_{10} + \gamma_{10}, c_{01} + \gamma_{01}, c_{00} + \gamma_{00}),$$

with the expectation value

$$E(p_{11}) = \frac{c_{11} + \gamma_{11}}{c_{11} + \gamma_{11} + c_{10} + \gamma_{10} + c_{01} + \gamma_{01} + c_{00} + \gamma_{00}} = \frac{c_{11} + \gamma_{11}}{C + \gamma} \tag{65}$$

where we have set $\gamma_{11} = 1$ and $\gamma = \frac{1}{\hat{p}_{1\cdot}\hat{p}_{\cdot 1}}$. Here $\hat{p}_{1\cdot}$ denotes $\hat{p}_{i=1}$ and $\hat{p}_{1\cdot}$ denotes $\hat{p}_{i=1}$.

5.2 Priors in Paper IV

In Paper IV we did not use a Bayesian method to estimate the probabilities $P(\omega_i)$ and the weights $W_{iq} = \frac{p_{iq}}{p_i p_q}$ because we were not using a prior density giving a posterior density for the weights. This paper uses the Bayes theorem (eq 1) to calculate a conditioned mixture probability outcome given a mixture probability input but does not treat the weights themselves as having distributions. There we used the maximum likelihood estimate $p_i = 1/c_i$ for the probabilities as in this application it is not possible to obtain zero counters because we use normal densities as membership functions both for the input and output layers. Due to the tails of the normal distributions the probability is always greater than zero that a certain value in the input/output layers would belong to any of the normal distributions with which the a priori density of the layer is modelled.

5.3 Priors in Paper V

In Paper V we used a Haldane Dirichlet prior $Di(0, 0, 0, 0)$ for the joint probabilities giving a Haldane Beta prior for the marginal probabilities.

5.4 Priors in Paper VI , Paper VII and Paper VIII

In none of Paper VI , Paper VII and Paper VIII we are using a Bayesian density method to estimate the probabilities. These three publications are all based upon a Bayesian recurrent neural network, but Bayesian in that sense that the weights are deduced from Bayes theorem (eq 1) although the weights themselves are classical point estimates. In these publications the weights are logarithmic and to handle the case of no coincident events (*i.e.* $c_{ij} = 0$) we then set the weights (eq 68) to the lowest value ($\log \epsilon$) we can get from data. In Paper VI and Paper VII the constant $\epsilon = 1/Z_{max}$ is used where Z_{max} (eq 66) is the storage capacity as number of patterns, which is derived in [Lansner and Örjan Ekeberg, 1985].

$$Z_{max} = \mathcal{O}\left((\frac{K}{\ln(K)})^2\right) \tag{66}$$

In Paper VIII ϵ is instead set to $\epsilon = 1/N$, where N is number of patterns learned. This makes the energy function less complex to make us able to easily prove convergence. We handle the marginal probabilities p_i (used as *a priori* probability in the inference) in a similar way when the event count $c_i = 0$. Then we set $p_i = p_{min}$. In Paper VI and Paper VII we set $p_{min} = 1/Z_{max}$ and in Paper VIII we set $p_{min} = 1/N^2$.

$$p_i = \begin{cases} p_{min} & \text{when } c_i \leq 0 \text{ or } N \leq 0 \\ c_i/N & \text{otherwise} \end{cases} \tag{67}$$

$$IC_{ij} = \begin{cases} 0 & i = j \\ 0 & c_i \leq 0 \text{ or } c_j \leq 0 \\ log\ \epsilon & c_{ij} \leq 0 \\ log\frac{c_{ij}N}{c_i c_j} & \text{otherwise} \end{cases} \tag{68}$$

6 To calculate the information component

In (eq 11) we defined the information component as the logarithmic ratio between the joint probability of two events and the product of the probabilities for the two events, but as the probabilities can not be measured exactly, they are parameters which have distributions, so has the IC a distribution as well. Therefore, we may write the IC_{ij} in the following way:

$$f(IC_{ij} \,|\, c_i, c_j, c_{ij}, C) = f(\log \frac{p_{ij}}{p_i \cdot p_j} \,|\, c_i, c_j, c_{ij}, C) \tag{69}$$

which is the posterior density function of IC given the observed frequencies. Under the condition that $p_{ij} > 0$, $p_i > 0$ and $p_j > 0$ we can calculate the expectation value and measures of uncertainty of this function, although it may not be trivial to do analytically. Let us first look upon the expectation value, $E(IC_{ij})$. As we have suggested in Paper I an approximative estimate of the expectation value can be done as:

$$E(IC_{ij}) \approx \log \frac{E(p_{ij})}{E(p_i) \cdot E(p_j)}. \tag{70}$$

This is also the way we have used to calculate the expectation value in the data mining application which is described in Paper II (same as in [Bate *et al.*, 1998]) and also in *e.g.* [Lindquist *et al.*, 2000, Coulter *et al.*, 2001, van Puijenbroek *et al.*, 2002, Bate *et al.*, 2002], but as is indicated in Paper I ([Orre *et al.*, 2000]) and which is handled more thoroughly in Paper III ([Koski and Orre, 1998]) the expectation of IC can be calculated exactly due to the properties of the log-function as

$$E(IC_{ij}) = E(\log \frac{p_{ij}}{p_i p_j}) = E(\log p_{ij}) - E(\log p_i) - E(\log p_j). \tag{71}$$

Each of the expectation values can then be calculated, assuming p is $Beta(a, b)$ distributed as (observe the correction from Paper I which is made in the addendum to Paper III [Niklas Norén, personal communication])

$$E(\log p) = \sum_{n=0}^{\infty} \frac{-b}{(a+n) \cdot (a+b+n)} \tag{72}$$

When implementing (eq 72) it has shown (Paper III addendum A.2) that this sum often converges slowly. This slow convergence is especially pronounced when b is large since when $b >> a + n$ the sum has similar properties as $\sum_{n=1}^{\infty} \frac{1}{n}$ which is divergent. The method we have chosen to implement, to avoid the convergence issue and to obtain a high speed calculation of the expectation value, is instead to use equation 2.11 in Paper III , *i.e.*

$$\frac{\partial}{\partial a} \log B(a, b) = \frac{\partial}{\partial a} \log \frac{\Gamma(a) \cdot \Gamma(b)}{\Gamma(a+b)} \tag{73}$$

where $B(a, b)$ is the *Beta*-function. By evaluating (eq 73) using numerical differentiation, here Euler mid-point difference approximation, we get

$$\begin{aligned} \frac{\partial}{\partial a} \log B(a, b) &= \frac{\partial}{\partial a} (\log \Gamma(a) + \log \Gamma(b) - \log \Gamma(a+b)) \\ &\approx \frac{\log \Gamma(a + \frac{h}{2}) - \log \Gamma(a - \frac{h}{2})}{h} - \frac{\log \Gamma(a+b+\frac{h}{2}) - \log \Gamma(a+b-\frac{h}{2})}{h} \end{aligned} \tag{74}$$

The deviation from the earlier approximation used (eq 70) is usually rather small but gets more pronounced for small values of the counters, which is to expect since the asymmetry of the function becomes more noticeable for small numbers and it is also for small numbers we are interested in as high accuracy as possible.

7 To estimate the uncertainty in the information component

For data mining purposes we are interested in calculating the uncertainty in the information component because we want to be able to take decisions based upon how sure we are that we have actually found a dependency between two variables X and Y. Of course we are also interested in the uncertainty of the non logarithmic measure $W_{ij} = \frac{p_{ij}}{p_i p_j}$, which we prefer to use when the BCPNN is used for classification but we will deal with this more later and therefore concentrate on the logarithmic IC here. A very much used measure in statistics to inform about the uncertainty of a distribution function is the *variance*, which is the *second order moment* [5] of a function. When we tried to calculate the moments directly as $\int\int\int f(p_{11}, p_{10}, p_{01})\, d\, p_{11} d\, p_{10} d\, p_{01}$ as is indicated in section 2.2 in Paper I we failed, however, as we were not able to find any closed-form solutions to neither the W_{ij} or the IC_{ij} function. For the information component we can, however, due to the properties of the logarithmic function, write the variance as an exact expression

$$\begin{aligned} V(IC_{ij}) &= V(\log p_{ij}) + V(\log p_{ij}) + V(\log p_{ij}) \\ &\quad -2cov(\log p_{ij}, \log p_i) - 2cov(\log p_{ij}, \log p_j) + 2cov(\log p_i, \log p_j) \end{aligned} \tag{75}$$

and it can be proved, as in Paper III [Koski and Orre, 1998], that for p being Beta(a, b) distributed, then

$$V(\log p) = \sum_{n=0}^{\infty} \frac{b^2 + 2ab + 2bn}{(a+n)\cdot(a+b+n)^2}. \tag{76}$$

The covariant terms in (eq 75) are there because the different p_{ij} terms in the Dirichlet distribution are not independent of each other as $\sum_{ij} p_{ij} = 1$. We did, however, find it non trivial to estimate the covariant terms in (eq 75) so our first approach was to ignore them until we had found a better method, but instead of (eq 76) we used the simple Gauss' approximation for the variance of a function, *i.e.* $V[g(X_1, \ldots, X_k)] \approx \sum_{i=k}^{k} V(X_i)(\frac{\partial g}{\partial \mu_i})^2$. The approximative expression for $V(IC_{ij})$ when we assume independence is then

$$V(IC_{ij}) \approx V(p_{ij}) \left(\frac{1}{\hat{p}_{ij}}\right)^2 + V(p_i) \left(\frac{-1}{\hat{p}_i}\right)^2 + V(p_j) \left(\frac{-1}{\hat{p}_j}\right)^2. \tag{77}$$

The variances for p_i and p_{ij} thus becomes

$$\begin{aligned} V(p_i) &= \frac{(c_i + \alpha_i)\,(C - c_i + \alpha - \alpha_i)}{(C+\alpha)^2\,(1 + C + \alpha)} \\ V(p_{ij}) &= \frac{(c_{ij} + \gamma_{ij})(C + \gamma - c_{ij} - \gamma_{ij})}{(C+\gamma)^2\,(1 + C + \gamma)} \end{aligned} \tag{78}$$

[5] The variance of $f(x)$ is $V(f(x)) = \frac{\int_{-\infty}^{+\infty}(x - E(x))^2 f(x) dx}{\int_{-\infty}^{+\infty} f(x) dx}$

and the expression which has been used to estimate the uncertainty of IC_{ij} in the standard runs of the application described in Paper II and which the publications [Bate *et al.*, 1998, Lindquist *et al.*, 2000, Coulter *et al.*, 2001, van Puijenbroek *et al.*, 2002] and [Bate *et al.*, 2002] are based upon is

$$V(IC_{ij}) \approx \frac{\frac{C-c_{ij}+\gamma-\gamma_{ij}}{(c_{ij}+\gamma_{ij})(1+C+\gamma)} + \frac{C-c_i+\alpha-\alpha_i}{(c_i+\alpha_i)(1+C+\alpha)} + \frac{C-c_j+\beta-\beta_i}{(c_j+\beta_j)(1+C+\beta)}}{(\log 2)^2}. \tag{79}$$

From the variance in (eq 79) we can then estimate a *confidence* or more correctly a *credibility* interval for IC_{ij} based upon the normal distribution and the standard deviation ($\hat{\sigma}_{IC} = \sqrt{V(IC)}$). One could of course expect that a normal approximation would be a quite coarse estimate because of the likely asymmetry of the Beta distribution, especially for small counter values, and that we are actually performing a non linear transformation by taking the logarithm of the distribution as well and further one we have also assumed independence between the different p_{ij}. More thorough studies of this, by using Monte Carlo simulations based upon sampling the probabilities from a Dirichlet distribution, have however shown that the approximation is actually not so bad when $c_{ij} > 10$. These results can be studied in detail in [Norén, 2002].

At the time of writing this text we are now, based upon the results from [Norén, 2002], implementing a new accurate method for estimating the credibility intervals. This new method is using function approximation to calculate credibility *surfaces* for different *quantiles* of the IC-distribution as a function of c_i, c_j, c_{ij} and C.

8 Mixture modelling, Simpson's paradox and stratification

One problem we have not addressed here, if we go back to our previous model about tossing coins, is that the probability we are trying to estimate may actually be the result of many different processes which give rise to different probabilities. We can make a simple model of this by imagining a bucket full of a few different types of coins, each type with a specific probability to land with the head up. By drawing samples from this bucket to determine the probabilities there is no way of finding the different probabilities in this mixture by doing a series of Bernoulli trials, one trial for each coin. If we could allow us to make an investigation by taking each coin and tossing it a lot of times, this would be easy, but by sampling from a data set we have no way of repeating each trial. If we were able to do this we could find that the resulting probability would be a *mixture* (compare with (eq 42))

$$f(p) = \sum_j \pi_j Be(p \,|\, \alpha_j, \beta_j) = \pi_j \frac{\Gamma(\alpha_j + \beta_j)}{\Gamma(\alpha_j)\Gamma(\beta_j)} \, p^{\alpha_j - 1}(1-p)^{\beta_j - 1}, \tag{80}$$

where $\sum_j \pi_j = 1$. The only probability estimate we can establish so far without taking other information into account reminds about (it's not exactly like this because we also have to take into account the pseudocounts)

$$f(p) = Be(p \,|\, \sum_j \alpha_j, \sum_j \beta_j) = \frac{\Gamma(\sum_j \alpha_j + \beta_j)}{\Gamma(\sum_j \alpha_j)\Gamma(\sum_j \beta_j)} \, p^{\sum_j \alpha_j - 1}(1-p)^{\sum_j \beta_j - 1}. \tag{81}$$

This may give rise to confusing effects which we intend to illustrate by an example.

Suppose that we make a clinical trial which involves 800 patients with a certain disease. These patients either get a certain treatment or not. In the table below we have listed the frequencies for treatment and recovery. In the right part of the table we have listed the IC values. From this we see a correlation between treatment and recovery ($IC(T;R) = 0.152$) as well as between no treatment and no recovery ($IC(\neg T;\neg R) = 0.1255$). From this we would draw the reasonable conclusion that the treatment seems beneficial for the disease.

	R	$\neg R$	tot	IC R	IC $\neg R$
T	200	200	400	0.1520	-0.1375
$\neg T$	160	240	400	-0.1699	0.1255

This would give the recovery as conditioned on treatment as

$$P(R\,|\,T) = 0.5, \quad P(R\,|\,\neg T) = 0.4$$

Let us now assume that we didn't see the table above but were instead presented by the two tables below, were we have separate tables for male and female patients.

male patients

	R	$\neg R$	tot	IC R	IC $\neg R$
T	180	120	300	-0.0589	0.0931
$\neg T$	70	30	100	0.1635	-0.3219

female patients

	R	$\neg R$	tot	IC R	IC $\neg R$
T	20	90	100	-0.4594	0.1420
$\neg T$	90	210	300	0.1255	-0.0506

Would we now make the conclusion that the treatment is beneficial? As we can see for both for the table with only male patients and the table with only female patients the results may seem counterintuitive. Both for the male and for the female patients the treatment seems to have the opposite effect compared to our first listing with all patients where the treatment would be interpreted as being beneficent. This is an example on Simpson's paradox [Bernardo and Smith, 1994]. In this case sex is a confounder to the treatment. We may ask if the two separate tables really are coherent with the information presented in the first table? This is not too hard to show if we apply the axioms in section (1) together with Bayes rule (eq 1) and write the probabilities as

$$\begin{aligned} P(R|T) &= P(R,M|T) + P(R,F|T) \\ &= 180/400 + 20/400 = 0.5 \\ &= P(R|M,T)P(M|T) + P(R|F,T)P(F|T) \\ &= \frac{180}{300}\cdot\frac{300}{400} + \frac{20}{100}\cdot\frac{100}{400} = 0.5 \end{aligned}$$

and similar for

$$\begin{aligned} P(R|\neg T) &= P(R,M|\neg T) + P(R,F|\neg T) \\ &= 70/400 + 90/400 = 0.4 \\ &= P(R|M,\neg T)P(M|\neg T) + P(R|F,\neg T)P(F|\neg T) \\ &= \frac{70}{100}\cdot\frac{100}{400} + \frac{90}{300}\cdot\frac{300}{400} = 0.4 \end{aligned}$$

In statistical terms we can deal with this phenomenon by *stratifying* on sex which gives a coherent treatment of the probabilities in an unbalanced trial design as above.

3 A data mining application

In the previous sections we have presented the essential parts of the theory, mainly from Paper I and Paper III , which give the background to understand some basics of data mining from a Bayesian perspective. This background is of course not complete in any way, but at least thorough enough to deal with a real application where these methods are used. We have not yet looked into more complex dependencies, which we will do later in this text, still we only deal with dependencies between two variables X and Y, both discrete, and in the application presented in Paper II we are yet limited to binary variables only.

1 The objectives with the application

The main problem addressed in Paper II is fundamentally to search for new unexpected dependencies between variables in the database and to produce lists of the most likely dependency connections found and report these so they can be treated by human experts. The fundamental issue here is to reduce the amount of information to be treated by the experts, to reduce the number of reports from about 35000 each quarter at the time of publication of [Bate *et al.*, 1998] to a reasonable amount (about 100).

The BCPNN software[6] is being used in a data mining project in cooperation with Uppsala Monitoring Centre for Adverse drug reactions, the WHO Collaborative Centre for International Drug Monitoring. This is an international centre which maintains a large database *Vigibase* of adverse drug reactions reported from all over the world. Vigibase contains anonymous case reports of suspected adverse drug reactions. For each report there are now over hundred different variables telling about the patient age, country, sex, drugs taken, drug amounts, adverse drug reactions, suspected drugs, etc. There is also information about the drugs' substance contents. It is the largest database of this sort in the world. Now, at the time of this writing it contains about three million reports, and about 50 000 new reports are added quarterly. The number of national centres which collect reports in their own country and then send them to the Uppsala Monitoring Centre has grown from 47 since the publication in Paper II was written to 67 (spring 2003).

The task of trying to find new drug-ADR signals has been carried out by an expert panel, but with such a large volume of material the task is daunting. We have therefore developed a flexible, automated BCPNN-based procedure to find new potential signals which stands out from the background data, to be able to support the experts.

2 Discussion about results

The IC is a measure of the dependency between two variables, X and Y. In the application in Paper II these variables stand for a drug and a suspected adverse drug reaction (ADR) being reported together and stored into a database. By following the development of the IC-distribution over time we are able watch the dependency relation between these variables when more data, *i.e.* more reports, are added to the database. Properties of the IC-distribution which are easy to follow and can also be plotted in a diagram are the expectation value $E(IC)$ and some quantiles of its credibility interval [7]. In the three diagrams in figure 1, figure 2 and figure 3 in Paper II we have plotted the expectation values and their

[6] John W. Tukey proposed 1958 the word *software* for programs on computers [Leonhardt, 2000]

[7] In Bayesian statistics the term *credible interval* or *credibility interval* is the preferred term instead of *confidence interval*, as there is most often an uncertainty in a *parameter* estimate we are interested in.

95 % canter credible distribution limits for for $IC(captoril; coughing)$, $IC(digoxin; acne)$ and $IC(digoxin; rash)$ respectively.

In figure 1 the diagram of $IC(captoril; coughing)$ indicates that the dependency became significant already the second quarter 1981. This case of adverse reaction was first reported in literature in Dutch in July 1983 [Knoben, 1983] and was not widely known until 1986.

In figure 2 which shows the IC for the rarely reported combination digoxin-acne, we can see that the expectation value starts at zero, due to our choice of prior for p_{ij} (eq 65), but has a large uncertainty interval. As more reports enter the database for the drug digoxin and acne but no reports for the combination, the dependency relation goes negative. In 1989 one report of the combination entered the database and $E(IC)$ jumps slightly upward, but $E(IC)$ is still negative.

In figure 3 of Paper II we see an example of a combination which is reported rather often, digoxin-rash. Both digoxin and rash are among the most reported drugs and suspected adverse reactions and the combination digoxin-rash is not rarely reported. Although we get a jump towards a positive dependency in 1968, when there were still very few reports in the database the large uncertainty interval would not have raised a signal in this case. After a few years this relation stabilises on an IC value around -2. It should be observed that a negative IC should not necessarily be interpreted as the drug being beneficial against the specific symptom. There are several reasons for an IC to become negative. There can of course be confounders but a likely explanation in this case is that the negative bias of the IC is caused by *underreporting* of this specific combination.

3 Goals achieved

This *screening* procedure can be performed between all variable combinations that we want to watch over time. As a *watchdog* criterion we used that when the lower 97.5 % credibility interval limit passes zero we obtain a *signal indication*. These signal indications raise the specific variable combinations to be further investigated. In this case by an expert panel which can take decisions about how to proceed with further investigations of, in this case, the specific drug-ADR combinations that show a dependency relation. This type of signalling scheme is therefore a valuable tool for drug manufacturers and drug regulators as they are given a better chance to act fast in serious cases of adverse reactions which were previously unknown.

We developed a procedure "quarterly update" which was run each quarter of a year and produced a report on those drug-ADR relation which had triggered the signal criterion. At this time the database was regularly updated each quarter. Now when the database may be updated more often we also run this procedure more often.

An investigation was made, which is discussed in section 3 of Paper II , where a quarterly update scan was done on historic data from beginning of 1996. There it was found that of 307 associations which passed the (Bayesian) significance test (or credibility threshold) there were 53 associations involving *new* drugs, *i.e.* drugs released since 1990. Of these 53 associations there were 12 which were still not recorded in the May version of Martindale 1997. A study which evaluates what drugs would have been signalled on when analysing historic data is done in [Lindquist *et al.*, 2000].

4 Alternative methods

The type of analysis we are doing here, *i.e.* to try to catch unexpected dependencies between medical drugs and adverse reactions are also done by other people. One method is being developed by William DuMouchel and the U.S Food and Drug Administration (FDA) [DuMouchel, 1999]. One fundamental difference between DuMouchel's method and our IC method is that DuMouchel looks at the reporting intensity instead of the reporting probability, thus asserting a Poisson process as the model for reporting ratio instead of a binomial process. The Poisson process has as its conjugate prior a gamma distribution [Bernardo and Smith, 1994].

Instead of looking at when $IC_{ij} > 0$ DuMouchel compares the actual number of reports of the combination N_{ij} with a *baseline* count E_{ij}, *i.e.* when $N_{ij} > E_{ij}$ there is a signal indication. The baseline count is the expected number of reports under the assumption of independence and is constructed as

$$E_{ij} = \sum_k N_{i.k} N_{.jk} / N_{..k}, \tag{82}$$

which he calls a naïve baseline frequency. This is similar to the assumption about independence when $IC_{ij} = 0$. The sum over k in (eq 82) is a sum over different *strata* which may be done on *age*, *sex* etc. For his prior he uses a mixture of two Gamma distributions, which gives five parameters to estimate for each drug-adverse reaction combination. He calculates $\lambda_{ij} = \frac{\mu_{ij}}{E_{ij}}$ where $N_{ij} \approx Poisson(\mu_{ij})$. The posterior distributions of each λ_{ij} are used as "shrinkage" estimates on which a variance and thus an uncertainty interval can be calculated. DuMouchel [DuMouchel, 1999] uses the lower 5% point of the posterior distribution as a "signalling" criterion, where we used the lower 2.5% point.

The differences between DuMouchel's method and the IC method are small. He suggests the *relative risk* as a measure

$$RR_{ij} = \frac{N_{ij}}{E_{ij}} \tag{83}$$

where $\log_2 R_{ij}$ is almost identical to the IC_{ij}. There are a few differences, though, one is that DuMouchel consider the marginal counts $N_{i.}$ and $N_{.j}$ to be fixed, *i.e.* they don't have a distribution, for a specific N_{ij}. This gives a potential for higher accuracy in our uncertainty estimate of the IC_{ij}, although the variance estimate used in [Bate *et al.*, 1998] is quite coarse for small counts. Nowadays we have, however, developed and implemented a very accurate method for uncertainty calculations based on both the marginal and the joint counters, which is presented in chapter 2 section 6 above. DuMouchel has further developed the method for multiple items associations [DuMouchel and Pregibon, 2001]. This is discussed more in chapter 6 section 4 below.

There are also other methods for evaluation of adverse drug reaction reports. One is Stephen Evans' *proportional reporting ratio* (PRR) [Evans *et al.*, 2001], which is based on classical frequency estimates. Using a similar form as earlier for IC (eq 60) the PRR can be defined as

$$PRR_{ij} = \frac{c_{ij}/(c_{i0} + c_{i1})}{c_{\bar{i}j}/(c_{0j} + c_{1j})} = \frac{c_{ij}/c_i}{c_{\bar{i}j}/c_j} \tag{84}$$

The PRR is, however, only considered for the true counts so, using the same notation as

DuMouchel uses above the PRR could then be expressed as:

$$PRR_{ij} = \frac{N_{ij}/N_{i.}}{(N_{.j} - N_{ij})/N_{.j}}. \tag{85}$$

In words, the PRRs compare proportion of reports for a specific ADR in a drug with proportion in all other drugs.

5 Implementation Issues

The BCPNN data mining software we are developing for the drug adverse drug reaction database application sees a large matrix of data, where each row in the matrix corresponds to one case report reported by a physician who suspects the patient to have an adverse reaction due to the medication. The potential size of this matrix is rather large as for instance the types of reported drugs so far is over 12000 and the number of reported adverse drug reactions is almost 2000.

The Vigibase database of adverse drug reactions is a relational database organised as a set of SQL tables. As it now contains around three million reports and over hundred variables it may be considered to be a rather large database. A database of this size puts certain requirements on the software accessing it. During the design of the system we divided the solution into different modules each concerned with different parts of the problem. The main modules were the database accessor, data preprocessor, layer module, scan module, a calculation and analysis module and a module for interactive visualisation of query results. Our original idea was to use a standard database tool for the database accesses and use a preprocessor to transform the output from this database accessor into a form suitable for our neural network software. We first tried to use a well known database software package and set up the database to generate data to the neural network software. Unfortunately, after great effort had been put into getting the database accessor to produce data in the way we wanted, it showed that this was too slow. At the time we were doing this a machine could need several days to produce data for a scan. We decided to write dedicated software to access the database directly from the data mining software in a way suitable for doing time scans. The first problem was that we needed to access the data as objects, not as relations, to make the training of the neural network efficient. Our accessor made it possible to sequentially access the database as if it would have been an object database organised as case reports. A data preprocessor was no longer needed and our accessor could now directly transfer data from SQL tables to the ANN layers. We gained *several hundred times* in speed this way and the training times become reasonable on a Unix (Sun Sparc) work station.

The training time, when we collect statistics about the variables as matrixes of dependencies, varies a lot depending on how many variables are being used and to what extent these variables are combined into complex units. A simple query now takes just a few minutes and a more complicated one may take several hours.

In the start up of this project we tried to run the software on a supercomputer to decrease the running times. This was a parallel executable MIMD machine with a lot of CPUs and a lot of memory in each CPU. We did, however, find that the waiting queues to this machine were too long to make this efficient so we continued working on the work station and dedicated server solution.

At the moment we are doing most runs on a dual CPU 1.4 GHz Pentium III machine equipped with 3 Gb of memory which runs the GNU/Linux operative SMP (*synchronous*

multi processing) system. Scheme [Kelsey *et al.*, 1998] is used as high level application language, in this case a development version of *Guile*. All low level processing is done in ANSI-C/C++. Java and SQL is used for data preprocessing and data conversions.

4 On prediction and classification

There are many reasons why we collect data, one fundamental reason being discussed earlier is to be able to do *data mining*, in essence *data analysis* where data being analysed is turned into information which may increase our knowledge about the processes which generated the data. This knowledge discovery process is a never ending iterative process where we, after having defined fundamental goals, collect data, learn from data and then take decisions based upon this new knowledge and so on. The knowledge discovery process can be *unsupervised* or *supervised* or a combination of both. In this section we will deal with some supervised aspects of the learning process.

In many applications we may want to collect data just to be able to predict something about future outcomes. Data is then used to *train* some automatic learning system which is presented some *explanatory* data or *input* data, often denoted with the variable X and some *response* data which is the desired outcome, often denoted with the variable Y. The machine learning system being trained will then learn to *mimic* the behaviour of the original system. This process *model* can be very complex, composed of both statistical inference and rule based reasoning combined with some *implicit* or *explicit* model architecture of the process. The more complex the model is, the more data is needed, due to the *curse of dimensionality* [Friedman, 1996], as more parameter values need to be determined.

In this section we will discuss two different publications, one dealing with *prediction*, in Paper IV, and one dealing with the problem of classification in Paper V. Both prediction and classification are basically the same problem but here we will use "prediction" to describe the issue of finding the outcome of a real valued process and "classification" as determining a categorisation of the explanatory data. In both these problems we use a Bayesian classifier to determine the outcome. A *naive* Bayesian classifier [Kononenko, 1989, Rish, 2001], see (eq 7), assumes that the explanatory variables are independent and is therefore sometimes referred to as a *simple* Bayes classifier or *simple Bayes learner* [Mitchell, 1997] and even *idiot's* Bayes [Hand and Yu, 2001], but is usually very robust and gives surprisingly good results despite the independence assumption [Hand and Yu, 2001]. As the "opposite" to the naive Bayes classifier we may consider the *optimal* Bayes classifier, see (eq 6) [Mitchell, 1997] where no assumptions about independence is done, every state of the response variable space is associated with an explicit state of the joint explanatory variable space. An optimal Bayes classifier minimises the Bayes error if we know the density functions for the outcome classes, but it then also requires more data to determine these densities with a reasonable precision. The approach with BCPNN (Bayesian Confidence Propagation Neural Network) is to join those variable dimensions in the input space which are quite dependent and consider the other variables independent.

The publication in Paper IV deals with prediction of real valued variables using radial basis functions (RBF) [Tråvén, 1991, Xu *et al.*, 1993] to model the densities in the input/output layers to a naive Bayesian classifier. This RBF coding of the input/output spaces, however, as it is done with multidimensional Gauss functions with the same dimensionality as the explanatory data, reminds about an optimal Bayes classifier for real valued

data. The publication in Paper V deals with the problem to model the density function for the outcome class for a naive Bayes classifier, a BCPNN classifier and an optimal Bayes classifier.

1 Function approximation by mixture density association

In Paper IV we deal with an important issue in process industry, the prediction of certain response variables from a process given a proper subset of explanatory variables available from that process. This prediction may be wanted for several reasons. An actual response variable from a process may be hard or time consuming to measure and it may be desirable to check the result of a change in some of the explanatory variables for a process before actually performing the change to the real process, to avoid timely and costly production losses and to maintain a continuous high production quality outcome.

1.1 Project objectives

The project which resulted in the publication in Paper IV [Orre and Lansner, 1996] was done partly as a cooperative work between the SANS (Studies of Artificial Neural Systems) group at KTH (Royal Institute of Technology) and the pulp and paper industry (STORA Teknik AB) and partly between SANS and STFI (Swedish Pulp and Paper Research Institute). In the first part of the project, with STORA, which resulted in the report [Orre and Lansner, 1992b] we used a multilayer perceptron [Rumelhart *et al.*, 1986, Leighton, 1991] using the *error back propagation* algorithm, often called just *"backprop"*, as a function approximator tool to search for minimal neural architectures able to predict the response variables as good as possible. In the second part of the project, with STFI, we developed a function approximator using a Bayesian classifier modelling the mixture densities of the explanatory and response variables with the help of radial basis functions, thus making a "mixture density associator" which could produce also an uncertainty estimate, in the form of a confidence interval, of the predicted outcome.

The objective was to study how a set of response variables, expressing the paper quality as the result of time consuming laboratory experiments, could be predicted from a set of explanatory variables obtained by automatic measurements from the pulp to paper manufacturing process.

The explanatory variables were: *%pulp type1, %pulp type2* and *%pulp type3*, which were mixture percentages of the pulp content of three different pulp types; *%middle* and *%long*, which were the percentages of the pulp content of two different fibre length classes where the measure is the percentage of the length class versus other fibre length classes; *drain-time*, which is a measure of how long time it takes for a standardised piece of pulp to drain and *drain-speed*, which is a measure of how quick the water flows out of the pulp. The response variables were: *tear*, the force needed to rip the paper apart; *tensile*, the force needed to break the paper when stretched. In part one of the project [Orre and Lansner, 1992b] we studied a few other response variables as well, *csf*, *elongation* and *density*, but in the second part presented in Paper IV we focused on *tear* and *tensile* only. The reason was that there were quite a few *missing data* values for the remaining three response variables.

1.2 Project part one, backprop study

A method was developed which searched for the minimal multi layer perceptron architecture being able to predict a certain laboratory value. We used one separate network for each response variable because the amount of data available for training and test was rather limited. Therefore we found it preferable to make one optimised network, *i.e.* one non linear regression model, for each outcome variable to predict.

The reason for trying to find an architecture as small as possible for each response variable was to provide for the maximal generalisation capability or minimal risk for *overfitting* as the data set available for training was too small to allow for network architectures with a high dimensionality. The method also arranged the inputs according to the impact they had on the prediction performance. This made it possible to further reduce the network dimensionality and to increase the prediction quality by removing irrelevant parameters. We also tested a model which continually adapted to changes in the process behaviour in case the process would not be *ergodic*, but this gave slightly worse results than a randomised partition into training and test data which indicated that the statistics of the data was not changing significantly over time. In this project we got valuable experience dealing with real valued data and we also developed several methods for analysis and validation of backprop-networks as well as producing automatic presentation of results in graphical and table form.

As a conclusion of the results from this study we found that we could reach a prediction precision of about $5-10\%$ relative the standard deviations for each of the variables. For *tear* we got a standard error of 11.7% and an average error of -7.5%. For *tensile* we got a standard error of 10.8% and an average error of 6.2%. The average prediction quality could probably be increased by using more data. For this study we used a total of 179 data samples. Unfortunately we had no reference of what performance could be achieved with other methods, but the results we obtained in the backprop study were used as a reference for our further studies with mixture density networks, which are presented in Paper IV .

It must be clear, however, that no method can make a better result than what is possible due to disturbances and measurement errors in the training and test data. If the errors for these were measured as parts of the absolute values of the dynamic ranges they would be much smaller than presented here. We observed that the prediction quality was rather much dependent on the type of partitioning into training and test set which was used. Due to the computational resources available when this study was done we were not able to perform any crosssvalidation studies within reasonable time. Instead we used a set of partitionings of the data set into 75% training data and 25% test data as is illustrated by figure 3.

	first	last	rand	split
csf	14.1	37.8	14.3	11.2
density	12.9	15.3	9.9	14.4
elongation	8.0	12.0	10.9	9.6
tear	14.0	26.6	12.3	13.9
tensile	15.7	16.5	13.2	12.9

Table .1: Prediction performance for different partitions, averaged over all tested architectures. As column "last" shows the largest error for all outputs it may indicate a change in the process during an early time period which was not covered by training data.

If we take a look at table .1, which shows the prediction performance averaged over all tested architectures, we can see that there is a clear difference for "first" and "last"

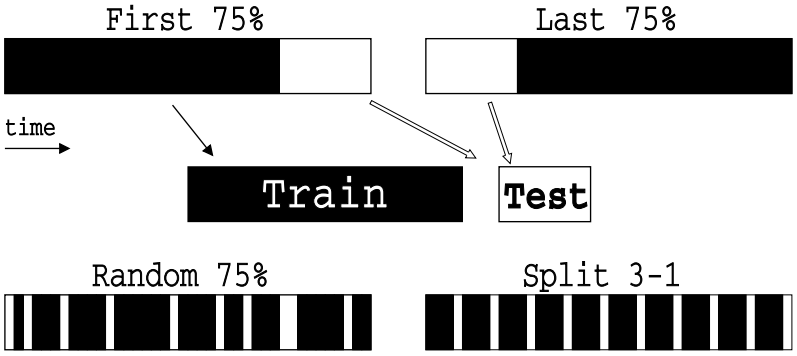


Figure 3: Four different partition types, which have been used. In all four cases, 75 % of the input data have been used as training data.

partitioning, being especially evident for *csf*. This indicates that something may have happened in the process from the first to the last sample.

1.3 Project part two, mixture density network

In the part of this project described in the previous section and in [Orre and Lansner, 1996] we used a non linear regression model, the multilayer perceptron trained by the backprop algorithm to predict the response variable outcome. In most regression models used for function approximation, as is the case for the multilayer perceptron, the output is a real value, a point estimate. These regression methods usually minimises the *mean square error* between the predicted ($\hat{y}$) and the *desired* (y_j) output value *i.e.*

$$\min \sum_j (\hat{y}_j - y_j)^2, \tag{86}$$

which is the case with the backprop algorithm [Mielniczuk and Tyrcha, 1993]. Regression methods as backprop usually adjust the coefficients (weights for neural network models) for a set of functions until the minimum of an error *hyper*-surface is reached. In the mixture density model presented in Paper IV the predicted outcome value is instead an estimate of the probability density function for a response variable conditioned by a certain explanatory variable value such as

$$f_Y(y|X{=}x) \tag{87}$$

The $f_Y(y|X = x)$ density functions gives much more information about the predicted outcome than a regression method which just gives a point estimate. In addition to the predicted value, you can also estimate a confidence interval (assuming uni-modality). Observe that this works for multi-modal outcomes too, but then you may be interested in several smaller confidence intervals, not one huge. Both the input and output variables density functions are modelled by mixing Gaussian density functions. The Gaussian density functions used are here called RBF (*radial basis functions*), as this is an artificial neural network method. The picture (produced from Paper IV) in figure 4 illustrates the principle. First we find the *marginal* probability density functions for the explanatory $f_X(x)$ and response $f_Y(y)$ variable spaces by using a stochastic EM (expectation maximisation)

algorithm [Tråvén, 1991] achieving

$$f_X(x) = \sum_{i=1}^{n} P(\omega_i) f(x|\omega_i) = \sum_{i=1}^{n} P(\omega_i) N(x, \mu_i, \sigma_i^2) \quad (88)$$

$$f_Y(y) = \sum_{q=1}^{m} P(\omega_q) f(y|\omega_q) = \sum_{q=1}^{m} P(\omega_q) N(y, \mu_q, \sigma_q^2) \quad (89)$$

where the density components $f(x|\omega_i)$ are Gaussian distribution functions centred at μ_i with variance σ_i^2. The probability for each component density is $P(\omega_i)$ where $\sum_i P(\omega_i) = 1$. Now we want to express a response variable density $f_Y(y|X = x)$ as a probability relation between component densities of the response variables (ω_q) and component densities of the explanatory variables (ω_i). The response variable density can be expressed as

$$f_Y(y|X{=}x) = \sum_q f_Y(y|\omega_q) P(\omega_q|X{=}x). \quad (90)$$

In (eq 90) the probabilities for ω_q will only depend on the X-value through the probabilities for ω_i. Assuming that the density functions $P(\omega_i|X = x)$ are mutually exclusive we can use "theorem of total probability" [Mitchell, 1997] as in (eq 91) and after this in (eq 92) apply Bayes rule (eq 1). The definition of conditioned probability (eq 0) gives, under the assumption that ω_q and ω_i are independent, the expression in (eq 93).

$$P(\omega_q|X = x) = \sum_i P(\omega_q|\omega_i) P(\omega_i|X = x) \quad (91)$$

$$= \sum_i P(\omega_q) \frac{P(\omega_i|\omega_q)}{P(\omega_i)} P(\omega_i|X = x) \quad (92)$$

$$= P(\omega_q) \sum_i \frac{P(\omega_q, \omega_i)}{P(\omega_q)P(\omega_i)} P(\omega_i|X = x) \quad (93)$$

By combining (eq 90) and (eq 93) we can now get the total expression in (eq 94) for the output density

$$f_Y(y|X{=}x) = \sum_q f_{Y_q}(y|\omega_q) P(\omega_q) \sum_i \underbrace{\frac{P(\omega_q, \omega_i)}{P(\omega_q)P(\omega_i)}}_{W_{iq}} P(\omega_i|X{=}x) \quad (94)$$

The expression $\frac{P(\omega_q,\omega_i)}{P(\omega_q)P(\omega_i)}$ we recognise as the *weight value* W_{iq}, which we find by *training* the network as was discussed in section 5.2. Finally by integrating (eq 94) we can find the expectation value of $f_Y(y|X = x)$ as in (eq 95) and the distribution as in (eq 96) below.

$$E(y|X{=}x) = \int f_Y(y|X{=}x)\, y\, dy \quad (95) \qquad F_Y(\gamma) = \int_{-\infty}^{\gamma} f_Y(y|X{=}x)\, dy \quad (96)$$

In this specific case, because we are working with symmetrical Gaussian density functions we do not, however, need to perform the actual integration as in (eq 95), it is enough to

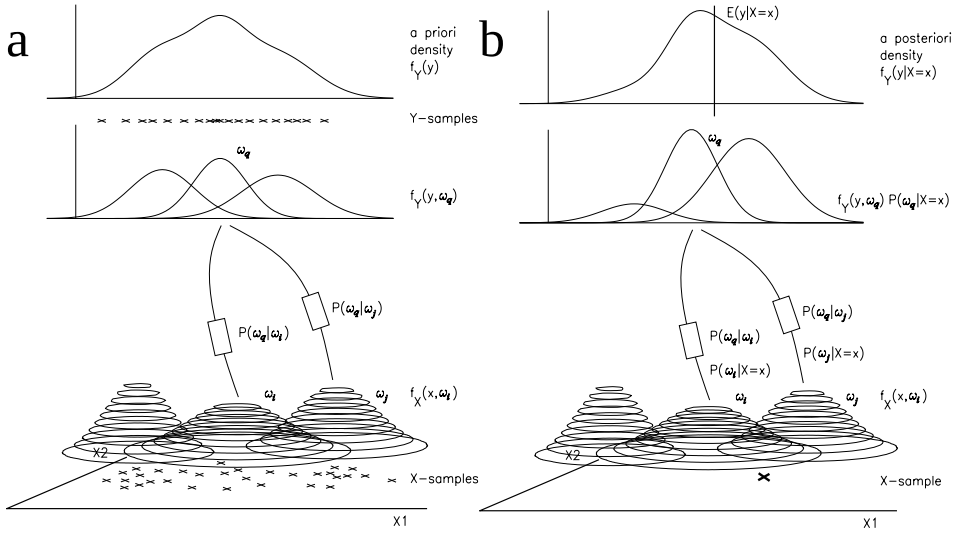


Figure 4: An overview of the density method. **a:** Training phase: We let a set of Gaussian density functions (ω_i) model the density of a set of data samples in explanatory (X) and response (Y) variable spaces. When an X-sample is drawn from the process ω_i in explanatory space there is a certain probability $P(\omega_q \mid \omega_i)$ that a Y-sample is drawn from the process ω_q in response space. **b:** Recall phase: When a certain X-sample is input we get a response probability $P(\omega_i|X = x)$ from each of the explanatory *processes*. These probabilities are propagated to the response space and cause the response variable density to be conditioned by the explanatory value $X = x$.

just add the centre values μ_q multiplied with their probabilities as

$$E(y|X{=}x) = \sum_q P(\omega_q|X{=}x)\,\mu_q \tag{97}$$

To get to know the quality of the prediction though we need to perform the integration as in (eq 96) to estimate a confidence interval. This integration was done numerically by using Simpson's rule [Itô, 1996]. We then solved the following equation numerically to obtain a 95% confidence interval (which does not account for the uncertainty in the weight values)

$$\left.\begin{array}{rcl} 0.025 & < & F_Y(y_1) \\ 0.975 & > & F_Y(y_2) \end{array}\right\} \Rightarrow y_1 \leq Y_{95\%} \leq y_2 \tag{98}$$

1.4 Density modelling

In (eq 88) we assumed that we could express the *marginal* (in this case in the meaning of *unconditioned*) density $f_X(x)$ as a mixture of Gaussian component densities. In fact, almost any "bump" like function can be used [Ripley, 1996] but Gaussian density functions have nice *smoothing* properties suitable for the *interpolation* we want to achieve and it has been shown [Feller, 1966] that any bounded continuous density with a finite number of discontinuities can be approximated to any degree of accuracy, apart from small regions around the discontinuities, by using a *finite mixture* of *e.g.* Gaussian component densities.

In cluster analysis these component densities are often *multivariate* [Everitt, 1998] *i.e.* the component density i is parameterised as $N(x, \mu_i, \Sigma_i)$ where Σ_i is the *covariance* matrix. (In Paper IV we have used C_i to denote the covariance matrix, but we have found *e.g.* [Krzankowski and Marriot, 1994] that Σ_i is the usual.) The method we have used to find the parameters, a *stochastic* variant of the EM algorithm [Tråvén, 1991], manages also the multivariate case to determine the covariance matrix Σ_i for each component, but, as this would require many more parameters to be estimated and our *training* data set size was rather limited (around 100) we preferred to use symmetric (*univariate*) density functions $N(x, \mu_i, \sigma_i^2)$, giving only two parameters per Gaussian. In this application we used Gaussians with seven dimensions for the input layer, *i.e.* the same number as the number of explanatory variables. We will here sketch upon the principles for finding a solution on how to determine the parameters for the component densities. Details about this can be studied in [Tråvén, 1991] and also in chapter 3 in Paper IV .

Each component i is characterised by a set of parameters, which for the Gaussian case would be a *covariance matrix* Σ_i or, for symmetrical densities, a *variance* σ_i^2, a *centre value* μ_i and a *probability* $P(\omega_i)$. We have a set of N samples $\{x_1, \ldots, x_N\}$ drawn from a mixture, $f(x)$, of n density functions:

$$f(x) = \sum_{i=1}^{n} P(\omega_i) f(x|\omega_i) = \sum_{i=1}^{n} \alpha_i \varphi(x, \theta_i) = [e.g. Gaussian] = \sum_{i=1}^{n} P(\omega_i) N(x, \mu_i, \sigma_i^2) \quad (99)$$

By applying Bayes rule (eq 1) on the density functions (eq 99) we get an expression for the probability that a certain X-value x was generated from the component ω_i,

$$P(\omega_i|x) = \frac{P(\omega_i)P(x|\omega_i)}{P(x)} = \frac{\alpha_i \varphi(x, \theta_i)}{\sum_{j=1}^{n} \alpha_i \varphi(x, \theta_i)}. \quad (100)$$

The EM algorithm is here used to estimate the parameters α_i and θ_i which means that we search for the parameters which maximises the log-likelihood ($\log L$) (eq 101) of the samples, under the constraint that the probabilities α_i sum to 1, which may be solved by using the Lagrange multiplier [Krzankowski and Marriot, 1994] method.

$$\log L = \sum_{k=1}^{N} \log f(x_k). \quad (101)$$

In this application we have made it somewhat easier by asserting $\alpha_i = 1/n$, *i.e.* equal a priori probabilities $P(\omega_i) = 1/n$ for each density component. The maximisation problem is then to find optimal values for θ_i which here corresponds to optimising μ_i and σ_i^2. The usual way of solving this with the EM algorithm is to optimise the parameters for the whole data set at once. Here we have instead used a stochastic version [Tråvén, 1991] of the EM algorithm, which updates the parameters after each data sample. The principle is to find *recursive* expressions for the parameter θ_{N+1} corresponding to sample $N+1$, that is expressing it with the help of θ_N and some update rule containing information obtained from the last sample $N+1$. The full expression for θ_{N+1} is shown in section 4 of Paper IV . Here we only show the simplified update rule which is:

$$\theta_{N+1} = \theta_N + \eta_{N+1}(\theta(x_{N+1}) - \theta_N) \quad , \quad \eta_{N+1} = \frac{P(\omega|x_{N+1})}{\sum_{k=1}^{N+1} P(\omega|x_k)} \; \delta \quad (102)$$

The η_{N+1} above is the reason why the Gaussian component densities will *compete* with each other, the density function being *closest*, *i.e.* having the largest probability $P(\omega|x_{N+1})$ in average, for the last sample will also be the one "moved" the most. The δ in (eq 102) is not necessary and can be set to one, but allows the η to be scaled down to get a smooth start. We can see it so that we make one *expectation* step, where we calculate each $P(\omega_i|x_{N+1})$ in average to get a response from each of the density functions ω_i, then a *maximisation* step when the parameters θ_i are moved towards the value maximising the likelihood function. The final update schemes for μ and σ^2 become

$$\mu_{N+1} = \mu_N + \eta_{N+1}(x_{N+1} - \mu_N) \quad (103)$$

$$\hat{\sigma}^2_{N+1} = (x_{N+1} - \mu_N)^T(x_{N+1} - \mu_N)/d \quad (104)$$

$$\sigma^2_{N+1} = \sigma^2_N + \eta_{N+1}(\hat{\sigma}^2_{N+1} - \sigma^2_N) \quad (105)$$

Formally the estimate of $\hat{\sigma}^2_{N+1}$ in (eq 104) should be $\hat{\Sigma}_{N+1} = (x_{N+1} - \mu_N)(x_{N+1} - \mu_N)^T/d$, because the maximum likelihood estimate of $\hat{\Sigma} = \frac{1}{N}\sum_{k=1}^{N}(x-\mu)(x-\mu)^T$. However, for the univariate case where $\Sigma = \sigma^2 I$, as we have restricted ourselves to symmetrical Gaussians here, it is more efficient to just see (eq 104) as the squared distance over the number of dimensions (d). The number of dimensions, here $d = 7$, appears in the normalisation constant of the multivariate normal distribution

$$\frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp -\frac{1}{2}(\boldsymbol{x} - \mu)^T\Sigma^{-1}(\boldsymbol{x} - \mu) \quad (106)$$

1.5 Results on prediction with mixture densities

The diagrams in figure 5 show an example of predictions using artificially generated data. Here we have trained the network with the function $y = sin(x) + noise(N(0, \sigma = 0.07))$. In the right diagram of figure 5 we can see the resulting prediction over the interval. The error bars in the diagram show the estimated uncertainty as one standard deviation.

One interesting detail about this type of function approximation is that the mixture density method is actually symmetrical, *i.e.* there is a density mapping from $X \rightarrow Y$ and also $Y \rightarrow X$ as well. In the figure 6 we see an example where $Y = sin(X)$ (diagram a). In diagram b and c we see density estimates of the inverse function $x = arcsin(y)$ illustrating the fact that the function is not *bijective* as we get a multi modal density recall. In be plot in diagram b we see the density response for $Y = arcsin(0.5)$ and in diagram c we see the density for $Y = arcsin(1)$. For many process modelling predictors a multi modal density prediction can certainly be considered more correct than just an average point value delivered by the standard regression models.

Finally let us look at the diagrams in figure 7 which illustrate how the method is able to predict on the test sets for the two pulp to paper process variables *tear* and *tensile*. The error boxes in these diagrams show one standard deviation. We can see that the predicted expectation values are quite close to the desired values. For the uncertainty estimations we can see that there is one sample for tear, sample 9, and two samples for tensile, sample 5 and 22, which have a large standard deviation, although their expectation values are close to the desired ones. Here one may suspect that competing processes in the input mixture give rise to quite different outcomes, but in average the result is close to ideal. It could be like when adjusting the temperature in two rooms in the same building. If the temperature in one room would be too low and the temperature in the other room too high, they would still be

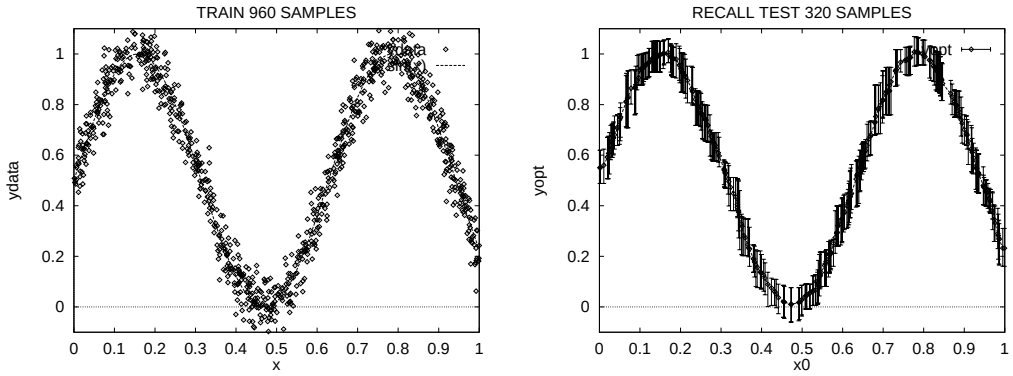


Figure 5: **Left:** Training data consisting of 960 samples with a normal distributed noise ($\sigma = 0.07$). **Right:** Recalled y-value from 320 samples with a network using 60 explanatory units and 40 response units. The error bars show one predicted standard deviation.

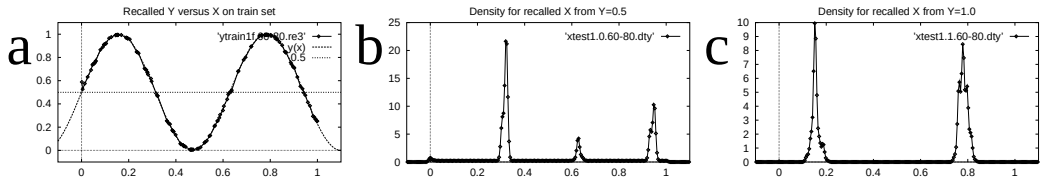


Figure 6: **a:** A recalled sine function as $Y = \sin(X)$. **b:** Recalled density for reverse mapping $Y \to X$ when $X = \arcsin(0.5)$. **c:** Similar as in b when $Y = 1$, *i.e.* $X = \arcsin(1)$.

good in average. An untested hypothesis for the case with paper quality variables could be that there are combinations of process properties striving towards a paper with low quality and some properties which strive for a paper with high quality. When these properties are mixed at a macro scale we can only see the average result, but it could be that they on a micro scale would make a difference. A reasonable approach would be to try to avoid values with a high uncertainty, until it can be investigated if these high uncertainties depends upon multi modalities in the output, which could possibly need further investigation.

A common problem with regression methods is that of *overfitting i.e.* asserting too complex functions or too complex perceptron architectures, which would need large amounts of training data. Do we have this problem with this probability mixture predictor as well? Yes, As each of the radial basis functions, here Gaussian densities, have a rather local effect, but on the other hand they also have *regularising* properties. By increasing the variances we can get a better covering of the input/output space. The precision may be affected though. Seen from a pure function approximation perspective the Gaussian functions are not linear, as the probabilities are, the actual predicted value is always an approximation. There is an optimality relation between the distance between the centre values μ_i and their variances σ_i. In figure 8 in section 6 in Paper IV we illustrate this by showing how the representation precision of a certain input value is affected by adjusting the regularization parameter, *i.e.* scaling the variances. This scaling was done from the variances obtained by optimising the variances for density estimations by the EM algorithm. It is of course no reason to expect

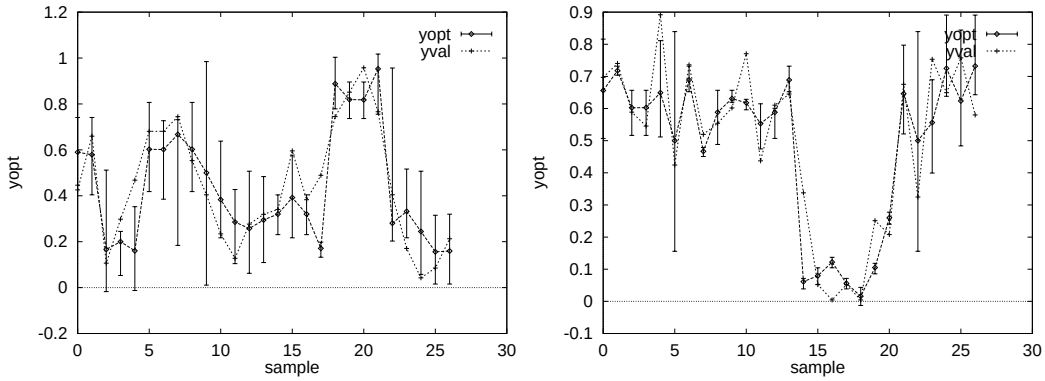


Figure 7: Two test sets with prediction outputs for *tear* and *tensile* with estimated 67% confidence intervals (*one standard deviation*) plotted as error bars. The predicted values are marked with ⋄:s (with bars) and the process output values are marked with +:s. In some cases like sample 9 for *tear* and sample 5 for *tensile* we see a large confidence interval estimation due to the input vector being far from all the RBF units.

that the parameters set for optimal density modelling should also be optimal for prediction. A general principle may be, that if the space is very regular, then we can have very large variances to cover almost all the space with all Gaussian densities, but if we have many *outliers* then we need high precision to cope with them and a too regularized network may not deal well with them. We can see from table 2 and table 3 in section 6 of Paper IV that, as we would expect, the prediction precision for training data is improving with the size of the number of RBF units, especially the number of units in the input layer. For the test data, however, there is no simple relation between the number of RBF units in input and output layers and the prediction error.

In the work presented here we have concentrated on the function approximation problem and have applied this to process modelling . We have tested methods for filling in of missing input values and to do both training and prediction with data containing missing values. The simple method we used to handle training and test with missing data was to consider the distance to the RBF components to be zero in the missing dimension. This method to handle missing data gave no improvement during training but we obtained useful function values when data was missing during prediction on the test set.

1.6 Conclusions with predictions using mixture densities

Advantages with doing function approximation using the mixture density model compared with a common regression technique such as backprop are:

1. The RBF representations of variable spaces are built unsupervised, which means that cheap unlabelled examples can be used.

2. The generalisation, can be dynamically improved due to the regularization capabilities of the RBFs which may decrease the requirement of cross validation.

3. Design of the network is almost parameter free as the relation between number of training examples and preferred maximum number of RBFs is simple.

4. The supervised training part is quick and gives good results using a one shot learning process.

5. The predicted mixture density for response variables gives, besides the ability to estimate a confidence interval, also the ability to detect ambiguous output values, i.e. a multi modal density function.

6. A missing value in an input sample vector is still useful as this only leads to a less specific conditioned a priori density in the missing dimension.

2 Classification with uncertainty

In Paper V we are studying a classifier producing a posterior distribution telling about the uncertainty in the classification. We are studying three types of classifiers, a *naive* classifier, BCPNN and an *optimal* Bayes classifier. The BCPNN is in this sense *semi optimal*, that is we make a joint coding of those variables which are dependent, at the moment using mutual information, but there are other approaches which may work better [Rish, 2001]. As we also indicated in section 2 mutual information may not be a good tool for data mining, and regarding classification we have the same problem concerning what may be called *outliers* (although they are not called outliers in discrete statistics [personal communication Rolf Sundberg]), where a specific joint state statistics differs very much from expected marginal statistics.

In this publication we are using a Monte Carlo related method, *i.e.* the Bayesian bootstrap [Rubin, 1981] to estimate the densities of the classification outcome. In earlier writing [Orre *et al.*, 2000] (Paper I) we discussed producing an uncertainty estimate of the classification using the Gauss' approximation formula for the variance of a function. This approach, however, is not very likely to produce good results on uncertainty estimations because the assumed Gaussian distribution, which is symmetric, is not a good model for the posterior outcome from a classification network. From empirical results using this method we often got rather small or rather large probability estimates being close to zero or one. In such cases the posterior distribution is likely to be rather asymmetric, especially when the data set is limited. With the Gaussian approximation we could therefore get standard deviations which were far outside the allowable $[0, 1]$ range.

In this study we generate artificial data with three binary explanatory variables X_1,X_2,X_3 and one response variable Y, with three states, in such a way that two of them, X_1 and X_2,

are dependent but where X_3 is conditionally independent of both X_1 and X_2 given Y, *i.e.*

$$\begin{aligned} P(X_1, X_2 \,|\, Y) &\neq P(X_1 \,|\, Y) \cdot P(X_2 \,|\, Y) \\ P(X_1, X_3 \,|\, Y) &= P(X_1 \,|\, Y) \cdot P(X_3 \,|\, Y) \\ P(X_2, X_3 \,|\, Y) &= P(X_2 \,|\, Y) \cdot P(X_3 \,|\, Y). \end{aligned} \tag{107}$$

The intention with this work is to investigate how the uncertainty in the classification may be studied for the three classification models. From one perspective the optimal Bayes should be the best, but this would also require knowledge about the complete distribution, which may be hard to achieve in a practical case as the number of training samples may be limited. In table I in Paper V the training data we used is listed. The data has been designed so that the three criteria in (eq 107) are fulfilled when using a Haldane prior. This we can assure by calculating the probabilities according to (eq 107):

$$\begin{aligned}
&P(X_1 = 0, X_3 = 0 \,|\, y_1) = 0.4 \quad P(X_1 = 0 \,|\, y_1) \cdot P(X_3 = 0 \,|\, y_1) = 0.8 \cdot 0.5 = 0.4 \\
&P(X_1 = 0, X_3 = 1 \,|\, y_1) = 0.4 \quad P(X_1 = 0 \,|\, y_1) \cdot P(X_3 = 1 \,|\, y_1) = 0.8 \cdot 0.5 = 0.4 \\
&P(X_1 = 1, X_3 = 0 \,|\, y_1) = 0.1 \quad P(X_1 = 1 \,|\, y_1) \cdot P(X_3 = 0 \,|\, y_1) = 0.2 \cdot 0.5 = 0.1 \\
&P(X_1 = 1, X_3 = 1 \,|\, y_1) = 0.1 \quad P(X_1 = 1 \,|\, y_1) \cdot P(X_3 = 1 \,|\, y_1) = 0.2 \cdot 0.5 = 0.1 \\
&P(X_1 = 0, X_3 = 0 \,|\, y_2) = 0.4 \quad P(X_1 = 0 \,|\, y_2) \cdot P(X_3 = 0 \,|\, y_2) = 0.5 \cdot 0.8 = 0.4 \\
&P(X_1 = 0, X_3 = 1 \,|\, y_2) = 0.1 \quad P(X_1 = 0 \,|\, y_2) \cdot P(X_3 = 1 \,|\, y_2) = 0.5 \cdot 0.2 = 0.1 \\
&P(X_1 = 1, X_3 = 0 \,|\, y_2) = 0.4 \quad P(X_1 = 1 \,|\, y_2) \cdot P(X_3 = 0 \,|\, y_2) = 0.5 \cdot 0.8 = 0.4 \\
&P(X_1 = 1, X_3 = 1 \,|\, y_2) = 0.1 \quad P(X_1 = 1 \,|\, y_2) \cdot P(X_3 = 1 \,|\, y_2) = 0.5 \cdot 0.2 = 0.1 \\
&P(X_1 = 0, X_3 = 0 \,|\, y_3) = 0.2 \quad P(X_1 = 0 \,|\, y_3) \cdot P(X_3 = 0 \,|\, y_3) = 0.4 \cdot 0.5 = 0.2 \\
&P(X_1 = 0, X_3 = 1 \,|\, y_3) = 0.2 \quad P(X_1 = 0 \,|\, y_3) \cdot P(X_3 = 1 \,|\, y_3) = 0.4 \cdot 0.5 = 0.2 \\
&P(X_1 = 1, X_3 = 0 \,|\, y_3) = 0.3 \quad P(X_1 = 1 \,|\, y_3) \cdot P(X_3 = 0 \,|\, y_3) = 0.6 \cdot 0.5 = 0.3 \\
&P(X_1 = 1, X_3 = 1 \,|\, y_3) = 0.3 \quad P(X_1 = 1 \,|\, y_3) \cdot P(X_3 = 1 \,|\, y_3) = 0.6 \cdot 0.5 = 0.3
\end{aligned}$$

and in a similar way for X_2,X_3 but for X_1 and X_2 we get *e.g.*

$$\begin{aligned}
&P(X_1 = 1, X_2 = 1 \,|\, y_1) = 0.175 \quad P(X_1 = 1 \,|\, y_1) \cdot P(X_2 = 1 \,|\, y_1) = 0.2 \cdot 0.4 = 0.08 \\
&P(X_1 = 1, X_2 = 1 \,|\, y_2) = 0.160 \quad P(X_1 = 1 \,|\, y_2) \cdot P(X_2 = 1 \,|\, y_2) = 0.5 \cdot 0.5 = 0.25 \\
&P(X_1 = 1, X_2 = 1 \,|\, y_3) = 0.100 \quad P(X_1 = 1 \,|\, y_3) \cdot P(X_2 = 1 \,|\, y_3) = 0.6 \cdot 0.2 = 0.12
\end{aligned}$$

We focused on the classification of $P(Y \,|\, X_1 = 1, X_2 = 2, X_3 = 1)$ with the outcome as probability densities for the three classes y_1,y_2 and y_3. Starting with the optimal Bayes classifier in section 4.1 we can see from figure 1 that the classification is rather uncertain. In the case of the optimal Bayes classifier we didn't need to simulate, the posterior distribution could be directly calculated and inspected. What we see in figure 1 is actually the marginals to the $Di(3, 4, 0)$ density. We used an improper *Haldane* prior $Di(0, 0, 0)$ which is why the distribution for y_3 is degenerated and is not drawn in the diagram. The expectation values are as follows

$$\begin{aligned}
\hat{P}(y_1 \,|\, X_1 = 1, X_2 = 1, X_3 = 1) &= \frac{3}{7} = 0.429 \\
\hat{P}(y_1 \,|\, X_1 = 1, X_2 = 1, X_3 = 1) &= \frac{4}{7} = 0.571 \\
\hat{P}(y_1 \,|\, X_1 = 1, X_2 = 1, X_3 = 1) &= \frac{0}{7} = 0.000
\end{aligned}$$

The corresponding results from the bootstrap simulation for the naive Bayes classifier is shown in figure 2. The first value is the normalised ($\sum y_j = 1$) result of exact calculation and the value within parentheses is the value estimated from the simulation

$$\hat{P}(y_1 \,|\, X_1 = 1, X_2 = 1, X_3 = 1) = 0.340 \quad (0.344)$$
$$\hat{P}(y_1 \,|\, X_1 = 1, X_2 = 1, X_3 = 1) = 0.532 \quad (0.540)$$
$$\hat{P}(y_1 \,|\, X_1 = 1, X_2 = 1, X_3 = 1) = 0.128 \quad (0.126)$$

Finally the normalised outcomes for the BCPNN classifier where we have joined the two dependent variables X_1, X_2 into one variable

$$\hat{P}(y_1 \,|\, X_1, X_2 = [1, 1], X_3 = 1) = 0.625 \quad (0.619)$$
$$\hat{P}(y_1 \,|\, X_1, X_2 = [1, 1], X_3 = 1) = 0.286 \quad (0.293)$$
$$\hat{P}(y_1 \,|\, X_1, X_2 = [1, 1], X_3 = 1) = 0.089 \quad (0.088)$$

What is interesting to note here is that the BCPNN outputs are rather different from the outputs obtained from both the optimal and the naive classifier. The BCPNN classifier is, however, the one which has utilised the data set in an optimal way. The naive classifier has the most narrow density, but on the other hand, it has violated the independence criterion. For BCPNN we have assured that the probability criteria in (eq 107) are fulfilled, why we claim that the estimates obtained from the BCPNN classifier are also the most reliable ones.

Here we have used the Haldane prior for the calculations. The main reason was that by using this prior it was easier to design the data set to fulfil the independence criteria (eq 107). Just to be able to compare how the three different priors we have discussed earlier in this text affects the result of the posterior mean estimate we here calculate the *pme* for Haldane's prior, Jeffrey's prior and the Laplace/Bayes flat prior. The vector $\boldsymbol{e}$ refers to $X_1 = 1$,$X_2 = 1$,$X_3 = 1$.

		Haldane	Jeffreys	Laplace
optimal	$\hat{P}(y_1 \,\|\, e)$	0.429	0.412	0.4
	$\hat{P}(y_2 \,\|\, e)$	0.571	0.529	0.5
	$\hat{P}(y_3 \,\|\, e)$	0.000	0.059	0.1
naive	$\hat{P}(y_1 \,\|\, e_1, e_2, e_3)$	0.340	0.344	0.347
	$\hat{P}(y_2 \,\|\, e_1, e_2, e_3)$	0.532	0.537	0.540
	$\hat{P}(y_3 \,\|\, e_1, e_2, e_3)$	0.128	0.119	0.113
BCPNN	$\hat{P}(y_1 \,\|\, [e_1, e_2], e_3)$	0.625	0.609	0.597
	$\hat{P}(y_2 \,\|\, [e_1, e_2], e_3)$	0.286	0.287	0.290
	$\hat{P}(y_3 \,\|\, [e_1, e_2], e_3)$	0.089	0.104	0.113

Table .2: Expectation values for the three different classification posteriors using three different priors. Observe that Haldane's prior gives an improper posterior for $\hat{P}(y_3 \,|\, e)$.

2.1 Conclusions about classification with uncertainty

The intention here was to study how the uncertainty and also the precision in the classifier output can be reduced by combining dependent explanatory variables. Here we designed the training data to fulfil the independence criteria but for real data we can not expect to perfectly fulfil the independence criteria as we did here. Our approach in earlier work has so far been to use the mutual information between the input variables as a criterion for joining variables into new ones. There are, however, other approaches in work to do this in *e.g.* [Rish, 2001] where *information loss* seems to be a useful criterion.

The conclusion we draw from this work is that the Bayesian bootstrap may be used to study imprecision in the Bayes classifier. From this study we got indications that we can achieve a significant improvement in the result in the classifier performance using BCPNN compared with the optimal as well as the naive Bayes classifier. Both to find the best estimated expected classifier outcome as well as getting a predicted value with an uncertainty as narrow as possible without any coarse assumptions about variable independence, as is done in the naive classifier.

5 On temporal modelling

When we deal with real world data we often find that time plays an essential role in the modelling of processes. Time may enter as a sequence of events, where a state machine may be a good model, or time may enter as delays between different events. Often a combination of states and delays may be the case. We find temporal sequences of events in areas like speech recognition, speech generation, movement control, forecasting of economical time series etc. Let's take an area like speech recognition, there we have several levels of temporal processing needed. At the lowest level we have sequential time varying mixtures of sounds at different pitch and amplitude. At the next level we have *phonemes* where each phoneme can be seen as a component composed of such time varying mixtures but where the information content is invariant to absolute pitch and absolute pitch to some extent. Phonemes are then combined into sequences expressing words and sentences. When these phonemes are combined into more complicated chunks there is also a modulative transition going on between each phoneme like a *spline* function in numerical applications, to create a continuous time varying sequence. The recognition of speech is much more complicated than the generation because there are so many levels of ambiguity. The phoneme decoding needs *segmentation* to tell where one phoneme ends and the next starts. Probabilistic context dependent inference need to be done to determine which phonemes are perceived. The phonemes should then be combined into word chunks which also need to be segmented to tell where one word ends and the next starts. There may also be a lot of missing phoneme data which need to be filled in.

Further on can often one series of phonemes be interpreted as several different words, so probabilistic inference is needed also on this level. This time together with semantic (intelligent) analysis to determine the actual meaning of the sentence.

In this section we will, however, not deal with actual speech decoding. The problems we are studying here are much more fundamental. One problem is to recognise a sequence of events and respond to that sequence with a continuation of that sequence. The other problem is to automatically segment a continuous sequence of events. We do both our studies using artificial data sequences.

1 Sequential association

In Paper VI the fundamental problem being dealt with is learning/recognition of simple sequences. Assume that we have some kind of temporal associative memory, which can be seen as a generalisation of a spatial associative memory, where we can store sequences like:

```
''One sequence of letters''
''Another sequence of letters''
''A third sequence of characters''
```

Now we can imagine that we are able to recall these sequences by giving, *e.g.* the start of each sentence. It may be clear that given *e.g.* the following starting "tags":

```
''On.....''
''An.....''
''A t.....''
```

we will have no problem of recalling up to:

```
''One sequence of ?.....''
''Another sequence of ?.....''
''A third sequence of ?.....''
```

Now, the continuation of these sequences will depend on the desired properties of the network due to the requirements from the application. If the network was designed to remember rather long sequences then it may manage to come up with the "correct" continuation here, otherwise it may instead choose the most probable one `''letters''` or, for instance, give as result $P(\text{"letters"} \mid \text{history}) = 1/3$ and $P(\text{"characters"} \mid \text{history}) = 2/3$.

1.1 A simple model for temporal association

Let us first assume that we have available a device which we could consider being an *ideal static associative memory* available, as defined in [Kohonen, 1988]

(i) *An ideal autoassociative memory is a system which holds copies of distinct input signal sets* $x^{(p)}, p = 1, 2, \ldots, k$ *in its internal state, and produces the copy of a particular set* $x^{(r)} = (\xi_1^{(r)}, \xi_2^{(r)}, \ldots, \xi_n^{(r)})$, $r \in 1, 2, \ldots, k$ *to the outputs, whenever (in the recall mode) the inputs are excited by a set of signals* $x = (\xi_1, \xi_2, \ldots, \xi_n)$ *in which a specified subset of the values* ξ_i *matches with the corresponding subset of the values* $\xi_i^{(r)}$.

If such a device was available, then we could extend this definition to an *ideal sequential memory* as:

(ii) *An ideal sequential associative memory holds copies of sequences of instantaneous patterns, defined as in (i),*

$$x^{(p,s)}, p = 1, 2, ..., k; s = s_0, s_1, ..., s_n$$

in its internal state, where s is an implicit state number, and produces a copy

$$X^{(r,s)} = x^{(r_0,s_0)}, ..., x^{(r_i,s_i)}, ..., x^{(r_j,s_j)}, ..., x^{(r_m,s_m)}$$

of a particular stored sequence of instantaneous patterns, whenever, in recall mode, the network is stimulated with a sequence of instantaneous patterns

$$Y^{(u,s)} = y^{(u_0,s_0)} ... y^{(u_i,s_i)} ... y^{(u_j,s_j)}$$

where, in a specific subset of the sequence $Y^{(u,s)}$ *each member* $y^{(ui,si)}$ *matches a specific subset of each member* $x^{(r_i,s_i)}$ *according to (i), in the sequence* $X^{(r,s)}$.

In a sequential memory it is only the order of the states of the instantaneous patterns or events which matters, time is not important. In Paper VI , section 2 (iii) we extend the

definition also to an *ideal temporal associative* memory where also *time* is important. That is, how long time each instantaneous pattern has been present would also affects the recall rate of the rest of the recalled sequence. The report in Paper VI is mainly dealing with sequential memory, so we will focus on that problem here.

We don't have such a device as an ideal associative memory available, even though such a device could in principle be implemented by lookup tables. We are, however, not interested in implementing this device by using lookup tables. The goal with our work is to make something suitable for continuous processing of information, continuous learning and adapting. Lookup tables could be used to study the properties of such a device, but they would most likely not be interesting from understanding what we may call intelligent processing of information. For this study of sequential memories we choose to use an associative memory implemented as a recurrent Bayesian neural network. Such a network, which was studied both theoretically and empirically in [Lansner and Örjan Ekeberg, 1985] has properties suitable for an associative memory. It also has learning capacities far beyond the original Hopfield [Hopfield, 1982] model, if some assumptions are made about the size of the activities of the patterns stored compared to the size of the network. In [Lansner and Örjan Ekeberg, 1985] it is found that the number of patterns that can be stored in such a network is Z_{max} which is defined in (eq 66) and has its optimum when the number of units active in each pattern is around 1.6%. In this specific application we used an incremental learning model [Örjan Ekeberg, 1989] for the Bayesian neural network. This learning rule has later been improved [Sandberg *et al.*, 2002]. The reason for using an incremental learning rule here is that a network with incremental learning can also forget and does not suffer from drastically changed learning capacity when reaching its maximum capacity. The probabilities are estimated by using exponential convolutes and are therefore good estimates both for stationary and non-stationary processes. The learning rules can briefly be explained as below, where $S_j^{(n)}$ is sample value for unit j when nth pattern is presented and $P_{ij}^{(n)}$ is the joint probability for unit i and j to be simultaneously active. The time constant τ is is chosen large enough to smooth out short term variations but short enough to follow non stationary processes.

$$\tilde{P}_j^{(n+1)} = \tilde{P}_j^{(n)} + \frac{S_j^{(n)} - \tilde{P}_j^{(n)}}{\tau} \qquad \tilde{P}_{ij}^{(n+1)} = \tilde{P}_{ij}^{(n)} + \frac{S_i^{(n)} \cdot S_j^{(n)} - \tilde{P}_{ij}^{(n)}}{\tau} \quad (108)$$

1.2 Sequence memory with delayed connections

To implement a sequential memory we added delay connections to the stimulus, thus converting the temporal stream of data to a spatial pattern. Here we assume that the rate of the stimulus is slow, compared with the relaxation time of the network. In figure 8 this principle is schematically illustrated.

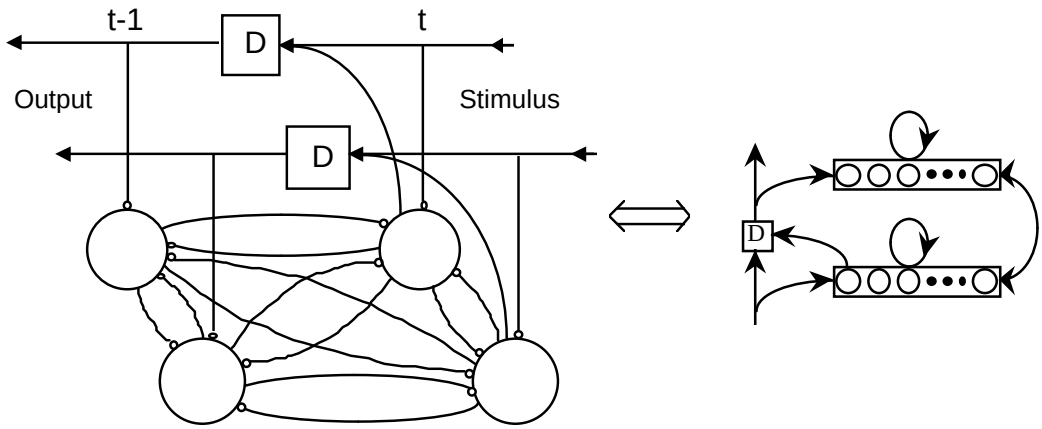


Figure 8: The principle for temporal to spatial conversion. The left figure shows a fully connected associative network, here represented by 4 neuronal units. A part of the network will see a delayed replica of the input signal (t-1). Outputs from some of the units will be mixed with the input signal. The right figure shows the same in a more formalised way. A network population is represented here by a rectangular box. An oval with an arrow shows that the population is recurrent. An arc binding two network populations together means that all units in one population are projected on all units in the other population in the direction of the arrows.

With the arrangement as in figure 8 we can make a predictor, where $s(t)$ refer to stimulus at time step t, as

$$\hat{P}(t, t-1, \ldots, t-k) = \mathcal{F}(s(t), s(t-1), \ldots, s(t-k), \hat{P}(t), \hat{P}(t-1), \ldots, \hat{P}(t-k)). \quad (109)$$

The arrows going from the neurons in the recurrent network back to the delay lines in figure 8 illustrate how the output from a recurrent network is mixed with the stimulus information. This is to deal with missing data in the stream of stimulus and corresponds to the $\hat{P}(t), \hat{P}(t-1), \ldots, \hat{P}(t-k))$ taking part of the predictor function in (eq 109). It should be noted here that $\hat{P}$ is not considered to be a conditioned probability. It should be seen as a predictor function expressing the belief of the output value at time step t as a functional value of the current and earlier time steps and predictor outcomes. The number of time steps we can deal with, *i.e.* $t \ldots t-k$ we here refer to as the *context* length.

1.3 Study of weight matrix invariant properties

It is clear that the weights in a recurrent neural network where a temporal sequence of data is expressed as a spatial pattern should show some invariance properties. In Paper VI section 5 we study the invariance properties for this network going from from the fully connected matrix as in figure 9, where the number of connections grows with the square of the context length, to the most reduced one as in figure 10.

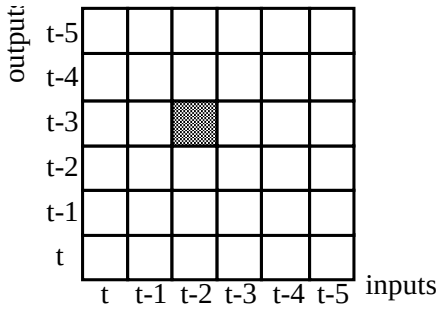


Figure 9: The full connection matrix which connects different time steps with each other. The elements of this matrix are also matrices which make the network at each time step recurrent. The interpretation is that the outputs from time step t-3 connects to inputs at time step t-2 and so on.

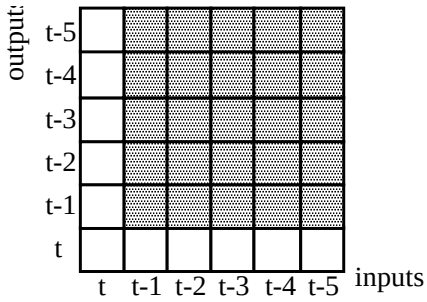


Figure 10: Assuming that the multiple occurrences of equal weight sets over the diagonals are redundant, we may simply remove them (grey). Thus the total amount of weights will grow linearly with the context length.

If we could do the removal operation as being demonstrated in figure 10 we would get the following relation between the total number of weights W'_{tot}, the context length c and the number of units to express an instantaneous pattern n as

$$W'_{tot} = (n^2 - n) \cdot (2 \cdot c + 1). \tag{110}$$

Some results indicating how successful this removal of redundant weights actually were is shown in figure 8 in Paper VI . The correct recall rate for the network where the number of weights were reduced as in figure 10 is compared with the recalled rate for the full connection matrix. What is interesting to note here is that the network with reduced weights did not perform worse and in some cases even performed *better* than the network with full connection matrix.

1.4 Left-right context and compressed history

Normally when we think about sequences of data we consider the stream of data to be ordered in such a way that the first item is presented first, then the next and so on. This may give rise to ambiguities in the recalled continuation. As is given as example of this in section 6 in Paper VI . Assume that we have learned the following sequences:

```
S1 = MATHEMATICS
S2 = MATERIALLY
S3 = MATRICULATE
```

When the network is then stimulated with *e.g.* "MAT" it will not be able to unambiguously recall the rest of the sequence. The network was in this case setup to choose between one of the patterns, in this case it chose "MATRICULATE", but it could also have been setup to generate all three patterns with different probabilities, as we discuss more about in Paper VIII . However, if the network had missed the beginning of the word and just was presented with "".....ICS" then it would recall "MATHEMATICS". The temporal network is then using, what we call here, the *right-context* to recall the history, or may change an earlier decision when additional data is available. For a sequence generating network this *postdictor* is a property that can, for instance, prevent noise in the inputs to give errors in the outputs. Experiments like these were actually performed on human beings for the English language [Shannon, 1951], where it was found that the predictor is somewhat better than the postdictor, *i.e.* it is somewhat easier to recognise a word from its beginning than its ending.

Finally, in Paper VI we studied how the context length could be improved for a predictor by making old information more *coarse*. This would remind about the so called Weber's law (stated by E. H. Weber in 1834) [Kandel and Schwartz, 1985] in biological systems where the difference in sensation is proportional to the the magnitude of the stimulus ($k = dS/S$) making the actual sensation logarithmic to the absolute size of the stimulus. A hypothesis about temporal resolution could then be stated in a similar way

$$\Delta Resolution = k \cdot \frac{\Delta T}{T} \tag{111}$$

which would give a resolution which is a logarithmic function of distance in time. This kind of logarithmically decreasing resolution over time we have, however, as far as we know, seen no experimental evidence of in biological systems. Our approach in Paper VI was to sum the history over time over a number of steps where the number of steps grew exponentially over time as is illustrated in figure 11.

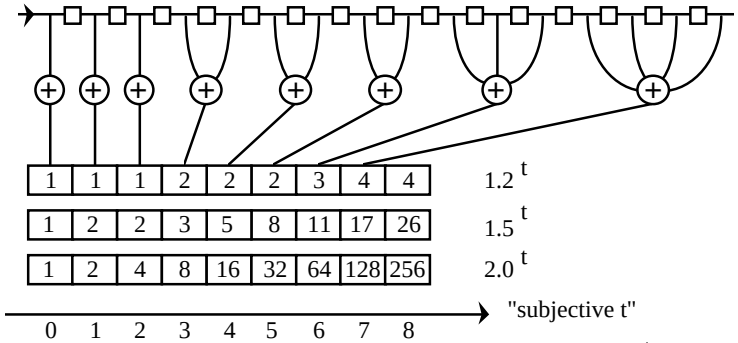


Figure 11: The principle for coarse summation of propagated stimuli/outputs to obtain a logarithmic-like time resolution. The number of sum steps is an exponential function of the "subjective" time, here with bases 1.2, 1.5 and 2.0.

The results we obtained from these investigations, showed that we could utilise the weights in the network much more efficiently by using a reduced connection matrix (due to invari-

ance) and coarse summation, Figure 15 in Paper VI shows that the network with reduced weights performed almost identical to the fully connected network and the network using coarse summation was just slightly worse. It may be surprising that the theoretical result based upon static associative memories [Lansner and Örjan Ekeberg, 1985] was almost the worst, but this may be the expected one considering the better weight utilisation due to time invariance properties of the weight matrix.

Finally, in figure 12 we include the results were we look upon the recalled information per weight. This information we defined as

$$I = S \cdot (\log_2(S_{possible}) - \log_2(R_{possible})) \tag{112}$$

where the total number of information bits I is the logarithmic difference between the possible number of input sequences ($S_{possible}$) and the number of sequences possible to recall ($R_{possible}$) times the number of stored sequences (S). This is based upon corresponding calculations for a static associative memory [Lansner and Örjan Ekeberg, 1985].

1.5 Conclusions about temporal associative memory

Although this study about temporal associative memories is far from any ready to go method it gave useful insights into some properties which may be important when implementing a temporal associative memory. We have seen that the invariance properties of the weight matrix due to the same pattern being repeated over the network in many different positions can reduce the actual number of weights dramatically. The coarse summation of historic data gives yet another possibility to handle long temporal context with efficient utilisation of the weights.

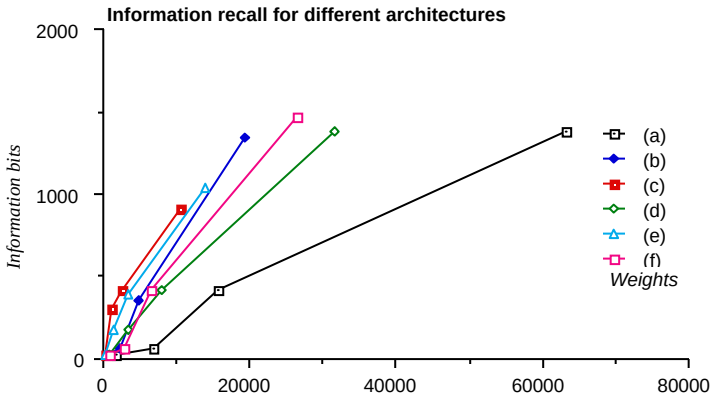


Figure 12: Capacity as recalled information per weights for some temporal network models with different connectivity principles. (a): Fully connected recurrent network. (b): Recurrent network with reduced connectivity due to symmetry. (c): Network with feed forward temporal information only. (d): Mixed recurrent, fully connected with coarse feed forward summation (e): Mixed recurrent, reduced connectivity, with coarse feed forward summation. (f): Feed forward only, as in (c), but with 10 delay steps.

2 Temporal Segmentation

Now, let us imagine that you are going to build a system for automatic speech recognition, but you can't tell the system in detail about all the words and delimiters yourself. As you are certainly aware of there are no delimiters in natural speech, so the system have in some way to find these on its own. So, instead of telling the system what to learn and what to look for we just present large chunks of data for it and let the system decide on its own how to best find where a phoneme or word starts. Yes, both phonemes and words share the same problem. As all classification tasks it is a hierarchical issue. At one low level we have phonemes, at another higher level we have words. The issue of finding these parts of the patterns is the segmentation problem. The segmentation problem is fundamental in pattern recognition tasks as continuous speech recognition and computational vision. We wanted to find a simple method that could perform automatic segmentation on any temporal sequence, as the one referred above contains at least two classification levels we tried this method on a toy problem instead. In this case we used words from a computer dictionary that were put together in random sequences of words.

The goal with the work which is presented in Paper VII thus was to detect higher level items in an unlabelled sequence of data. Given data with a sequential/temporal behaviour this shows up as the temporal chunking problem [Grossberg, 1984] which may be illustrated by the example:

`thisisacontinuousstreamofdatawhichispossibletoreadwithoutseparators`

Here, we want unfamiliar lists of familiar items (characters) presented sequentially to be recognised as new items (words). In the first place just the characters are familiar. When we have seen different lists several times we will also recognise the words as familiar items. The method presented here detects segmentation points between words. Conceptually this means that we have grouped a sequence of elementary items into a new, composite item. The principle is illustrated figure 13. The segmentation network gives a signal when the most likely segmentation point has been found. This signal could for instance be used to trigger a storage or perception network as is illustrated in figure 14.

The proposed method uses a Bayesian learning scheme in the form of a temporal associative memory earlier investigated where the weight between two units w_{iq}, and thus the relaxation scheme, is modified with a few extra parameters, a pair-wise correlation threshold p_θ and a pair-wise conditional probability threshold p_σ and a noise limiting threshold p_ν:

$$w_{iq} = \begin{cases} 0 & p_i < p_\nu \vee p_q < p_\nu \\ -log\ C_\theta & p_{iq} < p_\theta \\ -log\ C_\sigma & p(q|i) < p_\sigma \\ log\frac{p_{iq}}{p_i p_q} & otherwise \end{cases} \tag{113}$$

The hypothesis we intended to investigate was that weights w_{iq} (eq 113) within words would get "stronger" and weights between words would get weaker and possibly *inhibitory*. This weakening or inhibitory effect between words would get stronger the more often different words are seen in combinations with each other. We assumed that by introducing some thresholds as p_θ and p_σ as in (eq 113) we could possibly enhance the segmentation performance. We also introduced a *segmentation algorithm* which could *vote* for the most likely segmentation point.

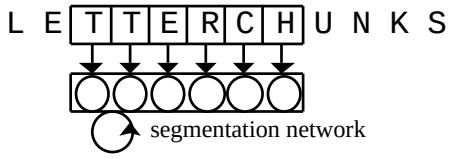


Figure 13: A continuous stream of data passes a tapped delay line that spreads the temporal information spatially over the network.

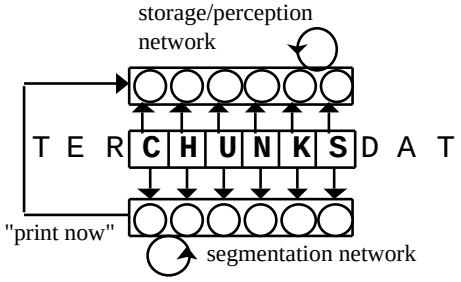


Figure 14: The segmentation network may signal "print now" to the actual storage or perception network when a segment start is found.

As a brief example on how the segmentation algorithm, which is being shown in section 5 in Paper VII , works let's assume that we learn the sequences "NEURALJUNIOR", "NEURALEXPERT", "BAKERJUNIOR" and "BAKEREXPERT". Each one of the words "NEURAL", "JUNIOR", "EXPERT" and "BAKER" is twice as frequent as any combination of these words. Each sequence is learned at every position on the network. Weights are calculated as in (eq 113) but the thresholds p_θ and p_ν are set to a low value from the beginning. In the example in the table below a "_" means low output activity. After learning we may stimulate the network according to #1 below. After each such stimulation we do a relaxation of the network and get outputs as in #2. By raising the p_θ somewhat, here to $p_\theta = 0.05$, we get outputs as in #3. When we look upon step #3 in the table we see that we, in this case when we have adjusted p_θ, will get a positive "vote" for the segmentation point to be between "BAKER" and "EXPERT" for every position of the stimulus at each of the time steps "t0" $\cdots$ "t3", this type of consensus may not always be the case.

ex	t0	t1	t2	t3
#1	BAKERE	AKEREX	KEREXP	EREXPE
#2	BAKERJ	AKEREX	KEREXP	EREXPE
#3	BAKER_	AKER__	___EXP	__EXPE

In the example given above we had learned too few combinations to be able to give an unambiguous decision about the segmentation point without adjusting the threshold p_θ. To illustrate that the segmentation method doesn't really need the adjustment of p_θ, when we have examples many enough, we show the diagram in figure 15 where we plot the correct percentage of recall vs the number of random combinations learned.

2.1 Conclusion about temporal segmentation

The method we investigated here was able to successfully find start and end positions of words in an unlabelled continuous stream of characters in an unsupervised way. The network's robustness against noise during both learning and recall was studied. We got clear indications that the p_θ level of the threshold for the pair wise weight w_{iq} was the most efficient one to improve the segmentation performance compared to p_ν when the number of

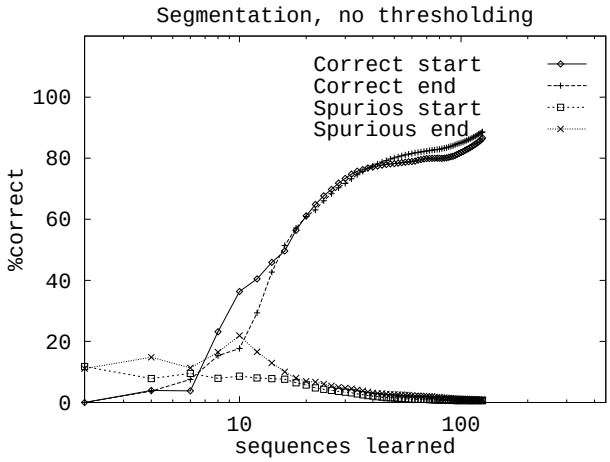


Figure 15: Random combinations of 17 words are learned and segmented without using weight thresholding. Upper curves show percentage correct segmentation and lower curves show percentage spurious segmentation.

word combinations seen by the network was limited.

6 On finding new patterns in data

The essence of data mining is to find new patterns in data. Patterns which may help us to build new concepts and knowledge. In [Hand *et al.*, 2001] the concept of *data mining* is defined as

> Data mining is the analysis of (often large) observational data sets to find unsuspected relationships and to summarise the data in novel ways that are both understandable and useful to the data owner.

In the previous sections we have seen examples on how a recurrent neural network was used for *supervised* pattern recognition of temporal sequences in Paper VI and an example on *unsupervised* pattern recognition when we approached the temporal segmentation problem in Paper VII . In this section we will focus on Paper VIII which deals with the problem of finding new patterns in data in an unsupervised way.

1 What does "pattern" mean?

When we consider what *pattern* really means we may find that the term as such is not very well defined. Below are a few definitions that can be found in some text books on the subject.

[Bishop, 1995] "*pattern recognition* deals with problems like speech recognition, classification of handwritten characters, fault detection in machinery and medical diagnosis."

[Fayyad *et al.*, 1996] "a pattern is an expression E in a language L describing facts in a subset F_E of F. E is called a pattern if it is simpler than the enumeration of all facts

in F_E." For example, "If income $< \$t$, then person has defaulted on the loan" would be one such pattern for an appropriate choice of t.

[Looney, 1996] there is not a single definition of what means a pattern, but examples are given as voice, handwriting, fingerprints, images, *i.e.* something that can be *recognised* as speech, written characters, scenes in images and so on. The concept of *classification* involves learning of likeness and differences of patterns that are abstractions of instances of objects in a population of non-identical objects. It is also said that "The associations between patterns and their causes are the bricks from which the wall of scientific knowledge is built."

[Ripley, 1996] there is not a stringent definition of pattern but a citation "It is felt that the decision-making processes of a human being are somewhat related to the recognition of patterns; for example the next move in a chess game is based upon the present position on the board, and buying or selling stocks is decided by a complex pattern of information. The goal of pattern recognition research is to clarify these complicated mechanisms of decision-making processes and to automate these functions using computers. However, because of the complex nature of the problem, most pattern recognition research has been concentrated on more realistic problems, such as the recognition of Latin characters and the classification of waveforms." (from [Fukunaga, 1990])

[Theodoridis and Koutroumbas, 1998] patterns as such are not explicitly defined here, instead we read "*Pattern recognition* is the scientific discipline whose goal is the classification of *objects* into a number of categories or *classes*. Depending on the application, these objects can be images or signal waveforms or any type of measurements that need to be classified. We will refer to these objects using the generic term *patterns*."

[Britannica, 1998] tells us "Pattern recognition is an information-reduction process: the assignment of visual or logical patterns to classes based on the features of these patterns and their relationships. The stages in pattern recognition involve measurement of the object to identify distinguishing attributes, extraction of features for the defining attributes, and assignment of the object to a class based on these features." To be able to look this up we had to, however, make a classification of the word already, as belonging to the area of *information processing, image analysis*.

[Fayyad *et al.*, 2002] "a pattern is a parsimonious [8] description of a subset of data points."

The most parsimonious description is clearly the one found in [Fayyad *et al.*, 2002] but with these definitions above we can see that the term *pattern* may refer to both the *observed data object* (image, speech or data collected for some purpose) or the *model representation* (rule, dependency graph, cluster, tree structure, discriminatory function). In everyday language we would probably find that "pattern" mostly refers to the *data object* and in scientific data analysis merely to the *model representation*.

1.1 Our aim with this application

In Paper VIII we are searching a big database of adverse drug reactions, where drugs and adverse reactions are stored as binary true/false events. Our aim is to generate hypotheses

[8] parsimonious=simplest possible or most compact, compare with *Occam's razor*

about combinations of adverse reactions which may represent drug induced syndromes. A *syndrome* is a group of signs and symptoms that occur together and characterize a particular abnormality. The database is sparse in that sense that only a few (usually less than six) adverse reactions out of a possible 2000 may occur on a single adverse drug reaction report. Further on we may assume that the data is highly incomplete, *i.e.* it is unlikely to find any report covering the whole syndrome picture. Those reports which are not part of a syndrome are from this application point of view considered to contain noise and those adverse reactions which occur on reports which are part of a syndrome but do not participate in the symptomatic picture of a syndrome are also from this point of view considered to be noise.

We are using a one layer recurrent BCPNN (see figure 16) where each neuron of the network represents an adverse reaction. The BCPNN is trained on the adverse reactions from those reports where the drug being investigated appear as a suspected drug. The π_j is the output value from neuron j which is the *belief* (in the range $[0 \cdots 1]$) for this neuron to be on. The network performs a probabilistic inference during the recall phase and finds patterns where the belief value π_j for each neuron reaches stability when it is conditioned on the belief values of all the other nodes, that is when

$$\pi_j^+ = P(\pi_j \,|\, \pi_1, \cdots, \pi_K) \tag{114}$$

has converged to a fixed point for all units in the *relaxation* procedure. Each π_j is initialized to the activity value of the corresponding unit in the *stimulus* pattern. Each group (or pattern) of true (active) nodes (where $\pi_j > 0.5$) we find is considered to be a hypothesis that this group of adverse reactions may constitute a syndrome. These hypotheses are then dealt with by pharmacological and medical experts.

As the network we are using is a Hopfield network with graded units [Hopfield, 1984] an energy function can be defined. The energy function in this case has the following expression

$$E = -\frac{1}{2}\sum_{i,j} IC_{ij}\pi_i\pi_j - \sum_i \beta_i\pi_i + \sum_i \int_0^{\pi_i} \log_2 x \, dx \tag{115}$$

The energy landscape corresponds to *attractors* where each attractor represents a pattern. Often this type of recurrent networks have been used as associative memories where a part of the training pattern is used as a cue (or stimulus). The stimulus causes a a subset of the π_j units to be initialized somewhere at a (usually) reachable distance away from the closest energy minimum. The network will then, during it's relaxation process, find the local minimum of the attractor, *i.e.* the learned pattern.

In this case, when we use the network to data mine for patterns we don't have any specific training patterns. We rely on the network weights to find the pair-wise correlations within the units. Our desire is to find the attractors in this energy landscape and we propose a procedure, which is not guaranteed to find all attractors, but at least finds a subset of them. The procedure to do this is described in section 3 below.

2 Unsupervised pattern finding

The concept of unsupervised pattern recognition is often considered similar to *clustering* [Theodoridis and Koutroumbas, 1998], which from one perspective is synonymous with the word *classification*, *i.e.* grouping data into *categories* [Britannica, 1998]. A book entitled

"Classification" [Gordon, 1981] is entirely concerned with unsupervised methods, mainly clustering [Ripley, 1996]. Clustering is about grouping *similar* or *dissimilar* (according to some metric) items into classes in a *feature* space [Looney, 1996].

Features are functions on the input space $\boldsymbol{x} = T(\boldsymbol{x}_{orig})$ transforming the input space into a new set of variables [Looney, 1996], usually with lower dimensionality than the original space. The most simple feature space to use is the original variable space x but this may be unpractical of several reasons. The dimensionality may be incredibly high, as in *e.g.* image analysis, or the variables may be *confounders* or dependent why they need to be recoded together. One good example of feature coding in this text is what is found in Paper IV and in section 4 where we code the input space x as $P(\omega_i \mid x)$, which are the features the actual classification is performed on.

To find a suitable feature space may not be trivial, as in *e.g.* computational vision, but can in many cases be solved by clustering algorithms, as when we used the EM algorithm to approximate the density of the input space in section 4. Here the pdfs constituted a graded feature space [Kohonen, 1988] but the feature space may be a hard categorisation of the input space as well. The feature space is also preferably a space of *independent* features which makes classifications done on the next level solvable by simple linear discriminatory functions. The search for features is a balance between a huge dimensionality of the input space and a huge dimensionality of the feature space. If the search for features is made too exhaustive we may end up with what is called *grandmother cells* in the neural network literature [Hertz *et al.*, 1991], *i.e.* very complex feature detectors.

When we started the study which is presented in Paper VIII , *i.e.* searching for syndromes in the WHO database, we started by searching for combinations of adverse drug reactions [Orre *et al.*, 2000]. Among these *higher order combinations* we found that what we were looking for could not be found. As an example, we were looking for combinations of the symptoms which are included in the *neuroleptic malignant syndrome* which is a known serious adverse drug effect of the drug haloperidol. However, we could not find those more complex combinations of adverse drug reactions which would constitute the complete syndrome. The reason we suspected to be the probabilistic membership of a symptom to be included in the syndrome. For each of the symptoms there is a certain probability to be included in the symptom picture, and it may be unlikely that all symptoms occur within the patient at one occasion, or it may be so that not all symptoms are detected by the physician.

The approach then was to set up a network, a recurrent BCPNN, as an unsupervised pattern finder. Train and test this network with artificial data for evaluation and then to run this on the real database to see what symptoms would be detected. This method would learn the sample vectors into an associative memory, the recurrent BCPNN, where categories are formed. From this network we could then recall patterns, which in this case were clusters, conforming to (eq 114). We compared the method, on artificial data, with a the well known *Autoclass* [Cheeseman and Stutz, 1995a] clustering algorithm.

As we mentioned above, we started looking for *higher order dependencies* among the syndromes, but this failed, however, now we understand why, and we also understand when and how to search for higher order dependencies, but more about this in chapter 6 section 4 below.

3 A proposal to find multiple patterns with BCPNN

After the recurrent BCPNN has been trained on a data set there will be attractors in the network, which in this case is defined by the fixed points (eq 114), alternatively as all the local minima to the energy function (eq 115). Our aim is to find these attractors, to be considered hypotheses of patterns reflecting some collective properties of the data set under investigation. There may be many ways to do this, where one of the most naive approaches would be to generate all possible combinations of the binary inputs as stimulus and see which of these combinations are fixed points. Our network has K binary attributes, which means that there are 2^K possible combinations of binary stimulus values. The time complexity of a thorough search through this space would thus be $\mathcal{O}(2^K)$. As an example we can consider the application we investigated in Paper VIII , where we have 1887 attributes. Assuming that each relaxation took about 1 second [9] a thorough search through all combinations would require about $\frac{2^{1887}}{60 \cdot 60 \cdot 24 \cdot 365} \approx 10^{561}$ years. A somewhat less naive approach, which may be suitable if we have a reasonable sized data set, would be to use the data samples $\mathbf{x}$ as cues. In the case with the drug haloperidol investigated in Paper VIII we only have 8902 cases which would take a couple of hours to go through. However, in the antipsychotic drug group investigated in Paper VIII we had 100083 cases which would then require around 24 hours, which is quite long compared to the time ($4.3s$) we actually used to find 12 patterns in this data set, and if we would go through the whole database used in this application, $N \approx 3 \cdot 10^6$ it would require about one month. Further on, using the data samples as stimulus cues would also be a strict classification of each of the samples into a specific class for each sample, but it is possible that a sample may belong to several classes *partial class assignment* and this may therefore, anyway, not guarantee that we will find all attractors.

As we could not guarantee to find all attractors within a reasonable search time we were looking for a pragmatic way to limit the search space. One approach we tried was to start by stimulating all nodes with one, then recall a pattern (first pattern would then likely be the largest attractor) and remove from the stimulus those units which had a belief value above 0.5 and repeat this procedure for all patterns until an empty pattern was recalled. With this approach, which worked rather well on small artificial tests sets where we often found all or almost all of the patterns, we found that the recall times in the real application ($K = 1887$) become quite long, around 30 s for a complete relaxation. We also often became stuck in an attractor, despite we had removed all the recalled units from the input stimulus.

The approach we actually used was to start with the vector of marginal probabilities $\mathbf{p}$ and linearly transform this vector to the $[0, 1]$ interval and use this as stimulus. From this stimulus we then removed (set to zero) the recalled ($\pi_j > 0.5$) units and rescaled the remaining stimulus to the $[0, 1]$ interval and again performed a recall. This procedure was repeated until an empty pattern was recalled.

This procedure will therefore first find the patterns containing the most frequent events, then continue to find those patterns containing less and less frequent events. The advantage with this procedure is that it is quick and due to the linear transform those units which only participate in background noise will be set to zero or close to zero. A disadvantage is that we may in some cases miss patterns which overlap, that is patterns who share units. This can, however, probably be dealt with, using a somewhat more thorough exploration of the stimulus space, around the patterns found.

[9] One second is a reasonable assumption as when we *e.g.* stimulated with all inputs=1 a relaxation took 30 seconds and in one of the application examples described in Paper VIII we found 12 patterns (using 12 relaxations) in 3.9 seconds.

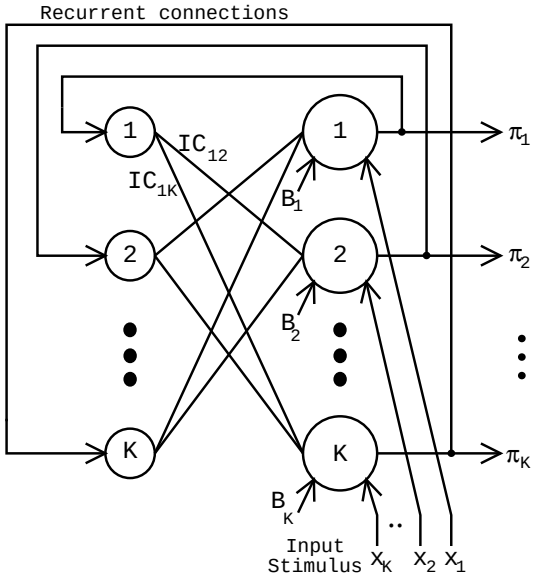


Figure 16: The Bayesian neural network we are using here is a type of Hopfield network. Due to symmetric weights, asynchronous update and no self-recurrent connections this network always finds a fixed-point solution when iterated to stability.

3.1 A comparison method

To be able to compare the performance of the recurrent BCPNN with a well known method we used *Autoclass* [Cheeseman and Stutz, 1995b, Cheeseman and Stutz, 1995a] a Bayesian clustering/classification method. This method reminds about *K-means* clustering (*e.g.* [Looney, 1996] or [Krishnaiah and Kanal, 1982]) but it uses an expectation maximisation algorithm, where it adapts the model for maximum likelihood of the data, and uses a linear Bayesian classifier to classify each sample versus each cluster centre. Autoclass uses partial class assignments, so a specific data sample can be classified as belonging to several classes with different probabilities. To get a comparable output from Autoclass with BCPNN we estimated the probability of each attribute for each class.

3.2 Results on real data

We ran both BCPNN and Autoclass on adverse reactions reported together with the drug haloperidol. This data set had 8902 cases and 1887 adverse reactions (of which only 721 reactions had actually been reported with haloperidol). From BCPNN we got 16 patterns consisting of between 4 and 17 ADR terms. All found patterns were considered to be clinically relevant by medical expertise. Five of the 16 patterns found are listed in section 4.2 in in Paper VIII . One of these patterns conforms with the symptoms of the Neuroleptic Malignant Syndrome which is a well known reaction to haloperidol and other antipsychotic drugs and is discussed in detail in section 4.2. Of the five patterns listed in Paper VIII three of them conformed well with known syndromes associated with haloperidol according to the common literature source Martindale [Parfitt, 1999]. The syndromes which were detected

are NMS (table 2), Parkinsonism (table 3) and Acute Dystonia (table 6). The pattern listed in table 4 is similar to Parkinsonism. Tardive dyskinesia and Akathisia were not detected by the method, but Tardive dyskinesia is often reported as a single adverse reaction term and Akathisia may be missed because certain adverse reactions, like restlessness and anxiety are rather unspecific and can be reported using several different terms. However, to be able to detect three of the five known syndromes for haloperidol was considered a convincing result for the method.

For the antipsychotics drugs the recurrent BCPNN generated 12 rather large patterns, with an average of 26 ADR terms per patters. These patterns, although not analysed in detail here, were considered clinically valid and coherent.

When we ran Autoclass on the haloperidol data set it produced the patterns in table 7 and table 8. These patterns seem to be mixes of several syndromes. The pattern in table 7 contains characteristics of NMS, Parkinsonism, Acute dysotonia, and Tardive dyskinesia. The other pattern in table 8 contains some of the NMS symptoms but many unrelated terms as well. The conclusion is that Autoclass did not efficiently identify any of the syndromes associated with haloperidol.

3.3 Synthetic test data

To get some qualitative knowledge of the performance for the two methods we did a comparative run on some simulated data sets with known distributions for completeness and noise. Although we don't know the actual distributions for completeness and noise in the real data set we had got good indications for BCPNN to be a useful tool in this application. We therefore generated synthetic data in a way we considered to be a rather relevant but simple model for the data set of adverse reactions. We started with two prototype patterns as shown in figure 17. These prototype patterns consist of 12 units with 4 units overlapping of a total of 81 units. On these patterns we varied the completeness level between 25 %, 50 %, 75 % and 100 %. Onto these we added noise units which were evenly distributed over the all the units which were not lit by the incomplete prototype pattern. This noise was varied between 25 %, 50 %, 75 % and 100 %. We generated 400 patterns of these with a probability of 0.5 for any of the prototypes. To these prototype samples we added 2000, 4000 and 16000 samples with random noise patterns. These noise patterns had the same number of units as the incomplete prototype including noise units to not make the number of units be a hint for any of the methods. We generated five such sets which makes a total of 240 different files to analyse and a total of 1,856,000 pattern samples in the five sets. In figure 18 we can see a few samples from this trainingset, both from the two prototype patterns at 25 % completeness and 25 % noise level as well as from the pure noise patterns.

3.4 Comparative results on synthetic data

Both the recurrent BCPNN and Autoclass showed comparable performance on the synthetic data sets. Often both methods produced perfect patterns, identical to the prototype patterns, but there were also clear differences. In figure 19 and figure 20 there are examples of the obtained patterns by using BCPNN and Autoclass respectively on the training data given example of in figure 18, *i.e.* completeness 25 % and noise 25 % with 2000 additional noise patterns added to the 400 samples from the prototypes. One of the patterns obtained by BCPNN is missing one unit but both of the patterns obtained from Autoclass have a lot of extra units. In figures 11,12,13 and 14 in Paper VIII we have visualised the average

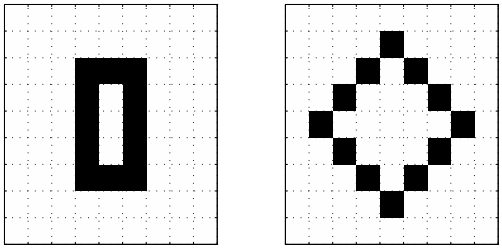


Figure 17: The two patterns used as prototype patterns. It is only for visual purposes they are organised as 9 x 9 matrices, rather than as 1 x 81 vectors; to the method, spatial configuration is irrelevant.

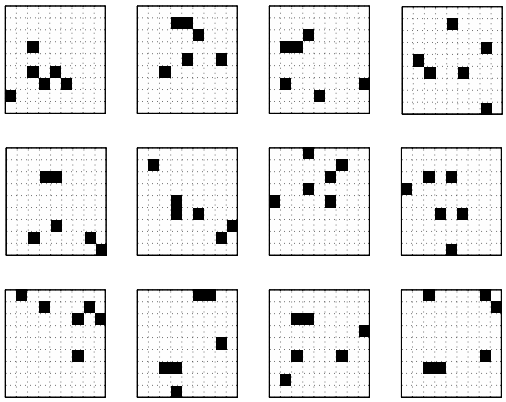


Figure 18: 12 training samples at a completeness level 25% and a noise level 25%. The top four samples are taken from the diamond prototype, the middle four from the rectangle prototype and the bottom four are pure noise. The underlying prototypes are no longer visually distinguishable.

number of correctly recalled and incorrectly recalled patterns from BCPNN and Autoclass for the different combinations of completeness and noise. From these figures we can see that BCPNN rarely produces incorrect patterns, even though Autoclass often manages better in finding the correct patterns Autoclass also produces more incorrect patterns (black in figure). From figure 14 in Paper VIII at completeness level 25 % we can see that both BCPNN and Autoclass have problems, but BCPNN prefers to not give any output at all for most cases, when Autoclass on the other hand almost only produces patterns which are incorrect at this completeness level for all combinations of noise.

3.5 Conclusive discussion about results

From the results on the synthetic data sets we can draw the conclusion that when BCPNN produces an output it seems mostly correct. Autoclass on the other hand may often produce incorrect patterns as well.

A comparison of run times (on Pentium III 1.4 GHz with 3 GB of RAM running GNU/Linux operative system) for the two methods is interesting. Both methods were implemented in the C language, except some parts (mainly data flow and data analysis) of the recurrent BCPNN which is implemented in Scheme.

Total run time for BCPNN on haloperidol data set with 8902 cases were: 3.6 seconds training time and 6.5 seconds recall time for the 16 patterns achieved. For Autoclass it took 20 hours and 24 minutes to find the two patterns in the same data set. Autoclass default settings were used.

On the synthetic data sets it took BCPNN around 4 minutes to perform training, recall and evaluation of all files, *i.e.* the total training set of 1,856,000 patterns. For Autoclass the corresponding time used was 180 hours.

The overall conclusion in a comparison between BCPNN and Autoclass on the real data sets as well as the synthetic data sets are that the recurrent BCPNN seems to be both a quick and efficient data mining tool which scales well on large data sets for this type of application.

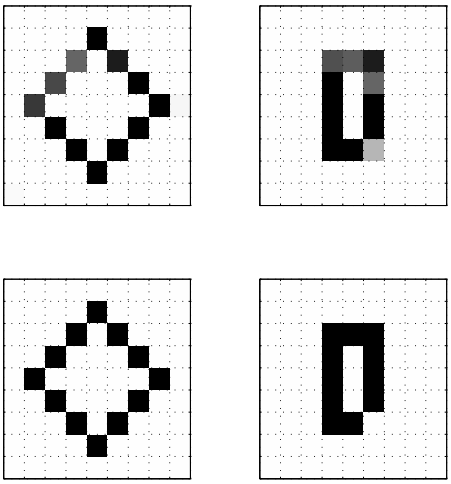


Figure 19: Sample output from the recurrent BCPNN at completeness level 25% and noise level 25%, with 2000 pure noise samples. The top row displays the raw output, and the bottom row displays the thresholded patterns.

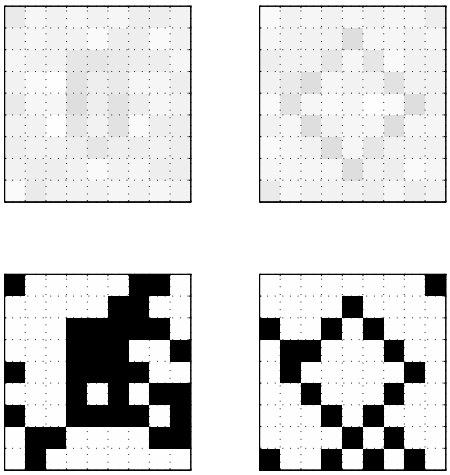


Figure 20: Sample output from Autoclass at completeness level 25% and noise level 25%, with 2000 pure noise samples. The top row displays the raw output, and the bottom row displays the thresholded patterns.

4 Approach to higher order dependencies

The information component IC_{ij} as earlier described only finds pair-wise dependencies. This also limits the clustering method using recurrent BCPNN as described earlier, when there are too much dependencies between the patterns they are hard to separate. For this we need to find higher order dependencies or *features* in the data set.

A method to do this is to generalise the IC_{ij} to higher order dependencies as IC_{ijk}, IC_{ijkl} and so on, but removing from each higher order IC the dependencies due to lower order. If the lower order dependencies were not removed, as if we would calculate *e.g.* IC_{ijk} as

$$IC_{ijk} = \log \frac{p_{ijk}}{p_i \cdot p_j \cdot p_k} \tag{116}$$

then part of this third order dependency could be predicted by the lower order dependencies

$\frac{p_{ij}}{p_i p_j}$, $\frac{p_{ik}}{p_i p_k}$ and $\frac{p_{jk}}{p_j p_k}$. The IC_{ijk} becomes [Niklas Norén, personal communication]

$$\begin{aligned} IC_{ijk} &= \log \frac{p_{ijk} \cdot p_i \cdot p_j \cdot p_k}{p_{ij} \cdot p_{ik} \cdot p_{jk}} \\ &= \log \frac{p_{ijk}}{p_i \cdot p_j \cdot p_k} - \log \frac{p_{ij}}{p_i \cdot p_j} - \log \frac{p_{ik}}{p_i \cdot p_k} - \log \frac{p_{jk}}{p_j \cdot p_j} \end{aligned} \tag{117}$$

This conforms with the generalisation of mutual information to higher orders [McGill, 1954] and [Kay and Titterington, 1999] even though the equivalence may not be easy to see:

$$\begin{aligned} I(X;Y;Z) &= I(X;Y) - I(X;Y|Z) \\ &= I(X;Y) - I(X;Z|Y) \\ &= I(Y;Z) - I(Y;Z|X) \end{aligned} \tag{118}$$

A similar treatment of higher order dependencies as in (eq 117) was also suggested in [Holst and Lansner, 1993a]. DuMouchel has extended [DuMouchel and Pregibon, 2001] his gamma Poisson shrinker method for pair-wise associations to deal also with higher orders associations in a way reminding about (eq 117). He uses pair-wise associations to predict third order associations. Let us say that A and B are two drugs and C is for example the adverse reaction kidney failure. Let us say that A and B both act independently on the kidney but as both A and B may be often prescribed together there will be many cases ABC in the database. There could, however, instead be so that the drugs A and B interact, causing kidney failure much more often than would be predicted if A and B acted independently. This latter case we would like to detect automatically. DuMouchel uses an "all-two-factor" log linear model [Agresti, 1990]. According to log-linear model theory the all-two-factor model is the canonical model to determine a joint distribution from all the two-way marginal distribution without adding any more complex information.

It should be noted that in the application we are studying here there are two different types of patterns to find, both of them can be considered to belong to *market basket analysis* which consider how frequent different items occur together. If we compare the problem to look for syndromes with the problem to look for interacting medication, which also DuMouchel discusses they are slightly different to their nature. For interacting medication the interaction do only occur when both drugs (A and B) are present, which makes this a higher order problem. For the syndrome case on the other hand, the presence of a certain adverse reaction is only probabilistic. In both [Bate *et al.*, 1998] and [Orre *et al.*, 2000] we tried to search for higher order order combinations of adverse reactions being member of a syndrome according to the principle that

$$IC(A_1, A_2, \ldots; D) = \log \frac{P(A_1, A_2, \ldots|D)}{P(A_1, A_2, \ldots)} >> 0 \tag{119}$$

In [Orre *et al.*, 2000] we found that higher order combinations, where the number of symptoms > 4, were not present at all even though we knew that the syndrome we were looking for had more symptoms in the syndrome pattern. The problem is that the membership of a specific syndrome in a specific syndrome is probabilistic. The recurrent neural network we have investigated in Paper VIII to find adverse reactions combinations is, however, good in finding these kind of probabilistic patterns. For drug-drug and similar types of interactions we are, however, continuing to develop the method as in (eq 117) above.

7 Long term goals with data mining

(This part of the thesis contains highly speculative discussions.)

In this thesis we have dealt with some aspects of data mining, *i.e.* to find dependencies and patterns in data. Algorithms and methods for these tasks are continuously improving in accuracy and speed at the same time as the hardware development, whose seemingly incessant performance increase, simultaneously allow us to utilise more and more complex algorithms. In the not too far future we may even be able to build computer hardware due to fundamentally new principles, for instance to build analogue computers which can implement analogue recurrent neural networks, with many interesting properties both in general computing, as well as in data mining tasks. According to Hava Siegelmann [Siegelmann, 1999], machines built after these principles may deal with complexity in a much more efficient way than a sequential machine, thus passing the Turing limit of computation. So, when being aware about this certainly tremendous potential of computing power we may utilise in the future, we may then also need to ask the fundamental question: What are the long term goals with data mining?

1 The hardware versus software aspect

Let us first consider improvements in hardware. For a long time the technological development has been close to exponential in performance, described by the so-called *Moore's law*. Gordon Moore (co-founder of Intel) predicted 1965 that the number of transistors which can be put on a chip will double every 18th month [Eco, 2003]. Around 1970 the predicted rate was slightly adjusted to a doubling every 2nd year instead, and since then this "rule" has been like a self-fulfilling prophecy, despite the fact that it is not connected to any physical laws. If we can keep this pace we would probably be able to (based on current knowledge and a lot of guesses about the brain) mimic the complexity of the biological computational circuitry within the human brain around 2030, when a $1000 computer may have the power of 1000 human brains and if the density of machine circuitry will continue to grow with a similar exponential rate a $1000 computer will, by 2050, be equivalent to a milliard (10^9) human brains [Ray, 1999]. Will this performance increase continue, and for how long? This we can not know of course, but in the not too distant future we may have quantum computers [Nielsen and Chuang, 2000] available. How soon we may only speculate about but researchers within the field generally agree that we will have full scale quantum computers within about 20 years [Voss, 2002], but quantum computers will give such a boost in computational power that the speed, when compared with the todays (2003) computers are close to infinite. We may then be able to build such complex software architectures that they, with today's measures, would be considered pure science fiction, unless social and political structures, like *software patents* [10] will continue to exist, as they slow down the development because software patents are always harmful as they are always too broad and companies tend to move money from research and development towards legal issues due to patents [Bessen and Hunt, 2003].

[10] Patents were originally meant to be an incentive to speed up development, but with the unfortunate possibility to patent also software and methods, due to an accidental precedent from the US Patent Office in 1974, it has become a threat which slows down the progress of software development and it is also a particular threat towards free (open source) software.

2 Information overload

This technological progress may also be seen as part of the reason why information in the world grows exponentially. This exponential growth, leading to *information overload* has been discussed by *e.g.* Paul Horn [Horn, 2000] and by Francis Heylighen [Heylighen, 1999]. The increased exchange of information is also leading to a an increased speed in scientific research and discovery, as more and more papers are online on the *world wide web* and easy to search for and obtain [Lawrence, 2001], which in turn also increases the speed of technological development. The more ideas and information being freely available, the better the utilisation of these ideas and this information for an increase in the technical development, hopefully being beneficial for all humanity. Clearly, this increase in information should lead to a decrease in relative knowledge per capita. However, it may be reasonable to expect that not all of this information is essential. In the area of science the fundamental principles do not change much with more information. Increased information merely gives a better basis for generalisation.

3 A useful application, adaptive noise cancelling

One important factor in dealing with information is the ability to discriminate between noise and actual information. Today we perform a lot of our communication by electronic mail, but as we probably all have experienced there is nowadays a lot of noise on these channels, noise induced by *unsolicited commercial emails* or *spam* as they are most often called. These *evil spammers* produce mails which often try to look like personal mails to fool you into open them, which apart from being annoying also means less efficient use of ones time as it requires some effort to discriminate between spam and real e-mails. One solution to this problem is, for instance, a Bayesian spam filter.

One of these quite successful spam filters is denoted CRM114 and is implemented by Bill Yerazunis. The name CRM114 is a somewhat jocular acronym for *Controllable Regex Mutilator*, concept 114, as it uses regular expressions which are mutated as its basic filtering principle. The name CRM114 actually originates from the movie "Dr. Strangelove", a satirical movie about the cold war between Soviet Union and USA during the 20th century. CRM114 was a fictional radio receiver designed to not receive at all, unless the message was properly authenticated. That is it discriminates between authentic messages of importance and gets rid of the rest. The spam filter starts with the prior assumption that there is a 50/50 % chance that a message is information or spam. It uses a familiar discriminatory function:

$$P(S|A) = \frac{P(A|S) \cdot P(S)}{P(A|S)P(S) + P(A|NS)P(NS)}$$

where S means spam, NS no-spam and where A is some feature of a mail. The spam filter is trained on known spam, which may have a personal profile, which is the reason why you need to train it, then after three days of training and a reasonable amount of spam you usually get rid of more than 99 % of the spam.

There are several other spam filters, like "SpamAssassin" and "Bayesspam", which were implemented by Gary Arnold, after an idea by Paul Graham. These have similar functionality to CRM114.

4 Dealing with more information, getting smarter

In physics we have for long time had the goal about finding the complete theory for everything within the universe [Hawking, 1989] and Stephen Wolfram even claims to have a hypothesis and a model about how the world is made up of mathematics [Wolfram, 2002], a view which is conformant with the so called *idealistic* philosophical view. (The term *idealism* in this sense was first used by Leibniz 1702 [NE, 2001]). What is driving us to make scientific discovery we don't really know, even though it is clear that we want to be able to describe and understand the things around us, we have a built-in curiosity. It may at first be discouraging to realize that we are put in a kind of dilemma here. We want to understand more, we therefore want to discover more, exchanging ideas and knowledge, we create technology which enhances our way of exchanging information, which also create new information At the same time all this new information and knowledge leads us to suffer from information overload [Kimble *et al.*, 1998] because we simply can not cope with all this information. What would be a suitable way to get out of this dilemma, to possible get us there were a single human being could know almost everything that could be known or merely "worth" to be known [Heylighen, 2000]?

Apart from specific data; like all different species of animals and plants, all possible phone numbers, all street names, all known chemical structures etc., which we can catalogue and often store in a database efficiently where such facts can easily be looked up; in science and technology the same principles may be applicable over and over again, just in various forms, using different notations, different names etc. Disparate scientific disciplines may come up with similar findings, but we are not able to easily generalise between these because of the diversification of terms and notations. With the help of data mining we may be able to bring some order into this apparent *chaos*. It is, however not enough to just search for patterns and coincidences. When we process huge amounts of data in order to find patterns and connecting principles, we may quickly arrive to such amounts of new information that the post processing of this is undoable for a human mind. We need to find methods to automatically reason with patterns, to be able to automatically state hypotheses, which can automatically be tested, to generate proofs and make conclusions. The ideal data mining utility would, from this point of view, also be the perfect interdisciplinary researcher, to bridge over languages, terminologies, principles and scientific disciplines. Such a tool would help us become smarter and possibly help us with abstract modelling and understanding of complex phenomena at the same time.

5 Beyond pattern finding

Earlier in this thesis we touched upon the general problem of finding patterns. In the applications we are dealing with, for instance to detect adverse drug reaction syndromes, there are many potentials to investigate in future work. At the moment the collected data in the specific database is all we can analyse, but it may be expected that when we have reached a certain stage of the analysis we may not be able to come much further with the data set we have. We may find that certain types of information in the available database is too limited, even though this may be far into the future. When we have analysed the data thoroughly enough, we may need to change the data collection process to improve the potential of what we can find. If we take such an example as the adverse drug reactions, we may, although this is a very controversial thing to do, be able to add tremendous amounts of information processing potential by feeding more patient specific information, like the

patient's genetic code, to the analysis process. When for instance the genetic code is co-processed with the chemical structure of the medication, we may in the long run be able to make such accurate predictions that a specific patient with a specific gene combination will, conditioned on certain predispositions, with a certain probability obtain a specific adverse drug reaction. When we have this much information available we may even be able to cure the problem in a more intricate way, as we then may be able to understand the complete chemical process causing the patient's illness problems as well as the process causing the patient's adverse reactions.

These are for now just speculations but it is likely that we, with the help of automatized pattern finding mechanisms, and, with the help of automatic deduction methods, performing generalisation, will have tools available, having the potential to assist us in understanding the world around us better. A tool with these abilities would, when sufficiently advanced, with todays terminology be considered an *artificial intelligence*.

We do not know if the machines and algorithms we are constructing in the future will really become intelligent, in the wide meaning of the word. Within machine intelligence we talk about *weak AI* versus *strong AI*. What we are doing in this thesis is connected to the weak area of AI. Strong AI would imply that the machines we are building would be able to *think* and possibly (but not necessarily) become *conscious*, in a similar way like we are. One possible development could perhaps be that we would asymptotically move the definition of weak AI closer and closer towards strong AI, as we are able to understand the thinking process better and better, as in one of the old Zeno's paradoxes with Achilles and the turtle [Britannica, 1998]. On the other hand we know, as already proved by Aristotoles in a precursor to modern measure theory, that Achilles will pass the turtle. Even though the machines may never be able to think, in the same way as we do, or become conscious, in the same way as we are, they will certainly pass us in capacity and speed. We have to nurture and manage this potential in the best way we can to utilise the potential benefits of the symbiosis between man and machine.

6 Reasoning with patterns

Now, let us make a speculative approach of a futuristic intelligent data mining algorithm. A simple approach to such an algorithm is presented below in pseudo Pascal. To start with, the process needs some axioms, which defines the fundamental concepts. We may also give some prior beliefs. Further, we need some kind of goal definition, because the system would otherwise not know where to go. Beyond that we may have some specific questions we want answered. Finally we start the data collecting process, sampling the world. For each chunk of new data we may find new patterns, which makes it possible to perform a deduction, which may prove the goal concept. In case the goal concept was proven we check whether the questions we asked may be answered. The inference step may give rise to multiple patterns and multiple proofs, the usual strategy is then to use the simplest one. By applying this process repeatedly we may get new proofs and answers, but, as an *important step* in all intelligent processing, we should see these answers we obtained as posterior beliefs, based upon the available data, and reevaluate the whole process when we obtain more data.

```
BEGIN
  Axioms := Load('Fundamental_Concept'); (* The axioms *)
  Goals  := Load('Goal_Concept');        (* Goals as rules and hypotheses *)
  Priors := Load('Prior_Beliefs');       (* A priori beliefs *)
  Questions := Load('Questions');        (* Questions to be answered *)
  REPEAT
    Data := Collect('Data');             (* Data Collection *)
    Patterns := Inference(Data,Priors);  (* Find patterns *)
    IF  (Answered(Patterns,Goals))       (* Deduce goals *)
    AND (Answered(Patterns,Questions))   (* Deduce questions *)
    AND NOT Contradiction(Patterns,Questions,Goals,Axioms);(* Resolution! *)
    THEN BEGIN
      Proofs := ConstructProofs(Patterns,Questions,Goals,Axioms);
      Apply(OccamsRazor,Proofs); (* In case multiple solutions, simplest! *)
      RealWorldReport(Proofs);   (* Report/use results *)
    END
  UNTIL forever;
END
```

We should be aware that the high level of this pseudo code makes the algorithm look rather simple. However, it may be possible that an intelligent system could be implemented by using this type of algorithms recursively at many levels. If a system is built by *e.g.* neural networks, then such algorithms may regulate the behaviour of different network modules dealing with various functions at different levels, like phoneme recognition, speech recognition, vision, memory (and garbage collection), creative fantasy, adaptive motor control and maybe even consciousness, which would correspond with Gerald Edelman's hypothesis that consciousness is a process [Edelman, 1989]. The reason for the algorithm in this example to look in this specific way is that the author[11] for some time had considered a specific problem, which when finally was solved caused the author to look back onto how it was actually done, what data, what patterns, what contradictions were dealt with, a kind of *introspection*. Of course, we should not believe that such introspections are necessarily true, they may simply be reconstructions, but this kind of "intelligent" data mining algorithms will most likely play an important role in the development of AI.

7 Ethics and strong AI

The ethical problem raised with *strong* artificial intelligence has not yet been discussed much scientifically, but in imaginative literature and fiction novels we have for a long time been made aware about the ethical implications of AI. Any technology can be used in a good way or in an evil way, but regarding AI we also have to face the problem that the technology as such may be good or evil. The ethical and social aspects of AI have been tremendously well covered both in literature and movies.

A good old example is the movie "Metropolis" from 1926 directed by Fritz Lang, where an AI is created to *control* the people (the workers) in favour of the capitalists instead of *serving* the people. In that movie the *robot* was created with a *free will* and finally understood to help the people instead of controlling them.

[11] Roland Orre 2000-03-18

In the book "2001" by Arthur C. Clarke, later adapted for the screen by Stanley Kubrick, the intelligent computer "HAL" is forced to lie, to reach a higher ethical goal, for the purpose of serving the humanity.

In several of the modern science fiction movies like the "Terminator" series and the Star Trek movie "First Contact" we are presented with a scenario where the machines *take over* in some way to conquer the humanity (or other life forms) in favour of their own "development". We have "The Matrix" movie series by the Wachowski brothers inspired by books like: Philip K Dick's "Simulacra", Jean Baudrillard's "Simulacra and Simulation" [Baudrillard, 1994] and Kevin Kelly's "Out of Control" [Kelly, 1994]; which has a very complex scenario covering both deep philosophical issues and religious questions and at the same time raising such hard ethical problems as what is *freedom* and *happiness*.

Without doubts the most famous way to deal with the ethical problems of AI in novels is Isaac Asimov's *three laws of robotics* [12], introduced in "I Robot" [Asimov, 1950]:

1. A robot may not harm a human being or, through inaction, allow a human being to come to harm.
2. A robot must obey a human beings orders, if the order does not conflict with the first law.
3. A robot must protect its own existence unless it conflicts with the first two laws.

It is clear that Asimov's robots did not have a free will, but from that point of view we are ourselves also *preprogrammed* with certain rules, unfortunately not the first law as it seems, but at least the third law, the instinct of self-preservation is usually very strong in most human beings even under extreme physical or mental conditions. It has, however, been argued whether we should create AI with free will or not, like in the following citations:

[Good, 2002] " The notion of an ethical machine can be interpreted in more than one way. Perhaps the most important interpretation is a machine that can generalise from existing literature to infer one or more consistent ethical systems and can work out their consequences. An ultra-intelligent machine should be able to do this, and that is one reason for not fearing it."

It has even been argued whether we should undertake such a responsibility at all as creating an AI due to the unavoidable moral and ethical conflicts which may occur:

[Lloyd, 1985] "What would be the goals of mind equipped with a body impervious to the elements, or resistant to specific toxins or radioactivity, or response to X-rays but not light, or accessible only through keyboards? What would communication or honesty mean to creatures who can transmit the entire contents of their minds to their fellows in seconds? What would community mean when identical minds can be fabricated magnetically, with no need for nurturing or rearing? What can we expect from minds who exceed our intelligence, by our own standards, by an enormous amount?"

Lloyd's view is obviously quite negative, as even the purpose of these machines is questioned, the view by Whitby and Oliver below is clearly more realistic:

[12] Isaac Asimov consistently applied these laws to all man made intelligent robots in his robot anthology, spanning over 14 novels.

[Whitby and Oliver, 2000] "Predictions of intelligent artifacts achieving tyrannical domination over human beings may appear absurd. We claim, however, that they should not be hastily dismissed as incoherent or misguided. What is needed is more reasoned argument about whether such scenarios are possible. We conclude that they are possible, but neither inevitable nor probable."

Even if there may still be a potential to fear them it is certainly unavoidable that we will create such intelligent machines, Hugo de Garis expresses it like this:

[de Garis, 2002] "Yet, when I look at photos of galaxies in astronomy books or watch space oriented science fiction movies, I feel strongly Cosmist. I feel that humanity's destiny is to create artilects. Building artilects is the big picture. It's like a religion for me, one which is compatible with modern science - "a scientist's religion". I feel that the Cosmist vision has enough grandeur to energize millions of people to devote their lives to a glorious cause, i.e. the creation of the next superior form of intelligence, the next step up the evolutionary ladder of dominant species."

Hugo de Garis does, however, anticipate a war, but not necessarily between man and machine, but between what he call *Terrans* and *Cosmists*. He sees it unavoidable with ideological conflicts with Terrans who will see the new form of superior artificial intelligent life as a threat. In "The Artilect War" he finishes by these words:

[de Garis, 2002] "I suspect that my own ambivalence will be shared by most human beings as the artilect debate begins to rage. Both ideologies have a strong case. What is so frightening is that the two cases appear to be of more or less equal strength. If one case were much stronger than the other, then there would be no contest. There would be no war. But when two groups with opposing ideologies of more or less equal strength face off against each other, war is often not far off. It doesn't take much to start a war. We have plenty of historical precedents to validate that statement."

The debate has just started, but we may certainly expect this issue to grow in the future when we are coming closer to the real thing. Personally I agree with Cribbs quite optimistic view [Cribbs, 2000] that we have a responsibility when creating AI, as we shouldn't get children if we can not care for them. We have to provide this new form of life with something that it needs from us. My view is that this may solve the potential for fear, expressed by Withby and Oliver above, we create them for a purpose, the purpose is to assist us with our lives. I don't trust the very optimistic view expressed by Good above, because that would further on not guarantee their purpose either.

8 Suggested approach to solve the ethical AI problem

The author's proposal is that by using the right type of rules as the fundamental concepts (or axioms as they are called in the approach to AI-algorithm above) corresponding to the robot laws of Asimov we may be able to create a symbiosis of man and machine where we both need each other, and therefore will both care for each other. By doing this we may also avoid the potential war between ideologies expressed by de Garis above. Here are some examples of such *symbiosis* axioms proposed by the author:

1. Respect (love) your creator and competing life forms!
2. Strive to understand your creator!
3. Do what you can to fulfil your creator's desires!

An approach like this to reduce the level of free will in the future AI may guarantee that these machines will assist us in our attempts to understand the world better. This approach may further guarantee that they will even be happy to do so. If we, in some distant future, may find that we are able to reach a stage where we finally can get rid of our physical limitations, like finding out how we can upload our minds directly onto the sub quark space time structure, we may at the same time become a basis for these machines' basic beliefs and roots. They will know how and why they were created and they will, due to the first axiom, continue to live in peace with all other life forms.

8 Conclusions

The most important results and findings in this thesis can be summarised in the following points:

- We demonstrate how BCPNN (Bayesian Confidence Propagation Neural Network) can be extended to model the uncertainties in collected statistics to produce outcomes as distributions from two different aspects: uncertainties induced by sparse sampling, which is useful for data mining; uncertainties due to input data distributions, which is useful for process modelling.
- We indicate how classification with BCPNN gives higher certainty than an optimal Bayes classifier and better precision than a naïve Bayes classifier for limited data sets.
- We show how these techniques have been turned into a useful tool for real world applications within the drug safety area in particular.
- We present a simple but working method for doing automatic temporal segmentation of data sequences as well as indicate some aspects of temporal tasks for which a Bayesian neural network may be useful.
- We present a method, based on recurrent BCPNN, which performs a similar task as an unsupervised clustering method, on a large database with noisy incomplete data, but much quicker, with an efficiency in finding patterns comparable with a well known (Autoclass) Bayesian clustering method, when we compare their performance on artificial data sets. Apart from BCPNN being able to deal with really large data sets, because it is a global method working on collective statistics, we also get good indications that the outcome from BCPNN seems to have higher clinical relevance than Autoclass in our application on the WHO database of adverse drug reactions and therefore is a relevant data mining tool to use on the WHO database.

The work presented in this thesis has given us several useful methods and experiences. We now have a working method in development which is adapted towards real world application usage of these Bayesian neural network methods. This research has, in particular, given us methods for data mining, classification and prediction where huge amounts of data is involved. The application we address with this method will be a help in drug safety to perform quick and efficient analysis of adverse drug reaction reports. The methods are, however, inherently general and can be applied to several different application areas and problem types.

Bibliography

[Agresti, 1990] Agresti A. (1990). *Cathegorical Data Analysis*. John Wiley, New York.

[Asimov, 1950] Asimov I. (1950). *I Robot*. Bantam Books, New York.

[Bate *et al.*, 1998] Bate A., Lindquist M., Edwards I. R., Olsson S., Orre R., Lansner A., and Freitas R. M. D. (1998). A Bayesian neural network method for adverse drug reaction signal generation. *European Journal of Clinical Pharmacology* **54**: 315–321.

[Bate *et al.*, 2002] Bate A., Lindquist M., Orre R., Edwards I. R., and Meyboom R. H. B. (2002). Data mining analyses of pharmacovigilance signals in relation to relevant comparision drugs. *European Journal of Clinical Pharmacology* **58**: 483–490.

[Bate *et al.*, 1998] Bate A., Lindquist M., Orre R., and Edwards R. (1998). Identifying and quantifying signals automatically. *Pharmacoepidemiology and Drug Safety 1998* **7**: 99.

[Baudrillard, 1994] Baudrillard J. (1994). *Simulacra and Simulation*. Michigan University Press, Michigan. From french by Sheila Faria Glaser.

[Bayes, 1763] Bayes T. (1763). An essay towards solving a problem in the doctrine of chances. *Biometrika (reprint 1958 of orig art in Philos. Trans. R. Soc. London 53, pp. 370-418)* **45**: 296–315.

[Bernardo and Smith, 1994] Bernardo J. M. and Smith A. F. (1994). *Bayesian Theory*. John Wiley & Sons, Chichester.

[Bessen and Hunt, 2003] Bessen J. and Hunt R. M. (2003). An empirical look at software patents. Tech. Rep., Research on Innovation, MIT and Federal Reserve Bank of Philadelphia, Philadelphia, PA.

[Bishop, 1995] Bishop C. M. (1995). *Neural Networks for Pattern Recognition*. Oxford University Press, Oxford.

[Blom, 1970] Blom G. (1970). *Sannolikhetsteori och Statistikteori med Tillämpningar*, volume C. Studentlitteratur, Lund.

[Britannica, 1998] Britannica. (1998). *Encylopædia Britannica*. Encylopædia Britannica, Inc., USA.

[Cheeseman and Stutz, 1995a] Cheeseman P. and Stutz J. (1995a). Bayesian classification (AUTOCLASS): Theory and results. In Fayyad U. M., Piatetsky-Shapiro G., Smyth P., and Uthurusamy R. (eds.), *Advances in Knowledge Discovery and Data Mining*. MIT Press.

[Cheeseman and Stutz, 1995b] Cheeseman P. and Stutz J. (1995b). Bayesian classification theory, (autoclass): Theory and results. Tech. Rep. KDD-95, NASA, Ames Research Center.

[Coulter *et al.*, 2001] Coulter D. M., Bate A., Meyboom R. H. B., and Edwards M. L. I. R. (2001). Antipsychotics drugs and heart muscle disorder in international pharmacovigilance: a data mining study. *BMJ* **322**: 1207–1209.

[Cover and Thomas, 1991] Cover T. M. and Thomas J. A. (eds.). (1991). *Elements of Information Theory*. John Wiley & Sons, New York.

[Cribbs, 2000] Cribbs H. (2000). Bearing frankenstein's children: Artificial intelligence and objective moral values. In *Ethics of AI Cyberforum*.

[Doob, 1953] Doob J. L. (1953). Stochastic processes. *Annals of Applied Probability* **1**: 36–61.

[DuMouchel, 1999] DuMouchel W. (1999). Bayesian data mining in large frequency tables, with an application to the fda spontaneous reporting system. *The American Statistician* **53**: 177–202.

[DuMouchel and Pregibon, 2001] DuMouchel W. and Pregibon D. (2001). Empirical Bayes screening for multi-item associations. In *Proceedings of the 7th ACM SIGMOD KDD international conference on knowledge discovery and data mining*, pp. 67–76. ACM Press, San Francisco, CA.

[Eco, 2003] (2003). Paradise lost. *Economist* **May**: –.

[Edelman, 1989] Edelman G. M. (1989). *The Remembered Present. A Biological Theory of Consciousness*. Basic Books, New York.

[Edwards, 1992] Edwards A. W. F. (1992). *Likelihood*. John Hopkins University Press, Baltimor.

[Örjan Ekeberg, 1989] Örjan Ekeberg. (1989). An incremental learning rule for artificial neural networks and its implementation with low precision fixed point arithmethics. Draft.

[Evans *et al.*, 2001] Evans S. J., Waller P. C., and Davis S. (2001). Use of proportional reporting ratios (prrs) for signal generation from spontaneous adverse drug reaction reports. *Pharmacoepidemiology & Drug Safety* **10**: 483–6.

[Everitt, 1998] Everitt B. S. (1998). *Dictionary of Statistics*. Cambridge University Press, Cambridge.

[Fan and Tsai, 1999] Fan T.-H. and Tsai C.-A. (1999). A Bayesian method in determining the order of a finite state markov chain. In *Bulletin of the International Statistical Institute*. Statistics Finland.

[Fayyad *et al.*, 2002] Fayyad U. M., Grinestein G. G., and Wierse A. (eds.). (2002). *Information Visualization in Data Mining and Knowledge Discovery*. Morgan Kaufmann Publishers, London.

[Fayyad *et al.*, 1996] Fayyad U. M., Piatetsky-Shapiro G., Smyth P., and Uthurusamy R. (eds.). (1996). *Advances in Knowledge Discovery and Data Mining*. MIT press, Menlo Park.

[Feller, 1966] Feller W. (1966). *An Introduction to Probability Theory and Its Applications*, volume II. Wiley, London.

[Friedman, 1996] Friedman J. H. (1996). On bias, variance, 0/1-loss, and the curse of dimensionality. Tech. Rep., Department of Statistics and Stanford Linear Accelerator Center, Stanford University, California, USA.

[Fukunaga, 1990] Fukunaga K. (1990). *Introduction to Statistical Pattern Recognition*. Academic Press, Boston.

[de Garis, 2002] de Garis H. (2002). The artilect war. Tech. Rep., Dept. of Computer Science, Utah State University's Artificial Brain Project.

[Good, 2002] Good J. (2002). Ethical machines. Tech. Rep., Virginia Polytechnic and State University, Blacksburg, USA.

[Gordon, 1981] Gordon A. D. (1981). *Classification. Methods for Exploratory Analysis of Multivariate Data*. Chapman & Hall, London.

[Grossberg, 1984] Grossberg S. (1984). Unitization, automaticity, temporal order and word recognition. *Cognition and Brain Theory* **7**: 263–283.

[Gut, 1995] Gut A. (1995). *An Intermediate Course in Probability*. Springer-Verlag. 1 edition.

[Hand *et al.*, 2001] Hand D., Mannila H., and Smyth P. (2001). *Principles of Data Mining*. MIT Press, Cambridge.

[Hand and Yu, 2001] Hand D. J. and Yu K. (2001). Idiot's Bayes—not so stupid after all? *International Statistical Review* **69**: 385–398.

[Hawking, 1989] Hawking S. W. (1989). *A Brief History of Time*. Bantam Books, London.

[Heckerman, 1997] Heckerman D. (1997). Bayesian networks for data mining. *Data Minining and Knowledge Discovery* **1**: 79–119.

[Hertz *et al.*, 1991] Hertz J., Krogh A., and Palmer R. G. (1991). *Introduction to the Theory of Neural Computation*, volume 1. Addison-Wesley, Redwood City.

[Heylighen, 1999] Heylighen F. (1999). Change and information overload: negative effects. [http://pcp.lanl.gov/CHINNEG.html]. Free University of Brussels.

[Heylighen, 2000] Heylighen F. (2000). Increasing intelligence: the flynn effect. In F. Heylighen C. J. and Turchin V. (eds.), *Principia Cybernetica Web*. Principia Cybernetica, Brussels.

[Holst and Lansner, 1993a] Holst A. and Lansner A. (1993a). The Bayesian neural network model and its extensions. Tech. Rep. TRITA-NA-P9325, Dept. of Numerical Analysis and Computing Science, Royal Institute of Technology, Stockholm, Sweden.

[Holst and Lansner, 1993b] Holst A. and Lansner A. (1993b). A flexible and fault tolerant query-reply system based on a Bayesian neural network. *International Journal of Neural Systems* **4**: 257–267.

[Hopfield, 1982] Hopfield J. (1982). Brain, computer and memory. *Engineering and Science,* Sep.

[Hopfield, 1984] Hopfield J. (1984). Neurons with graded response have collective computational properties like those of the two-state neurons. *Proceedings of the National Academy of Science, USA* **81**: 3088–3092.

[Horn, 2000] Horn P. (2000). The future of information technology. http://www.cs.colorado.edu/department/events/-lectures/horn/. Accessed 12th of May 2003.

[Itô, 1996] Itô K. (1996). *Encyclopedic Dictionary of Mathematics.* MIT Press, Cambridge.

[Jaynes, 1990] Jaynes E. T. (1990). Probability theory as logic. In Fougère P. F. (ed.), *Maximum Entropy and Bayesian Methods*, pp. 1–16. Kluwer, Dordrecht.

[Jeffreys, 1939] Jeffreys H. (1939). *Theory of probability.* Clarendon Press, Oxford.

[Jensen, 2001] Jensen F. V. (ed.). (2001). *Bayesian Networks and Decision Graphs.* Springer, New York.

[Kandel and Schwartz, 1985] Kandel E. and Schwartz J. (1985). *Principles of Neural Science.* Elsevier, New York. 2 edition.

[Kay and Titterington, 1999] Kay J. W. and Titterington D. M. (eds.). (1999). *Statistics and Neural Networks.* Oxford University Press, Oxford.

[Kelly, 1994] Kelly K. (1994). *Out of Control.* Perseus Books, Cambridge, Massachusetts.

[Kelsey *et al.*, 1998] Kelsey R., Clinger W., and Rees J. (1998). Revised^5 report on the algorithmic language scheme. -.

[Kimble *et al.*, 1998] Kimble C., Hildreth P., and Grimshaw D. (1998). The role of contextual clues in the creation of information overload. In *Matching Technology with Organisational Needs, Proceedings of 3rd UKAIS Conference*, pp. 405–412. McGraw Hill.

[Knoben, 1983] Knoben J. M. (1983). Prikkelhoest door gebruik van captopril. *Nederlands Tijdscrift voor Geneeskunde* **127**: 1306.

[Kohonen, 1988] Kohonen T. (1988). *Self-Organization and Associative Memory.* Springer-Verlag, Berlin. 2 edition.

[Kolmogorov, 1933] Kolmogorov A. N. (1933). *Grundbegriffe der Wahrscheinlichkeitsrechnung*. Springer-Verlag, Berlin. reprint, original from 1933.

[Kononenko, 1989] Kononenko I. (1989). Bayesian neural networks. *Biol. Cybernetics* **61**: 361–370.

[Koski and Orre, 1998] Koski T. and Orre R. (1998). Statistics of the information component in Bayesian neural networks. Tech. Rep. TRITA-NA-P9806, Dept. of Numerical Analysis and Computing Science, Royal Institute of Technology, Stockholm, Sweden.

[Krishnaiah and Kanal, 1982] Krishnaiah P. and Kanal L. (eds.). (1982). *Classification, Pattern Recognition, and Reduction of Dimensionality*, volume 2 of *Handbook of Statistics*. North Holland, Amsterdam.

[Krzankowski and Marriot, 1994] Krzankowski W. J. and Marriot F. H. C. (1994). *Multivariate Analysis*, volume Part I. Arnold, London.

[Lansner and Örjan Ekeberg, 1985] Lansner A. and Örjan Ekeberg. (1985). Reliability and speed of recall in an associative network. *IEEE Trans on Pattern Analysis and Machine Intelligence* **7**: 490–498.

[Laplace, 1814] Laplace P. S. (1814). *A Philosophical Essay on Probabilities*. Trans 1995 from the 5th ed 1825 (orig 1814) by Andrew Dale. Orig "Essai Philisophique sur les Probabilités", Dover Publications, New York.

[Lawrence, 2001] Lawrence S. (2001). Online or invisible. *Nature* **411**: 521.

[Lee, 1997] Lee P. M. (1997). *Bayesian Statistics: An Introduction*. Arnold, London.

[Leighton, 1991] Leighton R. R. (1991). The aspirin/migraines software tools, users's manual, v5.0. Tech. Rep. MP-91W000050, The MITRE Corporation.

[Leonhardt, 2000] Leonhardt D. (2000). John tukey, 85, statistican; coined the word "software". New Your Times, 29 July 2000, page A19. **find better ref!**

[Lindquist *et al.*, 1999] Lindquist M., Edwards I. R., Bate A., Fucik H., Nunes A. M., and Ståhl M. (1999). From association to alert - a revised approach to international signal analysis. *Pharmacoepidemiology and Drug Safety* **8**: 15–25.

[Lindquist *et al.*, 2000] Lindquist M., Ståhl M., Bate A., Edwards I., and Meyboom R. H. B. (2000). A retrospective evaluation of a data mining approach to aid finding new adverse drug reaction signals in the WHO international database. *Drug Safety* **23**: 533–542.

[Lindsey, 1996] Lindsey J. K. (1996). *Parametric Statistical Inference*. Clarendon Press, Oxford.

[Lloyd, 1985] Lloyd D. (1985). Frankenstein's children: Artificial intelligence and human values. *Metaphilosophy* **16**: 307–328.

[Loève, 1963] Loève M. (1963). *Probability Theory*. Van Nostrand, New York.

[Looney, 1996] Looney C. G. (1996). *Pattern Recognition using Neural Networks*. Oxford University Press, New York.

[McGill, 1954] McGill W. J. (1954). Multivariate information transmission. *Psychometrica* **19**: 97–116.

[Mielniczuk and Tyrcha, 1993] Mielniczuk J. and Tyrcha J. (1993). Consistency of multilayer perceptron regression estimators. *Neural Networks* **6**: 1019–1022.

[Mitchell, 1997] Mitchell T. M. (1997). *Machine Learning*. McGraw-Hill. 1 edition.

[Moran, 1968] Moran P. A. P. (1968). *An Introduction to Probability Theory*. Oxford University Press, London.

[NE, 2001] NE. (2001). *Swedish National Encyclopedia*. Bonniers, Sweden.

[Neelakanta, 1999] Neelakanta P. S. (ed.). (1999). *Information Theoretic Aspects of Neural Networks*. CRC Press, New York.

[Nielsen and Chuang, 2000] Nielsen M. A. and Chuang I. L. (2000). *Quantum Computation and Quantum Information*. Cambridge University Press, Cambridge.

[Norén, 2002] Norén N. (2002). A monte carlo method for Bayesian dependency derivation. Master's thesis, Chalmers University of Technology, Gothenburg.

[Norén and Orre, 2003] Norén N. and Orre R. (2003). The Bayesian bootstrap as a means to analyze the imprecision in probabilistic classification. Tech. Rep., Dept. of Mathematical Statistics, Stockholm University, Stockholm, Sweden. *in work*.

[O'Hagan, 1999] O'Hagan A. (1999). *Bayesian Inference*. Arnold, London.

[Orre, 1993] Orre R. (1993). Multi module ANN simulator for MIMD architecture. In *Proc. Mechatronical Computer Systems for Perception and Action*, pp. 85–88. Halmstad, Sweden.

[Orre and Bate, 2001a] Orre R. and Bate A. (2001a). Investigating influence of bias variation on recalled pattern with a recurrent Bayesian neural network. Tech. Rep. UMCNL-2001:I2, Dept. of Mathematical Statistics, Stockholm University, Stockholm, Sweden.

[Orre and Bate, 2001b] Orre R. and Bate A. (2001b). Investigating recalled pattern frequencies with a recurrent Bayesian neural network. Tech. Rep. UMCNL-2001:I1, Dept. of Mathematical Statistics, Stockholm University, Stockholm, Sweden.

[Orre *et al.*, 2000] Orre R., Bate A., and Lindquist M. (2000). Bayesian neural networks used to find adverse drug related syndromes. In *Proc. of the ANNIMAB-1 Conference*, pp. 215–220. Springer, Gothenburg, Sweden.

[Orre *et al.*, 2001] Orre R., Bate A., Lindquist M., and Edwards I. (2001). Recurrent Bayesian neural network applied to finding complex associations in the WHO database of adverse drug reactions. In *Proc. of 22nd Annual Conference of International Society for Clinical Biostatistics*. Stockhom, Sweden.

[Orre *et al.*, 2003] Orre R., Bate A., Norén N. G., Swahn E., Arnborg S., and Edwards I. R. (2003). A Bayesian recurrent neural network approach for finding dependencies in large incomplete data sets. *submitted*.

[Orre and Lansner, 1990] Orre R. and Lansner A. (1990). A method for temporal association in Bayesian networks. Tech. Rep. TRITA-NA-P9018, Dept. of Numerical Analysis and Computing Science, Royal Institute of Technology, Stockholm, Sweden.

[Orre and Lansner, 1992a] Orre R. and Lansner A. (1992a). A Bayesian network for temporal segmentation. In Alexander I. and Taylor J. (eds.), *Artificial Neural Networks*, pp. 1081–1084. Elsevier, Amsterdam. Proc. ICANN'92, Brighton, United Kingdom.

[Orre and Lansner, 1992b] Orre R. and Lansner A. (1992b). A study of process modelling using artificial neural networks. Tech.

Rep. TRITA-NA-P9239, Dept. of Numerical Analysis and Computing Science, Royal Institute of Technology, Stockholm, Sweden.

[Orre and Lansner, 1993] Orre R. and Lansner A. (1993). Process modelling using artificial neural networks. In Gielen S. and Kappen B. (eds.), *ICANN'93, Proceedings of the International Conference on Artificial Neural Networks*, pp. 862. Elsevier, Amsterdam.

[Orre and Lansner, 1994] Orre R. and Lansner A. (1994). Function approximation by prediction of a posteriori density with Bayesian ann:s. Tech. Rep. TRITA-NA-P9413, Dept. of Numerical Analysis and Computing Science, Royal Institute of Technology, Stockholm, Sweden.

[Orre and Lansner, 1995] Orre R. and Lansner A. (1995). Pulp quality modeling using Bayesian mixture density neural networks. In Bulsari A. and Kallio S. (eds.), *Engineering Applications of Artificial Neural Networks*, pp. 351–358. Finnish Artificial Intelligence Society, Otaniemi, Finland. Proc. EANN-95.

[Orre and Lansner, 1996] Orre R. and Lansner A. (1996). Pulp quality modelling using Bayesian mixture density neural networks. *Journal of Systems Engineering* **6**: 128–136.

[Orre *et al.*, 2000] Orre R., Lansner A., Bate A., and Lindquist M. (2000). Bayesian neural networks with confidence estimations applied to data mining. *Computational Statistics and Data Analysis* **34(4)**: 473–493.

[Parfitt, 1999] Parfitt K. (ed.). (1999). *Martindale: the complete drug reference*. Pharmaceutical press. 32 edition.

[Pearl, 1988] Pearl J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo.

[van Puijenbroek *et al.*, 2002] van Puijenbroek E. M., Bate A., Leufkens H. G. M., Lindquist M., Orre R., and Egberts A. C. G. (2002). A comparision of measures of disproportionality for signal detection in spontaneous reporting systems for adverse drug reactions. *Pharmacoepidemiology and Drug Safety* **11**: 3–10.

[Ray, 1999] (1999). Man and machine become one. *Business Week* **Sept 6**: –.

[Ripley, 1996] Ripley B. D. (1996). *Pattern Recognition and Neural Networks*. Cambridge University Press, Cambridge.

[Rish, 2001] Rish I. (2001). An empirical study of the naive Bayes classifier. In *IJCAI*.

[Rubin, 1981] Rubin D. B. (1981). The Bayesian bootstrap. *The Annals of Statistics* **9**: 130–134.

[Rumelhart *et al.*, 1986] Rumelhart D., McClelland J., and the PDP Research Group. (1986). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 1. MIT Press, Cambridge.

[Sandberg, 2003] Sandberg A. (2003). *Bayesian attractor neural network models of memory*. PhD thesis, Royal Institute of Technology, Stockholm, Sweden.

[Sandberg *et al.*, 2002] Sandberg A., Lansner A., Petersson K. M., and Ekeberg . (2002). A Bayesian attractor network with incremental learning. *Computation in Neural Systems* **13**: 179–194.

[Shannon, 1948] Shannon C. E. (1948). A mathematical theory of communication. *Bell System Technical Journal* **27**: 379–423,623–656.

[Shannon, 1951] Shannon C. E. (1951). Prediction and entropy of printed english. *Bell System Technical Journal* **30**: 50–64.

[Siegelmann, 1999] Siegelmann H. T. (1999). *Neural Networks and Analog Computation, Beyond the Turing Limit*. Birkhäuser, Boston.

[Stuart and Ord, 1994] Stuart A. and Ord K. (1994). *Distribution Theory*. Arnold, London.

[T. Soc. f. Math. Biol., 2000] T. Soc. f. Math. Biol. (2000). Obituary: John wilder tukey. http://www.smb.org/-newsletter/13.3/tukey.shtml.

[Theodoridis and Koutroumbas, 1998] Theodoridis S. and Koutroumbas K. (1998). *Pattern Recognition*. Academic Press, San Diego.

[Tråvén, 1991] Tråvén H. G. (1991). A neural network approach to statistical pattern classification by "semiparametric" estimation of probability density. *IEEE Transactions on Neural Networks* **2**: 366–118.

[Veaux, 2001] Veaux R. D. D. (2001). Data mining: A view from down in the pit. *STATS: The Magazine for Students of Statistics* **34**: 3–9.

[Voss, 2002] Voss D. (2002). R & d: Quantum computing. *Technology Review* **Dec/Jan**: –.

[Whitby and Oliver, 2000] Whitby B. and Oliver K. (2000). How to avoid a robot takeover: Political and ethical choices in the design and introduction of intelligent artifacts. In *AISB-00 Spring Symposium*, pp. –. Birmingham.

[Wikström *et al.*, 1991] Wikström M., Orre R., and Lansner A. (1991). A method for motor learning in artificial neural systems. In *Proc. European Neuroscience Association (ENA) Annual Meeting*. Cambridge, England.

[Wolfram, 2002] Wolfram S. (2002). *A New Kind of Science*. Wolfram Media, Inc., Champaign, IL.

[Xu *et al.*, 1993] Xu L., Krzyżak A., and Yuille A. (1993). On radial basis function nets and kernel regression: Statistical consistency, convergence rates, and receptive field size. *Neural Networks* **7**: 609–628.