Variational Inference in Probabilistic Models

Neil D. Lawrence

Computer Laboratory,

Cambridge University,

New Museums Site, Pembroke Street,

Cambridge, CB2 3QG, U.K.

Neil.Lawrence@cl.cam.ac.uk

PhD Thesis

January 2000

Contents

List of Figures

List of Tables

The University of Cambridge

Variational Inference in Probabilistic Models

Neil D. Lawrence

Computer Laboratory,

Cambridge University,

New Museums Site, Pembroke Street,

Cambridge, CB2 3QG, U.K.

Neil.Lawrence@cl.cam.ac.uk

PhD Thesis, 2000

Thesis Summary

One of the key objectives of modern artificial intelligence is the handling of uncertainty. Probability theory provides a framework for handling of uncertainty in a principled manner. Probabilistic inference is the process of reasoning under uncertainty and as such is vital for both learning and decision making in probabilistic models. For many models of interest this inference proves to be intractable and to make progress approximate methods need to be considered. This thesis concerns itself with a particular approximating formalism known as variational inference (Jordan *et al.*, 1998). Variational inference has the advantage that it often provides bounds on quantities of interest, such as marginalised likelihoods. In this thesis we describe a general framework for the implementation of the variational approach. We then explore the approach in several different probabilistic models including undirected and directed graphs, Bayesian neural networks and independent component analysis. In those models which have been handled using variational inference previously we show how we may improve the quality of our inference engine by considering more complex variational distributions.

Keywords: probabilistic models, variational methods, inference

Acknowledgements

First of all I would like to thank my supervisor Professor Christopher M. Bishop for his help and guidance.

Thanks also to Dr. William F. Clocksin and the Computer Laboratory for all the support they have given me.

I am indebted to Marta Milo for her influence both on this thesis, and in my life. I am, therefore, also indebted to the NATO Advanced Science Institute and the Newton Institute for enabling me to meet her.

I would also like to thank Niranjan and Neil Stephen for steering me in the right direction.

This thesis is much richer from the products of discussions with Mehdi Azzouzi, Alex Rogers and Mike Tipping.

I would also like to thank my family for supporting me both through my decision to leave my employment, and then throughout the years of working on this thesis.

It would be difficult to formulate a list of all those to who have influenced my research in some way. I can cover many of them by mentioning Aston University's NCRG and the people who attend David MacKay's inference group meetings.

I am grateful to the EPSRC for paying my tuition fees and for partial support of my living expenses through an EPSRC research studentship.

Neil Lawrence, January 2000

Declaration

The work in this thesis is the product of my efforts except where it is specified otherwise, it has not previously been submitted for any other degree or diploma.

Chapter 1

Introduction

Over the last twenty years, great inroads have been made in the field of computer science. Modern computational capabilities allow full virtual environments to be designed. However while we have made large advances in computational power, less progress has been made in the replication of human decision making skills and learning ability. This thesis is concerned with issues that humans handle with ease, but computers have difficulty with, uncertainty and learning.

1.1 Uncertainty

Uncertainty is an integral part of human decision making. We are repeatedly obliged to assess situations where many of the determining factors are unobserved. Such assessments occur on a daily basis within our daily lives and our societies most important institutions. Many legal systems formally acknowledge the presence of uncertainty within the judicial process. A good legal system provides us with an unambiguous set of rules by which to live our lives. Ideally infringements of these rules would lead automatically to prescribed punishments. In this case the role of the judicial system would be to determine to what degree the defendant's actions had infringed the law and what the resulting punishment should be. However, in reality, much of the judicial system is dedicated to the determination of the defendants actions. Judicial systems recognise the existence of uncertainty by requiring that guilt only be proved *beyond reasonable doubt*.

reasonable doubt The level of certainty a juror must have to find a defendant guilty of a crime. A real doubt, based upon reason and common sense after careful and impartial consideration of all the evidence, or lack of evidence, in a case.

Proof beyond a reasonable doubt, therefore, is proof of such a convincing character that you would be willing to rely and act upon it without hesitation in the most important of your own affairs. However, it does not mean an absolute certainty. – *Legal Lexicon's Lyceum, 1999.*

The lack of absolute certainty in the determination of guilt means that the law implicitly recognises that it allows itself to convict innocent people. Given that the importance of a fair legal system is

recognised in article 10 of the the Universal Declaration of Human Rights, we argue that the acknowledgement in many judicial systems of reasonable doubt is a social acknowledgement of uncertainty.

> *Everyone is entitled in full equality to a fair and public hearing by an independent and impartial tribunal, in the determination of his rights and obligations and of any criminal charge against him.*
>
> General Assembly of United Nations, 1948.

The problem of partial observability is a major one.

From a scientific point of view we know that at its most fundamental level, there is uncertainty within our universe. Heisenberg's uncertainty principle concerns the measurement of position and momentum of sub-atomic particles. Uncertainty, therefore, exists within the building blocks of our universe.

> *The more precisely the position is determined, the less precisely the momentum is known in this instant, and vice versa.*
>
> translation from Heisenberg, 1927.

In traditional artificial intelligence approaches uncertainty has often been ignored. The expert systems implemented between the late 1960s and the late 1980s normally worked around a knowledge base which was defined by `if ... then` rules (Crevier, 1993) and while such systems had some notable successes (Shortcliffe, 1976; McDermott, 1982), it became apparent that it was not satisfactory to ignore uncertainty. Purely rule-based approaches do not always provide the required functionality within the system.

Having acknowledged the need for representation of uncertainty, the first step is to decide upon the method of representation. For this we turn to probability theory.

1.2 Probability Theory

The classical or frequentist definition of probability considers probabilities to be the frequency of the truth of a proposition, a. Let us assume that we may query a system as to the truth of a. The probability of a being true is classically defined as:

$$P(a) = \lim_{N \to \infty} \frac{n}{N}, \tag{1.1}$$

where N is the number of times we query the system about a and n is the number of times a was found to be true.

An alternative definition of probability involves the idea of reasonable expectation. This definition of probability was formalised by Cox (1946) using Boolean algebra and two further axioms. The importance of this definition is that it allows us to consider a probability as a degree of belief. As such it provides us with a way of handling uncertainty in a principled manner. This has become known as the Bayesian school of thought. Utilising this approach allows us to make estimates of the chance of rain tomorrow. We are unable to repeat tomorrow a large number of times and count the number of

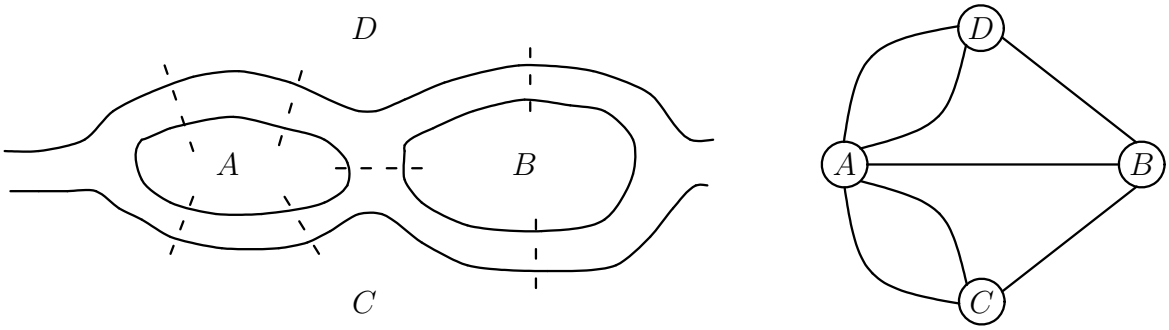


Figure 1.1: These two diagrams show the graphical representation of the Königsberg bridge problem due to Euler, 1736. The map on the left shows the two islands (A and B), the two banks (C and D) and the bridges of Königsberg (dashed lines). The figure on the right shows these features as a graph with connections representing the bridges and nodes representing the islands and banks.

times that it rains, however a trained meteorologist could still estimate the chances of rain as 60%. This is a reflection of their degree of belief.

Probability theory was suggested for handling uncertainty in expert systems as early as 1968 (Gorry and Barnett, 1968) but the lack of efficient algorithms meant that the computational load was too heavy. As a result researchers turned to heuristic methods such as the inclusion of 'certainty factors' within the rules, many of which could be given a probabilistic interpretation (Heckerman, 1986). The interest in probability theory for expert systems was re-aroused by Judea Pearl (Pearl, 1986; Pearl, 1988) who presented efficient algorithms for a particular class of probabilistic models. In this thesis we base our approach to handling uncertainty on probability theory.

1.3 Graphical Models

Now that we have described our approach to handling uncertainty we find it convenient to discuss a useful method of problem representation known as a graphical model. Graph theory dates back to Euler (1736). Euler discussed whether it was possible to walk around his home town of Königsberg crossing each of its bridges once starting and stopping at the same place. Euler used a graphical representation of the problem as shown in Figure 1.1. In the graph the nodes or vertices represent different regions of Königsberg. The edges or connections represent the bridges which connect those different regions. It wasn't until 200 years after Euler published this problem that the term 'graph' was coined by the German mathematician Dénes König. He made a systematic study of the properties of these systems (König, 1936). Graphical models have been used widely to represent topographical problems such as the four colour problem (Cayley, 1879; Appel and Haken, 1977; Appel *et al.*, 1977) and the traveling salesman problem (Hopcroft and Ullman, 1979).

Graphs may also be used to represent relationships between variables. This is how we intend to utilise them in this thesis.

Consider the following illustrative example involving the presentation of circumstantial evidence to a jury. We make use of the following definition of circumstantial evidence

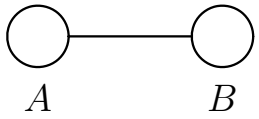


Figure 1.2: A two variable undirected graph showing a relationship between the presence of a suspect, B, and his fingerprints, A, at the scene of a crime.

circumstantial evidence Evidence which may allow a judge or jury to deduce a certain fact from other facts which have been proven. In some cases, there can be some evidence that can not be proven directly, such as with an eye-witness. And yet that evidence may be essential to prove a case. In these cases, the lawyer will provide the judge or juror with evidence of the circumstances from which a juror or judge can logically deduct, or reasonably infer, the fact that cannot be proven directly; it is proven by the evidence of the circumstances; hence, 'circumstantial' evidence. Fingerprints are an example of circumstantial evidence: while there may be no witness to a person's presence in a certain place, or contact with a certain object, the scientific evidence of someone's fingerprints is persuasive proof of a person's presence or contact with an object. – *Duhaime, 1998.*

Let us take binary variable A to be the discovery of a suspect's fingerprints at the scene of a crime. We then take binary variable B to be the suspect's guilt. We can represent the relationship between the discovery of fingerprints and a suspect's guilt graphically. The two variables are obviously not independent. The possibility of finding fingerprints is clearly dependent on whether or not the suspect was ever present at the crime scene. The variables therefore depicted with a connection between them, Figure 1.2. This is a graphical model of the relationship between the events.

1.4 Probabilistic Models and Graphical Models

The assessment of the implications of circumstantial evidence is a good example of a human reasoning process. In the field of probabilistic graphical models Figure 1.2 would be interpreted as representing the probability distribution $P(A, B)$. If this small model were part of a computerised jury system, we would be interested in estimating the values associated with that distribution. Inspired by our realisation that probabilities may be equated to reasonable expectations, we might query an expert witness.

It is generally believed that querying humans about joint distributions produces poor results. To fill in values for the joint distribution we would have to ask questions like: 'What are the chances of a man leaving his fingerprints at and burgling a house?'. Practitioners of expert systems suggest that a more accurate result might be found by asking the question 'What are the chances that a man burgling a house will leave his fingerprints there?'. This is a query about the conditional probability $P(A = \text{yes}|B = \text{yes})$. A relationship which is defined in terms of a conditional probability is represented by an arrow in a graphical model, Figure 1.3. This representation implies that our model is defined in terms of $P(A|B)P(B)$.

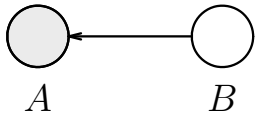


Figure 1.3: A two variable directed graph showing a directed relationship between the events.

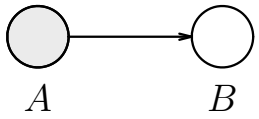


Figure 1.4: Inference in the two variable directed graph showing the arrow reversed.

Given our representative form, let us now fill the distributions with some plausible values. We will assume that a typical burglar is careful. An expert witness might tell us that there is about a 5% chance that he will leave his fingerprints. He might also point out that if the suspect hadn't visited the house then the chances of his fingerprints being there are very small, 1×10^{-4} %. Our conditional probability $P(A = \text{yes}|B = \text{yes})$ is therefore estimated as 0.05 and that of $P(A = \text{yes}|B = \text{no})$ is estimated as 1×10^{-6}. To specify the system fully we will also need to specify $P(B)$. This is our prior belief about B, which can be described as our belief as to whether the suspect committed the burglary before we have seen any evidence.

What about the alternative form of the conditional probability — $P(B|A)$? A question querying about the values of associated with this form would be 'What are the chances that a man leaving his fingerprints at a house is burgling it?'. This question perhaps seems no easier to estimate directly than a question about the joint distribution. However, this is the question our computer juror is trying to answer. Mathematically, the answer may be found through Bayes's rule:

$$P(B|A) \propto P(A|B)P(B), \tag{1.2}$$

where the constant of proportionality is the marginalised probability $P(A)$. The determination of this conditional distribution is sometimes known as inference. It is equivalent to reversing the direction of the arrow and propagating the observed evidence in the opposite direction. The equation is sometimes discussed in the following terms

$$\textbf{posterior} \propto \textbf{likelihood} \times \textbf{prior}. \tag{1.3}$$

Where the *prior* reflects our belief about B before we observe any evidence, the *likelihood* is determined by our model of the events and the *posterior* is our new belief about B after we have observed the evidence A.

For the example we have discussed above, let us assume that reasonable doubt is a 1 % chance[1]. Justice systems normally involve an assumption of innocence, so let us assume that in the absence of

[1]We can get an idea of a figure for reasonable doubt by considering the statement: 'It is better that 10 guilty men go free than one innocent man goes to jail.' This implies that the cost of sending an innocent man to jail is at least ten times that of not convicting a guilty may. This leads to an upper bound on reasonable doubt of 9.09%. Our use of 1% would be equivalent the cost of sending an innocent man to jail being 99 times that of not sending a guilty men to jail.

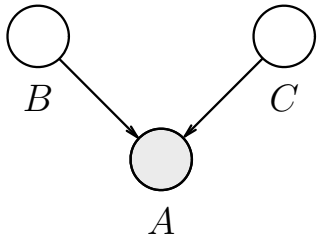


Figure 1.5: The additional variable, C, in this three variable directed graph represents the suspect visiting the scene of crime on legitimate business.

any evidence, our suspect is innocent of the crime, $P(B = \text{yes}) = 0.01$. Following through the process of inference described above we find that

$$P(B = \text{yes}|A = \text{yes}) = \frac{P(A = \text{yes}|B = \text{yes})P(B = \text{yes})}{\sum_B P(A = \text{yes}|B)P(B)} \quad (1.4)$$

$$= 0.9980, \quad (1.5)$$

which gives us a doubt of 0.2 % and a verdict of guilty. Note though that if we assessed the chances of his fingerprints arriving there without him burgling the house as 1×10^{-3} %, ten times higher than figure we used, the result would be 0.9806, or doubt of 1.94 %, enough to get our suspect off the hook.

The form of the conditional probability that we considered to be easier to estimate is consistent with the direction of causality. The fingerprints would be present as a result of the suspect burgling the house. The suspect would not be burgling the house as a result of his fingerprints being present. The arrow in Figure 1.3 may therefore sometimes be interpreted as representing the direction of causality. Much of the literature on belief networks makes causal interpretations (Pearl, 1995), more generally we interpret the arrow as simply showing the direction of the conditional probability we are considering. Often this may be that which is most easily determined, ease of determination may be dependent on the mathematics or on a human expert.

There is a weakness in the model of events we have outlined in our example. We haven't considered that the suspect's fingerprints may have arrived by means of a completely innocent visit to the house.

1.4.1 Explaining Away

Let us make our model more realistic. There is, of course, a chance that our suspect has visited the house on entirely legitimate business, we denote this event C. If this is the case then there is quite a high chance that he will have left his fingerprints. Our new model may be seen in Figure 1.5. This model will be useful in the discussion of a characteristic of directed graphs known as explaining away.

As a jury trying the case, when we are told by the prosecuting barrister that the suspect's fingerprints were discovered at the scene of crime, we are likely to be more inclined to believe that he committed the crime. However when we are then later told by the defendant's barrister that his client is a double glazing salesman who visited the premises two days prior to the crime we become less inclined to believe that he committed the crime. The evidence has been explained away. Explaining away means that although the events are not directly connected in the graph, there *is* a relationship

$P(A = \text{yes}\|B = \text{no}, C = \text{no})$	1×10^{-6}
$P(A = \text{yes}\|B = \text{yes}, C = \text{no})$	0.05
$P(A = \text{yes}\|B = \text{no}, C = \text{yes})$	0.7
$P(A = \text{yes}\|B = \text{yes}, C = \text{yes})$	0.2

Table 1.1: Probability table for the circumstantial evidence toy problem.

between the chances of him having burgled the house and whether he visited the house for innocent reasons. In terms of our symbols, the posterior probability of B is dependent on C if A is observed. If we assign probabilities to the various events we can step through this process of explaining away mathematically.

Inference in this model is slightly more complex. The probabilities now must be specified by a table, Table 1.1. We have taken

$$P(A = \text{yes}|B = \text{yes}, C = \text{yes}) < P(A = \text{yes}|B = \text{no}, C = \text{yes}), \tag{1.6}$$

i.e. we are assuming that if the suspect had burgled or was going to burgle the house, he might be more careful about leaving his fingerprints when he was visiting in any legitimate capacity. We need to define a prior probability for the suspect being at the house on legitimate business. We take $P(C = \text{yes}) = 0.01$, the same as $P(B = \text{yes})$.

We may now go through the process of explaining away numerically. We are interested in the probability of the suspect having visited the house legitimately and the probability of him having burgled it given that the finger prints have been observed, i.e. $P(C = \text{yes}|A = \text{yes})$ and $P(B = \text{yes}|A = \text{yes})$.

$$P(B = \text{yes}|A = \text{yes}) = \frac{\sum_C P(A = \text{yes}|B = \text{yes}, C)P(B = \text{yes})P(C)}{\sum_{C,B} P(A = \text{yes}|B, C)P(B)P(C)} \tag{1.7}$$

$$= 6.92\%. \tag{1.8}$$

A similar calculation leads to $P(C = \text{yes}|A = \text{yes}) = 93.3\%$. In other words, we become more confident that the suspect burgled the house and we become more confident that the suspect visited the house legitimately after we are told that his fingerprints are in the house. If we were then to gain direct evidence that the suspect had visited the house legitimately, our new belief about whether he burgled it, the conditional probability $P(B = \text{yes}|A = \text{yes}, C = \text{yes})$, can be computed as 0.287%. We are now less inclined to belief that the suspect committed the crime than we were before we saw any evidence at all. The presence of the fingerprints has been explained away.

In graph terminology C and B are known as parents of A and A is said to be a child of C and B. The variable C is described as a co-parent of B because they share a child. An important point to absorb from the above is that nodes which share a child are not probabilistically independent if that child is observed, even if there is no direct connection between them. The value of any node in a graph is dependent on its parents, its children and its co-parents. This group of nodes is known as the Markov blanket and is shown in Figure 1.6.

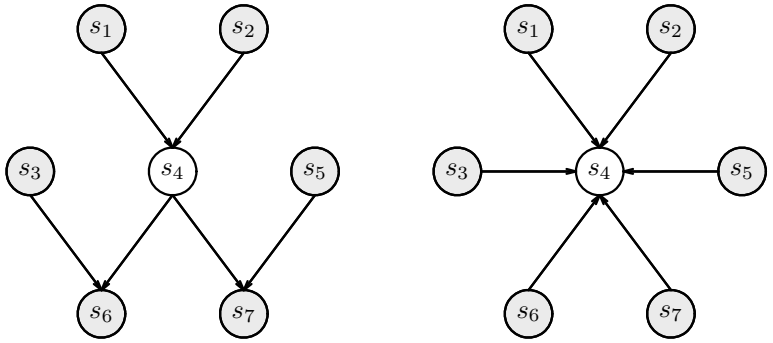


Figure 1.6: The Markov blanket of a node in a directed graph. The statistics of the node s_4 in the middle are dependent on the statistics of those nodes in its Markov blanket (shaded). (a) shows the interactions associated with s_4 and its co-parents in the original graph. (b) shows the conditional distribution of s_4 given all the other nodes in the graph. Note in particular the dependence on the co-parents s_3 and s_5.

1.4.2 Undirected Graphs and Cycles

The first graph we introduced, that of the Königsberg bridge problem in Figure 1.1, contained undirected links, but the graphs we have discussed so far, in the context of probabilistic models, have contained directed links which represented conditional probabilities. Recall that an undirected link is used to represent joint probability distributions. The graph in Figure 1.2, for example, would represent the joint probability $P(A, B)$. One consequence of an undirected relationship is a lack of explaining away, this means that to represent the probability distribution in Figure 1.5 we require an extra connection, Figure 1.7. The Markov blanket of a directed graph contains only those nodes

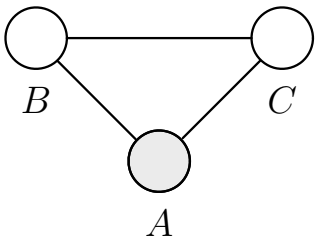


Figure 1.7: A three variable undirected graph. To represent the effect of explaining away in an undirected graph a further connection between the variables B and C is required.

which are neighbours in the graph as shown in Figure 1.8.

A final issue we should mention concerning graphs is the presence of a cycle within the model. A cycle may turn the graph into a dynamical system and the resulting model may well be useful for modeling of dynamical systems. However, the complexities of dealing with cycles are beyond the scope of this thesis. For this reason the directed models we study may be termed directed acyclic graphs (DAG), they are also known as belief networks or Bayesian networks.

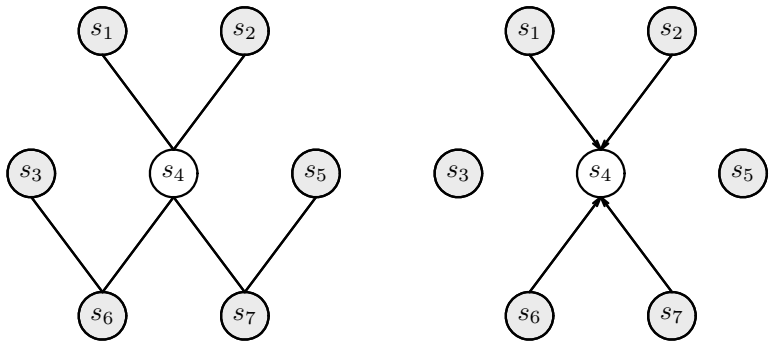


Figure 1.8: The Markov blanket of a node in an undirected graph. The statistics of the node s_4 in the middle are only dependent on those nodes which are neighbours in the graph. (a) shows the interactions associated with the shaded node and its co-parents. (b) shows the conditional distribution of s_4 given all the other nodes in the graph. Now there is no dependence on the nodes s_3 and s_5.

1.4.3 Latent Variables

In the above example whether or not the burglar did in fact commit the crime may be seen as an unobserved or latent variable. We do not have a witness. The fingerprints represent circumstantial evidence because of the relationship between the two events. The process of determining the likely value of the latent state is known as inference.

The latent variables in the circumstantial evidence example were associated with particular events. It is also common practice to introduce latent variables into our model purely to increase the representational power of the model. Factor analysis and hidden Markov models are examples of models which incorporate latent variables in this manner. In this thesis we study models mainly of this type.

In general we define a probabilistic model as representing a distribution of random variables, $\mathcal{S} = \{s_i\}$, which may be continuous or discrete. These variables may be further partitioned into groups of observed (or visible) variables, $\mathcal{V} = \{v_i\}$ and unobserved (or hidden or latent) variables, $\mathcal{H} = \{h_i\}$. The latent variables may correspond to real occurrences which were simply unobserved or may be fictional and exist purely to improve the representational power of the model.

Probabilistic models have been implemented to resolve issues involving uncertainty in materials sciences (Bhadeshia *et al.*, 1995; Gavard *et al.*, 1996; Fujii *et al.*, 1996; Cool *et al.*, 1997), the inferring of goals from free-text queries (Heckerman and Horvitz, 1998), software user modelling (Horvitz *et al.*, 1998), the grading of pork meat (Thodberg, 1996) and filtering junk e-mail (Sahami *et al.*, 1998).

1.5 Machine Learning

So far we have mentioned a few of the issues surrounding human-like decision making in computers, but we have assumed that the knowledge base could be obtained from a human expert. In practice, whilst we might hope to obtain our knowledge base through consultation with a human expert, it may be preferable if our system was able to determine the knowledge base from experience. This is

the aim of machine learning. We aim to optimise the parameters of our model, which in the above example were the probabilities provided by our expert, through the observation of data. To do this we look for a measure of the appropriateness of our model for the data. A commonly used measure is the likelihood of the data.

1.5.1 Maximum Likelihood

The likelihood function for a model without latent variables may be simply written

$$P(\mathcal{V}|\boldsymbol{\theta}) = \prod_{n=1}^{N} P(\mathcal{V}_n|\boldsymbol{\theta}), \tag{1.9}$$

where the data-set consists of a set of instantiations of the visible variables $\mathcal{V}_1, \ldots, \mathcal{V}_N$ which are assumed to be independently obtained and identically distributed.

To obtain the likelihood of a model containing latent variables we must first marginalise these variables.

$$P(\mathcal{V}|\boldsymbol{\theta}) = \prod_{n=1}^{N} \sum_{\mathcal{H}} P(\mathcal{H}, \mathcal{V}_n|\boldsymbol{\theta}), \tag{1.10}$$

where the discrete sum may be replaced with an integral in the case of continuous latent variables.

It is often convenient to deal with the log likelihood which may be written

$$\ln P(\mathcal{V}|\boldsymbol{\theta}) = \sum_{n=1}^{N} \ln \sum_{\mathcal{H}_n} P(\mathcal{H}_n, \mathcal{V}_n|\boldsymbol{\theta}) \stackrel{\text{def}}{=} L(\boldsymbol{\theta}). \tag{1.11}$$

In these equations we are implicitly assuming that it is the same set of variables which are observed in each pattern. The formalism is easily generalised to allow arbitrary combinations of missing and observed variables. From now on we suppress the summations over n to avoid cluttering the notation.

In maximum likelihood learning we maximise Eqn 1.11 with respect to the parameters $\boldsymbol{\theta}$. This approach is sometimes justified by an appeal to a measure of distribution discrepancy known as the asymmetric or Kullback-Leibler (KL) divergence (Kullback and Leibler, 1951; Kullback, 1959). The KL divergence between two distributions $P'(\mathcal{V})$ and $P(\mathcal{V}|\boldsymbol{\theta})$ is mathematically written as

$$\mathrm{KL}(P'||P) = \sum_{\mathcal{V}} P'(\mathcal{V}) \ln \frac{P'(\mathcal{V})}{P(\mathcal{V}|\boldsymbol{\theta})}. \tag{1.12}$$

It can be shown that this difference is always positive unless the two distributions are identical when it is zero. Consider $P'(\mathcal{V})$ to be the true distribution which generated the data, and we take $P(\mathcal{V}|\boldsymbol{\theta})$ to be our approximating distribution. If our data-set is independently sampled from $P'(\mathcal{V})$ then we may make a sample based approximation to the KL divergence,

$$\mathrm{KL}(P'||P) \approx \frac{1}{N} \sum_{n=1}^{N} \ln \frac{P'(\mathcal{V}_n)}{P(\mathcal{V}_n|\boldsymbol{\theta})} \tag{1.13}$$

$$= \frac{1}{N} \sum_{n=1}^{N} \ln P'(\mathcal{V}_n) - \frac{1}{N} \sum_{n=1}^{N} \ln P(\mathcal{V}_n|\boldsymbol{\theta}). \tag{1.14}$$

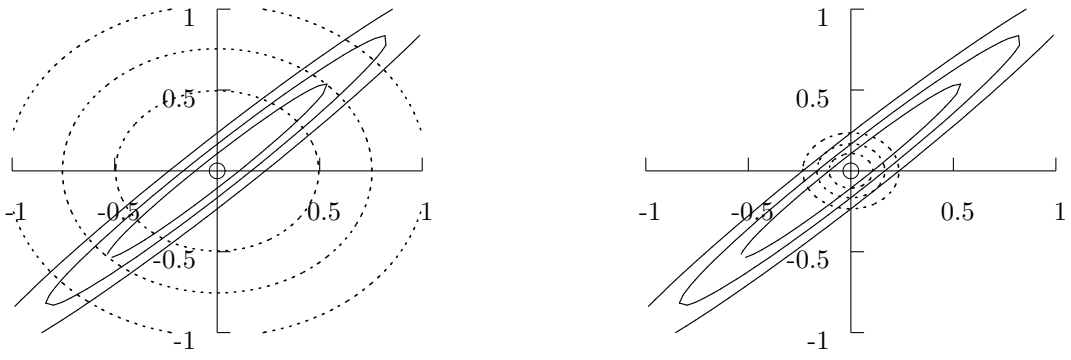


Figure 1.9: The asymmetry of the Kullback-Leibler divergence. These contour plots show the minimum KL divergence solution between two zero mean Gaussians. The first Gaussian, $\mathcal{N}_1$ has a covariance matrix with a very high correlation. Its eigenvalues are 1 and 0.01 and the eigenvectors are at forty-five degrees to the axis. In each plot three contours from the distribution are shown as solid lines. The second Gaussian, $\mathcal{N}_2$, is taken to be spherical and governed by a single variance parameter, σ^2. The variance parameter has been optimised in both diagrams so as to minimise the KL divergence. Three contours from the resulting distributions are shown by dotted lines. In diagram (a) $\int \mathcal{N}_1 \log \frac{\mathcal{N}_1}{\mathcal{N}_2}$ was used. Note the approximation has high probability where the true distribution has high probability. In (b) $\int \mathcal{N}_2 \log \frac{\mathcal{N}_2}{\mathcal{N}_1}$ was used, note the approximation has low probability where the true distribution has low value.

We note that the second term in Eqn 1.14 is the only term which is dependent on our adaptive parameters $\boldsymbol{\theta}$ and is equal to the negative of Eqn 1.11. We may therefore conclude that minimising the sample based approximation to the KL divergence is equivalent to maximising the likelihood.

1.5.2 The Asymmetry of the Kullback-Leibler Divergence

It is convenient at this point to point out some characteristics of the KL divergence. One important property is its asymmetry. The KL divergence we used in the previous section involved expectations under our true distribution $P'(\mathcal{V})$. The alternative form of the KL divergence considers expectations under the approximating distribution $P(\mathcal{V}|\boldsymbol{\theta})$. For the case where we have data points sampled from $P'(\mathcal{V})$ this form of the KL divergence is not calculable, but it will prove of interest later in this chapter.

Taking the expectation under the approximating distribution has the effect of encouraging the approximation to have low value where the true distribution has low value, conversely the alternative form of the KL divergence encourages the approximation to have high value where the true distribution has high value. This results in differing approximations. An example of this is given in Figure 1.9. Here we are attempting to approximate a highly correlated two-dimensional Gaussian with a spherical Gaussian. Depicted are the two results produced by minimising the two differing forms of the KL divergence.

1.5.3 Overview of Maximum Likelihood Learning

The aim of maximum likelihood is to develop a representative model, $\mathcal{M}$, of the data, D, with a particular parameterisation, $\boldsymbol{\theta}$. The model is optimised by considering the likelihood of a data-set,

D. This data-set will consist of N instantiations of some observed variables $\mathcal{V}$. The likelihood of the data-set can be viewed as a function of the parameters, $\boldsymbol{\theta}$, and therefore may be maximised to find a local minima in parameter space, $\hat{\boldsymbol{\theta}}$. The resulting model then has a high likelihood of producing the training data.

So far learning from data appears to be a fairly straight forward procedure. However, we haven't yet addressed the problem of model complexity. For example, it may be that our model $P(\mathcal{V}|\boldsymbol{\theta})$ had less representational power that the true distribution. In Figure 1.9(a), for example, we attempted to model a Gaussian with correlations in its covariance matrix with a spherical Gaussian. Clearly the resulting solution does not capture the full complexity true generating distribution. Error which is exhibited in this situation is sometimes known as bias. The solution to this problem seems obvious, we merely increase the representational power of our model. However, if we go too far down this road we face a further problem. Our model may over-fit the data. Consider the non-linear regression problem in Figure 1.10. The points were produced by the function $0.5 + 0.4\sin(2\pi x)$ with Gaussian noise of standard deviation 0.05 added. In 1.10(a) a linear model is used to interpolate between the data points. The model is clearly not rich enough to capture the complexity of the data. It exhibits bias. In 1.10(b) a polynomial of degree 20 was fitted to the data. The model is now too complex and over-fits the data. The over-fitting error is known as variance. Determining the correct complexity is known as the bias-variance dilemma, (Geman *et al.*, 1992).

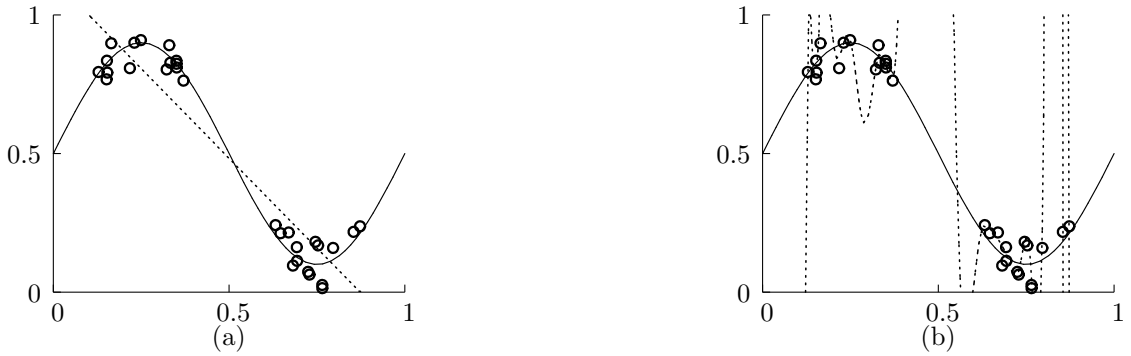


Figure 1.10: The bias-variance dilemma for a regression problem. The figures above show samples from the function $0.5 + 0.4\sin(2\pi x)$ (shown as a solid line) with added Gaussian noise of standard deviation 0.05. In figure (a) the data is approximated using a linear model. A linear model does not have enough complexity to represent the data well and the resulting error is known as bias. In figure (b) a polynomial of degree 20 has been used to fit the data. The polynomial has a high degree of flexibility and therefore 'over-fits' the data. Error which is caused by over-fitting is known as variance.

The principle of selecting the simplest explanation to explain data has a long history. An often quoted proponent of this philosophy is William of Ockham :

> *Pluralitas non est ponenda sine necessitas.*
> William of Ockham, 14th Century Franciscan philosopher

This statement could be loosely translated as 'Complexity is not warranted without necessity'. Because Ockham often used this principle to 'cut out' the arguments of others it became known as Ockham's razor. However, it pre-dates Ockham and may also be found in the writings of Aristotle.

A common approach to determining whether further complexity is warranted is through the use of validation sets. A validation set consists of a separate partition of the training data which is not used directly in the training process, but merely to evaluate the performance of trained models. The model which performs best may then be selected for use. There are two weaknesses with the validation approach. Firstly, if the available data is limited, setting aside a portion of the data for validation is wasteful. This first problem may be circumvented through cross validation (see Stone, 1974; Stone, 1978 and Wahba and Wold, 1975). The second problem though may prove more difficult to resolve. If we are exploring a large class of models and constraining complexity through the use of, for example, regularisation coefficients there may be many coefficients that require setting. Fixing many coefficients using cross validation is unlikely to be practical as it would require the searching of a high dimensional space. Each point in which would have to be evaluated by fitting the parameters to a new model. Instead we look to the Bayesian approach to resolve this issue.

1.5.4 Bayesian Inference

In maximum likelihood learning, we aim to determine the parameters of the model through finding a maxima of the likelihood function. In Bayesian inference we handle the parameters in a different manner. We treat the parameters themselves as stochastic latent variables. The model we have defined provides a conditional probability, e.g. in a continuous system $p(\mathcal{V}, \mathcal{H}|\boldsymbol{\theta})$, and to implement the parameters as stochastic variables we must specify a prior over our parameters. In the case of continuously varying parameters, this prior will take the form of a probability density function, $p(\boldsymbol{\theta})$. The use of this prior is the main area for controversy in the Bayesian approach. It implies that we can specify a probability using only our belief about the variable. As in Section 1.2 we may turn to Cox (1946) to obtain justification for this approach.

This approach is often refered to as Bayesian learning. In our terminology though, no learning of parameters is performed, they are all inferred. We obtain a posterior probability distribution for our parameters through the use of the likelihood, as defined by our model, and the prior, this is identical to how we perform inference in a latent variable model and it may be displayed graphically as in Figure 1.11. As part of determining the posterior we will be required to evaluate the marginalised

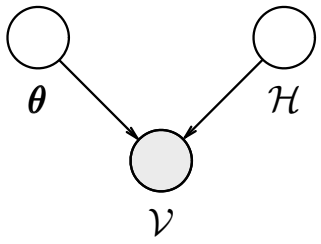


Figure 1.11: Treating the parameters as latent variables. This graph expresses a relationship between variables of the form $p(\mathcal{V}|\mathcal{H}, \boldsymbol{\theta})$.

likelihood. Ignoring the other latent variables for the moment, this will take the form

$$p(\mathcal{V}) = \int p(\mathcal{V}|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}. \tag{1.15}$$

Strictly speaking we should also include the conditioning on the model structure $\mathcal{M}$, giving us $p(\mathcal{V}|\mathcal{M})$ which we term the model likelihood.

Regularisation coefficients in maximum likelihood approaches often manifest themselves as *hyper-parameters* in the Bayesian approach. As such their values may be inferred through a hierarchical Bayesian framework.

1.6 Intractabilities in the Proposed Models

Unfortunately, in discrete systems, the summations over the variables $\mathcal{H}$ often involve an exponentially large number of terms, leading to an algorithm which is computationally intractable. There do exist general exact inference algorithms for these calculations (Jensen, 1996; Cowell, 1998) which take advantage the structure of the model to calculate expectations. However in general and for many distributions of interest the summations required in the marginalisations lead to an algorithm with NP-hard complexity (Dagum and Luby, 1993; Cooper, 1990; Chickering *et al.*, 1994). For systems involving continuous variables,

$$\ln p(\mathcal{V}|\boldsymbol{\theta}) = \sum_{n=1}^{N} \ln \int p(\mathcal{H}_n, \mathcal{V}_n|\boldsymbol{\theta})d\mathcal{H}_n, \tag{1.16}$$

the integrals required are high dimensional, non-linear and, with some exceptions (for example Williams, 1997 and Williams and Rasmussen, 1996) non-analytic.

As a result, for continuous, discrete and hybrid systems we must resort to approximations.

1.7 Sampling Approximations

It may be the case that we are not interested in actual values of the likelihood in Eqn 1.11 and Eqn 1.16, but only in expectations under $P(\mathcal{H}|\mathcal{V})$,

$$\langle f(\mathcal{H})\rangle_{P(\mathcal{H}|\mathcal{V})} = \sum_{\mathcal{H}} f(\mathcal{H})P(\mathcal{H}|\mathcal{V}), \tag{1.17}$$

where $f(\mathcal{H})$ is some arbitrary function of $\mathcal{H}$. We might look then to sampling approximations to make progress. Although it may not be possible to calculate $P(\mathcal{H}|\mathcal{V})$, it may be possible to obtain samples from this distribution. If enough independently distributed samples are obtained, expectations with respect to the posterior distribution of the latent variables may be accurately approximated with a finite sum across a set of independently obtained samples:

$$\langle f(\mathcal{H})\rangle_{P(\mathcal{H}|\mathcal{V})} \approx \frac{1}{S}\sum_{s=1}^{S} f(\mathcal{H}_s), \tag{1.18}$$

where $\{\mathcal{H}_s\}_{s=1}^{S}$ are samples from the posterior distribution $P(\mathcal{H}|\mathcal{V})$. The approximation becomes exact in the limit $S \to \infty$. The major difficulty relies on finding a representative set of samples. Markov chain Monte Carlo algorithms create Markov chains which converge to the desired distribution. Such an approach may be undertaken in discrete systems (for example Neal, 1992; Hinton and Sejnowski, 1983 and Geman and Geman, 1984) or continuous systems (for example Metropolis *et al.*, 1953). In fact the BUGS software (Spiegelhalter *et al.*, 1996) enables Gibbs sampling in almost arbitrary probabilistic models. MacKay (1998) provides a useful introduction to sampling techniques.

The use of sampling is particularly of interest in Bayesian inference. Neal (1996) demonstrated how such systems may be implemented for neural networks through the use of hybrid Monte Carlo sampling. Neal's implementation of a hierarchical Bayesian framework also bypasses the need for determination of the model likelihood in the interests of model selection.

Despite its advantages, problems remain with the sampling approach. It is difficult to judge when the Markov Chain has converged, and despite the utility of a hierarchical Bayesian framework, there may still be a need for estimation of the marginal likelihood. We return briefly to the question of estimating marginal likelihoods in sampling approaches in Chapter 2.

1.8 Functional Approximations to the Posterior

An alternative approach to sampling is to approximate the posterior distribution $p(\mathcal{H}|\mathcal{V})$ with a functional form which makes calculations tractable and is a useful representation of the posterior distribution.

1.8.1 The Laplace Approximation

In continuous systems, the Laplace approximation may be a viable option. The Laplace approximation involves finding a mode of the posterior distribution and constructing a Gaussian approximation using the second order Taylor expansion about this point.

$$p(\mathcal{H}|\mathcal{V}) \approx \frac{|\mathbf{H}|^{\frac{1}{2}}}{(2\pi)^{\frac{W}{2}}} \exp\left\{-\frac{1}{2}(\mathcal{H}-\mathcal{H}^*)^{\mathrm{T}}\mathbf{H}(\mathcal{H}-\mathcal{H}^*)\right\}, \tag{1.19}$$

where the matrix $\mathbf{H}$ is the Hessian about point $\mathcal{H}^*$,

$$\mathbf{H} = -\nabla\nabla(\ln p(\mathcal{H}|\mathcal{V})). \tag{1.20}$$

The Laplace approximation has been successfully applied in neural networks (MacKay, 1992) and other probabilistic models (Lindley, 1980; Chickering and Heckerman, 1996; Cheeseman and Stutz, 1996).

Unfortunately, since the approximation only takes account of the Hessian in the locality of a mode, it will often not be representative of the 'mass' of the posterior distribution, Figure 1.12.

Figure 1.12: The Laplace approximation to distributions with skew and kurtosis. These two plots show the Laplace approximation to (a) a sub-Gaussian distribution $p(x) \propto \exp(-x^4 - 0.1x^2)$ and (b) a skewed distribution $p(x) \propto \exp(-0.7x^4 + 1.5x^3 - 0.1x^2)$. In (a) the approximation over-estimates the tails of the distribution. In (b) it is unaffected by the region of probability to the left of the mode. The distributions are all shown subject to a constants of proportionality.

1.8.2 Variational Inference

Variational methods (Jordan *et al.*, 1998) provide a new approximating framework for inference in probabilistic models. The approximations to the posterior they provide are more responsive to the mass of the posterior than those of the Laplace approximation and often provide a rigourous bound on the marginal likelihood. In this thesis we aim to show how these approaches may be applied in a variety of probabilistic models.

In the general review of variational inference we provide here we have concentrated on discrete probabilities (denoted by capitals) and sums. However, the material covered may be applied equally to continuous probability density functions (which we denote with small letters) where integrals should replace the sums.

For many distributions of interest, the joint probability $P(\mathcal{H}, \mathcal{V}|\boldsymbol{\theta})$ is a Gibbs distribution which takes the following form:

$$P(\mathcal{H}, \mathcal{V}|\boldsymbol{\theta}) = \frac{\exp(-E(\mathcal{H}, \mathcal{V}|\boldsymbol{\theta})}{Z}, \tag{1.21}$$

where Z^{-1} is a normalisation factor and $E(\mathcal{H}, \mathcal{V}|\boldsymbol{\theta})$ is a function of the discrete variables $\{\mathcal{H}, \mathcal{V}\}$ which is dependent on the parameters $\boldsymbol{\theta}$. The variational approach is to apply a refined form of Jensen's inequality which introduces an approximating distribution $Q(\mathcal{H}|\mathcal{V})$:

$$\ln P(\mathcal{V}|\boldsymbol{\theta}) = \ln \sum_{\mathcal{H}} P(\mathcal{H}, \mathcal{V}|\boldsymbol{\theta}) \tag{1.22}$$

$$= \ln \sum_{\mathcal{H}} Q(\mathcal{H}|\mathcal{V}) \frac{P(\mathcal{H}, \mathcal{V}|\boldsymbol{\theta})}{Q(\mathcal{H}|\mathcal{V})} \tag{1.23}$$

$$\geq \sum_{\mathcal{H}} Q(\mathcal{H}|\mathcal{V}) \ln \frac{P(\mathcal{H}, \mathcal{V}|\boldsymbol{\theta})}{Q(\mathcal{H}|\mathcal{V})}. \tag{1.24}$$

Substituting in Eqn 1.21 as the form of our joint probability distribution we obtain

$$\begin{aligned} \ln P(\mathcal{V}|\boldsymbol{\theta}) \quad &\geq \quad -\sum_{\mathcal{H}} Q(\mathcal{H}|\mathcal{V}) E(\mathcal{H},\mathcal{V}|\boldsymbol{\theta}) \\ &\quad -\sum_{\mathcal{H}} Q(\mathcal{H}|\mathcal{V}) \ln Q(\mathcal{H}|\mathcal{V}) \\ &\quad -\ln Z \qquad (1.25) \\ &\stackrel{\text{def}}{=} \quad \mathcal{L}^{+} - \ln Z. \qquad (1.26) \end{aligned}$$

Leaving aside the normalisation term $-\ln Z$ for the moment, we can compute the first two terms of the bound 1.26 efficiently if, firstly, the form of $Q(\mathcal{H}|\mathcal{V})$ is such that its entropy is computable or may be lower bounded in polynomial time and secondly, the form of $E(\mathcal{H}|\mathcal{V},\boldsymbol{\theta})$ is such that its negative expectation with respect to the distribution $Q(\mathcal{H}|\mathcal{V})$, or a lower bound thereupon, is computable in polynomial time.

It is straightforward to calculate that the difference between the bound and the log likelihood is the KL divergence

$$\mathrm{KL}(Q||P) = -\sum_{\mathcal{H}} Q(\mathcal{H}|\mathcal{V}) \ln \frac{P(\mathcal{H}|\mathcal{V})}{Q(\mathcal{H}|\mathcal{V})}. \qquad (1.27)$$

Note that the form of the KL divergence differs from that in Eqn 1.14, as the expectation is under the approximating distribution, i.e. it corresponds to Figure 1.9(b).

The use of the KL divergence in the form Eqn 1.14 is often justified by pointing out that it is more important for the approximating distribution to be close to the data in regions where the data is more likely to be found. Why do we not seek to minimise this form of the KL divergence in the variational approach? First of all minimisation of $\mathrm{KL}(Q||P)$ is naturally ordained through seeking a bound on the marginal likelihood. Secondly, minimising $\mathrm{KL}(P||Q)$ involves maximising the entropy of the true posterior, $P(H|V)$. If this were tractable, then we would not need to turn to approximate methods. Application of a generalised expectation-maximisation algorithm (Dempster *et al.*, 1977) would be possible. Finally we present a more fuzzy argument. Optimisation of $\mathrm{KL}(P||Q)$ would lead to a approximation which may have high probability in regions where the true distribution has low probability, see Figure 1.9(a). In other words, sampled values of $\mathcal{H}$ from this distribution may well be from regions which, in truth, are very unlikely. Conversely utilising $\mathrm{KL}(Q||P)$ leads to an approximation which may have low probability in regions where the true distribution has high probability, see Figure 1.9(b). Sampling from this distribution will produce samples which, whilst they may not be truly representative of the entire posterior distribution, have a reasonable likelihood of being produced by the true posterior. In other words, when the approximating distribution is not well matched to the true posterior, the former case is likely to produce a very implausible inference. The latter case will only produce plausible inference, in legal terms it would find the truth, nothing but the truth, but not the whole truth.

It is true that utilising the form of the KL divergence which considers expectations under the approximating distribution may lead to an approximation which is only locally valid. However it is

still likely to give a more representative approximation than the Laplace approximation. In Figure 1.13 we show variational approximations of Gaussian distributions to the distributions we studied with the Laplace approximation in Section 1.8.1. They can be seen to respond to the mass of the true distributions far better than those in Figure 1.12.

Figure 1.13: The variational Gaussian approximation to distributions with skew and kurtosis. These two plots show a variational Gaussian approximation (dotted line) to (a) a sub-Gaussian distribution $p(x) \propto \exp(-x^4 - 0.1x^2)$ and (b) a skewed distribution $p(x) \propto \exp(-0.7x^4 + 1.5x^3 - 0.1x^2)$. In both (a) and (b) the approximations can be seen to be more responsive to the true mass of the distribution than in Figure 1.12. The distributions are again shown subject to a constant of proportionality.

Q-distribution Selection

In minimising the KL divergence in Eqn 1.27 the choice of the functional form for the Q-distribution is important. When selecting our Q-distribution, we seek a form which is simple enough to fulfill the tractability conditions mentioned above but also allows the bound 1.24 to be tight.

One set of distributions which may satisfy both the required conditions is that of the tractable sub-structures of the model. We consider a tractable sub-structure to be a sub-set of the variables and their interactions for which exact inference is tractable. Ghahramani and Jordan (1997) gave an example of how tractable sub-structures may be implemented explicitly as the Q-distribution for DAGs based on probability tables. By explicit implementation we mean that they define the approximating q-distribution as having a structure identical to that of the sub-structure. We show later in this section that for the graphical models they were considering this sub-structure can be shown to provide the best possible bound under the factorisability constraints they impose.

The recursive formalism of Jaakkola (1997) provides a method of eliminating variables (or nodes in the graphical model) recursively for a large class of graphical models know as *chain graphs*, these are models that contain both undirected and directed links. When variables are eliminated they are replaced with variational parameters until such a point that the remaining variables in the model form a tractable sub-structure.

Rather than explicitly defining a Q-distribution as Ghahramani and Jordan did we can simply

constrain its form by forcing it to be factorisable across sub-sets, $\mathcal{H}_i$, of the latent variables

$$Q(\mathcal{H}) = \prod_i Q(\mathcal{H}_i). \tag{1.28}$$

Using only this independence assumption, we may perform a free form optimisation over the factors of $Q(\mathcal{H})$. Our bound may be re-written

$$\begin{aligned} \ln P(\mathcal{V}) &\geq -\sum_{\mathcal{H}} \prod_i Q(\mathcal{H}_i) \sum_j \ln Q(\mathcal{H}_j) + \sum_{\mathcal{H}} \prod_i Q(\mathcal{H}_i) \ln P(\mathcal{H}, \mathcal{V}) && (1.29) \\ &= -\sum_j \sum_{\mathcal{H}_j} Q(\mathcal{H}_j) \ln Q(\mathcal{H}_j) \\ &\quad + \sum_{\mathcal{H}_j} Q(\mathcal{H}_j) \sum_{\mathcal{H}_{i\neq j}} \prod_{i\neq j} Q(\mathcal{H}_i) \ln P(\mathcal{H}, \mathcal{V}) && (1.30) \\ &\stackrel{\text{def}}{=} \mathcal{L} && (1.31) \end{aligned}$$

which may be rewritten as:

$$\begin{aligned} \mathcal{L} &= -\sum_{i\neq j} \sum_{\mathcal{H}_i} Q(\mathcal{H}_i) \ln Q(\mathcal{H}_i) + \ln Z' \\ &\quad -\mathrm{KL}(Q(\mathcal{H}_j) || \exp \langle \ln P(\mathcal{H}, \mathcal{V}) \rangle_{\prod_{i\neq j} Q(\mathcal{H}_i)} / Z'), \end{aligned} \tag{1.32}$$

where Z' is a normalisation constant. The first two terms are constant in $\mathcal{H}_j$. Therefore a maximisation of the bound in equivalent to maximising the third term, the negative KL divergence, or equivalently minimising the KL divergence. It is known that the minimum KL divergence occurs between two distributions when they are identical. This term is therefore maximised by setting

$$Q(\mathcal{H}_j) \propto \exp \langle \ln P(\mathcal{H}, \mathcal{V}) \rangle_{\prod_{i\neq j} Q(\mathcal{H}_i)}. \tag{1.33}$$

Having chosen the sub-structures, we may redraw our graph as a super-graph showing interactions between the sub-structures. If the resulting interactions are all directed, as in Figure 1.14, then the joint distribution of the resulting graph will be

$$P(\mathcal{S}) = \prod_i P(\mathcal{S}_i | \mathrm{pa}(\mathcal{S}_i)), \tag{1.34}$$

where $\mathrm{pa}(\mathcal{S}_i)$ are the parents of $\mathcal{S}_i$ in the super-graph. Substituting this joint distribution into Eqn 1.33 we obtain

$$Q(\mathcal{H}_j) \propto \exp \left(\left\langle \sum_i \ln P(\mathcal{H}_i | \mathrm{pa}(\mathcal{H}_i)) \right\rangle_{\prod_{k\neq j} Q(\mathcal{H}_k)} \right). \tag{1.35}$$

First we may remove all terms which are not dependent on $\mathcal{H}_j$ which leads to

$$Q(\mathcal{H}_j) \propto \exp \left(\langle \ln P(\mathcal{H}_j | \mathrm{pa}(\mathcal{H}_j)) \rangle_{\prod_{k\neq l} Q(\mathcal{H}_k)} + \sum_{i \in \mathrm{ch}(\mathcal{H}_j)} \langle \ln P(\mathcal{H}_i | \mathrm{pa}(\mathcal{H}_i)) \rangle_{\prod_{k\neq j} Q(\mathcal{H}_k)} \right), \tag{1.36}$$

where $\mathrm{ch}(\mathcal{H}_i)$ denotes variables which are children of node $\mathcal{H}_i$ in the super-graph and $\mathrm{pa}(\mathcal{H}_i)$ denotes variables which are parents of node $\mathcal{H}_i$. Note now that our approximating distribution is dependent

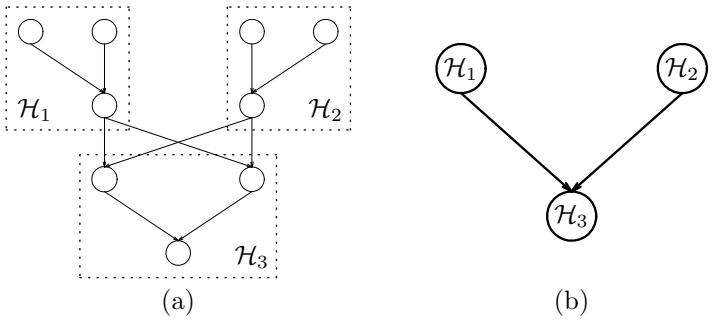


Figure 1.14: A graph and its super-graph after sub-sets of the variables have been selected. (a) shows the original graph with the sub-set selections. (b) shows the resulting super-graph which in this case is a directed graph.

on expectations of variables from the Markov blanket of $\mathcal{H}_j$. The first term in the exponent contains all the parents of node $\mathcal{H}_j$, and the second term includes the children and the co-parents.

The example above showed us that for sub-set selections which lead to directed super-graphs the only parts of the joint distribution in 1.33 which have a dependence on $\mathcal{H}_j$ are those which form the relationships in the Markov blanket of $\mathcal{H}_j$. The same is true for sub-set selections which lead to undirected super-graphs. We may therefore state that the distribution $Q(\mathcal{H}_j)$ will have the same form of the sub-structure of the model $P(\mathcal{H}_j|\text{pa}(\mathcal{H}_j))$ if those variables in its Markov blanket are *conjugate* to $Q(\mathcal{H}_j)$.

The notion of conjugacy here is similar to that for conjugate priors in Bayesian systems. In Eqn 1.36 it is required that the elements of the sum in the second term of the logarithm will have identical functional forms in the variables $\mathcal{H}_j$ as $P(\mathcal{H}_j|\text{pa}(\mathcal{H}_j))$. If this is the case then our variational approximation will have the same functional form as $P(\mathcal{H}_j|\text{pa}(\mathcal{H}_j))$. This is the condition necessary for the explicit imposition of the structure $Q(\mathcal{H}_j)$ (as implemented by Ghahramani and Jordan) to be the best possible under the factorisation across the sub-sets they declare. If we restrict ourselves to DAGs based on probability tables, where conjugacy is guaranteed, the use of the tractable sub-structures as our Q-distribution is the best approximation we can make given the factorisation assumption (Lawrence and Lerner, 2000).

In the context of continuous systems Eqn 1.33 has been used to provide approximations for Bayesian variations of several different models including neural networks (MacKay, 1995a, Barber and Bishop, 1998 and see also Chapter 6, principal component analysis (Bishop, 1999). For discrete systems the formalism has been applied to Bayesian networks by Haft *et al.*, 1999, who restricted themselves to sub-structures containing only one variable, and to hidden Markov models (MacKay, 1997).

There may be circumstances in which we wish to implement a Q-distribution whose form is not a sub-structure of the graph. This may occur when there are no tractable sub-structures in the model. If this were the case we would have to consider Q-distributions whose only justification is that they make the calculations tractable. These Q-distributions may then be explicitly implemented in our model.

Normalisation Factor — Handling the Normalisation Factor in General

Much of the above assumes we may handle the computation of $\ln Z$. This is the case, for example, in DAGs based on probability tables where $Z = 1$ and for Gaussian distributions where the partition function is well known. However, in its most general form the normalisation factor involves marginalisation across all the variables in the model,

$$-\ln Z = -\ln \sum_{\mathcal{H},\mathcal{V}} e^{-E(\mathcal{H},\mathcal{V}|\boldsymbol{\theta})}, \tag{1.37}$$

and may require further treatment to achieve tractability. One approach is to apply the variational framework to this term also by introducing an approximating distribution $Q^-(\mathcal{H},\mathcal{V})$ over the joint space. In this case we obtain an lower bound on $\ln Z$ of the form

$$\begin{aligned} \ln Z &= \ln \left\{ \sum_{\mathcal{H}} \sum_{\mathcal{V}} \exp\left(-E(\mathcal{H},\mathcal{V}|\boldsymbol{\theta})\right) \right\} \\ &\geq -\sum_{\mathcal{H}} \sum_{\mathcal{V}} Q^-(\mathcal{H},\mathcal{V}) E(\mathcal{H},\mathcal{V}) \\ &\quad -\sum_{\mathcal{H}} \sum_{\mathcal{V}} Q^-(\mathcal{H},\mathcal{V}) \ln Q^-(\mathcal{H},\mathcal{V}) \qquad (1.38) \\ &\stackrel{\text{def}}{=} \mathcal{L}^-. \qquad (1.39) \end{aligned}$$

If we select a distribution Q^- for which this computation is tractable our objective function, is now the difference between two lower bounds and therefore is not itself a bound.

$$\mathcal{L}^+ - \mathcal{L}^- \stackrel{\text{def}}{=} \mathcal{F}, \tag{1.40}$$

The absence of a rigourous bound is a consequence of the unrestricted form of the normalisation factor. The difference, $\mathcal{E}$, between $\mathcal{F}$ and the true log likelihood $\ln P(\mathcal{V}|\boldsymbol{\theta})$ is given by

$$\begin{aligned} \mathcal{E} &= \mathcal{F} - \ln P(\mathcal{V}|\boldsymbol{\theta}) \\ &= -\sum_{\mathcal{H}} Q^+(\mathcal{H}|\mathcal{V}) \ln \frac{P(\mathcal{H}|\mathcal{V},\boldsymbol{\theta})}{Q^+(\mathcal{H}|\mathcal{V})} \\ &\quad +\sum_{\mathcal{H}} Q^-(\mathcal{H},\mathcal{V}) \ln \frac{P(\mathcal{H},\mathcal{V}|\boldsymbol{\theta})}{Q^-(\mathcal{H},\mathcal{V})}, \end{aligned} \tag{1.41}$$

Where we have utilised the notation Q^+ to represent the conditional distribution approximation. The error is recognised as the difference between two Kullback-Leibler divergences. One between the true and approximating joint distributions, the other between the true and approximating conditional distributions.

$$\begin{aligned} \mathcal{E} &= \mathrm{KL}(Q^+ \parallel P(\mathcal{H}|\mathcal{V},\boldsymbol{\theta})) \\ &\quad -\mathrm{KL}(Q^- \parallel P(\mathcal{H},\mathcal{V}|\boldsymbol{\theta})). \end{aligned} \tag{1.42}$$

The KL divergence between two distributions is always positive. An approximation $\mathcal{F}$ can therefore be found through minimisation of both KL divergences. This may be achieved by minimising $\mathcal{F}$

with respect to the distribution Q^- and in turn maximising $\mathcal{F}$ with respect to the distribution Q^+. Thus the two distributions Q^- and Q^+ become approximations to, respectively, the joint distribution $P(\mathcal{H}, \mathcal{V}|\boldsymbol{\theta})$ and the conditional distribution $P(\mathcal{H}|\mathcal{V}, \boldsymbol{\theta})$. If a free-form optimisation were to be performed across all forms of Q^+ and Q^- we would recover $Q^+ = P(\mathcal{H}|\mathcal{V}, \boldsymbol{\theta})$ and $Q^- = P(\mathcal{H}, \mathcal{V}|\boldsymbol{\theta})$ and the approximation 1.40 would become exact. As before, we may perform free-form optimisations for these distributions under factorisability constraints. For the conditional distribution this will again lead to Eqn 1.33, for the joint distribution we again make use of a separability assumption,

$$Q(\mathcal{S}) = \prod_i Q(\mathcal{S}_i), \tag{1.43}$$

to perform the free-form optimisation resulting in

$$Q(\mathcal{S}_j) \propto \exp \langle \ln P(\mathcal{S}) \rangle_{\prod_{i \neq j} Q(\mathcal{S}_i)}. \tag{1.44}$$

Normalisation Factor — Upper Bounding Normalising Factors

We mentioned earlier that there will be some models for which the normalisation constant is tractable and we have described how we may deal with situations where the partition function involves a completely intractable marginalisation. There is however a third case which occurs when we may upper bound the logarithm of the partition function. An upper bound on the partition function leads to a lower bound on $\ln P(\mathcal{H}, \mathcal{V})$. This in turn leads to a lower bound on the KL divergence.

For these cases an implicit optimisation may proceed in the normal manner only with the lower bound on the log probability substituted in for $\ln P(\mathcal{H}, \mathcal{V})$. An example of this type of optimisation is given in the next chapter.

Note on Convergence

It is normally the case that we are able to monitor the convergence of the variational bounds on the posterior by observing bound 1.26 or approximation 1.40. We may also utilise these bounds and approximations in learning. Rather than maximising the true log-likelihood we choose instead to maximise a bound on, or an approximation to, the true log-likelihood.

1.9 Overview

Artificial intelligence has been summarised as consisting of three elements, Figure 1.15 (Sage, 1990; Haykin, 1999), representation, reasoning and learning. We are proposing that probabilistic models handle the knowledge representation. Reasoning then takes place within the inference engine and learning may take place through maximisation of the true log-likelihood or of bounds and approximations thereupon. This thesis aims to extend the toolkit available to the artificial intelligence practitioner through the introduction of novel techniques for achieving these three goals. We start in Chapter 2 where we investigate inference within in a DAG known as the sigmoid belief network. We

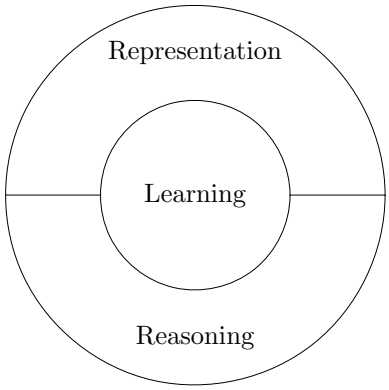


Figure 1.15: The three key components of an artificial intelligence system

build on work by Saul *et al.* (1996) and introduce two new forms for approximating Q distributions, one based on mixture distributions (Jaakkola and Jordan, 1998; Bishop *et al.*, 1998) and a second based on Markov chains (Frey *et al.*, 1998).

In 1943 McCulloch and Pitts produced one of the founding papers of artificial intelligence and neural computing. Their model of the biological neuron, also known as a linear threshold unit, is studied in Chapter 3 by considering the zero temperature limit of the sigmoid belief network. In this limit the graph becomes multi-layer network of Rosenblatt's perceptrons (Rosenblatt, 1962) which are based upon the McCulloch and Pitts neuron. We show that it is possible to perform exact inference in such a system, and by combining our approach with an appropriate learning algorithm, we demonstrate the algorithm on some standard data-sets.

We turn to undirected graphs in Chapter 4, in particular the Boltzmann machine. For this model the unwieldy partition function provides particular problems which have proved insurmountable in previous works (Galland, 1993). We overcome these problems in two ways, we show how judicious initialisation of the variational distribution improves the inference engine and we implement a more advanced variational approximation based on mixture distributions (Lawrence *et al.*, 1998).

We mentioned in Section 1.5.4 how some of the role of the learning process may be adopted by the inference engine through Bayesian techniques. The remainder of the thesis is concerned with this approach. In Chapter 5 we implement a mixture based Q-distribution in a Bayesian neural network (Lawrence and Azzouzi, 1999). We show how some of the problems associated with the Laplace approximation (Figure 1.12) do not arise for variational approximations in these models.

Chapter 2

Variational Inference in Belief Networks

2.1 A Brief Introduction

In the course of the introduction we discussed directed acyclic graphs (DAGs). In particular we focused on particular implementations of these systems based around discrete variables with their inter-relations defined by probability tables. In the case of large densely connected graphs however, such a representation is not plausible. In this chapter we present a graph in which the conditional probabilities are based around potential functions of the parents. We review work of Saul *et al.* (1996) who implemented a mean field approximation in a model of this type to obtain a lower bound. We then show how this bound may be tightened through the use of richer variational approximations based on mixture representations and Markov chains.

2.2 Belief Networks Based on Potential Functions

Consider the graph in Figure 2.1. We may quickly calculate the size of a probability table that would be required to specify a general relationship between the parents of s_7 ($s_1 \dots s_6$) and the node s_7. If the variables are taken to be binary the parents may take on $2^6 = 64$ possible combinations, for each

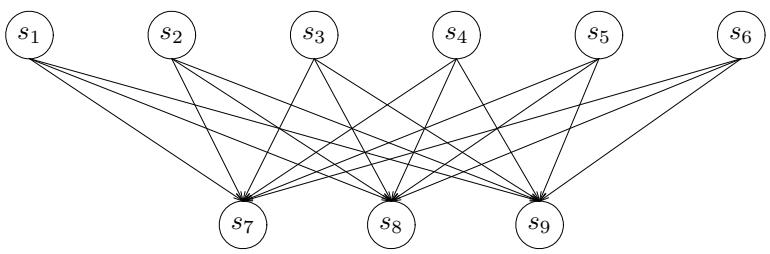
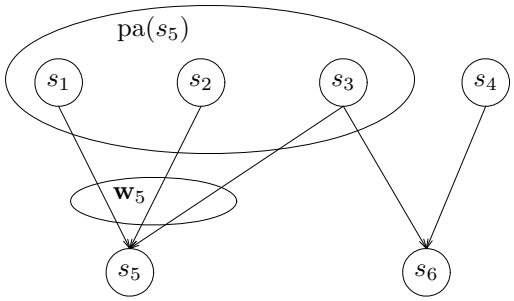


Figure 2.1: A densely connected belief network.



$$P(s_5|\text{pa}(s_5)) \propto \exp(-E(s_5, \mathbf{w}_5, \text{pa}(s_5))$$

Figure 2.2: The use of potential functions in a graph.

of which $P(s_7)$ must be specified. In general a probability table for a node and its P parents, where all nodes have K possible states requires $(K-1)K^P$ values. Obviously for large densely connected belief networks we will run into problems specifying such large tables. An alternative approach is to define the conditional probability of the child node in terms of a potential function. Figure 2.2 shows a form this relationship may take. A parameter w_{ij} is associated with every connection between a parent j and a child i. We may re-write the parameters as a row vector of parameters $\mathbf{w}_i$ associated with each child. The conditional probability of the child variables are then dependent on a function of these parameters and the parent values.

Sigmoid belief networks (Neal, 1992) are a form of belief network in which the conditional probabilities are based on potential functions. Normally the variables are taken to be binary and here we take $s_i \in \{0, 1\}$. For the sigmoid belief network a node's potential function is taken to be as follows:

$$E_i(s_i, \mathbf{w}_i, \text{pa}(s_i)) = \left(\sum_j w_{ij} s_j + w_{i0} \right) s_i \tag{2.1}$$

where w_{i0} is termed a 'bias'. We take $w_{ij} = 0$ if node j is not a parent of node i. Henceforth we shall absorb the 'bias' parameters into the connectivity matrix, $\mathbf{W}$, by introducing an additional variable $s_0 = 1$.

Through normalisation of the exponential of this potential function we may write down the conditional probability of the node

$$P(s_i|\text{pa}(s_i)) = \frac{\exp\left[\left(\sum_j w_{ij} s_j\right) s_i\right]}{1 + \exp\left[\sum_j w_{ij} s_j\right]}. \tag{2.2}$$

The probability of any node being on is therefore simply a logistic sigmoid[1] function of the parent nodes times the connecting parameters or weights.

$$P(s_i = 1|\text{pa}(s_i)) = \sigma\left(\sum_j w_{ij} s_j \right). \tag{2.3}$$

[1]The logistic sigmoid function is given by $\sigma(x) = \frac{1}{1+\exp(-x)}$.

Observe that we could now use these equations and the associated parameters to fill the probability tables we have discussed in Chapter 1. Of course though, the representations we would be able to provide will not be as rich as those we could have implemented with the full probability table.

Taking the product of the conditional probabilities leads to the joint probability of the combined states of the model,

$$P(\mathcal{S}) = \prod_i P(s_i|\text{pa}(s_i)) \tag{2.4}$$

$$= \prod_i \frac{\exp\left[\left(\sum_j w_{ij}s_j\right)s_i\right]}{1+\exp\left[\sum_j w_{ij}s_j\right]}. \tag{2.5}$$

We may partition the variables $\mathcal{S}$ into a visible set $\mathcal{V} = \{v_i\}$ whose values are observed, and a hidden or latent set $\mathcal{H} = \{h_i\}$ whose values are unobserved. The marginal probability of the observed states would then be obtained by summing over the latent variables to give

$$P(\mathcal{V}) = \sum_{\mathcal{H}} P(\mathcal{H}, \mathcal{V}). \tag{2.6}$$

A training set, D, then consists of a set of observations of the visible variables $\mathcal{V}_1, \ldots, \mathcal{V}_N$ and the log likelihood is a sum over patterns

$$\ln P(\mathcal{V}) = \sum_{n=1}^{N} \ln\left\{\sum_{\mathcal{H}_n} P(\mathcal{H}_n, \mathcal{V}_n)\right\}. \tag{2.7}$$

From now on we suppress the summations over n to avoid cluttering the notation.

For densely connected sigmoid belief networks with large numbers of latent nodes the computation of Eqn 2.7 is intractable.

2.3 Gibbs Sampling

The traditional manner in which inference may be performed in such systems is through the use of Gibbs sampling. In systems where we may easily determine the conditional distribution of each variable given all the other variables, $P(h_i|\{h_{j\neq i}\}, \mathcal{V})$, we may obtain samples from the relevant posteriors through Gibbs sampling. In Gibbs sampling for sigmoid belief networks we first initialise the latent nodes of the network with arbitrary values. We then cycle through each latent node in a random order and select a new value for the node by sampling from $P(h_i|\{h_{j\neq i}\}, \mathcal{V})$. Geman and Geman (1984) showed that such a system forms a Markov chain sampler. As such, our sampler has the property that it will converge to $P(\mathcal{H}|\mathcal{V})$ as an equilibrium distribution. The derivatives of Eqn 2.7 with respect to the weights, $\{w_{ij}\}$, may be expressed as expectations under the $P(\mathcal{H}|\mathcal{V})$ and therefore can be approximated using Gibbs sampling. Learning in sigmoid belief networks may therefore be undertaken using Gibbs sampling, and this was the approach described by Neal (1992).

We mentioned some problems with sampling approaches in the introduction. They apply for this model. We recall that it is difficult to determine when the sampler has reached equilibrium and

computing approximations to the marginal likelihood of the observed variables, $P(\mathcal{V})$, may also prove hard (see Neal, 1993, and Raftery, 1996). The marginal likelihood may be of particular interest if we intend to use these models for density modeling. An additional problem which applies in this context is that separate samplers must be run to convergence for each instantiation of the observed variables. Whilst this is also true for the variational approach discussed in the next section, the variational convergence may be monitored. As a result we need not spend any more computational effort than necessary at each observation.

Due to the problems described above, we look to variational methods as an alternative way of performing inference in these systems.

2.4 Variational Inference

Variational methods were introduced in the last chapter. In the context of this directed graph, our objective will be to maximise a lower bound on the logarithm of the marginal likelihood,

$$\mathcal{L} = \sum_{\mathcal{H}} Q(\mathcal{H}|\mathcal{V}) \ln \frac{P(\mathcal{H}, \mathcal{V})}{Q(\mathcal{H}|\mathcal{V})}. \tag{2.8}$$

We may substitute the model's joint distribution, Eqn 2.5, into Eqn 2.8 to obtain a lower bound on the likelihood. To utilise this bound we must make some assumptions about the form of the Q-distribution.

The factorising constraint leading to the simplest form of Q-distribution is the one which forces all variables to be independent,

$$Q(\mathcal{H}) = \prod_i Q(h_i). \tag{2.9}$$

This is known as naïve mean field theory. In this thesis we refer to it simply as mean field theory. This approach was first undertaken for the sigmoid belief network by Saul *et al.* (1996). In the next section we review their approach.

2.4.1 Mean Field Theory

For their approximating Q-distribution Saul *et al.* explicitly implemented a product of Bernoulli distributions of the form

$$Q(\mathcal{H}) = \prod_i \mu_i^{h_i} (1 - \mu_i)^{1 - h_i}, \tag{2.10}$$

in which we have introduced mean field parameters μ_i. A product of Bernoulli distributions is the most general distribution of binary variables which is factorisable across the individual variables. Therefore it will be expected to give the most flexibility and tightest bound in this class. In our review of mean field theory though, we will not explicitly define the form of our Q-distribution but instead look to a free-form optimisation in the manner of that described in Section 1.8.2.

To maintain clear notation in this derivation it is convenient to assume that all the variables are latent. We can later fix the expectation of instantiated variables to their observed values.

First of all we observe that the expectation of the logarithm of the joint distribution,

$$\langle \ln P(\mathcal{H}, \mathcal{V}) \rangle \quad = \quad \sum_{ij} \langle s_i \rangle w_{ij} \langle s_j \rangle - \left\langle \sum_i \ln \left[1 + \exp \sum_j w_{ij} \langle s_j \rangle \right] \right\rangle, \tag{2.11}$$

will not be tractable. The size of the sum in the exponent of the second term is dependent on the number of parents of the node i. We follow Saul *et al.* in the use of an upper bound

$$\langle \ln[1 + \exp(z_i)] \rangle \leq \xi_i \langle z_i \rangle + \ln \langle \exp(-\xi_i z_i) + \exp((1 - \xi_i) z_i) \rangle, \tag{2.12}$$

which leads to a lower bound on the expectation of $\ln P(\mathcal{H}, \mathcal{V})$,

$$\begin{aligned} \langle \ln P(\mathcal{H}, \mathcal{V}) \rangle \quad &\geq \quad \sum_{ij} \langle s_i \rangle w_{ij} \langle s_j \rangle - \sum_{ij} \xi_i w_{ij} \langle s_j \rangle \\ &\quad - \sum_i \ln \left\langle \exp \left(-\xi_i \sum_j w_{ij} s_j \right) + \exp \left((1 - \xi_i) \sum_j w_{ij} s_j \right) \right\rangle \qquad (2.13) \\ &\stackrel{\text{def}}{=} \quad \Lambda(\mathcal{H}, \mathcal{V}), \qquad (2.14) \end{aligned}$$

where the expectations are under all the factors of the Q-distribution. This in turn yields a lower bound on the likelihood of the form

$$\mathcal{L}\left[Q_{\text{mft}}(\mathcal{S})\right] = \Lambda(\mathcal{H}, \mathcal{V}) + \sum_i S(Q(s_i)), \tag{2.15}$$

where $S(P(\ \cdot\ ))$ is the entropy of the distribution $P(\cdot)$. We now perform a free-form optimisation to determine $Q(s_k)$. Substituting the lower bound 2.13 into Eqn 1.33 gives

$$Q(s_k) = \frac{\exp(s_k E'(k, \mathbf{W}, \mathcal{S}))}{1 + \exp(E'(k, \mathbf{W}, \mathcal{S}))}, \tag{2.16}$$

where

$$E'(i, \mathbf{W}, \mathcal{S}) = \sum_j w_{ji} \langle s_j \rangle + \sum_j w_{ij} \langle s_j \rangle + \phi_i \tag{2.17}$$

Here ϕ_i arises from the expectation of the third term in bound 2.13, the calculation of which we have relegated to Appendix A. We can see, therefore, that the expectation of any variable under its own distribution is

$$\langle s_i \rangle = \sigma\left(E'(i, \mathbf{W}, \mathcal{S})\right) \tag{2.18}$$

which is equivalent to μ_i in the product of binomials that Saul *et al.* explicitly implemented. As we might expect, given the flexibility of the binomial distribution they defined, a comparison between our Eqn 2.18 and the update equations for μ_i in the paper of Saul *et al.* reveals the same equation.

In the inference engine Eqn 2.18 is dependent on the expectation of the other variables and the values of the variational parameter ξ_i. The quality of the approximation may be improved by adjustment of each variational parameter ξ_i to maximise the lower bound on the likelihood, intermingled with updates of the variables' expectations $\langle s_i \rangle$ through Eqn 2.18.

Finally, learning may be achieved through evaluation of the lower bound's derivatives with respect to the w_{ij} and w_{i0}. They may then be used in a simple gradient ascent algorithm.

In summary, the algorithm involves presenting training patterns to the network, and for each pattern adapting the $\langle s_i \rangle$ and ξ_i to give the best approximation to the true posterior within the class that conforms to the assumption of factorisation across the variables. The gradients of the log likelihood bound 2.15 with respect to the model parameters w_{ij} and w_{i0} can then be evaluated for this pattern and used to adapt the parameters by taking a step in the gradient direction.

2.4.2 Mixture Based Q-distributions

Although mean field theory leads to a tractable algorithm, the assumption of a completely factorised distribution is a very strong one. In particular, such representations can only effectively model posterior distributions which are uni-modal. Since we expect multi-modal distributions to be common, we seek a richer class of approximating distributions which nevertheless remain computationally tractable. We consider an approach based on mixture representations.

We constrain our variational distributions to be mixtures of factorised distributions,

$$Q_{\text{mix}}(\mathcal{H}) = \sum_{m=1}^{M} Q(m) \prod_i Q(\mathcal{H}_i|m), \tag{2.19}$$

and in the case of sigmoid belief networks. Utilising this new Q-distribution we obtain:

$$\mathcal{L}[Q_{\text{mix}}] = \sum_m Q(m) \left[\Lambda(\mathcal{H}, \mathcal{V}) - \sum_{\mathcal{H}} \prod_i Q(\mathcal{H}_i|m) \ln \prod_i Q(\mathcal{H}_i|m) \right] + I(m, \mathcal{H}), \tag{2.20}$$

where $I(m, \mathcal{H})$ is the mutual information between the component label m and the set of latent variables $\mathcal{H}$, and is given by

$$I(m, \mathcal{H}) = \sum_m Q(m) \sum_{\mathcal{H}} \prod_i Q(\mathcal{H}_i|m) \ln \frac{\prod_i Q(\mathcal{H}_i|m)}{Q_{\text{mix}}(\mathcal{H})}. \tag{2.21}$$

The first term in bound 2.20 is simply a convex combination of standard mean field bounds and hence is no greater than the largest of these and so gives no useful improvement over a single mean field distribution. It is the second term, the mutual information, which characterises the gain in using mixtures. Since $I(m, \mathcal{H}) \geq 0$ the mutual information increases the value of the bound and hence improves its quality.

We may again attempt to perform a free form maximisation of bound 2.20, this time with respect to a factor of each component $Q(\mathcal{H}_i|m)$. However we would encounter problems with the mutual information term. We must first find a tractable lower bound on the mutual information.

Smoothing Distributions

As it stands, the mutual information itself involves a summation over the configurations of latent variables, and so is computationally intractable. In order to be able to treat it efficiently we follow

the approach of Jaakkola and Jordan (1998) and we rewrite the mutual information Eqn 2.21 in the form

$$\begin{aligned} I(m,\mathcal{H}) &= \sum_m \sum_{\mathcal{H}} Q(m)Q(\mathcal{H}|m)\ln R(\mathcal{H}|m) - \sum_m Q(m)\ln Q(m) \\ &\quad - \sum_m \sum_{\mathcal{H}} Q(m)Q(\mathcal{H}|m)\ln\left\{\frac{R(\mathcal{H}|m)}{Q(m)}\frac{Q_{\rm mix}(\mathcal{H})}{Q(\mathcal{H}|m)}\right\}, \end{aligned} \tag{2.22}$$

where we have introduced a set of 'smoothing' distributions $R(\mathcal{H}|m)$. It is easily verified that Eqn 2.22 is equivalent to Eqn 2.21 for arbitrary $R(\mathcal{H}|m)$. We next make use of the inequality

$$-\ln x \geq -\lambda x + \ln\lambda + 1 \tag{2.23}$$

to replace the logarithm in the third term in Eqn 2.22 with a linear function (conditionally on the component label m). This yields a lower bound on the mutual information given by $I(m,\mathcal{H}) \geq \mathcal{I}(m,\mathcal{H})$ where

$$\begin{aligned} \mathcal{I}(m,\mathcal{H}) &= \sum_m \sum_{\mathcal{H}} Q(m)Q(\mathcal{H}|m)\ln R(\mathcal{H}|m) - \sum_m Q(m)\ln Q(m) \\ &\quad - \sum_m \lambda_m \sum_{\mathcal{H}} R(\mathcal{H}|m)Q_{\rm mix}(\mathcal{H}) + \sum_m Q(m)\ln\lambda_m + 1. \end{aligned} \tag{2.24}$$

With $\mathcal{I}(m,\mathcal{H})$ substituted for $I(m,\mathcal{H})$ in bound 2.20 we again obtain a rigourous lower bound on the true log likelihood given by

$$\mathcal{L}_\lambda[Q_{\rm mix}(\mathcal{H})] = \sum_m Q(m)\left[\Lambda(\mathcal{H},\mathcal{V}) - \sum_{\mathcal{H}}\prod_i Q(\mathcal{H}_i|m)\ln\prod_i Q(\mathcal{H}_i|m)\right] + \mathcal{I}(m,\mathcal{H}). \tag{2.25}$$

The summations over latent configurations $\mathcal{H}$ can be performed analytically if we assume that the smoothing distributions $R(\mathcal{H}|m)$ factorise across the latent variables sub-sets,

$$R(\mathcal{H}|m) = \prod_i R(\mathcal{H}_i|m). \tag{2.26}$$

If we were to take $R(\mathcal{H}|m) \propto Q(\mathcal{H}|m)/Q_{\rm mix}(\mathcal{H})$ and maximise over the variational parameters λ_m, we would recover the mutual information exactly. However such a selection would lead to no reduction in computational tractability. In particular, we have to consider the following two summations over latent variable configurations

$$\sum_{\mathcal{H}} R(\mathcal{H}|m)Q(\mathcal{H}|m') = \prod_i \sum_{\mathcal{H}_i} R(\mathcal{H}_i|m)Q(\mathcal{H}_i|m') \stackrel{\rm def}{=} \pi_{R,Q}(m,m') \tag{2.27}$$

$$\sum_{\mathcal{H}} Q(\mathcal{H}|m)\ln R(\mathcal{H}|m) = \sum_i \sum_{\mathcal{H}_i} Q(\mathcal{H}_i|m)\ln R(\mathcal{H}_i|m). \tag{2.28}$$

We note that the left hand sides of Eqn 2.27 and Eqn 2.28 represent sums over exponentially many latent configurations, while on the right hand sides these have been re-expressed in terms of expressions requiring only polynomial time to evaluate by making use of the factorisation of $R(\mathcal{H}|m)$.

It should be stressed that the introduction of a factorised form for the smoothing distributions still yields an improvement over standard mean field theory. To see this, we note that if $R(\mathcal{H}|m) = \text{const.}$ for all $\{\mathcal{H}, m\}$ then $I(m,\mathcal{H}) = 0$, and so optimisation over $R(\mathcal{H}|m)$ can only improve the bound.

The resulting bound may now be utilised to perform free-form optimisations over $Q(\mathcal{H}_i|m)$. Once again, to simplify notation, we will assume all variables are latent. A free-form optimisation of each component in the mixture distribution now leads to:

$$Q(s_k|m) \propto \exp\left(E'(k, \mathbf{W}, \mathcal{S}, m)s_k + \ln R(\mathcal{S}_k|m) - \lambda_m R(\mathcal{S}_k|m)\right), \tag{2.29}$$

where the extra index in the $E'(k, \mathbf{W}, \mathcal{S}, m)$ specifies the component. Since we are focusing on mixtures of mean field distributions we take each sub-structure, $\mathcal{S}_i$, to consist of only one variable and obtain

$$Q(s_k|m) = \frac{\exp\left(E''(k, \mathbf{W}, \mathcal{S}, m)s_k\right)}{1 + \exp\left(E''(k, \mathbf{W}, \mathcal{S}, m)\right)}, \tag{2.30}$$

where

$$\begin{aligned} E''(k, \mathbf{W}, \mathcal{S}, m) &= E'(k, \mathbf{W}, \mathcal{S}, m) + \ln \frac{R(s_k = 1|m)}{R(s_k = 0|m)} \\ &\quad -\lambda_m \left[R(s_k = 1|m) - R(s_k = 0|m)\right] \end{aligned} \tag{2.31}$$

A variables expectation under a single component of its own Q-distribution can now be determined through an update equation of the form

$$\langle s_k \rangle_{Q(s_k|m)} = \sigma\left(E''(k, \mathbf{W}, \mathcal{S}, m)\right). \tag{2.32}$$

We note that had we considered sub-sets which consisted of more than one variable we may not have been able to fully represent our smoothing distributions, the table could be too large. We would then have to have considered a form based on potential functions.

Optimising the Variational parameters

We are now provided with an equation with which we may determine the approximate expectations of $\mathcal{S}$. The update equation is dependent on other variational parameters. In order to obtain the tightest bound within the class of approximating distributions we also must maximise the bound 2.20 with respect to the component mean field distributions, the mixing coefficients $Q(m)$, the smoothing distributions $R(\mathcal{H}|m)$ and the variational parameters ξ_{im} and λ_m. We now consider the last four of these in turn.

Consider the optimisation with respect to the mixing coefficients $Q(m)$. Since all of the terms in bound 2.25 are linear in $Q(m)$, except for the entropy term, we can write

$$\mathcal{L}_\lambda[Q_{\text{mix}}(\mathcal{H})] = \sum_m Q(m)(-E_m) - \sum_m Q(m) \ln Q(m) + 1 \tag{2.33}$$

where we have used bound 2.24 and defined

$$\begin{aligned} -E_m &= \mathcal{L}[Q_{\text{mft}}(\mathcal{H}|m)] + \sum_{\mathcal{H}} Q(\mathcal{H}|m) \ln R(\mathcal{H}|m) \\ &\quad + \sum_k \lambda_k \sum_{\mathcal{H}} R(\mathcal{H}|k) Q(\mathcal{H}|m) + \ln \lambda_m. \end{aligned} \tag{2.34}$$

Maximising bound 2.33 with respect to $Q(m)$, subject to the constraints $Q(m) \geq 0$ and $\sum_m Q(m) = 1$, we see that the mixing coefficients are given by the Boltzmann distribution

$$Q(m) = \frac{\exp(-E_m)}{\sum_k \exp(-E_k)}. \tag{2.35}$$

We next maximise the bound bound 2.25 with respect to the smoothing marginals $R(h_j|m)$. Differentiating with respect to the smoothing marginal we may obtain a fixed point equation of the form

$$R(h_j|m) = \frac{Q(m)Q(h_j|m)}{\lambda_m} \left[\sum_k \alpha_k \pi^j_{R,Q}(m,k) Q(h_j|k) \right]^{-1} \tag{2.36}$$

in which $\pi^j_{R,Q}(m,k)$ denotes the expression defined in Eqn 2.27 but with the j term omitted from the product.

Unfortunately it is not possible to determine a fixed point equation for the variational parameters introduced in bound 2.12, ξ_{im}. We therefore follow Saul *et al.*, 1996 and optimise them independently using one-dimensional function minimisation techniques.

Finally, we optimise the bound with respect to the λ_m, to give

$$\frac{1}{\lambda_m} = \frac{1}{Q(m)} \sum_k \pi_{R,Q}(m,k). \tag{2.37}$$

Since the various parameters are coupled, and we have optimised them individually keeping the remainder constant, it will be necessary to maximise the lower bound iteratively until some convergence criterion is satisfied. Having done this for a particular instantiation of the visible nodes, we can then determine the gradients of the bound with respect to the parameters governing the original belief network, and use these gradients for learning.

2.4.3 Markov Chain Q-distribution

We now consider an alternative approach to that of determining the variational solutions through free-form optimisation. In this section we consider an explicitly defined variational distribution which is described by a Markov chain. The Markov chain model is more general than the factorised distribution of mean field theory and provides an alternative approach to the mixture of mean field models described in the previous section. We consider Markov chains that decouple the different layers in the network. We effectively use one Markov chain per layer of latent nodes. We can express the variational model graphically and compare it with alternative approaches as shown in Figure 2.3. This form of variational distribution requires that a particular ordering of the latent nodes be chosen. However, in the types of networks that we are considering there is an interchange symmetry with respect to the latent nodes *before* learning. Exchanging the labels of any two latent nodes gives an equivalent network structure with an identical joint distribution over the observed variables. The specific choice of ordering effectively breaks this symmetry.

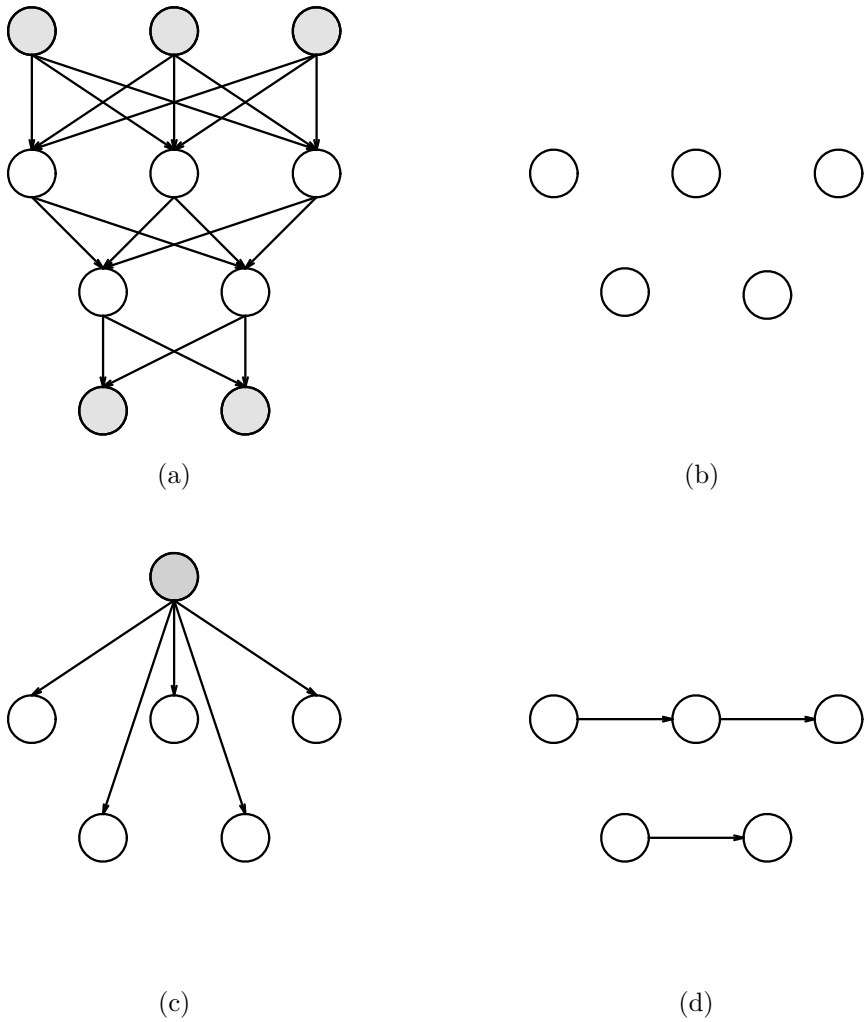


Figure 2.3: Some possible variational distributions. (a) A densely connected layered belief network. Observed input and output nodes are grey, while latent nodes are white. (b) Mean field theory assumes a fully factorised distribution over latent nodes. (c) In the mixed mean field approach there is an additional latent variable governing component selection shown in dark grey. (d) The Markov chain Q-distribution.

We consider a sigmoid belief network with $\mathcal{N}$ nodes, we therefore consider a Q-distribution of the following form.

$$Q(\mathcal{H}|\mathcal{V}) = Q_1(s_1)\prod_{i=2}^{\mathcal{N}} Q_i(s_i|s_{i-1}). \tag{2.38}$$

Where for the sake of notational simplicity we have assumed that the Markov chain weaves it way through all the layers of the network and $Q_i(s_i|s_{i-1})$ is

$$Q_i(s_i|s_{i-1}) = \left\{\mu_{i1}^{s_i}(1-\mu_{i1})^{1-s_i}\right\}^{s_{i-1}}\left\{\mu_{i0}^{s_i}(1-\mu_{i0})^{1-s_i}\right\}^{1-s_{i-1}}. \tag{2.39}$$

If $s_i \in \mathcal{V}$, we fix $\mu_{i0} = \mu_{i1}$ to the observed value. If s_i is the first unit in a layer, we decouple it from the previous layer by constraining $\mu_{i0} = \mu_{i1}$.

We therefore have a Markov transition matrix for s_i given s_{i-1}

$$\mathbf{A}_i \equiv \begin{bmatrix} 1-\mu_{i0} & 1-\mu_{i1} \\ \mu_{i0} & \mu_{i1} \end{bmatrix}. \tag{2.40}$$

To account for the beginning of the chain, we define $\mathbf{A}_0 \equiv \begin{bmatrix} 0 \\ 1 \end{bmatrix}$.

Evaluation of the Bound

To compute the bound on the likelihood we again must evaluate expectations of s_i, $s_i s_j$, $e^{-\xi_i z_i}$ and $e^{(1-\xi_i)z_i}$ where $z_i = \sum_j w_{ij} s_j$.

The marginal distribution of the latent variable s_i may be evaluated using the standard forward algorithm:

$$\begin{aligned} Q(s_i|s_{j\neq i}) &= \sum_{s_1} \cdots \sum_{s_{i-1}} Q_i(s_i|s_{i-1}) Q_{i-1}(s_{i-1}|s_{i-2}) \ldots Q_2(s_2|s_1) Q_1(s_1) \\ &= \sum_{s_{i-1}} Q_i(s_i|s_{i-1}) \sum_{s_{i-2}} Q_{i-1}(s_{i-1}|s_{i-2}) \cdots \sum_{s_1} Q_2(s_2|s_1) Q_1(s_1). \end{aligned} \tag{2.41}$$

We may express this as a series of matrix multiplications:

$$\begin{bmatrix} Q(s_i = 0|\mathcal{V}) \\ Q(s_i = 1|\mathcal{V}) \end{bmatrix} = \prod_{j=i}^{0} \mathbf{A}_j, \tag{2.42}$$

where $\prod_{j=i}^{0}$ indicates that matrix $\mathbf{A}_i$ appears on the far left and vector $\mathbf{A}_0$ appears on the far right. The computation of Eqn 2.42 takes time which is linear in the length of the chain, and hence is tractable. We then have

$$\langle s_i \rangle = [0 \;\; 1] \prod_{j=i}^{0} \mathbf{A}_j, \tag{2.43}$$

where the row vector $[0 \;\; 1]$ extracts the second component of the distribution.

Now consider the computation of $\langle s_i s_j \rangle$. As a consequence of the layered structure of the belief network, this expectation is only required if nodes i and j are in different layers. Since we constrain the layers to be decoupled in the Markov chain, we need only consider i and j for which $\langle s_i s_j \rangle = \langle s_i \rangle \langle s_j \rangle$.

We will also be faced with the evaluation of $\left\langle e^{-\xi_i z_i} \right\rangle$ and $\left\langle e^{(1-\xi_i) z_i} \right\rangle$. Once again these are easily evaluated using a forward propagation along the chain:

$$\left\langle e^{-\xi_i z_i} \right\rangle = \left\langle \prod_{j=0}^{\mathcal{N}} e^{-\xi_i w_{ij} s_j} \right\rangle \tag{2.44}$$

$$\begin{aligned} &= \sum_{s_\mathcal{N}} e^{-\xi_i w_{i\mathcal{N}} s_\mathcal{N}} \sum_{s_{\mathcal{N}-1}} Q(s_\mathcal{N}|s_{\mathcal{N}-1}) e^{-\xi_i w_{i\mathcal{N}-1} s_{\mathcal{N}-1}} \ldots \\ &\quad \times \sum_{s_1} Q(s_2|s_1) e^{-\xi_i w_{i1} s_1} Q(s_1) e^{-\xi_i w_{i0}} \end{aligned} \tag{2.45}$$

which, defining

$$\mathbf{K}_{ij} \equiv \begin{bmatrix} 1 & 0 \\ 0 & e^{-\xi_i w_{ij}} \end{bmatrix} \tag{2.46}$$

$$\tilde{\mathbf{K}}_{ij} \equiv \begin{bmatrix} 1 & 0 \\ 0 & e^{(1-\xi_i) w_{ij}} \end{bmatrix}, \tag{2.47}$$

leads to

$$\left\langle e^{-\xi_i z_i} \right\rangle = [1 \;\; 1] \prod_{j=\mathcal{N}}^{0} \mathbf{K}_{ij} \mathbf{A}_j \tag{2.48}$$

$$\left\langle e^{(1-\xi_i) z_i} \right\rangle = [1 \;\; 1] \prod_{j=\mathcal{N}}^{0} \tilde{\mathbf{K}}_{ij} \mathbf{A}_j. \tag{2.49}$$

Finally, we will need to evaluate the entropy of the approximating distribution, $S(Q) = -\langle \ln Q(\mathcal{H}|\mathcal{V}) \rangle$. To do this we make use of the Markov chain property to obtain

$$S(Q) = S(\mu_{11}) + \sum_{i=2}^{\mathcal{N}} \left[\langle s_{i-1} \rangle \, S(\mu_{i1}) + (1 - \langle s_{i-1} \rangle) S(\mu_{i0}) \right], \tag{2.50}$$

where $S(\mu_{ij})$ is the binary entropy of μ_{ij}. The time needed to compute the likelihood bound scales linearly with the connectivity.

The inference step consists of optimisation of the bound $\mathcal{L}$ in the parameters of the Q-distribution. We achieve this through gradient optimisation methods. The necessary gradients are listed in Appendix B.

Learning consists of optimisation of the parameters $\mathbf{W}$. Once again this may be achieved through gradient based optimisation methods, again the gradients are listed in Appendix B.

2.5 Results

We now assess the performance of the two formalisms by considering their application to the sigmoid belief network.

2.5.1 Inference

We first investigate the extent to which the our approaches yield improvements in the lower bound on the log likelihood compared with standard mean field theory. To do this, we followed Saul *et al.* (1996) and considered layered networks having 2 nodes in the first layer, 4 nodes in the second layer and 6 nodes in the third layer, with full connectivity between layers. In all cases the six final-layer nodes were considered to be visible and have their states clamped at zero. We generated 5000 networks with parameters $\{w_{ij}, w_{i0}\}$ chosen randomly with uniform distribution over $(-1, 1)$. The number of latent variable configurations is $2^6 = 64$ and was sufficiently small that the true log likelihood could be computed directly by summation over the latent states. We can therefore compare the value of the lower bound $\mathcal{L}$ with the true log likelihood L, using the relative error $(L - \mathcal{L})/L$. Figure 2.4 shows histograms of the relative error for various numbers of mixture components and for the Markov chain distribution, together with the mean values taken from the histograms. The mean values for the mixture distributions are also plotted in Figure 2.5. These show a systematic improvement in the quality of the approximation as the number of mixture components is increased.

We also explored the effect of increasing the span, W, across which there weights were distributed. We studied the quality of the bound for W between 0.5 and 4 at intervals of 0.5 where the weights

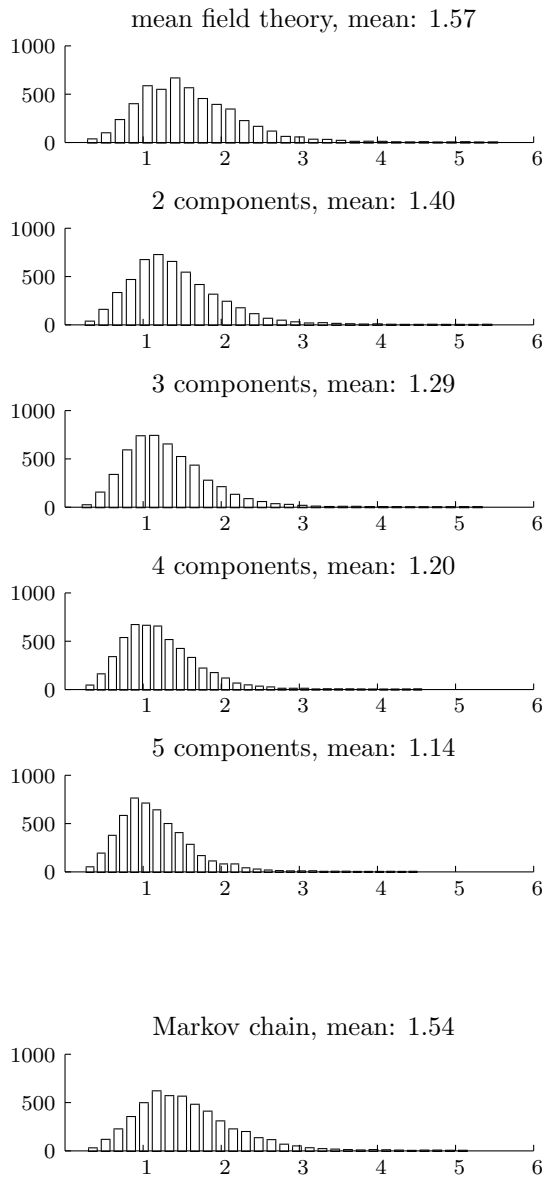


Figure 2.4: Plots of histograms of the relative error between the true log likelihood and the lower bound, for various numbers of mixture components, and for the Markov chain Q-distribution.

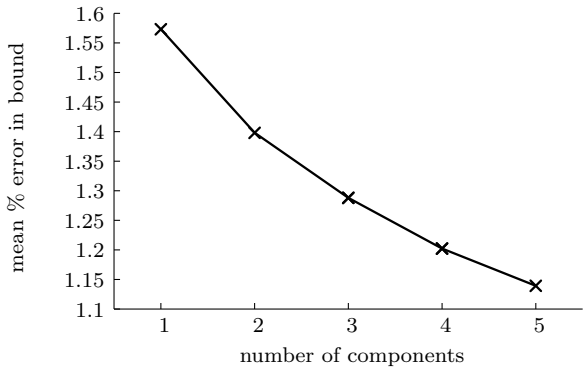


Figure 2.5: Graph of mean percentage relative error for different numbers of components in the mixture distribution.

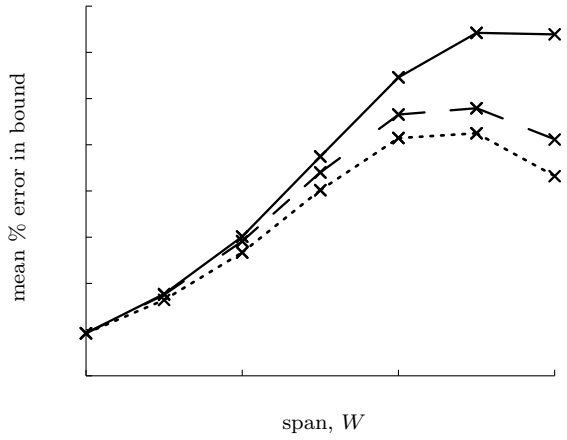


Figure 2.6: Graph of mean percentage relative error for different 'spans' of the weights in the random networks. The solid line represents mean field theory. The dashed line is the Markov chain Q-distribution and the dotted line is a two component mixture model.

were chosen from a uniform distribution over $(-W, W)$. We generated 1000 networks and investigated the performance of a mixture approximation containing two components and the Markov chain approximation. The results are shown in Figure 2.6. We note here how as the span starts to increase the mean error in the bound also increases. However, the relative performance benefit for using the more complex variational distributions increases. This is perhaps due to greater complexities in the posterior as the weight parameters increase in magnitude.

Note that in all the experiments we undertook for the Markov chain approximation, the scaled conjugate gradients algorithm we used appeared to be susceptible to local minima. As a result we initialised the variational parameters with a mean field theory solution before performing the optimisation.

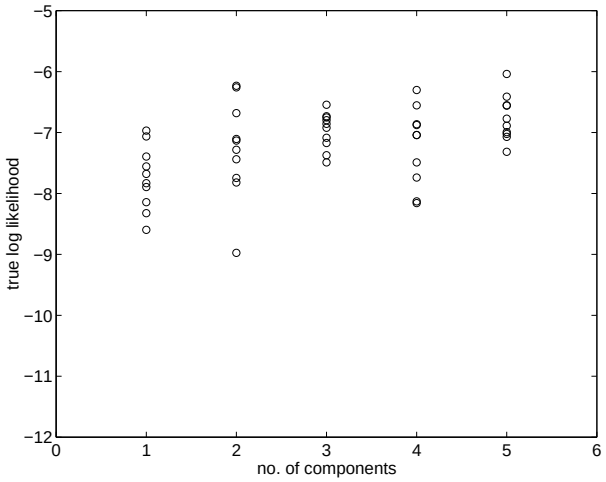


Figure 2.7: True log likelihood (divided by the number of patterns) versus the number of mixture components, M, for the 'bars' problem indicating a systematic improvement in performance as M is increased.

2.5.2 Learning

Whilst an improvement in performance was seen with all the proposed Q-distributions, the mixture approach seems to generally outperform the Markov chain approach, even when only two components are used. Therefore we restrict our studies of learning to only consider the impact of using mixture distributions. To explore this we use a small-scale problem introduced by Hinton *et al.* (1995) involving binary images of size 4×4 in which each image contains either horizontal or vertical bars with equal probability, with each of the four possible locations for a bar occupied with probability 0.5. We trained networks having architecture 1–8–16 using distributions having between 1 and 5 components. This structure was chosen as the smallest structure of sigmoid belief network that can represent the probability distribution which generates the data faithfully. The node in the first layer determines whether bars should be orientated horizontally or vertically whilst those in the second layer determine the positioning of the bars. Randomly generated patterns were presented to the network for a total of 500 times, and the μ_{mi} and ξ_{mi} were initialised from a uniform distribution over $(0, 1)$. Again the networks are sufficiently small that the exact log likelihood for the trained models can be evaluated directly. Figure 2.7 shows a plot of the true log likelihood versus the number of components in the mixture, M, for a set of experiments in which, for each value of M, the model was trained 10 times starting from different random parameter initialisations. These results indicate that, as the number of mixture components is increased, the learning algorithm is able to find a set of network parameters having a larger likelihood, and hence that the improved flexibility of the approximating distribution is indeed translated into an improved training algorithm. However the improvement in performance is perhaps not great enough to justify the extra computational effort required.

2.6 Discussion

In this chapter we reviewed variational inference in the context of sigmoid belief networks. We showed how the mean field approximation could be improved upon by the consideration of more complex Q-distributions. The results for approximate inference were particularly promising. In the case of the random networks with a span of 1 the use of five component mixture distributions resulted in a 27% decrease in the error. As span was increased the performance gain was seen to increase also. The Markov chain Q-distribution did not really demonstrate its worthiness over a two component approximation in the context of inference.

Unfortunately the results did not translate into dramatic improvements in learning. We suggest that this may be because in the learning task we studied the mean field distribution was able to sufficiently represent the true posterior.

A weakness with the mixture approach is the computational performance penalty associated with the mixture distribution, which is approximately linear in the number of components. There may be models, for example more sparsely connected graphs, where the utility of the mixture approach might justify the computational penalties.

Although we have concentrated on sigmoid belief networks in this chapter, our results may be applied in the context of other directed acyclic graphs such as noisy-OR belief networks (Pearl, 1988) or standard belief networks based on probability tables.

The mixtures framework we have described is not restricted to directed acyclic graphs. We shall see in Chapter 4 how it may be applied in the context of an undirected graph known as the Boltzmann machine and in Chapter 5 we develop a continuous version and implement it in a neural network.

In related work Haft *et al.* (1999) implemented a mixture formalism in the context of table-based DAGs. They dealt with the entropy of the mixture distribution using a 'minimum overlap assumption'. In other words they assume that there is no overlap between the components of the mixture distribution. This assumption leads to an upper bound on the entropy (see Appendix C, Eqn C.14) though and therefore destroys the integrity of the lower bound on the likelihood. Their work also highlights how discovering different modes in the network helps with interpretation of the network. The interpretability of the formalism is an advantage of the mixture representation over other forms of Q-distribution.

A further point of interest concerns the computation of the marginal likelihood (Eqn 2.7) through sampling methods. For discrete systems we may use a set of samples to compute lower bounds on the marginal. If we have S samples we simply consider our set of samples to be a mixture approximation to the posterior where the mixing coefficients are simply $1/S$. We can now calculate a lower bound on the marginal using the mixture formalism. Additionally if we replace duplicated samples with one sample with a coefficient of K/S, where K is the number of duplications, we may use the minimum

overlap assumption to compute the mutual information as

$$I = -\sum_{m=1}^{S} Q(m) \ln Q(m). \tag{2.51}$$

This avoids the use of the smoothing distribution.

Acknowledgements

Some of the work undertaken in this chapter was based on papers written in collaboration with Christopher M. Bishop, Tommi Jaakkola and Michael I. Jordan (Bishop *et al.*, 1998). The derivation of the mixture update equations are my own. The work on Markov chain distribution is based on a collaboration with Brendan J. Frey and Christopher M. Bishop and is described in a technical report (Frey *et al.*, 1998). The original idea of utilising Markov Chains is due to Brendan J. Frey, but the implementation is my own.

Chapter 3

Linear Threshold Units

3.1 A Brief Introduction

Linear threshold units were originally proposed as models of biological neurons. They were widely studied in the context of the perceptron (Rosenblatt, 1962). Due to the difficulties of finding a general algorithm for networks with hidden nodes, they never passed into general use. In this chapter we derive an algorithm in the context of graphical models and show how it may be applied in multi-layer networks of linear threshold units. We demonstrate the performance of the algorithm on three data-sets.

3.2 Linear Threshold Unit Networks

Linear threshold units were proposed by McCulloch and Pitts (1943) as a simple model of the biological neuron. The state of a node, s_i, depends on its parents' states and the parameters $\mathbf{W} = \{w_{ij}\}$ in the following way

$$s_i = \text{sign}\left(\sum_j w_{ij} s_j + w_{i0}\right). \tag{3.1}$$

When nodes of this type are to be implemented in pattern recognition, the model is often chosen to be one with a layered structure such as in Figure 3.1. In this example there is a layer of input nodes, which represent input variables, that can be real valued as well as binary. This layer is then connected to a layer of hidden nodes, whose activation function takes the form of Eqn 3.1, which are in turn connected to a layer of output nodes with the same activation function. The structure can be more general, containing more layers of hidden nodes and across layer connections (for example between input and output nodes), and our theoretical results apply to this whole set of models. For simplicity though, we constrain our practical investigations to models of the type depicted in Figure 3.1. It is also possible to envisage cases where some or all of the input variables are unobserved. We only investigate the case where they are all observed, which is sometimes known as supervised learning. Models of this

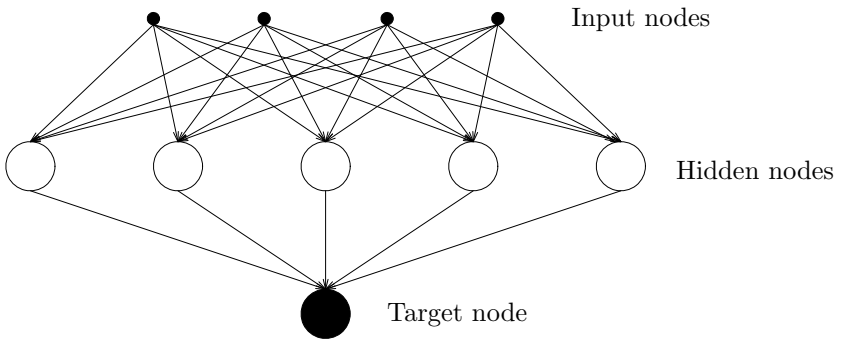


Figure 3.1: A linear threshold unit network with one layer of hidden units.

type were implemented by Rosenblatt in his perceptron. Perceptrons, however, were limited by the use of fixed basis functions which could not be adapted to the particular data-set under consideration. In other words the parameters determining the output of the hidden nodes where taken to be fixed. The generalised delta rule popularised by Rumelhart *et al.* (1986) allowed networks to have adaptive basis functions through the use of multiple layers of adaptable weights. Unfortunately this rule may not be applied when the basis functions are linear threshold units. A common misconception is that this is due to the basis functions' discontinuity; a larger problem is that they have a gradient of zero at all points except at the discontinuity. This means that any attempt to find the optimal parameters in such a network gains no useful information from the gradient: the gradient of the error surface as computed by the generalised delta rule is zero at almost all points.

One approach to learning in these networks has been to use 'noisy weights' (Hinton and van Camp, 1993). This gives a probability of a threshold unit being active which is a smooth function of the weights. However, the approach leads to some restrictions on the structure of the network. In this work the more general algorithm we derive is inspired by the variational approach of mean field theory, in particular the work of Saul *et al.* (1996). Bounds used in our mean field approach can become exact in the limits of interest and present a tractable learning algorithm for networks of linear threshold units.

3.3 From Sigmoid Belief Networks ....

Consider again the sigmoid belief network described in the previous chapter. We now take the bi-valued nodes to be $s_i \in \{-1, 1\}$. Once again the probability of a node being equal to one given the states of its parents can be seen to be

$$P(s_i = 1|\mathrm{pa}(s_i)) = \sigma\left(\sum_j \frac{w_{ij}}{T} s_j + \frac{w_{i0}}{T}\right), \tag{3.2}$$

where $\mathrm{pa}(s_i)$ is the set of nodes which are parents of node i, and the parameters $\mathbf{W}$ are a matrix of weights and biases. The 'temperature' parameter T in Eqn 3.2 is in principle redundant since it can be absorbed into the connectivity matrix, however it will prove convenient to separate it. For

bi-valued nodes the equivalent potential function of that in Eqn 2.1 is

$$E_i(s_i, \mathbf{w}_i, \mathrm{pa}(s_i), T) = \left(\sum_j \frac{w_{ij}}{T} s_j \right) \frac{(s_i + 1)}{2}, \tag{3.3}$$

where we have absorbed the biases into the weight matrix $\mathbf{W}$ in the usual manner by introducing a dummy variable $s_0 = 1$. The probability of the combined states of the model is therefore:

$$P(\mathcal{S}) \quad = \quad \prod_i P(s_i|\mathrm{pa}(s_i)), \tag{3.4}$$

where we may write the conditional probability as

$$P(s_i|\mathrm{pa}(s_i)) \quad = \quad \frac{\exp\left[E_i(s_i, \mathbf{w}_i, \mathrm{pa}(s_i), T)\right]}{1 + \exp\left[E_i(1, \mathbf{w}_i, \mathrm{pa}(s_i), T)\right]} \tag{3.5}$$

$$\equiv \quad \frac{\exp\left[E_i(s_i, \mathbf{w}_i, \mathrm{pa}(s_i), T)\right]}{\sum_{s_i} \exp\left[E_i(s_i, \mathbf{w}_i, \mathrm{pa}(s_i), T)\right]}. \tag{3.6}$$

The equation differs slightly to the equivalent in Chapter 2 due to a slightly different choice of potential function arising from the differing values of the states of s_i. The variables may be partitioned into a visible set $\mathcal{V} = \{v_i\}$, and a hidden set $\mathcal{H} = \{h_i\}$. In the context of supervised learning, our observed data, $\mathcal{V}$, may be further split into two subsets: input data, $\mathcal{I}$, and data labels or output data, $\mathcal{O}$. When the input data has been fully observed it is unnecessary to infer its distribution, we therefore need only consider the distribution $P(\mathcal{H}, \mathcal{O}|\mathcal{I})$. Marginalising over the latent variables of this distribution gives $P(\mathcal{O}|\mathcal{I})$ which in supervised learning is the distribution of interest.

$$P(\mathcal{O}|\mathcal{I}) = \sum_{\mathcal{H}} P(\mathcal{H}, \mathcal{O}|\mathcal{I}). \tag{3.7}$$

A training set consists of a set of observations of the input variables, $\mathcal{I}_1, \ldots, \mathcal{I}_N$, and the output variables, $\mathcal{O}_1, \ldots, \mathcal{O}_N$. The log likelihood of the data is a sum over patterns

$$\ln P(\mathcal{O}|\mathcal{I}) = \sum_{n=1}^{N} \ln \left\{ \sum_{\mathcal{H}_n} P(\mathcal{H}_n \mathcal{O}_n|\mathcal{I}_n) \right\}. \tag{3.8}$$

Once again we suppress the summations over n to avoid cluttering the notation. Additionally maintain simple notation though we will denote the observed values of the individual variables, s_i for $i \in \mathcal{I}$ or $i \in \mathcal{O}$, as an expectation of that variable under the $Q(\mathcal{H})$, $\langle s_i \rangle_{Q(\mathcal{H})}$ or more concisely $\langle s_i \rangle$.

The size of the sum in Eqn 3.7 becomes very impractical for large networks. We must therefore seek an alternative approach to learning without performing the sum explicitly.

3.3.1 Variational Inference

We have already shown in Chapter 1 how if we introduce a distribution $Q(\mathcal{H}|\mathcal{V})$, which we regard as an approximation to the true posterior distribution, then we may bound the likelihood below by

$$\ln P(\mathcal{O}|\mathcal{I}) \quad \geq \quad \sum_{\mathcal{H}} Q(\mathcal{H}|\mathcal{O}, \mathcal{I}) \ln \frac{P(\mathcal{H}, \mathcal{O}|\mathcal{I})}{Q(\mathcal{H}|\mathcal{O}, \mathcal{I})}. \tag{3.9}$$

The aim of this approach being to choose an approximating distribution which leads to computationally tractable algorithms and yet which is also flexible enough to permit a good representation of the true posterior. We also briefly reviewed mean field theory in sigmoid belief networks (Section 2.4.1). For networks containing bi-polar valued nodes these probability distributions will differ slightly in form. Additionally we will choose to handle the expectation of the normalising constant (the denominator in Eqn 3.6) in a slightly different manner. Recall from the introduction Eqn 1.36. Given a subset selection where $\mathcal{H}_j = s_j$,

$$Q(\mathcal{H}|\mathcal{O},\mathcal{I}) = \prod_{i\in\mathcal{H}} Q(s_i), \tag{3.10}$$

this leads to

$$Q(s_j) \propto \exp\left(\langle \ln P(s_j|\mathrm{pa}(s_j))\rangle_{\prod_{k\neq l} Q(s_k)} + \sum_{i\in\mathrm{ch}(s_j)} \langle \ln P(s_i|\mathrm{pa}(s_i))\rangle_{\prod_{k\neq j} Q(s_k)}\right). \tag{3.11}$$

Substituting in the form of our conditional distributions from Eqn 3.6 we obtain

$$\begin{aligned} Q(s_j) \quad \propto \quad & \exp\Bigg(\langle E_j(s_j,\mathbf{w}_j,\mathrm{pa}(s_j))\rangle_{\prod_{k\in\mathrm{pa}(s_j)} Q_k} \\ & - \sum_{i\in\mathrm{ch}(s_j)} \left\langle \ln \sum_{s_i} \exp\left[E_j(s_i,\mathbf{w}_i,\mathrm{pa}(s_i))\right]\right\rangle_{\prod_{k\in\mathrm{pa}(i),k\neq j} Q_k}\Bigg), \end{aligned} \tag{3.12}$$

when, in fact, it becomes exact. Here we have used Q_k as shorthand for $Q(s_k)$. The first term in this equation is straightforward to evaluate, the second term, however, proves more difficult. The cause is the log of the sum. We choose to again apply a variational lower bound 3.9 to this term.

$$\begin{aligned} \ln \sum_{s_i} \exp\left[E_i(s_i,\mathbf{w}_i,\mathrm{pa}(s_i),T)\right] \quad \geq \quad & \sum_{s_i} Q^-(s_i|\mathrm{pa}(s_i)) E_i(s_i,\mathbf{w}_i,\mathrm{pa}(s_i),T) \\ & - \sum_{s_i} Q^-(s_i|\mathrm{pa}(s_i)) \ln Q^-(s_i|\mathrm{pa}(s_i)). \end{aligned} \tag{3.13}$$

It is straightforward to show, through free form optimisation, that this bound is maximised by setting

$$Q^-(s_i|\mathrm{pa}(s_i)) = \frac{\exp(E_i)}{\sum_{s_i}\exp(E_i)}, \tag{3.14}$$

where we have used the shorthand E_i to represent $E_i(s_i,\mathbf{w}_i,\mathrm{pa}(s_i),T)$[1].

Expectations under $Q(\mathcal{H})$

The expectation of a variable under the distribution $Q^-(s_i|\mathrm{pa}(s_i))$ can easily be seen to be

$$\langle s_i\rangle_{Q_i^-} = \tanh\left(E_i(1,\mathbf{w}_i,\mathrm{pa}(s_i),T)\right). \tag{3.15}$$

we rewrite Eqn 3.12 as

$$\begin{aligned} Q(s_j) \quad \propto \quad & \exp\Bigg(\langle E_j(s_j,\mathbf{w}_j,\mathrm{pa}(s_j))\rangle_{\prod_{k\in\mathrm{pa}(s_j)} Q_k} \\ & - \sum_{i\in\mathrm{ch}(s_j)} \left\langle \left(\frac{w_{ji}}{T}s_j\right)\frac{(\langle s_i\rangle_{Q_i^-}+1)}{2}\right\rangle_{\prod_{k\in\mathrm{pa}(i),k\neq j} Q_k} + \sum_{i\in\mathrm{ch}(s_j)} S\left(Q^-(s_i)\right)\Bigg), \end{aligned} \tag{3.16}$$

[1] In fact $Q^-(s_i|\mathrm{pa}(s_i))$ in Eqn 3.14 is recognised as the true posterior distribution of s_i given its parents. Thus when Q^- is set as shown in Eqn 3.14, the bound 3.13 becomes an equality. This may be seen by deriving $P(s_i|\mathrm{pa}(s_i))$ from Eqn 2.1.

where $S(Q^-(s_i))$ is the entropy of the distribution $Q^-(s_i)$. The argument of the exponential in Eqn 3.16 can be written in terms of a function of s_j, which we denote $E'_j(s_j)$, a constant and the entropy term. Substituting Eqn 3.3 into Eqn 3.16 and collecting terms in s_j we write

$$\log Q(s_j) = E'_j(s_j) + \sum_{i \in \mathrm{ch}(s_j)} S\left(Q^-(s_i)\right) + \text{const.} \tag{3.17}$$

where the function $E'_j(s_j)$ can be seen to be

$$E'_j(s_j) = s_j \left(\sum_i \frac{w_{ij}}{2T} \langle s_i \rangle + \sum_{i \in \mathrm{ch}(s_j)} \frac{w_{ji}}{2T} \left(\langle s_i \rangle - \left\langle \langle s_i \rangle_{Q^-} \right\rangle \right) \right). \tag{3.18}$$

Now note that the expectation of s_i under Q^- is also dependent on s_j (Eqn 3.15). We therefore rewrite this expectation to bring out the dependence on s_j,

$$\begin{aligned}
\langle s_j \rangle_{Q^-} &= \tanh\left(\sum_k \frac{w_{ik}}{T} s_k \right) \\
&= \frac{1+s_j}{2} \tanh\left(\sum_{k \neq j} \frac{w_{ik}}{T} s_k + \frac{w_{ij}}{T} \right) + \frac{1-s_j}{2} \tanh\left(\sum_{k \neq j} \frac{w_{ik}}{T} s_k - \frac{w_{ij}}{T} \right) \\
&= \frac{s_j}{2} \left(\tanh\left(\sum_{k \neq j} \frac{w_{ik}}{T} s_k + \frac{w_{ij}}{T} \right) - \tanh\left(\sum_{k \neq j} \frac{w_{ik}}{T} s_k - \frac{w_{ij}}{T} \right) \right) + \text{const.} \\
&\stackrel{\text{def}}{=} s_j \phi_{ij} + \text{const.},
\end{aligned} \tag{3.19}$$

where const. is a term constant in s_j. Substituting this representation back into Eqn 3.18 we obtain,

$$E'_j(s_j) = s_j \left(\sum_i \frac{w_{ij}}{2T} \langle s_i \rangle + \sum_j \frac{w_{ji}}{2T} \left(\langle s_i \rangle - \frac{1}{2} \langle \phi_{ij} \rangle \right) \right), \tag{3.20}$$

leading to

$$Q(s_j) = \frac{\exp\left(s_k E'(s_j)\right)}{\exp\left(E'(1)\right) + \exp\left(E'(-1)\right)}, \tag{3.21}$$

3.4 ... to Linear Threshold Units

The sigmoid belief network may be converted into a network of linear threshold units by taking the limit of Eqn 3.2 as the temperature goes to zero. The marginal likelihood will then become discontinuous being equal to 1 for a subset of the set of all possible values of $\mathcal{S}$, and 0 at all other times. This subset is the patterns which the network has stored. The output of each node is then deterministic as in Eqn 3.1 and our network is a multi-layer network of linear threshold units.

We can now consider the behaviour of the distribution in Eqn 3.21 in the zero temperature limit, or more specifically we discuss the behaviour of the expectation of s_i under this distribution in that limit. Using the distribution in Eqn 3.21 the expectation of any node can be seen to be

$$\langle s_k \rangle = \lim_{T \to 0} \tanh\left(E'(s_j) + \sum_{i \in \mathrm{ch}(s_j)} S(Q^-(s_i)) \right), \tag{3.22}$$

When we take the limit the entropy term goes to zero and, through substitution of Eqn 3.18 for $E'(s_j)$, we find

$$\langle s_k \rangle \quad = \quad \text{sign}\left(\sum_i w_{ij} \langle s_i \rangle + \sum_i \frac{w_{ji}}{2} \left(\langle s_i \rangle - \langle \phi_{ij} \rangle \right) \right). \tag{3.23}$$

Additionally the expectation $\langle \phi_{ij} \rangle$ also becomes tractable,

$$\begin{aligned} \langle \phi_{ij} \rangle &= \lim_{T \to 0} \frac{1}{2} \left\langle \tanh\left(\sum_{k \neq j} \frac{w_{ik}}{T} s_k + \frac{w_{ij}}{T} \right) - \tanh\left(\sum_{k \neq j} \frac{w_{ik}}{T} s_k - \frac{w_{ij}}{T} \right) \right\rangle \\ &= \frac{1}{2}\text{sign}\left(\sum_{k \neq j} w_{ik} \langle s_k \rangle + w_{ij} \right) - \frac{1}{2}\text{sign}\left(\sum_{k \neq j} w_{ik} \langle s_k \rangle - w_{ij} \right). \end{aligned} \tag{3.24}$$

Updates of each $\langle s_j \rangle$ for $j \in \mathcal{H}$ may then be undertaken for each pattern n in the training set. Upon presentation of each example Eqn 3.23 may be applied to each node in random order until a stable solution is reached.

3.5 Learning

In the last section we discussed how inference of latent variables (or hidden units) may be performed in networks of LTUs. We now turn to the learning of the parameters $\mathbf{W}$.

The variational lower bound on the log likelihood may be written (from bound 3.9 substituting in Eqn 3.4, Eqn 3.6 and Eqn 3.10)

$$\ln P(\mathcal{O}|\mathcal{I}) \geq \sum_i \langle E_i \rangle - \sum_i \langle \ln\left(1 + \exp(E_i(s_i))\right) \rangle + \sum_i S\left(Q(s_i)\right) \stackrel{\text{def}}{=} \mathcal{L}. \tag{3.25}$$

We may treat the second term of the bound as in bound 3.13. Substituting in Eqn 3.3 and noting that the entropy term will always go to zero in the zero temperature limit. We therefore obtain

$$\mathcal{L} = \lim_{T \to 0} \sum_{ij} \frac{w_{ij}}{2T} \langle s_j \rangle \left(\langle s_i \rangle - \left\langle \langle s_i \rangle_{Q^-} \right\rangle \right). \tag{3.26}$$

Clearly this limit does not exist unless $\langle s_i \rangle = \left\langle \langle s_i \rangle_{Q^-} \right\rangle$ for all i.

Consider the function $T\mathcal{L}$

$$T\mathcal{L} = \sum_{ij} \frac{w_{ij}}{2} \langle s_i \rangle \langle s_j \rangle - \sum_{ij} \frac{w_{ij}}{2} \langle s_i \rangle \left\langle \langle s_j \rangle_{Q^-} \right\rangle. \tag{3.27}$$

First of all note that both terms take the form $\sum_{ij} \langle s_i \rangle w_{ij} x_j$. This form is maximised, for bi-valued x, if $x_j = \text{sign}\left(\sum_i w_{ij} \langle s_i \rangle\right)$. Now recall that the form of $\left\langle \langle s_j \rangle_{Q^-} \right\rangle = \text{sign}\left(\sum_i w_{ij} \langle s_i \rangle\right)$. Therefore Eqn 3.27 and Eqn 3.26 are bounded from above by zero. As we mentioned before, the limit in Eqn 3.26 only exists if $\langle s_i \rangle = \text{sign}\left(\sum_i w_{ij} \langle s_i \rangle\right)$ for all nodes in the latent set, $\mathcal{H}$, and the output (or target) set, $\mathcal{O}$. In other words the limit only exists when the pattern is classified correctly and all hidden nodes have their expected values. If this is the case the true likelihood for that pattern is 1, and our bound on the log-likelihood (which is $0 = \ln 1$) is exact. The objective in learning therefore is to adapt the

parameters $\mathbf{W}$ so that this is the case. Of course, we wish to make this the case for every pattern in the training set so, reintroducing the index over the N patterns, we really wish to optimise

$$\mathcal{L} = \lim_{T \to 0} \sum_{n=1}^{N} \sum_{ij} \frac{w_{ij}}{2T} \left\langle s_j^{(n)} \right\rangle \left(\left\langle s_i^{(n)} \right\rangle - \left\langle \left\langle s_i^{(n)} \right\rangle_{Q^-} \right\rangle \right). \tag{3.28}$$

We can envisage situations where the limit cannot exist for any parameterisation of $\mathbf{W}$. Consider, for example, the case where an identical input pattern has two different labels. We make it our objective, therefore, to satisfy the above equation for as many patterns as we can given the constraints of our chosen model structure.

Optimisation of the weight parameters in Eqn 3.28 is in effect a constrained optimisation with N constraints imposed on the weight parameters associated with each hidden and output node where, as long as the constraints are all satisfied, the value of the objective function is constant. For values of $\mathbf{W}$ that satisfy the constraints the likelihood is one, for all other values it is zero.

3.5.1 Perceptron Learning Rule

The constraints imposed lead to an optimisation problem at each node which is identical to that of the perceptron with Eqn 3.27 providing the perceptron criterion for each node (given the inferred values of $\langle s_i \rangle$). To form a historical connection, we first simply derive the perceptron learning rule for each node.

Gradient based optimisation of the weight parameters could be achieved through differentiation of Eqn 3.27 with respect to w_{kl}. Ignoring the dependence of $\left\langle \langle s_i \rangle_{Q^-} \right\rangle$ upon the w_{kl} we obtain

$$\frac{\partial T\mathcal{L}}{\partial w_{kl}} = \frac{\langle s_k \rangle \langle s_l \rangle - \left\langle \langle s_l \rangle_{Q^-} \right\rangle \langle s_l \rangle}{2}. \tag{3.29}$$

If the weights are adjusted by simple gradient descent the algorithm can be recognised as the perceptron learning rule.

The solution to the constrained optimisation is not unique[2], see Figure 3.2, and may not exist, see Figure 3.3. If the solution does not exist then the perceptron learning rule will fail to converge. Ideally we seek the solution which we expect to generalise best on unseen data. Therefore, rather than utilising the perceptron learning rule to optimise our weights, we utilise a maximum margin approach to finding a solution.

3.5.2 Maximum Margin

A maximum margin classifier, also known as the optimal perceptron, is a technique which can be justified from the perspective of statistical learning theory. In this work we only give a brief overview of these classifiers, for a more detailed introduction see Burges (1998).

[2]The objective function is zero once all constraints are satisfied, therefore any solution which satisfies the constraints is equally valid.

Figure 3.2: These two examples have decision boundaries which would both represent solutions to the linear programming problem.

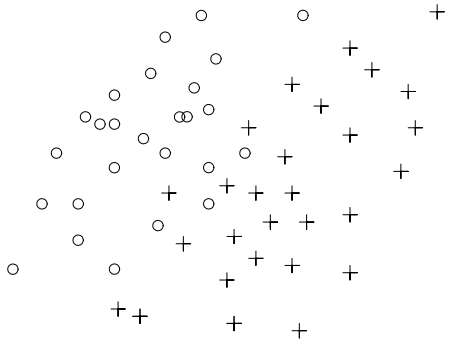


Figure 3.3: No linear decision boundary can be found which solves the linear constraints imposed by this data.

Statistical Learning Theory

Statistical learning theory relies on notions of capacity which reflect the number of patterns that a classifier may store (Vapnik, 1982). It is possible to place bounds on the expected generalisation error of such classifiers which hold with a certain confidence. These bounds are functions of the model capacity. The principle of structural risk minimisation is to minimise these bounds, through capacity reduction, to obtain the best classifier. This is the underlying theory of maximum margin classification. The margin size can be defined as the minimum distance the classification boundary needs to be moved for an example to be mis-classified. The model capacity is known to be inversely related to margin size. Increasing the margin decreases the capacity and hence should lead to a classifier with better generalisation properties. Additionally, the objective function in the region where the constraints has been satisfied becomes convex, and it can be shown that the solution for the weight parameters becomes unique.

The maximum margin solution, shown in Figure 3.4, is sometimes defined by data-points which lie on that margin, known as support vectors, and by Lagrange multipliers associated with those points. It may be found through quadratic programming.

There remains the issue of separability. If the problem is not linearly separable, there will be no solution for the standard formulation of the quadratic programming problem. Fortunately the

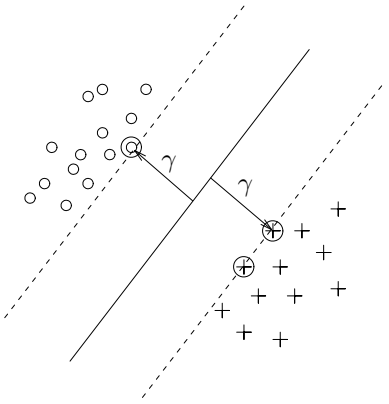


Figure 3.4: In linearly separable data, maximum margin provides a unique solution. Circled data-points indicate the support vectors and γ indicates the size of the margin.

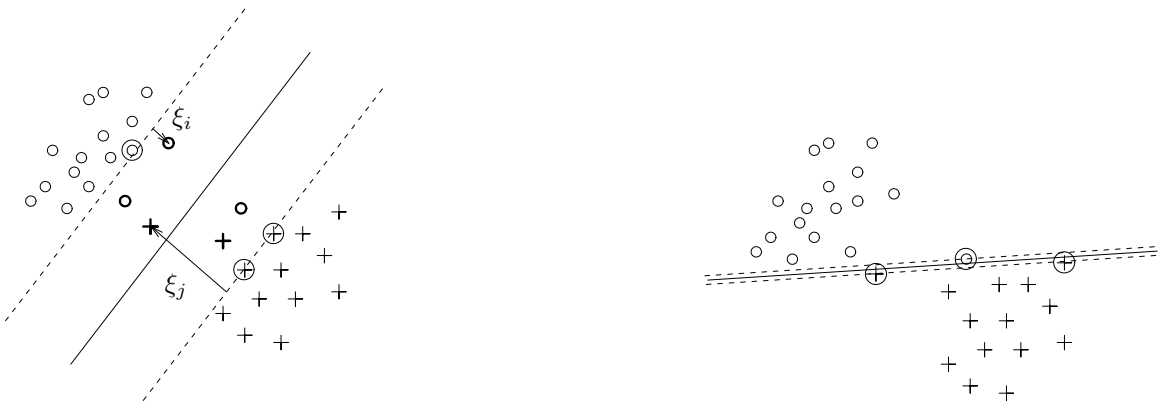


Figure 3.5: Depending on the penalty assigned to errors, C, different solutions will be found. The left hand figure is a low error penalty, the right hand figure is a high error penalty. Bold data-points indicate 'errors' and ξ_i represents the extent of the error on data-point i.

quadratic programming problem may be reformulated to allow errors through the introduction of slack variables and a parameter, C, known as the error penalty. A slack variable, ξ_n, represents the extent of the error on data-point n. A penalty term of the form $C\sum_{n=1}^{N}\xi_n$ may then be introduced into the quadratic programming problem (Figure 3.5). The solution found will naturally then be sensitive to the choice of C.

3.6 Algorithm Overview

The algorithm as we have described it involves two steps. The first step involves updating the expected value of the hidden nodes, $\left\langle s_i^{(n)} \right\rangle$, for each data point, n. The next step is to update the weight parameters, $\{w_{ij}\}$, by a quadratic programming problem, given the expected values of the variables.

3.7 Some Algorithm Heuristics

Having described the fundamental theory behind our algorithm, we also wish to mention some heuristics which were found useful in the practical implementation of the algorithm.

Simulated annealing

The updates of Eqn 3.23 are taking place in a highly discontinuous space. It may be advantageous to reintroduce the temperature parameter and perform *deterministic annealing* during the updates of $\langle s_j \rangle$, gradually reducing the temperature to zero (see Waugh *et al.* (1990) for some theoretical justification of this approach).

Expectation Initialisation

The values for $\langle s_j \rangle$ for $j \in \mathcal{H}$ may be initialised with the values of s_j obtained from a 'forward pass' through the network.

Convergence

The algorithm converges when every training pattern has been classified exactly. However, it is quite possible for the training error, in terms of the number of mis-classifications, to increase after the inference and learning steps. This may occur despite the overall objective function going down as it may coincide with an increase in the number of hidden nodes which are 'classified correctly'. It would therefore seem prudent to store the network best training error achieved at all times during learning and, in the case that learning doesn't converge (e.g. for the reasons outlined in Section 3.5) to utilise this network for the solution.

Weight Normalisation

The inference step for a node (Eqn 3.23) involves a sum across the weights which propagate out from the node and a sum across weights which feed into the node. The positioning of the separating hyper-plane which is defined by the weights which feed into the node is invariant to a rescaling of the weights. Envisage, therefore, a situation where the magnitude of the weights which feed into the node is much larger than the magnitude of the weights which propagate out (this can be achieved without affecting the mapping which the network represents between the input set and the output set). The inferred value of the node as given by Eqn 3.23 will then be dominated by the variables which feed into the node. This situation is far from ideal. In our experiments, therefore, we implemented a heuristic where the values of the weights which propagate into a node are normalised by the magnitude of the vector of the weights which propagate out from the node.

Relaxing Quadratic Programming Convergence Conditions

Maximising the margin can be an expensive computation. In the early stages of the algorithm, when we expect our inference about the values of the latent nodes to be 'less correct', we may not wish to spend undue computational effort on precise optimisation of the margin. In our implementation we relaxed the conditions for convergence of the maximum margin solutions for the first four iterations. To be precise, we normally required the Karus-Kuhn-Tucker[3] (KKT) conditions to be satisfied to an accuracy of 1×10^{-4}, but for the first four iterations we replaced this equation with $1 \times 10^{-(k-1)}$ where k is the iteration number. Note that for the first iteration, this is convergence of the KKT to within 1. In other words no maximisation of the margin is required for the first iteration.

3.8 Results

In this section we apply the algorithm to three data-sets. The first two data-sets are well known benchmarking data-sets. The third data-set is from a medical imaging problem involving the analysis of fluorescent in-situ hybridisation (FISH) images. For these experiments, updates were undertaken in the form of a full inference step, i. e. updates of $\{s_i^{(n)}\}$ to convergence, followed by learning of the parameters $\mathbf{W}$ to convergence using the sequential minimal optimisation (SMO) algorithm (Platt, 1998) to solve the quadratic programming problem. The process of inference and learning was repeated seven times for all data-sets.

The Pima Indians Diabetes Data

This well known benchmark data-set is demonstrated here with the partitions used by Ripley (1996). The objective is to create a classifier which predicts whether the subjects are diabetic. The data consists of seven inputs and a set of class labels. Ripley split the data into a training set of 200 points and a test set of 332 points. We normalised the data to be zero mean and lie between -1 and 1. We then trained networks with differing error penalties, $C = 10^k$ for $k = 0, 1$ and 2, and different numbers of hidden units, $H = 10, 15$ and 20. Ten different initialisations were used for each parameterisation. We limited the margin optimisation to a maximum of 100,000 iterations of Platt's SMO algorithm. Simulated annealing was implemented for updating hidden unit expectations. The temperature was initialised at 1000 and the decay rate was set to 0.8, i. e. at iteration k the temperature was $1000 \times 0.8^{k-1}$. Once the temperature dropped below 0.001 it was taken to be zero. Once annealing was complete, a maximum of a further 70 passes updating the expectations of the latent nodes were permitted.

Results are summarised in Figure 3.6 and Table 3.1. The other presented results are taken from Ripley (1996) and Barber and Williams (1997).

The results on this data appear encouraging. Note though the variability in performance with different initialisations shown in Figure 3.6. However the average performance was still in line with

[3]See (Burges, 1998) for details of these convergence conditions.

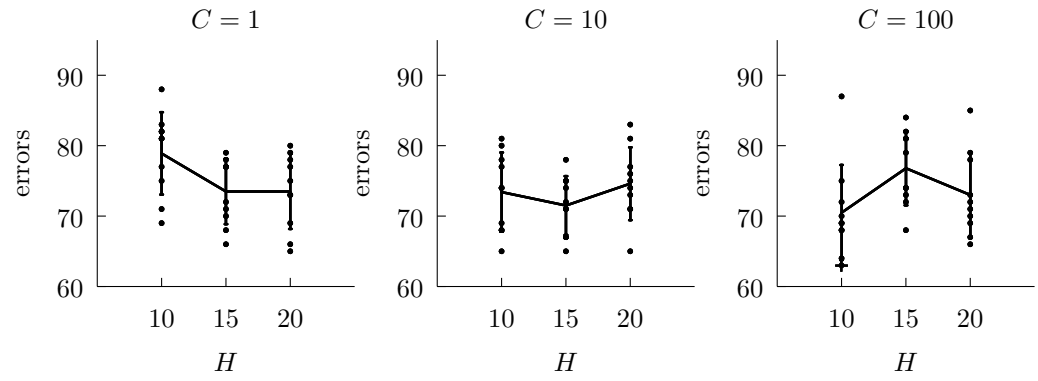


Figure 3.6: Results for the networks on the Pima Indians data-sets for $C = 1$, $C = 10$ and $C = 100$. Dots represent each of the ten different networks initialisations. The network which achieved the overall minimum error is marked with +. Also shown is the mean and error-bars at one standard deviation for each parameterisation.

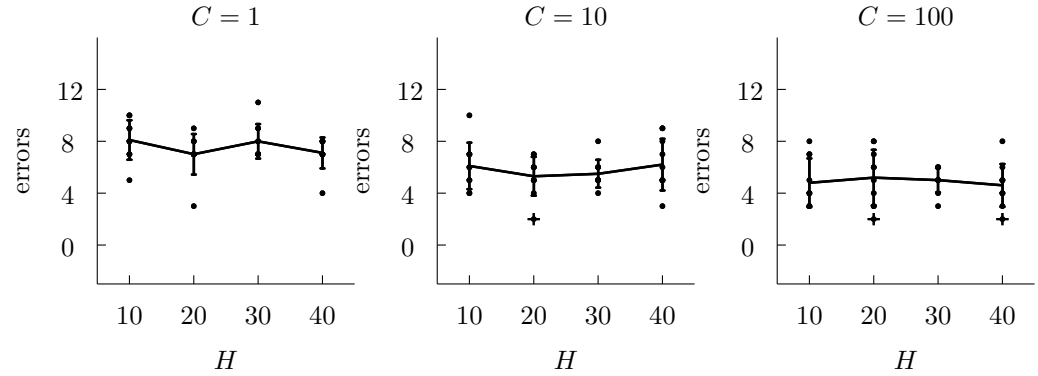


Figure 3.7: Results for the networks on the Leptrograpsus crabs data-sets for $C = 1$, $C = 10$ and $C = 100$. Dots represent each of the ten different networks initialisations. The networks which achieved the overall minimum error are marked with +. Also shown is the mean and error-bars at one standard deviation for each parameterisation.

that of the neural network results. Some caution has to be exercised with interpretation of these results. The best linear threshold unit classifier had a better performance on the test set than on the training set. This gives rise to some suspicion that these results may be somewhat partition sensitive due to the low number of samples.

The Leptrograpsus Crabs Data

This is another benchmark data-set. The partitions implemented here are from Ripley (1994). Five input features are used to describe crabs of two different species. The objective is to predict the sex of the crab given these features. The data consisted of 80 training points and 120 test points. The data was tested in the same manner as the Pima Indians data-set, but with different numbers of hidden units $H = 10, 20, 30$ and 40. The results are depicted in Figure 3.7, and the error rates are shown in Table 3.1.

Similar conclusions may be drawn from the results on the crabs data as were drawn for those on the Pima Indians data but with the same reservations also. For a better evaluation of the algorithm

model	Pima	crabs
Neural Network	75+	3
Linear Discriminant	67	8
Logistic Regression	66	4
MARS (degree = 1)	75	4
PP (4 ridge functions)	75	6
2 Gaussian Mixture	64	-
Gaussian Process Classifier	68	3
Average Linear Threshold Unit	73.97	6.08
Best Linear Threshold Unit	63	2

Table 3.1: Classification rates for the Pima Indians and Leptrograpsus crabs data-sets. The table shows some of the obtained results on the Pima Indians and Leptrograpsus crabs data-sets. The results labeled 'Best Linear Threshold Unit' are those which obtained the minimum errors on the test sets. Their parameterisations are depicted in Figure 3.7 and Figure 3.6. The result labeled 'Average Linear Threshold Unit' is the average of all the networks' classification errors.

we turned to a larger data-set from a medical application.

FISH Data

The FISH data-set was acquired from 400 images of cell nuclei. The objective is to classify signals produced by fluorescent in-situ hybridisation as either real signals, or artifacts of the slide preparation. The data consists of twelve features of the images which are based on shape and colour. The data were hand labelled by a trained cytologist, the problem is more fully described in Lerner and Lawrence (1999). We followed Lerner and Lawrence in partitioning the data into 2200 training points and 944 test points. For training the linear threshold units we made use of a validation set by further sub-dividing the training data. The validation set contained 314 points. Cross validation was not a viable option for the LTU models as the results are initialisation dependent. The inputs were normalised to have zero mean and unit standard deviation for these experiments. The results of the FISH experiments are summarised in Table 3.2 and Figure 3.8.

The results for the linear threshold units stand up well in comparison to the other classifiers. The best performing linear threshold unit network gives results in line with that of the support vector machine. The committee of networks also gives a competitive result. The single network selected by utilising the validation set gives equivalent performance to a standard neural network. The linear threshold units are placed at a disadvantage with respect to the other techniques which were trained on all the available training data. There is some chance that performance could be improved by further training of the selected models on the validation set.

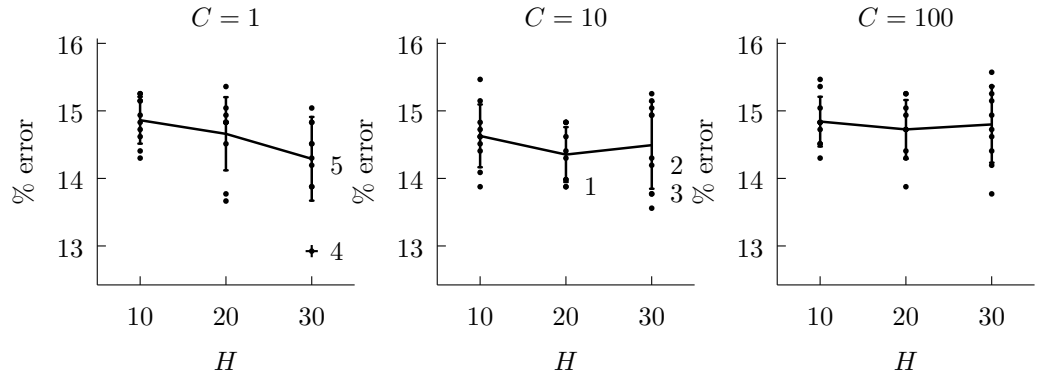


Figure 3.8: Results for the networks on the FISH data-sets for $C = 1$, $C = 10$ and $C = 100$. Dots represent each of the ten different networks initialisations. The network which achieved the overall minimum error is marked with +. The networks with the five lowest validation set error are marked in order 1 to 5, with 1 having the lowest validation error. Also shown is the mean and error-bars at one standard deviation for each parameterisation.

model	Classification Rate
Naive Bayesian Network	17.0 %
Neural Network	13.6 %
Bayesian Neural Network	11.8 %
Support Vector Machine	12.8 %
Linear Threshold Unit (Best)	12.9 %
Linear Threshold Unit (Average)	14.6 %
Linear Threshold Unit (Validation Set)	13.9 %
Linear Threshold Unit (Committee)	13.2 %

Table 3.2: Classification rates for the FISH data. The table shows some of the obtained results on the FISH data-set. The result labelled 'Best' is that which obtained the minimum error on the test sets, the 'Average' result was the average error for all networks, the result labelled 'Validation Set' is that selected through the validation set and finally that labelled 'Committee' is a result from an unweighted committee of five networks selected according to the validation set error.

3.9 Discussion

In this chapter we have presented an algorithm for learning in networks of linear threshold units. We demonstrated the algorithm's implementation with a variety of data-sets.

The algorithm has an interesting parallel with the expectation-maximisation (EM) algorithm (Dempster *et al.*, 1977), the 'expectation' step would be the determination of the parameters $\{\langle s_i \rangle\}$ and the 'maximisation' step is a quadratic program in $\mathbf{W}$. It not strictly correct, however, to call the fixed point equation update of $\left\langle s_i^{(n)} \right\rangle$ an expectation step because the function $Q(\mathcal{H}|\mathcal{V})$ is no longer probabilistic. This is analogous to the relationship between K-means clustering and EM optimisation of Gaussian mixture models (Bishop, 1995). K-means clustering may be seen as the limit as the variance goes to zero of EM update of a Gaussian mixture model. This method of learning in networks of linear threshold units is merely the limit as the temperature goes to zero of mean field theory for sigmoid belief networks.

We have considered only linear threshold units in this work. However it is possible to start with probabilistic graphs containing *soft-max* nodes and *linear-Gaussian* nodes and to arrive at networks containing *winner-takes-all* and *linear* activation functions. For the case of linear activation functions in the input rows and winner-takes-all activation functions in the output layer the derived equations are identical to the above. For linear inputs the value of $\langle s_i \rangle$ for $r = 1$ becomes continuous.

For the case of a network with linear inputs and only one layer of weights the product $T\mathcal{L}$ is identical to the perceptron criterion (Rosenblatt, 1962) and the weight update is the associated learning algorithm.

Finally this approach may provide a method of performing inference in expert systems which use `if ... then` rules for their knowledge base. Such a system can be viewed as the zero temperature limit of a DAG based on probability tables, just like networks of linear threshold units are the zero temperature limit of the sigmoid belief network. We could therefore apply the this approach to performing inference in such systems.

Acknowledgements

We would like to thank Boaz Lerner for providing the FISH data-set used in these experiments.

Chapter 4

Boltzmann Machines

4.1 A Brief Introduction

Thus far in this thesis we have concentrated on directed graphical models. In Chapter 2 we discussed a stochastic graph known as the sigmoid belief network and in Chapter 3 we investigated the same model in a zero temperature limit. In this chapter we turn to undirected graphs based around discrete variables.

A general undirected graphical model of a discrete set of variables would be based around probability tables. We discussed in Chapter 2 in the context of the sigmoid belief network how the size of these tables can explode as models grow in connectivity and size. We also presented a solution: rely on potential functions to describe the probabilistic relationships. The same problem and solution applies in the context of undirected graphs. The focus of this chapter is a graph which makes use of potential functions and is known as the Boltzmann machine.

4.2 The Boltzmann Machine

The Boltzmann machine (Ackley *et al.*, 1985) is an undirected graphical model whose nodes correspond to two-state stochastic variables. The choice of potential function is identical to that of the sigmoid belief network

$$E_i(s_i, \mathbf{w}_i, s_{i\neq j}) = \left(\sum_j w_{ij} s_j + w_{i0} \right) s_i, \tag{4.1}$$

in which $\mathcal{S} = \{s_i\}$ denotes the set of stochastic variables, and the parameters $\mathbf{W}$. Here $w_{ij} = 0$ for nodes which are not neighbours on the graph. Two further constraints are placed on $\mathbf{W}$. A node may not be a neighbour of itself, $w_{ii} = 0$ and the matrix must be symmetric. This last constraint ensures that the graph contains a stable probabilistic solution rather representing a dynamic system. As in Chapter 2, for notational convenience, we will absorb the bias parameters w_{i0} into the weight matrix $\mathbf{W}$ by introducing an additional variable $s_0 = 1$. The variables, $\mathcal{S}$, are normally taken to be binary.

Either, $s_i \in \{-1, 1\}$, or $s_i \in \{0, 1\}$. The joint distribution over all states is then given by

$$P(\mathcal{S}) = \frac{\exp\left(\sum_i E_i(s_i, \mathbf{w}_i, s_{i\neq j})/T\right)}{Z}. \tag{4.2}$$

where Z is a partition function. The temperature parameter T in Eqn 4.2 is in principle redundant since it can be absorbed into the weights. However, as in the previous chapter it can play a useful role through deterministic annealing, although for the moment we set $T = 1$. Normalisation of the sigmoid belief network was facilitated by its definition as a directed acyclic graph. This meant we could normalise the distribution through consideration of each variable individually . In the case of the Boltzmann machine however, we must consider a more general normalisation. This is a consequence of the graphs undirected nature. The normalisation constant Z is given by summing the numerator over all states

$$Z = \sum_{\mathcal{S}} \exp(E_i(s_i, \mathbf{w}_i, s_{i\neq j})/T). \tag{4.3}$$

The distribution in Eqn 4.2 is thus recognised as a Boltzmann distribution whence the model derives is name.

If there are N variables in the model, the number of configurations of states is 2^N, and so evaluation of Z may require exponential time (e.g. in fully connected models) and hence is, in the worst case, computationally intractable.

4.2.1 Boltzmann Machine Learning

As for the sigmoid belief network, we partition the variables $\mathcal{S}$ into a visible set $\mathcal{V} = \{v_i\}$, and a latent set $\mathcal{H} = \{h_i\}$. The probability of the observed states may then be determined by marginalisation across the latent states.

$$P(\mathcal{V}) = \sum_{\mathcal{H}} P(\mathcal{H}, \mathcal{V}), \tag{4.4}$$

A training set, D, will again consist of visible variables $\mathcal{V}_1, \ldots, \mathcal{V}_N$ and we recall that the log likelihood is therefore a sum over patterns

$$\ln P(\mathcal{V}) = \sum_{n=1}^{N} \ln \left\{ \sum_{\mathcal{H}} P(\mathcal{H}, \mathcal{V}_n) \right\}. \tag{4.5}$$

Learning in the Boltzmann machine is then achieved by maximising the log likelihood Eqn 4.5 with respect to the parameters $\mathbf{W}$ using gradient methods. Substituting Eqn 4.2 into Eqn 4.5 and differentiating with respect to w_{ij} we obtain

$$\frac{\partial \ln P(\mathcal{V})}{\partial w_{ij}} = \langle s_i s_j \rangle_{\mathrm{C}} - \langle s_i s_j \rangle_{\mathrm{F}}, \tag{4.6}$$

where $\langle \cdot \rangle_{\mathrm{C}}$ denotes an expectation with respect to the conditional distribution $P(\mathcal{H}|\mathcal{V})$ while $\langle \cdot \rangle_{\mathrm{F}}$ denotes expectation with respect to the joint distribution $P(\mathcal{H}, \mathcal{V})$. In the case of the expectation under the conditional distribution, each s_i in Eqn 4.6 corresponding to a visible variable would simply be set to its observed value.

Supervised Learning

So far in this chapter we have been considering learning a *joint* probability, $P(\mathcal{V})$, which represents the data-set. In the last chapter we considered the supervised learning problem where our observed data, $\mathcal{V}$, is split into input data, $\mathcal{I}$, and output data, $\mathcal{O}$. We are often interested in learning only the *conditional* probability $P(\mathcal{O}|\mathcal{I})$ rather than the joint probability which is equivalent to $P(\mathcal{O}, \mathcal{I})$. If this is the case then we need marginalise only across the latent variables and the output variables to obtain our partition function, no marginalisation across the input data is required. In Eqn 4.6 we would then be interested in expectations under the distribution $P(\mathcal{H}, \mathcal{O}|\mathcal{I})$ rather than $P(\mathcal{H}, \mathcal{V})$.

4.3 Gibbs Sampling

Evaluation of the expectations in Eqn 4.6 requires summing over exponentially many states, and so is intractable for densely connected models. The original learning algorithm for Boltzmann machines made use of Gibbs sampling to generate separate samples from the joint and marginal distributions, and used these to evaluate the required gradients. Some limitations of this approach were mentioned in Chapter 2. A further difficulty of this approach in the context of the Boltzmann machine is that the gradient is expressed as the difference between two Monte Carlo estimates and is thus very prone to sampling error.

4.4 Variational Inference

In an attempt to resolve these difficulties, there has been considerable interest in approximating the expectations in Eqn 4.6 using deterministic methods based on mean field theory (Peterson and Anderson, 1987). Although in principle this leads to a relatively fast algorithm, it often fails to find satisfactory solutions for many problems of practical interest.

We now derive mean field theory for the Boltzmann machine within the context of the variational framework. We discuss the limitations imposed by the fully factorised distribution. We suggest that one reason for the failure of mean field theory is the inconsistency of the mean field solutions found during learning. This problem was also mentioned by Hinton (1989) and Galland (1993). We show how some of the problems that Galland encountered may be eliminated in the supervised learning problems he was considering by judicious initialisations of the mean field distributions. We then discuss an alternative solution based on mixture representations for approximating the joint distribution. Experimental results on toy problems (including a review of the work in Galland, 1993), and on a problem involving hand-written digits, are presented in Section 4.5.1 and Section 4.5.2.

4.4.1 Mean Field Theory

Mean field theory is normally derived in the context of the Boltzmann machine by considering the explicit implementations of products of binomials for the approximating distributions. However, in

the manner of this thesis so far we shall seek to derive it by undertaking free-form optimisations over the variables.

A consequence of the undirected nature of the Boltzmann machine is that we are unable to compute or lower bound $-\ln Z$. Therefore we seek to approximate the joint distribution, $P(\mathcal{H},\mathcal{V})$, as well as the conditional distribution, $P(\mathcal{H}|\mathcal{V})$. We are therefore required to make two separability assumptions. We consider

$$\begin{aligned} Q^+(\mathcal{H}|\mathcal{V}) &= \prod_i Q(h_i), & (4.7)\\ Q^-(\mathcal{H},\mathcal{V}) &= \prod_i Q(h_i)\prod_i Q(v_i) \\ \equiv Q^-(\mathcal{S}) &= \prod_i Q(s_i). & (4.8) \end{aligned}$$

The derivation of the form of the approximating distribution is similar for both approximating distributions. We therefore concentrate on that of the joint distribution. From Eqn 1.44 and making use of Eqn 4.1 we may obtain

$$Q^-(s_k) = \frac{\exp\left(s_k \sum_j w_{kj} \left\langle s_j \right\rangle^-\right)}{\exp\left(\sum_j w_{kj} \left\langle s_j \right\rangle^-\right) + \exp\left(-\sum_j w_{kj} \left\langle s_j \right\rangle^-\right)}, \tag{4.9}$$

leading to

$$\left\langle s_k \right\rangle^- = \tanh\left(\sum_j w_{kj} \left\langle s_j \right\rangle^-\right), \tag{4.10}$$

where $\left\langle \cdot \right\rangle^-$ denotes an expectation with respect to the approximating joint distribution $Q(\mathcal{S})$. A similar equation may be derived for the conditional distribution:

$$\left\langle s_k \right\rangle^+ = \tanh\left(\sum_j w_{kj} \left\langle s_j \right\rangle^+\right), \tag{4.11}$$

where $\left\langle \cdot \right\rangle^+$ denotes an expectation with respect to the approximating conditional distribution $Q(\mathcal{H}|\mathcal{V})$ and $\left\langle s_j \right\rangle^+$ should be replaced with the instantiated value when the jth node is observed.

Once the mean field parameters have been determined we can update the model parameters using gradient based optimisation techniques. We may wish to make use of the objective function for monitoring convergence. The objective function is an approximation to the likelihood. From Eqn 1.40 we obtain

$$\begin{aligned} \mathcal{F}_{\text{mft}} &= \mathcal{L}^+ - \mathcal{L}^- & (4.12)\\ &= \sum_{ij} \left\langle s_i \right\rangle^+ w_{ij} \left\langle s_j \right\rangle^+ + \sum_i S(Q^+(s_i)) & (4.13)\\ &\quad - \left\langle s_i \right\rangle^- w_{ij} \left\langle s_j \right\rangle^- - \sum_i S(Q^-(s_i)), & (4.14) \end{aligned}$$

the derivatives of the objective function with respect to w_{ij} are then given by

$$\frac{\partial \mathcal{F}_{\text{mft}}}{\partial w_{ij}} = \left\langle s_i \right\rangle^+ \left\langle s_j \right\rangle^+ - \left\langle s_i \right\rangle^- \left\langle s_j \right\rangle^- . \tag{4.15}$$

Thus we see that the gradients have been expressed in terms of simple products of mean field parameters, which themselves can be determined by iterative solution of deterministic equations. The resulting learning algorithm is computationally efficient compared with stochastic optimisation of the true log likelihood. Comparison of Eqn 4.15 with Eqn 4.6 shows how the expectations have been replaced with deterministic approximations.

In a practical setting it is often useful to introduce a temperature parameter as in Eqn 4.2. For large values of T the true distribution of the parameters is smoothed out and the variational optimisation is simplified. The value of T can then be slowly reduced to $T = 1$ through deterministic annealing while continuing to update the mean field parameters.

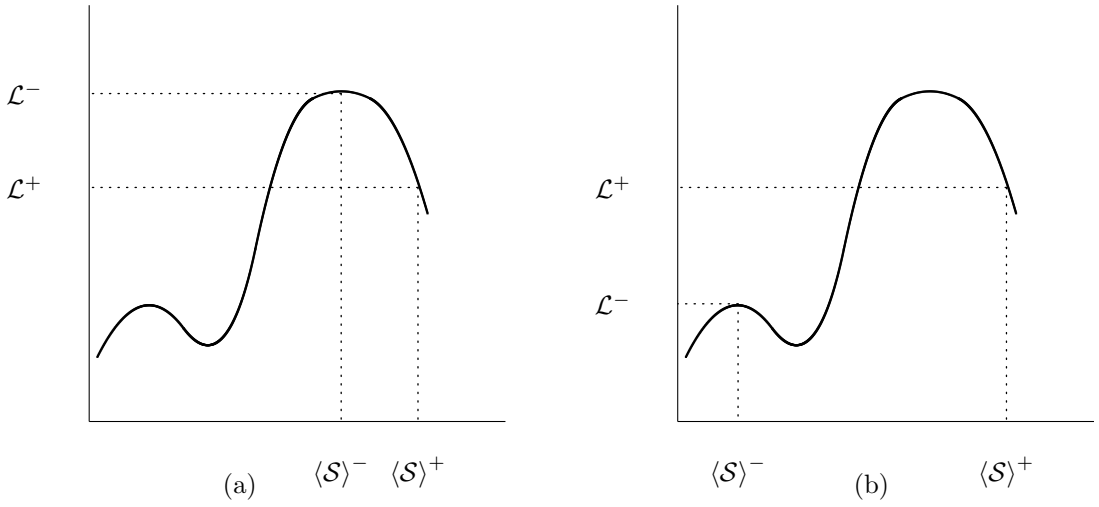


Figure 4.1: These figures show an embedded direction within the optimisation space of the expectations of the variables $\langle\mathcal{S}\rangle^-$. They show the positions of the expectations, $\langle\mathcal{S}\rangle^-$ and $\langle\mathcal{S}\rangle^+$, after optimisation through Eqn 4.11 and Eqn 4.10. Figure (a) shows the desired situation where $\langle\mathcal{S}\rangle^+$ and $\langle\mathcal{S}\rangle^-$ are in the same mode and the value associated with $\mathcal{L}^-$ is higher than that of $\mathcal{L}^+$ because the optimisation $\langle\mathcal{S}\rangle^+$ is taking place within an embedded sub-space of $\langle\mathcal{S}\rangle^-$. Figure (b) shows an unfortunate situation which can arise when $\langle\mathcal{S}\rangle^+$ and $\langle\mathcal{S}\rangle^-$ do not share the same mode. The resulting approximation $\mathcal{F}$ would be positive.

Note that our log likelihood approximation 4.14 could become positive. This would imply an approximation to $P(\mathcal{V})$ which is greater than one. The origin of this problem is depicted in Figure 4.1. It is a side effect of a lack of consistency between the mean field solutions for the joint and conditional distributions. In Section 4.5.2 we show the problems associated with this inconsistency may be resolved if we initialise the joint distribution parameters $\langle\mathcal{S}\rangle^-$, with values taken from the conditional approximation, $\langle\mathcal{S}\rangle^+$. First however we consider an alternative approach based on mixture representations.

In Section 2.4.2 we saw how the mixture representations are effective in improving the approximations to conditional distributions, we expect this effect to be more marked when we use the mixture to approximate a joint distribution. The joint distribution is more likely to be multi-modal than the conditional, especially when we consider data-sets which consist of clusters.

4.4.2 Mixture Based Q-distributions

As we are deriving the mixture based representations in the context of the joint distribution all expectations in this section, except where it is specified otherwise, are taken to be with respect to this distribution. We consider an approximating distribution of the form

$$Q_{\text{mix}}(\mathcal{S}) = \sum_{m=1}^{M} Q(m) \prod_i Q(s_i|m). \tag{4.16}$$

The mixing coefficients $Q(m)$ satisfy $Q(m) \geq 0$ and $\sum_m Q(m) = 1$. Optimisation of the variational approximation proceeds in a similar manner to that described in Chapter 2 in the context of sigmoid belief networks. Using the variational distribution (Eqn 4.16) we can express $\mathcal{L}^-$ from bound 1.39 in the form (Jaakkola and Jordan, 1998)

$$\mathcal{L}^-(Q_{\text{mix}}) = \sum_{m=1}^{M} Q(m)\mathcal{L}^-[Q(\mathcal{S}|m)] + I(m, \mathcal{S}), \tag{4.17}$$

where as in Chapter 2, $I(m, \mathcal{S})$ is the mutual information between the component label m and the variables $\mathcal{S}$ given by

$$I(m, \mathcal{S}) = \sum_m \sum_{\mathcal{S}} Q(m)Q(\mathcal{S}|m) \ln \left\{ \frac{Q(\mathcal{S}|m)}{Q_{\text{mix}}(\mathcal{S})} \right\}. \tag{4.18}$$

Once again, the first term is simply a linear combination of mean field contributions, and provides no improvement over the simple mean field bound. It is the second, mutual information, term which allows the mixture representation to give an improvement relative to mean field theory. However, the mutual information again involves an intractable summation over the states of the variables. In order to be able to treat it efficiently we again follow Jaakkola and Jordan (1998) and introduce the set of 'smoothing' distributions $R(\mathcal{S}|m)$, rewriting the mutual information Eqn 4.18 in the form

$$\begin{aligned} I(m, \mathcal{S}) &= \sum_{m,\mathcal{S}} Q(m)Q(\mathcal{S}|m) \ln R(\mathcal{S}|m) - \sum_m Q(m) \ln Q(m) \\ &\quad - \sum_{m,\mathcal{S}} Q(m)Q(\mathcal{S}|m) \ln \left\{ \frac{R(\mathcal{S}|m)}{Q(m)} \frac{Q_{\text{mix}}(\mathcal{S})}{Q(\mathcal{S}|m)} \right\}. \end{aligned} \tag{4.19}$$

As before we next make use of the following inequality

$$-\ln x \geq -\lambda x + \ln \lambda + 1 \tag{4.20}$$

to replace the logarithm in the third term in Eqn 4.19 with a linear function (conditionally on the component label m). This yields the same lower bound on the mutual information as was given in Chapter 2,

$$\begin{aligned} \mathcal{I}(m, \mathcal{S}) &= \sum_m \sum_{\mathcal{S}} Q(m)Q(\mathcal{S}|m) \ln R(\mathcal{S}|m) - \sum_m Q(m) \ln Q(m) \\ &\quad - \sum_m \lambda_m \sum_{\mathcal{S}} R(\mathcal{S}|m)Q_{\text{mix}}(\mathcal{S}) \\ &\quad + \sum_m Q(m) \ln \lambda_m + 1. \end{aligned} \tag{4.21}$$

Like in Chapter 2, summations over configurations of $\mathcal{S}$ in bound 4.21 can be performed analytically if we make factorisability assumptions about the smoothing distributions $R(\mathcal{S}|m)$.

In order to obtain the tightest bound within the class of approximating distributions, we proceed as in the last chapter and perform a free-form maximisation of the bound with respect to the variational distributions $Q(s_i|m)$ revealing

$$Q(s_k|m) = \frac{\exp\left(E'(k,m)s_k\right)}{\exp\left(E'(k,m)\right) + \exp\left(-E'(k,m)\right)}, \tag{4.22}$$

where

$$\begin{aligned} E'(k,m) &= \sum_j w_{kj} \left\langle s_j \right\rangle_m + \ln \frac{R(s_k = 1|m)}{R(s_k = -1|m)} \\ &\quad -\lambda_m \left[R(s_k = 1|m) + R(s_k = -1|m)\right]. \end{aligned} \tag{4.23}$$

A variables expectation under a single component of its own distribution can now be determined through an update equation of the form.

$$\left\langle s_k \right\rangle_m = \tanh\left(E'(k,m)\right), \tag{4.24}$$

where $\left\langle \ \cdot \ \right\rangle_m$ signifies the expectation with respect to the mth component of the mixture distribution.

We can handle optimisation of the mixing coefficients, $Q(m)$, the smoothing distributions, $R(\mathcal{S}|m)$, and the remaining variational parameters, λ_m, through fixed point equations with the same form as those listed in Chapter 2.

Since the various parameters are coupled, and we have optimised them individually keeping the remainder constant, it will be necessary to maximise the lower bound iteratively until some convergence criterion is satisfied. Having done this for a particular instantiation of the visible nodes, we can then determine the gradients of the bound with respect to the parameters governing the original belief network, and use these gradients for learning.

Once the variational approximations to the joint and conditional distributions have been optimised, the derivatives of the cost function are evaluated using

$$\frac{\partial \mathcal{F}_{\text{mix}}}{\partial w_{ij}} = \left\langle s_i \right\rangle^+ \left\langle s_j \right\rangle^+ - \sum_m Q(m) \left\langle s_i \right\rangle_m^- \left\langle s_j \right\rangle_m^-. \tag{4.25}$$

These derivatives are then used to update the weights using a gradient-based optimisation algorithm. The learning algorithm then alternates between optimisation of the variational approximation (analogous to an E-step) and optimisation of the weights (analogous to an M-step).

4.5 Results

4.5.1 Inference

Our variational framework allows expectations of the form $\langle s_i s_j \rangle_{\text{F}}$ to be approximated by deterministic expressions involving variational parameters, of the form $\langle s_i s_j \rangle^- = \sum_m Q(m) \left\langle s_i \right\rangle_m^- \left\langle s_j \right\rangle_m^-$, in

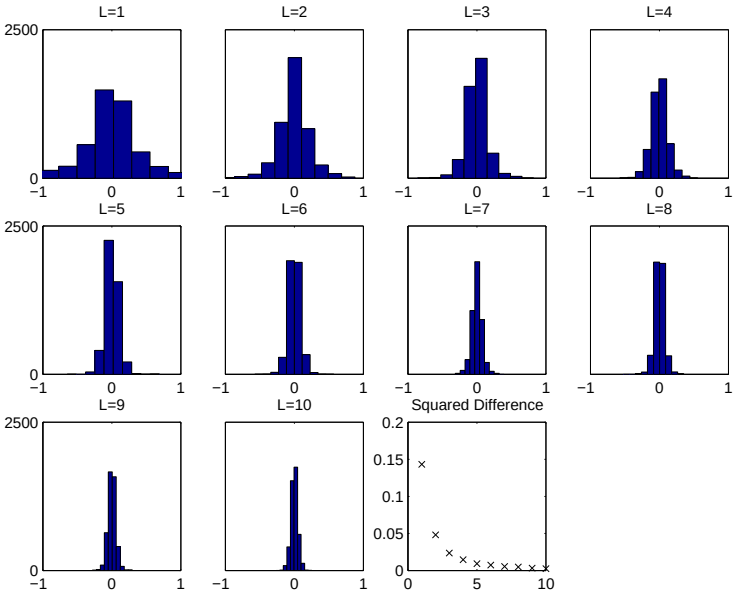


Figure 4.2: Histograms of the differences between true and approximate expectations for 100 randomly generated networks each having 55 independent parameters, for different numbers M of components in the mixture approximation, together with a summary of the dependence of the mean sum-of-squares of the differences on M.

which standard mean field theory corresponds to the case of just one component in the mixture. We now investigate how well this approach is able to approximate the true expectations, and how the approximation improves as the number of components in the mixture is increased.

For this purpose we consider small networks such that the (exponentially large) summation over states can be performed exactly, thereby allowing us to compare the variational approximation to the true expectation. The networks have ten variables and are fully interconnected, and hence have 55 independent parameters including biases. None of the units are observed. Evaluation of the expectations involves summing over $2^{10} = 1024$ configurations. We have generated 100 networks at random in which the weights and biases have been chosen from a uniform distribution over $(-1, 1)$. For each network we approximate the joint distribution of variables using mixture distributions involving M components, where $M = 1, \ldots, 10$, and the temperature parameter T was annealed in 8 steps from $T = 60$ to $T = 1$. In Figure 4.2 we show plots of the histograms of the differences between the approximate and exact expectations, given by

$$\sum_{m=1}^{M} Q(m) \langle s_i \rangle_m^- \langle s_j \rangle_m^- - \langle s_i s_j \rangle_{\mathrm{F}}, \tag{4.26}$$

together with a summary of the behaviour of the mean sum-of-squares of the differences (averaged over all 100 networks) versus the number of components in the mixture, M. We see that there is a clear and systematic improvement in the accuracy with which the expectations are approximated as the number of components in the mixture is increased.

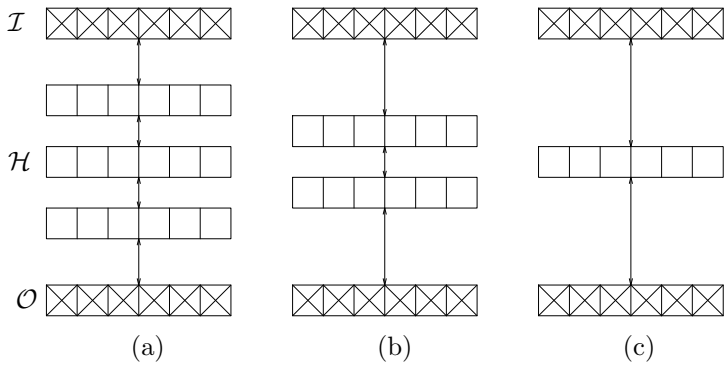


Figure 4.3: The structure of the networks used for the binary vector association task. Each node is indicated by a separate box. Instantiated nodes are indicated by a crossed box. In these diagrams we have arbitrarily chosen the topmost layer to be that of the input variables $\mathcal{I}$. (a) Three hidden layers. (b) Two hidden layers. (c) One hidden layer.

4.5.2 Learning

Mean Field Theory

We first review part of the work of Galland (1993). In this work Galland applied mean field theory to a supervised learning problem which involved the associations between ten randomly matched six bit input-output vector pairs. They considered networks upon which he had imposed a layered structure, where they allowed full connectivity between adjacent layers and no connectivity otherwise. Each layer contained six nodes. The resulting structures are shown in Figure 4.3. Galland made use of the cross-entropy error,

$$\mathcal{F}_{\text{obs}} = -\sum_{i \in \mathcal{O}} \frac{1 + \langle s_i \rangle^-}{2} \log \frac{v_i + 1}{2} + \frac{1 - \langle s_i \rangle^-}{2} \log \frac{v_i - 1}{2}, \tag{4.27}$$

to monitor the performance of the model during the learning phase. He points out that $\mathcal{F}_{\text{obs}}$ should be identical to $\mathcal{F}$ if the variables in the graph are truly independent in the joint distribution and therefore expects a good correspondence between the two error measures when the mean field approximation is valid.

We recreated Galland's mean field theory experiments. He made use of a temperature annealing scheme and damped asynchronous updates of the mean field parameters. He applied simple gradient ascent learning and made use of a very gradual learning rate, $\eta = 0.001$. He considered the learning to have converged when the values of the expectations $\{\langle s_{i \in \mathcal{O}} \rangle^-\}$ were within 0.2 of their target values. The mean field parameters for both phases were initialised at zero. We recovered similar results to those he presents. Figure 4.4(a) shows the result for the three-layered network. In common with Galland we found this to be the worst performing result. In Figure 4.4(b) we present results for the same experiments, but this time with the joint distribution approximation initialised from the conditional. We see a marked improvement in these results. The network has 'converged' after only 98 iterations. Note the strength of the correlation between the two curves. As the model converges the lines overlay each other. This result is obtained despite a learning rate ten times higher than

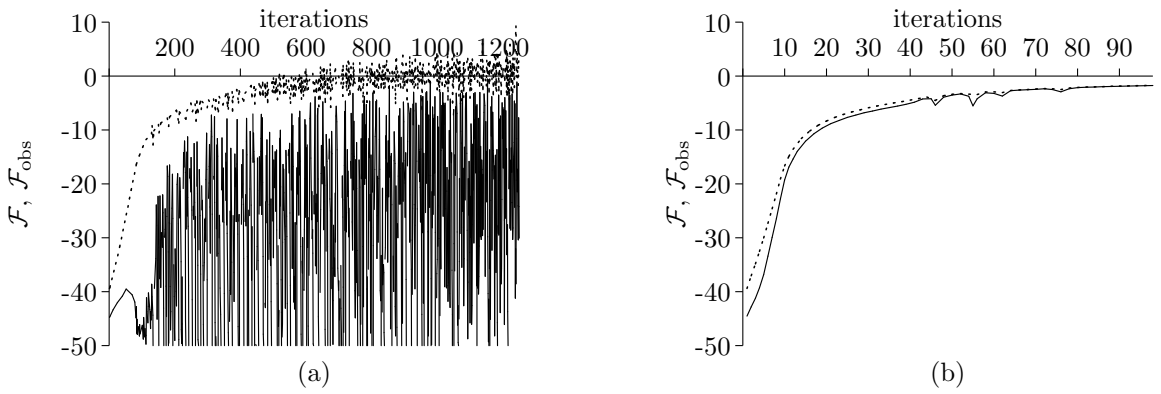


Figure 4.4: Learning curves from the binary vector association task. Figure (a) shows the curves from a recreation of Galland's test. The objective function is shown as a dotted line while the cross-entropy error is shown as a solid line. Figure (b) shows learning curves for the same task but when the approximation to the joint distribution is initialised from the approximation to the conditional. The learning rate was a factor of ten higher in for this test (note the difference in the x-axis scales).

that used in the previous experiment. A larger learning rate is expected to hinder the stability of the system (Hinton, 1989).

Mixture Representations

In the previous Section 4.5.2 we saw how the use of mixture representations can lead to improved accuracy of inference compared with standard mean field theory, but with mean field theory giving such impressive results why would we still look to the mixture formalism? If we consider unsupervised learning problems we note that the joint distribution will be identical for all presented patterns. In a batch learning context, we may therefore want to utilise the same approximation to the joint distribution for learning many different patterns. Even in an on-line learning context we may feel that some consistency in our approximation may be appropriate. This rules out initialisations based on the conditional approximation in the manner we have discussed. Instead, we turn to the mixture formalism. Once again, for simplicity we use gradient ascent learning, with gradients evaluated using Eqn 4.15 or Eqn 4.25. First we consider a simple toy problem designed to have a multi-modal joint distribution, and we then apply our approach to a problem involving images of hand-written digits.

Toy Problem

As a simple example of a problem leading to a multi-modal distribution we follow Kappen and Rodriguez (1998) and consider a network consisting of two visible nodes and no hidden nodes, together with a data-set consisting of two copies of the pattern $(1, 1)$ and one copy of the pattern $(-1, -1)$. In this case the approximation to the joint distribution needs to be bi-modal for the network to have learned a solution to the problem. Due to the small size of the network, comparison with exact results is straightforward. We apply standard mean field theory, and compare it with a mixture model having two components, and with learning using the exact log likelihood gradient. In the inference stage,

the variational parameters are iteratively updated until the cost function $\mathcal{F}_{\text{mft}}$ changes by no more than 0.01% (up to a maximum of 20 iterations). No annealing or damping was used. The network weights were initialised using parameters drawn from a zero-mean Gaussian distribution having a standard deviation of 0.1, and learning is by gradient ascent with a learning rate parameter of 0.25. The variational approximation to the joint distribution at each iteration was initialised to that of the previous iteration. The results are shown in Figure 4.5.

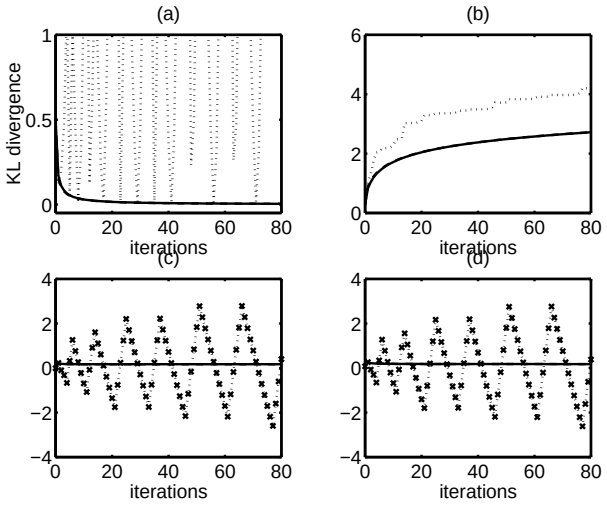


Figure 4.5: Results from learning in the toy problem. (a) The KL divergence between the true distribution of the training data and the distribution represented by the network as a function of the number of iterations of learning. The results from mean field theory (dotted curve) are wildly unstable whereas the results obtained using a mixture model (dashed curve) are well behaved and indistinguishable from those found using the true log likelihood (solid curve). The remaining figures show the corresponding evolution of the weight parameter (b) and the two bias parameters (c) and (d) for the three approaches.

Mean field theory seen to be quite unstable during learning. In particular, the bias parameters undergo systematic oscillations[1]. To investigate this further we plot an expanded region of the training curve from Figure 4.5 (c) in Figure 4.6 together with the mean field parameters at each learning step. We see that the uni-modal approximating distribution of mean field theory is oscillating between the two potential modes of the joint distribution, as the algorithm tries to solve this multi-modal problem.

This phenomenon can be analysed in terms of the stability structure of the mean field solutions. We find that, for the first few iterations of the learning algorithm when the weight value is small, the mean field equations Eqn 4.11 and 4.10 exhibit a single, stable solution with small values of the mean field parameters. However, once the weight value grows beyond a critical value, two stable solutions (and one unstable solution) appear whose values depend on the bias parameters. Evolution of the bias parameters modifies the shape of the stability diagram and causes the mean field solution to oscillate between the two stable solutions, as the parameter vector repeatedly 'falls off' the edges of the cusp bifurcation (Parisi, 1988).

[1]The oscillations are systematic because we have initialised the joint approximation at each iteration with that of the previous iteration. In other experiments where the mean field solution was initialised randomly, the oscillations occurred but in a more random manner.

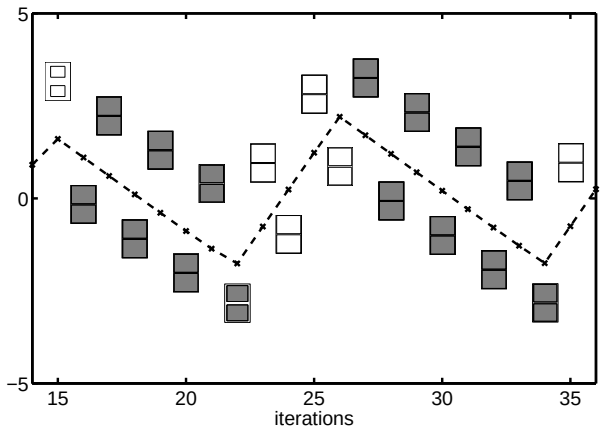


Figure 4.6: Expanded plot of the evolution of one of the bias parameters during training with mean field theory. Also shown are plots of the mean field parameters, represented as 'Hinton' diagrams in which each of the two parameters is denoted by a square whose area is proportional to the parameter value and white denotes negative values, while grey denotes positive values.

Handwritten Digits

As a second example of learning with mixture representations we turn to a more realistic problem involving hand-written digits[2] which have been pre-processed to give 8×8 binary images. We extracted a data-set consisting of 700 examples of each of the digits 0 through 9. Examples of the training data are shown in Figure 4.7.

Figure 4.7: Examples of the hand-written digits from the training set.

The networks consisted of 64 visible nodes in an 8×8 grid, with each visible node connected to its neighbours on both diagonals and in the horizontal and vertical directions. The network also had ten hidden nodes which are fully connected with each other and with all the visible nodes. Additionally all nodes had an associated bias. An annealing schedule was used during the inference involving 7 successive values (100, 50, 25, 10, 5, 2 and 1) of the temperature parameter, T. One pass of updates was performed at each value until 1 was reached, updates where then continued until convergence. Once again the joint distribution approximation was initialised at each step with the parameters from the previous step. The conditional approximation parameters $\{\langle s_{i\in\mathcal{H}}\rangle^{+}\}$ were then initialised with the expected states under the joint distribution $\{\langle s_{i\in\mathcal{H}}\rangle^{-}\}$. Contrast this with the initialisation proposed in Section 4.5.2 and utilised earlier in this section. Since we are using different numbers of components

[2]Available on the CEDAR CDROM from the U.S. Postal Service Office of Advanced Technology.

in the two approximations, it is not possible to follow the previously suggested scheme.

Learning was achieved through 30 iterations of gradient ascent in the parameters w_{ij}, with a learning rate of $0.1/N$ where $N = 7,000$ is the total number of patterns in the training set.

Due to the size of the network it is no longer possible to perform exact calculations for the joint distribution. We therefore compare standard mean field theory with a mixture distribution having ten components. Figure 4.8 shows the evolution of the cost functions $\mathcal{F}_{\text{mft}}$ and $\mathcal{F}_{\text{mix}}$. Again we see

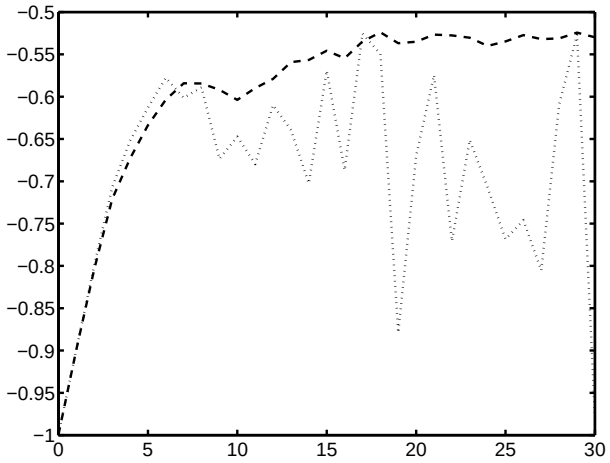


Figure 4.8: Evolution of the cost functions $\mathcal{F}_{\text{mft}}$ (dotted curve) and $\mathcal{F}_{\text{mix}}$ (dashed curve) for the digits problem.

that mean field theory is relatively unstable compared to the mixture model.

Figure 4.9 shows the evolution of the variational parameters from the approximation to the joint distribution (plotted for the visible units only), as this provides insight into the behaviour of the algorithm. We see that simple mean field theory exhibits substantial 'mode hopping', while the components of the mixture distribution are much more stable (although some tendency to mode hop is still evident, suggesting that a larger number of components in the distribution may be desirable).

4.6 Discussion

We have seen how, in the context of a simple supervised learning problem, some of the limitations associated with mean field learning in the Boltzmann machine may be overcome through appropriate initialisations. We have also shown how intuitively more sensible joint distribution approximations based on mixture representations may be implemented, particularly in the context of unsupervised learning problems.

There is a relationship between our approach of initialising the joint distribution approximation with that of the conditional and the independently inspired approach of Hinton's product of experts (Hinton, 1999). In his product of experts Hinton investigates systems where the conditional distribution, $P(\mathcal{H}|\mathcal{V})$, may be determined exactly. He uses this distribution to initialise a Gibbs sampler which approximates the joint distribution. To be sure that the samples from the joint distribution are related to the conditional distribution, Hinton only utilises one step in his Gibbs sampler.

Figure 4.9: Variational parameters from $Q_{\mathrm{mft}}(\mathcal{H}, \mathcal{V})$ in which successive rows correspond to successive iterations of learning running from 0 to 30. The right most column corresponds to mean field theory while the first ten columns correspond to the ten components in the mixture model. In each case only the parameters corresponding to the visible variables are shown.

The approach of utilising the conditional distribution to initialise the joint distribution could prove useful in a variety of probabilistic models where the partition functions are intractable. Traditionally the intractabilities of the partition function have been a major handicap in the implementation of graphical models containing undirected links such as Markov random fields.

Acknowledgements

Some of the work in this chapter was undertaken in collaboration with Christopher M. Bishop and Michael I. Jordan and formed the basis of a conference paper (Lawrence *et al.*, 1998).

Chapter 5

Bayesian Neural Networks

5.1 A Brief Introduction

Bayesian inference provides a natural framework for handling issues such as model selection. As a result there has been great interest in Bayesian approaches to probabilistic models. Neural networks can be viewed as probabilistic models and therefore may be viewed from the Bayesian perspective. Unfortunately, since exact inference in Bayesian neural networks is non-analytic to compute, approximate methods such as the evidence procedure, Monte Carlo sampling and variational inference must be utilised. In this chapter we present a general overview of the Bayesian approach, with a particular emphasis on the variational procedure. We then present an approximating distribution based on *mixtures* of Gaussian distributions and show how it may be implemented for regression neural networks. We present results on a simple toy problem and on two real world data-sets.

First we briefly review the different Bayesian approaches to neural networks (Section 5.3) and then we discuss in more depth the variational perspective (Section 5.4). We derive in general the rigourous lower bound on the marginal likelihood $p(D|\mathcal{M})$ and then demonstrate the implementation of this bound with the mixture of Gaussians as an approximating distribution. Finally in Section 5.5 we apply the approach to both synthetic and real-world data and compare the performance of our approximation with other well established methods.

5.2 Bayesian Inference

We described in the introduction how learning from data consists of developing a model, $\mathcal{M}$, with a particular parameterisation $\boldsymbol{\theta}$, which provides a good representation of a set of data, D, containing N instantiations. In the case of a regression problem, this data consists of inputs $\{\mathbf{x}_1 \dots \mathbf{x}_N\} \in \mathbb{R}^I$ and targets $\{\mathbf{t}_1 \dots \mathbf{t}_N\} \in \mathbb{R}^P$ which are assumed to have come from some generating function $y(\mathbf{x})$. We mentioned in Chapter 1 that learning may be undertaken by adapting the parameters, $\boldsymbol{\theta}$, through optimisation of an information theoretic criteria such as the likelihood. This single estimate, $\boldsymbol{\theta}^*$, may

then used to make predictions. We also mentioned the bias-variance dilemma. This means that when the number of data points, N, is 'small' the parameters are poorly estimated by this approach. A complex model will typically better fit the training data without being able to give good predictions on unseen data, i. e. it will exhibit poor generalisation capabilities. This phenomenon is known as *over-fitting* and we must look to techniques such as regularisation in order to solve this problem. Such techniques often use an extra partition of the data known as a validation set in order to evaluate the generalisation capabilities of the model during the training phase.

In this chapter, we follow the alternative approach known as Bayesian inference. We have discussed already how in the Bayesian approach we consider the parameters vector, $\boldsymbol{\theta}$, of the model to be random variables. We seek to infer the form of the conditional distribution of $\boldsymbol{\theta}$ given the training data, D, and the model's structure, $\mathcal{M}$. We must therefore define a *prior* distribution $p(\boldsymbol{\theta})$, and the required conditional or *posterior* distribution, $p(\boldsymbol{\theta}|D, \mathcal{M})$, may then be determined via Bayes's theorem:

$$p(\boldsymbol{\theta}|D, \mathcal{M}) = \frac{p(D|\boldsymbol{\theta}, \mathcal{M})p(\boldsymbol{\theta})}{p(D|\mathcal{M})}, \tag{5.1}$$

where $p(D|\boldsymbol{\theta}, \mathcal{M})$ is the likelihood of the data given the parameter vector as defined by our model. Predictions for unseen data can then be made by considering expectations under the posterior distribution.

Regularisation can be given a natural interpretation within the Bayesian framework and regularisation coefficients may be determined without the need to resort to a validation set.

Unfortunately in many cases the integrals required for determining $p(D|\mathcal{M})$ are non-analytic. We have already presented variational approaches for handling intractablilities arising from discrete sums. In this chapter we will demonstrate how the variational framework may be applied to resolve the analytic intractabilities present in the integrals required for exact Bayesian inference. The multi-layer perceptron provides one example of a model for which exact Bayesian solutions are not possible. In response approximate methods such as the Laplace approximation (MacKay, 1992; Gull, 1988), and Monte Carlo sampling (Neal, 1996) have been successfully applied for both regression and classification problems.

The variational approach was introduced in the context of the multi-layer perceptron by Hinton and van Camp (1993). Once again it involves the selection of a class of distributions which may represent good approximations to the posterior, yet are also simple enough to allow necessary integrals to be performed. The posterior distribution of interest will now be a continuous probability density function, but its parameterisation may still be determined through minimisation of the Kullback-Leibler (KL) divergence between the true posterior and the approximating distribution. Hinton and van Camp used a Gaussian distribution with a diagonal covariance matrix for the approximation, Barber and Bishop (1998) extended their approach and demonstrated how a Gaussian with a full covariance matrix could be utilised. In this chapter, we review the variational approach for this continuous system and demonstrate its implementation with an approximation based on mixtures of Gaussian distributions which uses a continuous implementation of the framework we outlined in Chapter 2.

5.3 Bayesian Inference in Neural Networks

We define our model to be a two-layer feed-forward neural network with I input nodes, H hidden nodes and a single output node. The network function may be written as:

$$f(\mathbf{x}, \mathbf{w}) = \sum_{h=1}^{H} v_h g(\mathbf{u}_h^{\mathrm{T}} \mathbf{x}), \tag{5.2}$$

where $\mathbf{w} = \{\mathbf{u}_1 \dots \mathbf{u}_H, \mathbf{v}\}$ is a vector representing the parameters or 'weights' of the network in which the input to hidden weights are represented by H vectors $\mathbf{u}_h$, each vector being the weights that 'fan-in' to hidden unit h and $\mathbf{v}$ is the vector of the hidden to output weights, consisting of H elements v_h. We account for hidden layer 'biases' by considering additional input and hidden nodes whose values are taken to be one at all times. The activation function, g, is often taken to be a hyperbolic tangent, for reasons of tractability though, we follow Barber and Bishop (1998) in our use of the cumulative Gaussian distribution function:

$$g(x) = \sqrt{\frac{2}{\pi}} \int_0^x \exp(-t^2)\ dt, \tag{5.3}$$

which has a similar form to the hyperbolic tangent.

Given our data-set $D = \{\mathbf{x}_n, t_n\}_{n=1}^N$ our task is to interpolate through the data. We model the data as being derived from a underlying true function $y(\mathbf{x})$ with Gaussian noise added. This leads us to consider a likelihood function of the form:

$$p(D|\mathbf{w}, \beta) = \left(\frac{\beta}{2\pi}\right)^{N/2} \exp(-\beta E_D(D, \mathbf{w})), \tag{5.4}$$

where β is the inverse noise variance and $E_D(D, \mathbf{w})$ represents the error on the data-set:

$$E_D(D, \mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} (t_n - f(\mathbf{x}_n, \mathbf{w}))^2. \tag{5.5}$$

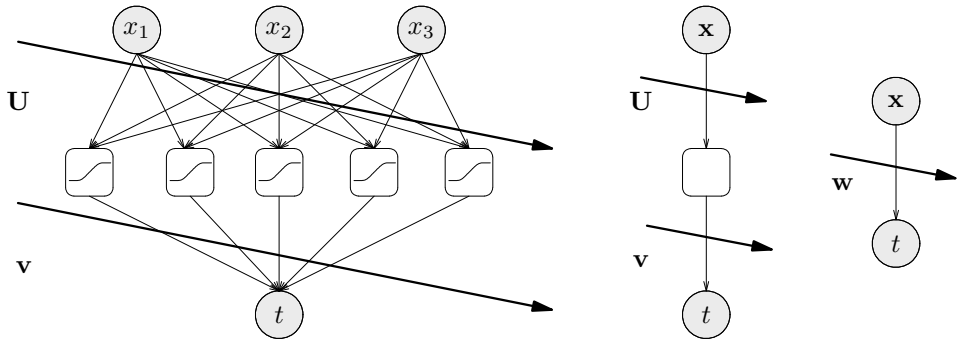


Figure 5.1: Maximum likelihood learning for for single hidden layer neural networks. The hidden layer is dependent on the inputs and the adaptable parameters **U** in a deterministic manner. The outputs in turn are dependent on the hidden layer and the adaptable parameters **v** in a deterministic manner. The bias parameters are omitted for clarity. The diagram on the left may be more concisely represented by that on the right, where **w** represents a set of parameters containing **U** and **v**. In the Bayesian approach we are undertaking, the adaptive parameters will be considered as an extra node in the graph — see Figure 5.2.

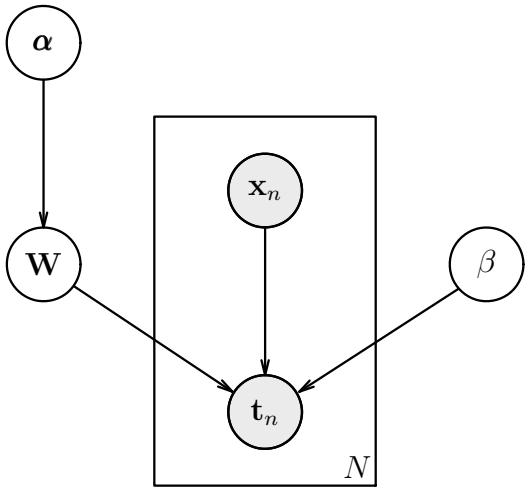


Figure 5.2: The super-graph of the proposed Bayesian formalism for the neural network. Note how the parameters are now represented as stochastic variables which require priors.

We define a prior distribution $p(\mathbf{w}|\alpha)$ which is taken to be a zero mean Gaussian distribution:

$$p(\mathbf{w}|\alpha) = \left(\frac{\alpha}{2\pi}\right)^{K/2} \exp(-\alpha E_w(\mathbf{w})), \tag{5.6}$$

where α is a hyper-parameter governing the prior, K is the number of weights in the model and $E_w(\mathbf{w}) = \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w}$. This choice of prior corresponds to a belief that positive and negative weights are equally probable and that small weights are more probable than larger ones, leading to smoother interpolations. More complex priors are possible in which the weights grouped according to their connectivity. Hyper-parameters are then associated with each separate grouping.

We also require a prior over the inverse noise variance β. As it represents the inverse variance of a Gaussian distribution a conjugate choice of prior is the gamma distribution

$$p(\beta) = \mathrm{gam}(\beta|a_\beta, b_\beta). \tag{5.7}$$

Where we take the gamma distribution to be:

$$\mathrm{gam}(\tau|a, b) = \frac{b^a}{\Gamma(a)}\tau^{a-1}\exp\left(-b\tau\right), \tag{5.8}$$

where a and b are constants, and $\Gamma(\cdot)$ is the Gamma function. The ratio a/b is the expectation of τ under this distribution and the value of a/b^2 is the variance of τ under the distribution.

The second level of inference will be concerned the hyper-parameter α. A full Bayesian treatment at this level makes use of hyper-priors. Since α represents the inverse variance of a Gaussian distribution we represent our hyper-prior by the gamma distribution:

$$p(\alpha) = \mathrm{gam}(\alpha|a_\alpha, b_\alpha). \tag{5.9}$$

The first level of inference deals with the posterior distribution,

$$p(\mathbf{w}, \beta|D, \alpha) = \frac{p(D|\mathbf{w}, \beta)p(\mathbf{w}|\alpha)p(\beta)}{p(D|\alpha)} \tag{5.10}$$

$$= \frac{1}{Z(\alpha)}\exp(-\alpha E_w(\mathbf{w}) - \beta E_D(D, \mathbf{w}, \beta) - E_\beta(\beta)), \tag{5.11}$$

where we have dropped the conditioning on the model, $\mathcal{M}$ to avoid cluttering the notation and $Z(\alpha) = \int \exp(-\alpha E_w(\mathbf{w}) - \beta E_D(D, \mathbf{w}))d\mathbf{w}$ is a normalisation constant.

The second level of inference concerns the *hyper-posterior* distribution

$$p(\alpha|D) = \frac{p(D|\alpha)p(\alpha)}{p(D)}. \tag{5.12}$$

The term $p(D|\alpha)$ is sometimes called the *evidence* for α, and under certain conditions may be considered a good approximation to the *model evidence* which refers to $p(D|\mathcal{M})$. We refer to $p(D|\alpha)$ as the *type II likelihood* and the model evidence as the *model likelihood*. The type II likelihood is the denominator of Eqn 5.10 and can be evaluated by integration over $\mathbf{w}$:

$$p(D|\alpha) = \int p(D|\mathbf{w}, \beta)p(\mathbf{w}|\alpha)p(\beta)d\mathbf{w}d\beta. \tag{5.13}$$

Finally network predictions on unseen data are obtained by looking at the expected value of the output under the posterior distribution,

$$\langle f(\mathbf{x}, \mathbf{w})\rangle_{p(\mathbf{w},\beta|D)} = \int f(\mathbf{x}, \mathbf{w})p(\mathbf{w}, \beta|D)d\mathbf{w}d\beta, \tag{5.14}$$

for which, since the network function is not dependent on β, we require the posterior distribution of the weights given the data

$$p(\mathbf{w}|D) = \int p(\mathbf{w}|D, \alpha, \beta)p(\alpha)p(\beta)\ d\alpha\ d\beta. \tag{5.15}$$

The variance of predications, $\left\langle f(\mathbf{x}, \mathbf{w})^2\right\rangle - \langle f(\mathbf{x}, \mathbf{w})\rangle^2$ may also be of interest for placing error-bars on the predictions.

The Bayesian framework provides a principled way in which to handle generalisation and confidence intervals on predictions. Unfortunately exact Bayesian learning requires the computation of expectations under the posterior distribution $p(\mathbf{w}|D)$ which involves a high dimensional non-linear integral.

5.3.1 Hybrid Monte Carlo Sampling

Neal (1996) applied hybrid Monte Carlo sampling techniques to obtain samples from the posterior distribution for Bayesian neural networks. If a representative set of samples, $\{\mathbf{w}_s\}_{s=1}^S$ from $p(\mathbf{w}|D)$ are obtained, sample based approximations to the required expectations may be made.

$$\langle f(\mathbf{x}, \mathbf{w})\rangle_{P(\mathbf{w}|\mathcal{V})} \approx \frac{1}{S}\sum_{s=1}^{S} f(\mathbf{x}, \mathbf{w}_s) \tag{5.16}$$

Although these samples may be more representative of the posterior than a simple functional approximation, issues remain such as convergence of the sampler, and the lack of an approximation to the model likelihood.

5.3.2 The Laplace Approximation

The evidence procedure (MacKay, 1992; Gull, 1988) involves making the Laplace approximation to $p(\mathbf{w}|D, \alpha, \beta)$ and makes use of type II maximum likelihood for determining the hyper-parameters.

The Laplace approximation involves finding a single local optimal point $\mathbf{w}^*$ in weight space and constructing a full covariance Gaussian approximation to the posterior distribution around this point.

$$p(\mathbf{w}|D, \alpha, \beta) \approx \frac{|\mathbf{H}|^{\frac{1}{2}}}{(2\pi)^{\frac{W}{2}}} \exp\left\{-\frac{1}{2}(\mathbf{w} - \mathbf{w}^*)^{\mathrm{T}}\mathbf{H}(\mathbf{w} - \mathbf{w}^*)\right\}, \tag{5.17}$$

where the matrix $\mathbf{H}$ is the Hessian about point $\mathbf{w}^*$,

$$\mathbf{H} = \nabla\nabla(\beta E_D(D, \mathbf{w}) + \alpha E_w(\mathbf{w})). \tag{5.18}$$

The treatment of the hyper-parameter, α, and the parameter, β, involves also an approximation of the posterior probability distribution $p(\alpha, \beta|D)$. It is assumed that the distribution is sharply peaked around the most probable values α^* and β^*. It is then possible to get re-estimation formulae for the hyper-parameters as a function of the eigenvalues of the Hessian matrix $\mathbf{H}$. A review of the approach can be found in MacKay (1995b).

The evidence procedure is a two-step procedure. First, the maximum a posteriori value for the weights is obtained by maximising the penalised log-likelihood. This local maximum is then used for estimating the form $p(D|\alpha, \beta)$. Re-estimation equations for α and β are obtained by maximising $p(D|\alpha, \beta)$. The whole process is repeated until convergence. If the posterior distribution $p(\beta|D, \alpha)$ is strongly peaked around β then $p(D|\alpha, \beta)$ may be considered a good approximation to the type II likelihood, $p(D|\alpha)$, which in turn will be a good approximation to the model likelihood if the type II likelihood is strongly peaked around α. Practical details of the implementation and some results for a regression problem can be found in Thodberg (1996).

While the Laplace approximation provides a quick and easy method of approximating the posterior distribution, the approximation is only locally valid, it is concentrated at a mode of the posterior. We saw in the introduction that, even if the posterior is uni-modal, this approximation may not be representative of the mass of the distribution. We therefore look to a variational approximation which is more responsive to the true mass of the distribution.

5.4 Variational Inference

Variational inference was first applied in the context of neural networks by Hinton and van Camp (1993), where in was introduced through the minimum description length principle. The approximation to the posterior in this approach is more responsive to the mass of the distribution than that of the evidence procedure and it has the additional advantage of providing a rigorous lower bound on the model likelihood.

5.4.1 Variational Techniques

Consider the model likelihood for a Bayesian neural network

$$p(D) = \int p(D, \mathbf{w}, \alpha, \beta) d\mathbf{w}\ d\alpha\ d\beta \tag{5.19}$$

$$= \int p(D|\mathbf{w}, \beta) p(\mathbf{w}|\alpha) p(\alpha) p(\beta)\ d\mathbf{w}\ d\alpha\ d\beta \tag{5.20}$$

As we saw for the discrete systems we have studied so far, the likelihood may be bounded from below through the introduction of a variational distribution $q(\mathbf{w}, \alpha, \beta)$,

$$\ln p(D) \geq \int q(\mathbf{w}, \alpha, \beta) \ln \frac{p(D, \mathbf{w}, \alpha, \beta)}{q(\mathbf{w}, \alpha, \beta)}\ d\mathbf{w}\ d\alpha\ d\beta. \tag{5.21}$$

We may then make progress by considering factorisability constraints. We attempt to group the variables into tractable sub-structures. We assume

$$q(\mathbf{w}, \alpha, \beta) = q_w(\mathbf{w}) q_\alpha(\alpha) q_\beta(\beta), \tag{5.22}$$

which corresponds to a factorised approximating distribution. We may now rewrite the bound

$$\ln p(D) \geq \int q_w(\mathbf{w}) q_\alpha(\alpha) q_\beta(\beta) \ln p(D, \mathbf{w}, \alpha, \beta)\ d\mathbf{w}\ d\alpha\ d\beta + S(q_w(\mathbf{w})) + S(q_\beta(\beta)) + S(q_\alpha(\alpha)), \tag{5.23}$$

where $S(p(x))$ denotes the entropy of the distribution $p(x)$.

It remains to determine the form of our approximating distributions. We first turn to the approximating distribution $q_w(\mathbf{w})$, the other distributions will be dealt with in Section 5.4.3.

5.4.2 Gaussian q-distributions

Generally in this thesis we have used a free-form optimisation to determine our choice of approximating distribution. We may attempt the same strategy for the Bayesian neural network. Doing so reveals

$$q_w(\mathbf{w}) \propto \exp\left[\frac{1}{2} \langle\beta\rangle \sum_{n=1}^{N} (t_n - f(\mathbf{x}_n, \mathbf{w}))^2 - \frac{1}{2} \langle\alpha\rangle \mathbf{w}^{\mathrm{T}} \mathbf{w}\right]. \tag{5.24}$$

Unfortunately determining the normalisation constant of this distribution is not straightforward, it is a high-dimensional non-linear integral. In other words the parameters $\mathbf{w}$ are not governed by a tractable sub-structure. We therefore opt for the strategy of explicitly choosing a functional form for our q-distribution and optimising the parameters of our approximating distribution through maximisation of the lower bound 5.23. Performing this optimisation is equivalent to minimising the KL divergence between our approximating distribution and the actual best approximating distribution in 5.24.

Hinton and van Camp (1993) selected a diagonal covariance Gaussian for the approximating distribution q_w. The advantage of this choice is the simplicity of its implementation. In particular, if the activation function g of the network is chosen to be the cumulative Gaussian all but one of the required integrals are analytically tractable. In Appendix C we show how the remaining integral may be approximated efficiently. However, such a simple representation can not account for correlations between network weights present in the posterior.

5.4.3 Mixture Based q-distributions

In order to effectively model a multi-modal posterior distribution, we propose to apply the mixture formalism:

$$q_w(\mathbf{w}) = \sum_{m=1}^{M} Q_{\mathbf{w}}(m) q_w(\mathbf{w}|m), \tag{5.25}$$

where we take each component to be a diagonal covariance Gaussian, $q_w(\mathbf{w}|m) = \mathcal{N}(\bar{\mathbf{w}}_m, \mathbf{\Sigma}_{q_m})$. This representation is able to capture correlations between the weights and would also be able to represent skewness or kurtosis in the posterior. In practice however, strong correlations between weights may require many components for a good approximation to the posterior distribution and it may not be practical to optimise such large numbers of components. In particular, we might expect to underperform an approach based on full covariance Gaussians when these correlations are present in the posterior distribution. Barber and Bishop (1998) implemented a full covariance Gaussian for their variational approximation to the posterior. However the efficient implementation of this approach remains an unresolved issue. The framework requires the use of either a one dimensional numerical integral or a three-dimensional look up table to evaluate the lower bound. We discuss the covariance structures which may be handled under the approximation we applied later in the chapter.

Lower bound on the entropy

The use of the mixture distribution does not complicate the calculation of the first term in bound 5.23 beyond what has been tackled in previous works. Once again it is the entropy of the mixture distribution that presents problems, namely we are required to evaluate the expectation of the logarithm of a sum. As for the discrete case, we make progress by considering the mutual information, $I(m; \mathbf{w})$, between the component label m of the mixture distribution and an observed weight vector. The entropy may be rewritten as:

$$\begin{aligned} S(q_w(\mathbf{w})) &= -\sum_{m=1}^{M} Q_{\mathbf{w}}(m) \int q_w(\mathbf{w}|m) \ln \left[\frac{q_w(\mathbf{w})}{q_w(\mathbf{w}|m)} q_w(\mathbf{w}|m) \right] d\mathbf{w} \\ &= \sum_{m=1}^{M} Q_{\mathbf{w}}(m) S(q_w(\mathbf{w}|m)) + I(m; \mathbf{w}). \end{aligned} \tag{5.26}$$

The first term in Eqn 5.26 is a sum of entropies and is analytically tractable for Gaussian distributions[1]. The second term, the mutual information, takes the form

$$I(m; \mathbf{w}) = \sum_{m=1}^{M} Q_{\mathbf{w}}(m) \int q_w(\mathbf{w}|m) \ln \frac{q_w(\mathbf{w}|m)}{q_w(\mathbf{w})} \, d\mathbf{w}. \tag{5.27}$$

[1] the entropy of each component is simply $S(q_w(\mathbf{w}|m)) = \frac{1}{2} \ln |\mathbf{\Sigma}_{q_m}| + \frac{K}{2}(1 + \ln 2\pi)$.

As in the discrete model, we are still left with a logarithm of a sum in this second term and as before we look to a further lower bound

$$\begin{aligned} I(m;\mathbf{w}) &\geq \sum_{m=1}^{M} Q_{\mathbf{w}}(m) \int q_w(\mathbf{w}|m) \ln r(\mathbf{w}|m)\ d\mathbf{w} \\ &\quad - \sum_{m,m'}^{M} Q_{\mathbf{w}}(m)\lambda_{m'} \int r(\mathbf{w}|m') q_w(\mathbf{w}|m)\ d\mathbf{w} \\ &\quad - \sum_{m=1}^{M} Q_{\mathbf{w}}(m) \ln Q_{\mathbf{w}}(m) + \sum_{m=1}^{M} Q_{\mathbf{w}}(m) \ln \lambda_m + 1, \end{aligned} \tag{5.28}$$

where we have introduced the variational distributions $r(\mathbf{w}|m)$ and the variational parameters λ_m for tightening the bound. These parameters and distributions correspond to those used in the discrete implementation in Chapter 2. Once again selection of the smoothing distributions $r(\mathbf{w}|m)$ is important. Again if we were to take $r(\mathbf{w}|m) \propto q_w(\mathbf{w}|m)/q_w(\mathbf{w})$ and maximise over the variational parameters λ_m, we would recover the mutual information exactly. As before though, such a choice would lead to no reduction in complexity of the calculations, and we opt instead to use a Gaussian form, $r(\mathbf{w}|m) \propto \mathcal{N}(\bar{\mathbf{r}}_{r_m}, \mathbf{\Sigma}_{r_m})$ leading to the following bound (see Appendix C)

$$\begin{aligned} I(m;\mathbf{w}) \geq & -\frac{1}{2}\sum_{m=1}^{M} Q_{\mathbf{w}}(m) \left\{ \mathrm{tr}(\mathbf{\Sigma}_{q_m}\mathbf{\Sigma}_{r_m}^{-1}) + (\bar{\mathbf{w}}_m - \bar{\mathbf{r}}_{r_m})^{\mathrm{T}} \mathbf{\Sigma}_{r_m}^{-1} (\bar{\mathbf{w}}_m - \bar{\mathbf{r}}_{r_m}) \right\} \\ & - \sum_{m,n}^{M} \frac{Q_{\mathbf{w}}(m)\lambda_{m'}}{|\mathbf{I} + \mathbf{\Sigma}_{q_m}\mathbf{\Sigma}_{r_{m'}}^{-1}|^{\frac{1}{2}}} \exp\left\{ -\frac{1}{2}(\bar{\mathbf{w}}_m - \bar{\mathbf{r}}_{m'})(\mathbf{\Sigma}_{q_m} + \mathbf{\Sigma}_{r_{m'}})^{-1}(\bar{\mathbf{w}}_m - \bar{\mathbf{r}}_{m'}) \right\} \\ & - \sum_{m=1}^{M} Q_{\mathbf{w}}(m) \ln Q_{\mathbf{w}}(m) + \sum_{m=1}^{M} Q_{\mathbf{w}}(m) \ln \lambda_m + 1 \end{aligned} \tag{5.29}$$

$$\equiv \mathcal{I}(m;\mathbf{w}). \tag{5.30}$$

As in the discrete case, the smoothing distributions $r(\mathbf{w}|m)$ control the tightness of the bound. Optimisation with respect to the parameters of these distributions increases the bound. In our experiments, we used diagonal covariance matrices for r although the extension for full covariance is straightforward.

Note that the maximum possible value for the mutual information is $-\sum_{m=1}^{M} Q_m(m) \ln Q_m(m)$ (Section C.2). This upper bound corresponds to the case where there is no overlap between components. Since this term is in turn upper bounded by $\ln M$ we may only improve our bound by a maximum of $\ln 2$ by moving from one component to two.

Lower bound on the log-likelihood

We now write the lower bound on the marginal log-likelihood using $p(\mathbf{w}, \alpha, \beta|D) \propto p(D|\mathbf{w}, \beta)p(\mathbf{w}|\alpha)p(\alpha)p(\beta)$,

$$\begin{aligned} \ln p(D) &\geq \left\langle \ln p(D|\mathbf{w},\beta) \right\rangle_{q_w q_\beta} + \left\langle \ln p(\mathbf{w}|\alpha) \right\rangle_{q_w q_\alpha} \\ &\quad + \left\langle \ln p(\alpha) \right\rangle_{q_\alpha} + \left\langle \ln p(\beta) \right\rangle_{q_\beta} \\ &\quad + \sum_{m=1}^{M} Q_{\mathbf{w}}(m) S(q_w(\mathbf{w}|m)) + \mathcal{I}(m;\mathbf{w}) + S(q_\alpha(\alpha)) + S(q_\beta(\beta)) \end{aligned} \tag{5.31}$$

$$\equiv \mathcal{L}(\mathbf{w}, \alpha, \beta). \tag{5.32}$$

For diagonal covariance Gaussian components of the approximating distributions, $q_w(\mathbf{w}|m)$, all these expectations but one are analytically tractable. The intractable expectation concerns the square of the output function $\left\langle f(\mathbf{x},\mathbf{w})^2 \right\rangle_{q_w}$ for which the approximation we have already mentioned is required. For clarity, we relegate the mathematical derivations of these integrals to Appendix C.

The Eqn 5.32 may be rewritten

$$\begin{aligned}
\mathcal{L}(\mathbf{w},\alpha,\beta) &= \sum_{m=1}^{M} Q_{\mathbf{w}}(m) \int q_w(\mathbf{w}|m) q_\beta(\beta) \left\{ -\frac{\beta}{2} E_D(D,\mathbf{w}) + \frac{N}{2}\ln\beta + \ln p(\beta) \right\} dw\ d\beta \\
&+ \sum_{m=1}^{M} Q_{\mathbf{w}}(m) \int q_w(\mathbf{w}|m) q_\alpha(\alpha) \left\{ -\frac{\alpha}{2} E_w(\mathbf{w}) + \frac{K}{2}\ln\alpha + \ln p(\alpha) \right\} d\mathbf{w}\ d\alpha \\
&+ \sum_{m=1}^{M} Q_{\mathbf{w}}(m) S(q_w(\mathbf{w}|m)) + I(m;\mathbf{w}) + S(q_\alpha(\alpha)) + S(q_\beta(\beta)) + \text{const} \qquad (5.33)
\end{aligned}$$

where K is the total number of weight parameters, in the model. Consider first the dependence of $\mathcal{L}(\mathbf{w},\alpha,\beta)$ on $\mathbf{w}$

$$\begin{aligned}
\mathcal{L}(\mathbf{w}) &= \sum_{m=1}^{M} Q_{\mathbf{w}}(m) \int q_w(\mathbf{w}|m) \left\{ -\bar{\beta} E_D(D,\mathbf{w}) - \frac{\bar{\alpha}}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} \right\} d\mathbf{w} \\
&+ \sum_{m=1}^{M} Q_{\mathbf{w}}(m) S(q_w(\mathbf{w}|m)) + I(m;\mathbf{w}) + \text{const}, \qquad (5.34)
\end{aligned}$$

where we have introduce the expectations $\bar{\alpha} = \int \alpha q_\alpha(\alpha) d\alpha$ and $\bar{\beta} = \int \beta q_\beta(\beta) d\beta$. Evaluating the derivatives of the above expression with respect to the parameters of the variational distributions can be undertaken analytically (Section C.3). Together with the model likelihood bound, $\mathcal{L}(\mathbf{w})$, these derivatives may be used in a non-linear optimisation algorithm.

Optimising the other approximations

So far, the distributions q_α and q_β have not been specified. Yet we need at minimum to estimate the moments $\bar{\alpha}$ and $\bar{\beta}$ in order to determine $q_w(\mathbf{w})$. The free-form optimisation over all possible forms of q_β as described in Section 1.8.2 leads us to the following result

$$q_\beta \propto \exp \left\langle \ln p(D|\mathbf{w},\beta) p(\beta) \right\rangle_{q_w}. \qquad (5.35)$$

Substituting the appropriate form of q_w we find

$$q_\beta(\beta) = \text{gam}(\beta|\bar{a}_\beta, \bar{b}_\beta) \qquad (5.36)$$

where

$$\begin{aligned}
\bar{a}_\beta &= \frac{N}{2} + a_\beta, \qquad (5.37) \\
\bar{b}_\beta &= \frac{\left\langle E_D(D,\mathbf{w}) \right\rangle_{q_w}}{2} + b_\beta. \qquad (5.38)
\end{aligned}$$

A similar treatment for α reveals

$$q_\alpha \propto \exp \left\langle \ln p(D|\mathbf{w},\beta) p(\mathbf{w}|\alpha) \right\rangle_{q_w}. \qquad (5.39)$$

Substituting in the forms of the other distributions we obtain

$$q_\alpha(\alpha) = \text{gam}(\alpha|\bar{a}_\alpha, \bar{b}_\alpha) \tag{5.40}$$

where

$$\bar{a}_\alpha = \frac{K}{2} + a_\alpha, \tag{5.41}$$

$$\bar{b}_\alpha = \frac{\left\langle \mathbf{w}^{\mathrm{T}}\mathbf{w} \right\rangle_{q_w}}{2} + b_\alpha. \tag{5.42}$$

Variational inference in Bayesian neural networks can be undertaken as a two stage optimisation. First, we minimise the KL divergence with respect to the approximating distribution $q_w(\mathbf{w})$, secondly we re-estimate the distributions $q(\alpha)$ and $q(\beta)$. The whole process is re-iterated until convergence of the model likelihood bound.

Other Forms of the Prior

In the next section, to aid comparison of the different Bayesian approaches, all the models are implemented with priors which place weights within four groups: input to hidden layer weights, hidden biases, hidden to output layer weights and output biases. A hyper-parameter, α_g, is then associated with each group, $\mathbf{w}_g$. Our prior then takes the form

$$p(\mathbf{w}|\boldsymbol{\alpha}) = \prod_{g=1}^{G} \left(\frac{\alpha_g}{2\pi}\right)^{\frac{K_g}{2}} \exp\left(-\frac{\boldsymbol{\alpha}_g}{2}\mathbf{w}_g^{\mathrm{T}}\mathbf{w}_g\right) \tag{5.43}$$

where K_g is the number of weights in group g. In its most flexible form such a prior might contain a hyper-parameter for every parameter, alternatively the groupings appropriate to an automatic relevance determination (ARD) prior may be chosen. In an ARD prior (Neal, 1996; MacKay, 1995b) the input-hidden weights are split into a further I groups. Each weight is grouped according to the input node from which it arises. This allows the relevance of each input to be determined through the optimisation of the corresponding hyper-parameters.

Our hyper-prior must now take a slightly different form, a convenient choice of hyper-prior is a product of gamma-distributions:

$$p(\boldsymbol{\alpha}) = \prod_{g=1}^{G} \text{gam}(\alpha_g|a_g, b_g). \tag{5.44}$$

Substituting in the grouping prior from Eqn 5.43 and the hyper-prior from Eqn 5.44 into Eqn 5.40 we obtain

$$q_{\alpha_g} = \text{gam}(\alpha_g|\bar{a}_g, \bar{b}_g), \tag{5.45}$$

where

$$\bar{a}_g = \frac{K_g}{2} + a_g \tag{5.46}$$

$$\bar{b}_g = \frac{1}{2}\left\langle \mathbf{w}_g^{\mathrm{T}}\mathbf{w}_g \right\rangle + b_g. \tag{5.47}$$

The recovery of the gamma distribution from the free-form optimisation is due to the selection of a prior which is a log-linear function of the hyper-parameter and a gamma distributed hyper-prior. These distributions are therefore conjugate.

5.5 Results

In this section, we present our approach on both synthetic and real-world data. We also compare the performance of the approximation with hybrid Monte Carlo sampling and the evidence procedure. We take the components of the mixture distribution for the variational approach to be diagonal covariance Gaussians as well as the variational smoothing distributions.

5.5.1 Toy Problem

To try and get a handle on the performance of our approximation, we first consider its application to a simple toy regression problem. We consider samples from the function

$$h(x) = 0.5 + 0.4\sin(2\pi x) \tag{5.48}$$

with additive Gaussian noise of standard deviation 0.05. We generated thirty data points. The x position of the data points was determined by sampling fifteen points from a Gaussian with mean 0.125 and a further fifteen points from a Gaussian with mean 0.625. Both Gaussians where chosen to have a standard deviation of 0.25.

We used networks containing the activation functions from Eqn 5.3. The networks contained five hidden nodes. For the hyper-parameters $\boldsymbol{\alpha}$ we considered the more complex form where the weights are placed in four groups: input-hidden weights, hidden biases, hidden-output weights and output biases. Each group was then associated with its own hyper-parameter α_g.

We first trained a network using the evidence procedure. Each hyper-parameter was initialised as $\alpha_g = 0.1$ and the parameter governing the inverse noise variance, β was initialised as 50. The weights were then trained for a maximum of 500 iterations in a scaled conjugate gradient optimiser followed by an update of the $\boldsymbol{\alpha}$ and β. This process was repeated ten times to try and obtain a good approximation to the posterior distribution of the weights as well as estimating the values of $\boldsymbol{\alpha}$ and β.

We then initialised a diagonal covariance variational approximation with a mean equal to the mean of the Laplace approximation and variances of 1×10^{-6}. We trained the resulting model for 500 iterations of a scaled conjugate gradient optimiser. We used the values of $\boldsymbol{\alpha}$ and β learnt in the evidence procedure.

For the mixture approximation we used five components. Each component's mean was initialised with that learnt from the diagonal variance approximation, with Gaussian noise added. The standard deviation of the added noise was taken to 1% of the absolute value of the mean. If this was not done, the components of the mixture model still separated but took longer to do so. We trained the resulting network in the following way. We first maximised the objective function with respect to the

the parameters of the smoothing distributions and the parameters $\{\lambda_m\}$ for fifty iterations of scaled conjugate gradient. The next step was to perform fifty iterations of maximisation with respect to all the variational parameters associated with the mixture model except the mixture coefficients. The mixture coefficients were then updated using a fixed point equation. This procedure was repeated ten times.

The final method we applied was hybrid Monte Carlo sampling. We initialised the sampler at the mode found in the Laplace approximation and used the values for the hyper-parameters found also by the evidence procedure. We then obtained samples from the posterior distribution using hybrid Monte Carlo sampling without persistence. We obtained 300 samples using 100 leap-frog steps. No samples were rejected from the start of the chain because we were interested in samples near the mode found by the Laplace approximation. The leap-frog steps utilised a step size of 2×10^{-3}.

The results of the experiments are shown in Figure 5.3. Figure 5.3(e) shows the samples from the the true posterior using hybrid Monte Carlo techniques. This should be contrasted with the samples from the diagonal covariance approximation (Figure 5.3(d)), those from the Laplace approximation, (Figure 5.3(c)) and those from the mixture distribution (Figure 5.3(b)). The samples from the Laplace approximation in Figure 5.3(c) appear to be completely unrepresentative of the true posterior. We investigated the Laplace approximation further by plotting the logarithm of the approximation along the eigenvectors of its covariance matrix. We compared these values to the true logarithm of the posterior. The results are shown in Figure 5.4. We note that for larger eigenvalues the approximation is very poor. To better represent the mass of the distribution the eigenvalues of the approximation should be smaller. This explains the poor samples we observe in Figure 5.3(c). Also shown in that figure is an output from a network which has the mean of the Laplace approximation as its weights (long dashed line) and the expected value of the output under the Laplace approximation (this is shown as a solid line which is off-scale for much of the figure). This value may be calculated for networks with cumulative Gaussians as the activation function:

$$\langle f(\mathbf{w},\mathbf{x})\rangle_{q(\mathbf{w}|D)} = \sum_{h=1}^{H} \sqrt{\frac{2}{\pi}} \frac{\boldsymbol{\Sigma}_{v_h \bar{\mathbf{u}}_h}\mathbf{x}}{\sqrt{1+\mathbf{x}^{\mathrm{T}}\boldsymbol{\Sigma}_{\mathbf{u}_h\bar{\mathbf{u}}_h}\mathbf{x}}} \exp\left(-\frac{1}{2}\frac{\mathbf{x}^{\mathrm{T}}\bar{\mathbf{u}}_h}{1+\mathbf{x}^{\mathrm{T}}\boldsymbol{\Sigma}_{\mathbf{u}_h\mathbf{u}_h}\mathbf{x}}\right) + \bar{v}_h g\left(\frac{\mathbf{x}^{\mathrm{T}}\bar{\mathbf{u}}_h}{\sqrt{1+\mathbf{x}^{\mathrm{T}}\boldsymbol{\Sigma}_{\mathbf{u}_h\mathbf{u}_h}\mathbf{x}}}\right) \tag{5.49}$$

where we have used the following notation: $\boldsymbol{\Sigma}_{\mathbf{xy}}$ indicates the part of the covariance matrix associated with the interactions between weights $\mathbf{x}$ and $\mathbf{y}$ and the notation; $\mathbf{u}_h$ denotes weights which fan into node h. Barber and Bishop (1998) show how the required integrals to obtain this expectation may be performed. This equation may be contrasted with Eqn C.32 which represents the special case where the covariance of the Gaussian approximation is constrained to be diagonal.

When the Laplace approximation is implemented in practice it is rarely the case that we are able to compute Eqn 5.49 as we are here. This may be due to an alternative choice of activation function, or the use of networks with more hidden layers. In these cases, the required expectation is approximated by the output of the network at the mean of the Laplace approximation. As Eqn 5.49 shows though, this may be a poor approximation. This is particularly true if $\mathbf{x}^{\mathrm{T}}\boldsymbol{\Sigma}_{\mathbf{u}_h\mathbf{u}_h}\mathbf{x}$ is large, as

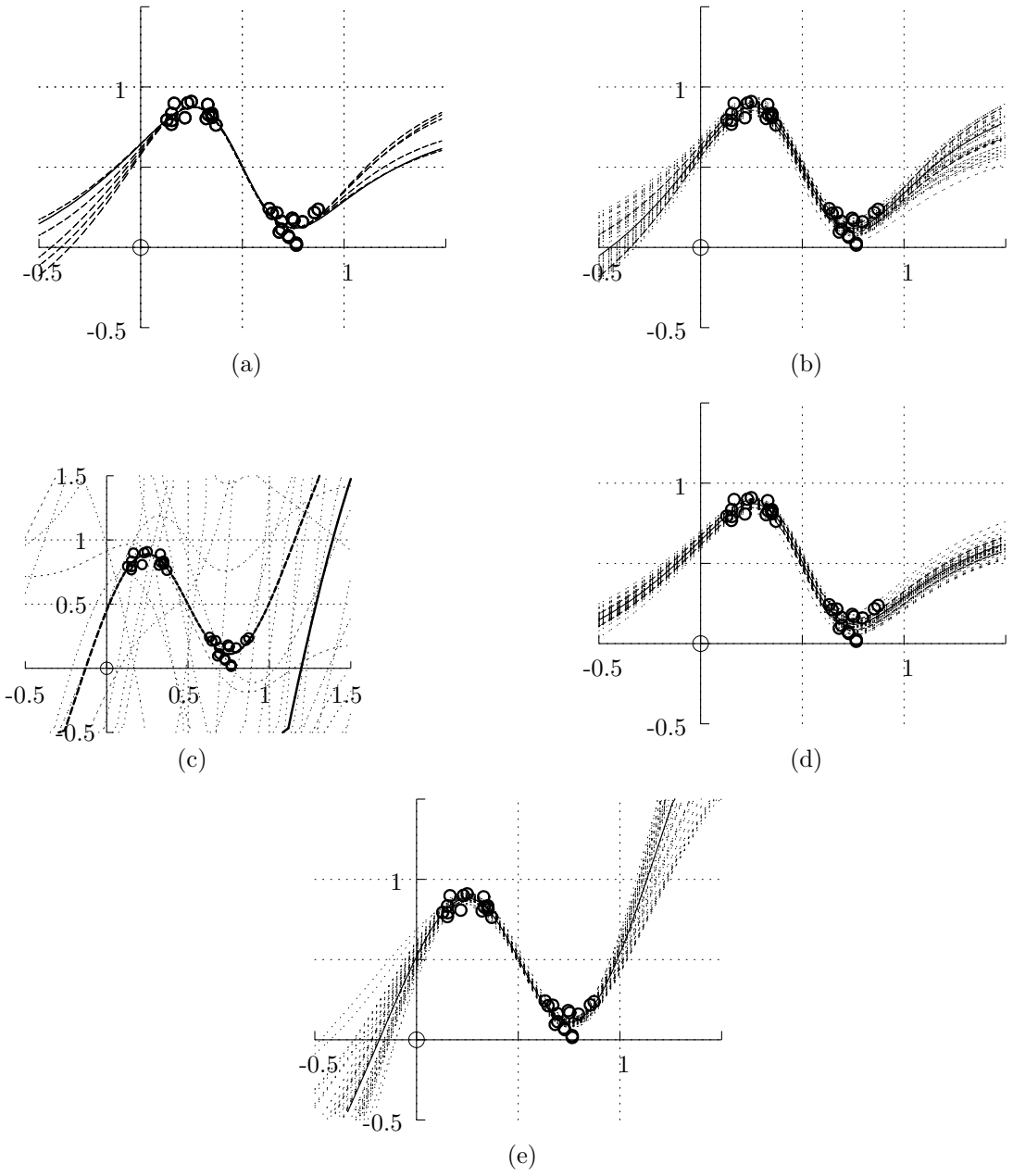


Figure 5.3: The various approximations to the Bayesian approach interpolating data generated from a noisy sine wave. All figures show the examined data points as circles. (a) shows the expected output under each component of the variational mixture approximation (dashed lines) along with the expected output from the diagonal Gaussian approximation (solid line) from which they were initialised. (b) shows samples from the mixture approximation (faint dotted lines) along with the expected output (solid line) and error-bars (dashed lines) signifying one standard deviation. (c) shows samples from the Laplace approximation (faint dotted lines) along with the expected network output under the Gaussian approximation (solid line - most of which is off scale) and the output from a network using the mean value of $\mathbf{w}$ from the Laplace approximation (long dashed lines). (d) shows samples from the diagonal covariance Gaussian approximation (faint dotted lines) along with the expected output (solid line) and error bars at one standard deviation (dashed lines). (e) shows the samples obtained using the hybrid Monte Carlo approach (faint dotted line). Also shown is the mean of the samples (solid line).

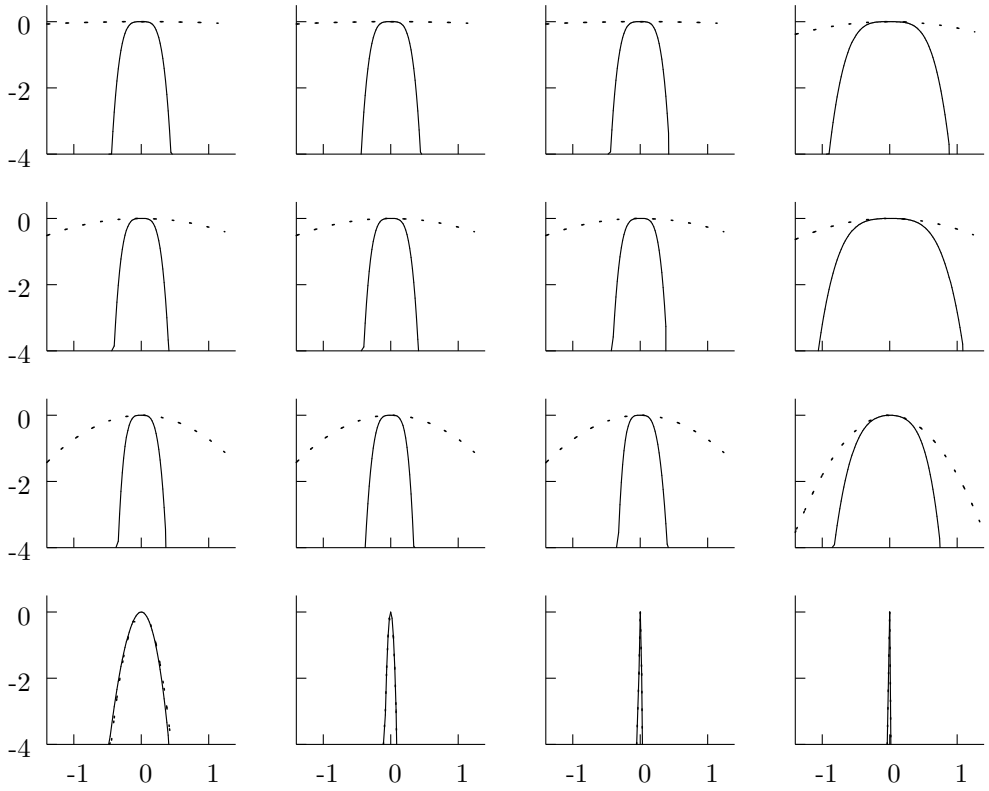


Figure 5.4: The logarithm of the Laplace approximation (dotted line) plotted alongside the logarithm of the true posterior for a regression neural network (solid line). The plots show the logarithms plotted along the eigenvalues of the covariance matrix for the Laplace approximation. The plots are displayed in the order of highest eigenvalue first, lowest last. In the last four plots the approximation overlays the true value. The curves are shown with an offset so that they are visible.

it appears to be for our toy example. However the output of the network at the mean of the Laplace approximation is likely to represent the data well, as it was minimised with this objective in mind. So we see that although it is apparent that the Laplace approximation is not a good representation of the posterior, because we do not use the approximation to predict the expected network output the approach is self-redeeming. We suspect a similar logic applies to error bars obtained through this approach. It is clear that error-bars computed from samples such as those in Figure 5.3(c) are not going to be good reflections of our data-set. However when the approach is applied there is also a further approximation required to compute the expectation of the output squared. It involves an assumption of local linearity in the network output. We suggest that this approximation also causes the approach to be self-redeeming.

Figure 5.3(d) shows samples and expectations from the diagonal covariance approximation. It does not express the uncertainty present in the regression in areas where the training data density was low. We suggest this is due to the fact that this approximation is unable to express correlations between the weights.

Figure 5.3(b) shows samples from the variational mixture distribution. Finally Figure 5.3(a) shows

the expected output of the network under the diagonal covariance distribution as well as the expected output the network under each component of the mixture distribution. Note how the components expectations have spread out from where they were initialised in regions of low data density. This would not have occurred if we had used the minimum overlap assumption. The separation of the components is also apparent in Figure 5.5 which plots the logarithm of the mixture distribution alongside the logarithm of the true posterior. At then end of training the lower bound on the mutual information

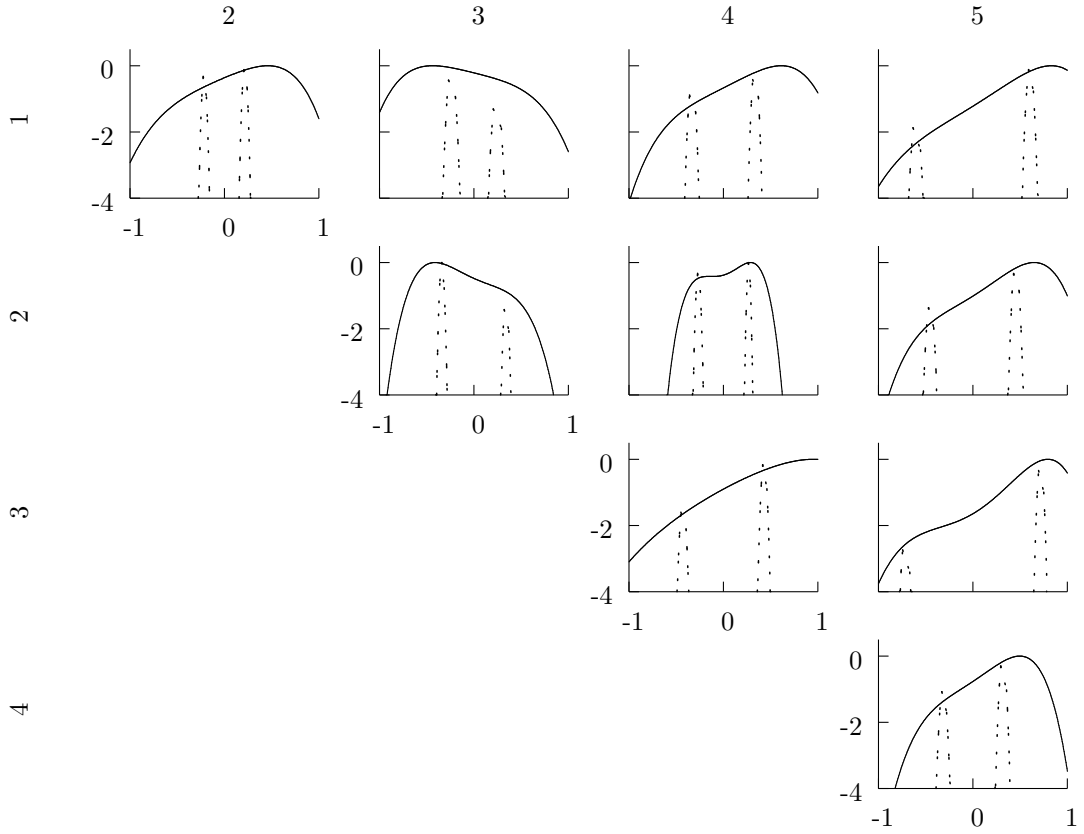


Figure 5.5: The logarithm of the variational mixture approximation (dotted line) plotted alongside the logarithm of the true posterior for a regression neural network (solid line). The lines are plotted along and embedded direction in the space of $\mathbf{w}$, namely between the means of each possible combination of two of the approximating components. The numbers across the top and along the left of the plots specify the components across which the logarithms are plotted, i.e. the plot in the top left corner shows the line which passes through the means of component 1 and component 2.

was calculated as 1.5229 this compares with an upper bound, calculated using the minimum overlap assumption of 1.5614. In Figure 5.5 the approximations have spread out within the mode. There appears to be no overlap between any components so this suggests an error in our bound on the mutual information of about 2.46%. The fact that the components are all fairly narrow indicates that the posterior is narrow across other embedded directions, preventing each component from better filling the posterior. This in turn suggests strong correlations in the posterior. We would require many more components to represent the posterior accurately. This effect can be seen clearly if we try to minimise the KL divergence between a highly correlated two dimensional Gaussian and a mixture of two

diagonal covariance Gaussians. Figure 5.6 shows the minimum KL divergence solution. The form of

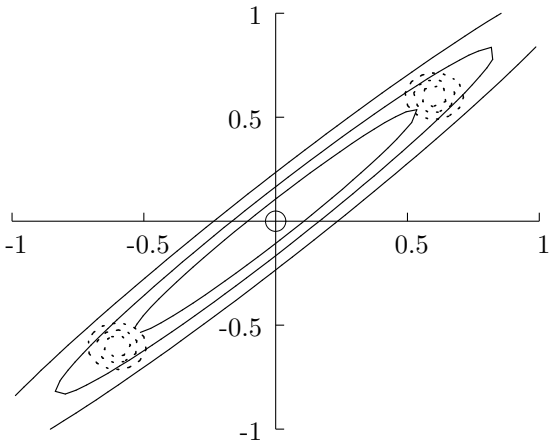


Figure 5.6: This diagrams shows the minimum KL divergence solution between a Gaussian and a mixture of two Gaussians. The first Gaussian has a covariance matrix with a very high correlation, its eigenvalues are 1 and 0.01 and the eigenvectors are at forty-five degrees to the axis. Three contours from this Gaussian are shown with solid lines. The Gaussians in the mixture model are constrained to have diagonal covariance. Three contours from the mixture model are shown with dotted lines.

the KL divergence we are using involves the expectation under the mixture distribution. This results in two widely separated components which sit within the correlated Gaussian. Note in particular the narrow widths associated with the components of the mixture. This is the same effect as that visible in Figure 5.5.

5.5.2 Real Data

We now choose to investigate two real world problems to evaluate our approach. The first is a benchmark time series and the second is a ten input non-linear regression problem. In these experiments where hyper-priors on $\boldsymbol{\alpha}$ were used, we set $a_\alpha = 3 \times 10^{-4}$, $b_\alpha = 1 \times 10^{-3}$. and took the hyper-prior to be a product of gamma distributions. Where a prior on the inverse noise variance β was implemented we set $a_\beta = 0.02$, $b_\beta = 1 \times 10^{-4}$. In approaches where use was made of type II maximum likelihood for determining $\boldsymbol{\alpha}$ and β. Each element of $\boldsymbol{\alpha}$ was initialised at 0.3 and β was initialised as 200.

Sunspot Data

Our first real world data-set involves the annual average[2] of sunspots from 1700 to 1920. The time series shows a strong seasonality with a time varying amplitude and has served as a benchmark in the statistical literature (Tong, 1995). The average time between maxima is 11 years and to date no principled theory exits for describing this behaviour, although it is known that sunspots are related to other activities of the sun.

[2]The data are daily, monthly and annually reported by the Royal Observatory of Belgium and can be found at `http://www.oma.be/KSB-ORB/SIDC/sidc_txt.html`.

We propose to model this time series within a Bayesian framework as the number of training points is relatively small. We compare results from three different Bayesian approaches and that of a standard neural network trained by 'early stopping'. The goal of the simulations is to compare Bayesian techniques on a time series data-set. To facilitate comparison, the number of hidden nodes is taken to be fixed at eight and the input window is chosen arbitrarily to be 12, i. e. we modelled $x_n = f(x_{n-1}, \ldots x_{n-12})$. The yearly sunspot data from 1700 through 1920 (221 data points) was used for training and the data from 1921 to 1979 (59 data points) for evaluating the generalisation performance. These choices correspond to those of Weigend *et al.* (1990) who studied the performance of early stopping techniques on this time series. The data-set was normalised to have a zero mean and unit variance.

For the hybrid Monte Carlo sampling, 200 samples of the posterior distribution were taken by sampling with persistence using a decay rate of 0.95. Predictions were then made by averaging the network samples from the posterior distribution, ignoring the first 100 samples.

For the Laplace approximation, we trained ten models with different random initialisations. The error function was minimised by a quasi-newton optimisation[3] routine. The optimisation was terminated when both the weights and the error was changing by less than 1×10^{-7} or when a 500 iterations were exceeded. Type II maximum likelihood was then used to update $\boldsymbol{\alpha}$ and maximum likelihood for β. This cycle was completed ten times. We then selected the best network according to the approximation to the type II likelihood of the training data.

For the variational inference we took the following approach. We trained a one component model with five different initialisations. The parameters of the distribution, q_w, were again optimised using the quasi-newton method. However we used slacker termination criteria. The optimisation was terminated when changes both the lower bound on the log-likelihood and the parameters of q_w were less than 1×10^{-5} or when 200 iterations were exceeded. The parameters of the approximation to the posterior of β were updated according to Eqn 5.36 and the parameters of the approximation to the hyper-posterior of $\boldsymbol{\alpha}$ were updated as in Eqn 5.46 and Eqn 5.46. We present results from the model which exhibited the largest lower bound on the model-likelihood. We also used this model to initialise the variational mixture. We studied distributions containing 5 components. To ensure a tight bound on the mutual information was maintained we alternated a general parameter update of $q_w(\mathbf{w})$ with updates of the only parameters which are specific to the bound on the mutual information. Fifty iterations of the quasi-newton optimiser were applied to each. The mixing coefficients, $Q_m(m)$, were updated via a fixed point equation after these two steps. The parameters of $q_\alpha(\boldsymbol{\alpha})$ and $q_\beta(\beta)$ where then updated and the entire process repeated ten times.

Table 5.1 reports the normalised mean squared error (NMSE) obtained by each technique on the test set. The first result was reported by Weigend *et al.* (1990).

While the data-set is too small to draw absolute conclusions, the results seem to indicate the utility of the Bayesian approach in general and show the competitiveness of the methods based on

[3]We used the quasi-newton routine available in the NETLAB package (Nabney, 1998)

Method	Test error
Early stopping	0.262
Variational (diagonal)	0.178
Variational (mixture)	0.172
Evidence	0.171
Hybrid Monte Carlo	0.131

Table 5.1: Normalised mean squared error on the test set for the five techniques. For the evidence and ensemble learning techniques, we selected the best network according to the type II likelihood and the model likelihood respectively. The variational committee is composed of five components.

variational techniques. The mixture of diagonal covariance Gaussians exhibits an slight improvement in performance over that of the simple diagonal covariance Gaussian.

Tecator Data

We also assessed the performance of our approach on a regression problem, the Tecator data-set (Thodberg, 1996). The data are recorded on a Tecator Infratec Food and Feed Analyser working in the wavelength range 850 - 1050 nm by the Near Infrared Transmission (NIT) principle. Each sample contains finely chopped pure meat with different moisture, fat and protein contents. The task is to predict the fat content of a meat sample on the basis of its near infrared absorbance spectrum. For each meat sample, the data consists of a 100 channel spectrum. the spectra are pre-processed using principal component analysis and the first ten principal components are used as inputs to a neural network. Thodberg demonstrated the benefits of the evidence approach compared to the early stopping technique.

The training data-set contains 172 samples and the test set is composed of 43 samples[4]. In his paper, Thodberg reported the square root of the mean square error and in order to have comparable results we used the same error measure. Note however that in this previous work, the neural networks had a direct connection for the inputs to the output. Our models do not contain such connections. Thodberg determined the maximum evidence was associated with models containing 3 hidden units. We however choose to investigate networks containing 8 hidden units as we expect the Bayesian approach to provide the required capacity control through the hyper-parameters $\boldsymbol{\alpha}$.

Table 5.2 reports the test error obtained by each technique. The first four results were reported by Thodberg (1996). The Bayesian approaches outperform the 'early stopping' technique. The variational committee once again performs well.

[4]The data-set is available from `http://temper.stat.cmu.edu/datasets/Tecator`

Method	Test error
Linear regression	2.78
Quadratic regression	0.80
Early stopping	0.65
Evidence	0.55
Variational (diagonal)	0.526
Variational (mixture)	0.519
Hybrid Monte Carlo	0.481

Table 5.2: Square root of the mean squared error on the test set for the five techniques. For the evidence and ensemble learning techniques, again, we selected the best network according to the likelihood on the training set. The variational committee contained five components.

5.6 Discussion

In this chapter we have reviewed the Bayesian framework for regression neural networks with a particular focus on variational approaches. We introduced a new variational distribution based on mixtures of diagonal covariance Gaussians. This approximation enables the modelling of non-Gaussian posterior distributions.

The results on the toy problem showed the importance of accounting for correlations in the posterior and demonstrated that the mixture distribution would not be able to fulfill this requirement without resorting to many components. The results on real world data-sets showed that the approach was competitive. However any improvement in performance afforded by the use of mixture distributions is accompanied by a computational penalty. The time required for learning will increase at least linearly in the number of components.

We mentioned that other covariance structures were tractable using the approximation we applied. In fact we may consider covariances which account for all the correlations within the hidden-output layer and the correlations between the layers. However we are unable to account for correlations between all the weights in the input-hidden layer without resorting to the techniques described in Barber and Bishop (1998). We may take account of correlations between weights that fan into the same node, but not across weights that fan into different nodes. The resulting covariance matrix would be of the following structure

$$\mathbf{\Sigma} = \begin{bmatrix} \mathbf{\Sigma}_{\mathbf{u}_1\mathbf{u}_1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{\Sigma}_{\mathbf{u}_1\mathbf{v}} \\ \mathbf{0} & \ddots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{\Sigma}_{\mathbf{u}_h\mathbf{u}_h} & \mathbf{0} & \mathbf{0} & \mathbf{\Sigma}_{\mathbf{u}_h\mathbf{v}} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{\Sigma}_{\mathbf{u}_H\mathbf{u}_H} & \mathbf{\Sigma}_{\mathbf{u}_H\mathbf{v}} \\ \mathbf{\Sigma}_{\mathbf{v}\mathbf{u}_1} & \cdots & \mathbf{\Sigma}_{\mathbf{v}\mathbf{u}_h} & \cdots & \mathbf{\Sigma}_{\mathbf{v}\mathbf{u}_H} & \mathbf{\Sigma}_{\mathbf{v}\mathbf{v}} \end{bmatrix}. \tag{5.50}$$

Perhaps accounting for correlations of this sort may go some way to alleviating the difficulties associ-

ated with diagonal covariance components.

This framework is also applicable to classification networks, although this requires the use of further variational bounds.

Acknowledgements

The journal paper on which this chapter is based was completed in collaboration with Mehdi Azzouzi. The idea to implement mixture distributions in this manner was my own, as was the idea of utilising Gaussians for the smoothing distributions. The actual derivation of the bound on mutual information (Appendix C) was performed by Mehdi Azzouzi.

The experiments utilised Radford Neal's implementation of hybrid Monte Carlo sampling (Neal, 1999) and Ian Nabney's NETLAB package (Nabney, 1998).

Chapter 6

Discussion

6.1 Review

In the introduction we described a general approach to the handling of uncertainty. The approach was based on probability theory. Probability theory provides a framework for the handling of uncertainty in computational devices. Unfortunately many of the required marginalisations, whether they are discrete sums or integrals, turn out to be intractable. We mentioned two separate well used approaches to handling the intractabilities: sampling methods and the Laplace approximation. We then introduced a third approach, variational methods.

The framework we suggest for implementation of a variational approach involves three steps. First split the model variables into exclusive sub-sets. Ideally each of the sub-sets should form a tractable sub-structure in which exact inference may be performed. In other words it must be possible to both compute the joint distribution of the variables and to marginalise across any or all of the variables in the sub-sets. The sub-sets, at their smallest, may only contain one variable, in which case the approach is known as mean field theory. At the other extreme a sub-set may contain all the variables in the model. In this case exact inference is tractable in the model. Once the sub-sets have been selected a super-graph may be drawn. This super-graph will show the inter-dependencies of the variational approach through the Markov blanket of the sub-sets.

6.1.1 Conjugacy

The best Q-distributions may now be determined via a free-form optimisation if the variational distributions representing each node are conjugate. In other words, in the graph shown in Figure 6.1, if the distribution $p(\mathcal{H}_1|\text{pa}(\mathcal{H}_1))$ has a particular functional dependence on $\mathcal{H}_1$ then to be conjugate, the distributions governing its children, $\mathcal{H}_2$ and $\mathcal{H}_3$ need to share the form of that functional dependence. If that dependence was quadratic, for example, and if the variables were continuous $p(\mathcal{H}_1|\mathcal{H}_{i\neq 1})$ would be a Gaussian distribution and $p(\mathcal{H}_2|\text{pa}(\mathcal{H}_2))$ and $p(\mathcal{H}_3|\text{pa}(\mathcal{H}_3))$ would have either quadratic or linear dependence on $\mathcal{H}_1$. If this condition of conjugacy is fulfilled then the resulting Q-distribution for any

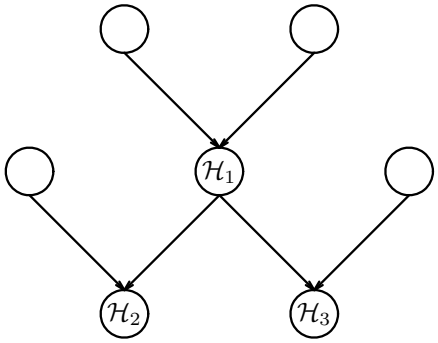


Figure 6.1: Conjugacy in probabilistic models

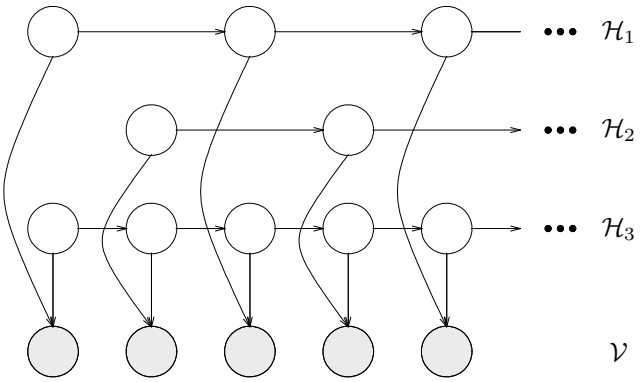


Figure 6.2: A frame hopping hidden Markov model. The model is similar to the factorial hidden Markov model (Ghahramani and Jordan, 1997), but matrices of transition probabilities may skip time frames.

particular sub-set will have the same functional form as the interactions within that sub-set.

6.2 Overview of Method Implementation

Conjugacy is guaranteed for a large class of models which are of interest. That is chain graphs of discrete variables. As we mentioned in the introduction a chain graph is one which contains both directed and undirected connections. This means that we now have a powerful approach to handling approximate inference in such models. We may now step through two examples. These two examples demonstrate the core elements of the approach.

For our first example we consider a frame hopping hidden Markov model (FHHMM). The FHHMM has multiple hidden chains, some of which have transition probability matrices which skip across time frames. This might be useful in the modelling of, for example, music. Music contains structure which cannot be well represented by normal hidden Markov models but might be well handled by the FHHMM. The model in Figure 6.2 represents a FHHMM with second order dependencies. In some respects the FHHMM is similar to the factorial hidden Markov model (FHMM) of Ghahramani and Jordan, 1997. The factorial hidden Markov model however does not consider skipping of time frames.

In our implementation of approximate inference for the FHHMM we could be inspired by the sub-set selection that Ghahramani and Jordan utilised. We may associate a sub-set with each of the hidden

chains leading to the super-graph such as the one in Figure 6.3(a). If we assume our hidden Markov

Figure 6.3: The super-graph, (a), of the frame hopping hidden Markov model and its mean field approximation, (b).

model is one based on probability tables, conjugacy between these variable sub-sets is guaranteed. We finally should implement a factorisation assumption, this is equivalent to removing the interactions from the super-graph, the mean field approximation to the super-graph, Figure 6.3(b).

For our second example consider the densely connected graph in Figure 6.4. With the given sub-set

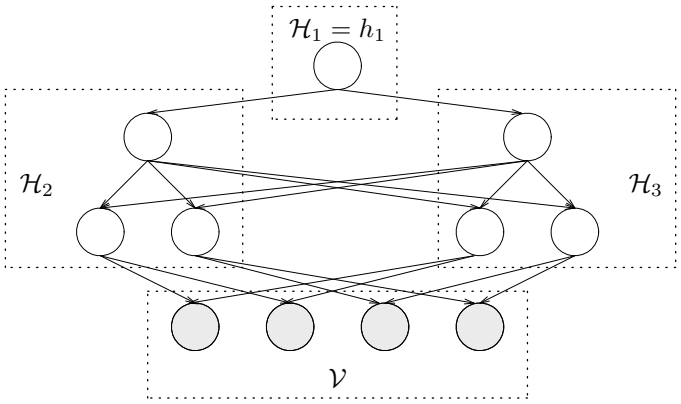


Figure 6.4: A densely connected graph with possible sub-set selections.

selections our super-graph may be drawn as in Figure 6.5. Once again if conjugacy between these variable subsets is guaranteed we may apply the machinery of mean field inference to the super-graph in a straightforward manner.

6.3 Implementation in this Thesis

The above examples demonstrate the simplicity with which our framework may be sometimes applied. In this thesis we attempted to apply this approach to a variety of models, as well as extending it in some instances.

In Chapter 2 we extended the approach by considering a mixture of mean field approximations

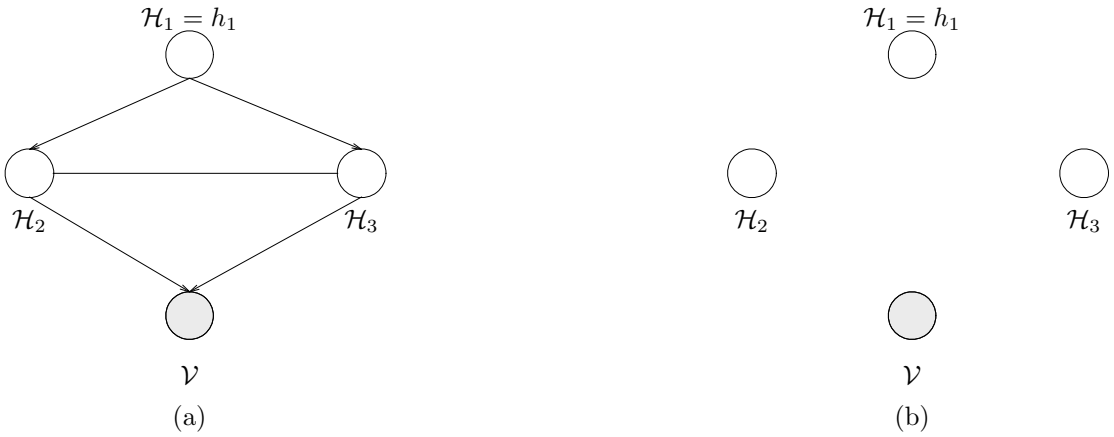


Figure 6.5: The super-graph, (a), of the densely connected graph and its mean field approximation, (b).

for the super-graph. This approach appeared promising in our implementation. However, we only considered the framework with sub-sets containing only one variable, to implement this approach with larger sub-sets, an alternative method of handling the smoothing distributions, $R(\mathcal{H}_i|m)$, efficiently would have to be found. We had chosen to specify this distribution by a table, but clearly if $\mathcal{H}_i$ contained many variables this table could be prohibitively large. To implement this approach in general would require the use of potential functions in a similar manner to that described in Section 2.2.

The first chapter covered two further issues. Namely we were unable to find tractable sub-structures of our original model. Even by resorting to mean field theory we found that we needed to perform a sum which was exponentially dependent on the number of parents a node had. As a result we were forced to find an additional lower bound expectations of the joint probability distribution $P(\mathcal{H}, \mathcal{V})$ as part of the approach. Finally we implemented another alternative to mean field theory based on Markov chains. The three approaches to handling the super-graph (which in the context of Chapter 2 is identical to the original graph) were summarised in Figure 2.3. In conclusion we decided that whilst richer approximating distributions indubitably lead to tighter lower bounds, the extra computational complexity may discourage their use.

The ideas in Chapter 3 were stimulated by observing that mean field inference in sigmoid belief networks is exact in the limit as the connectivity parameters go to infinity. The inference algorithm we derived is of historical interest due to the McCulloch and Pitts model of the neuron and Rosenblatt's perceptron. Indeed it was thought that such an algorithm could not be found (Minsky and Papert, 1969). In modern computer science, the approach allows us to perform inference in rule based systems. As such it may belong to the field of inductive logic programming (Muggleton and de Raedt, 1994) which aims to develop rule based systems from data.

In Chapter 4 we studied the implementation of the framework in the Boltzmann machine, an undirected graphical model. We discussed in the introduction how in such models we must resort to making two variational approximations. One to the conditional distribution $P(\mathcal{H}|\mathcal{V})$ and a further approximation to $P(\mathcal{H}, \mathcal{V})$. This results in an approximation to the likelihood which we utilised for

learning. We also showed how judicious initialisation of the variational approximations enabled mean field theory to learn representations which were unrealisable in previous works (Galland, 1993). We demonstrated the utility of approximating the joint distribution with an distribution based upon mixtures, taking advantage of the techniques we discussed in Chapter 2. For approximations to the joint distribution in an unsupervised learning task the computational penalty associated with updating the more complex variational distribution is diminished. This is because if we utilise a batch learning[1] framework we only need update the variational distribution once for each presentation of the data-set.

We mentioned, in the introduction, how Bayesian inference may be seen as a latent variable model. Thus it is amenable to application of the variational approaches we described above. We demonstrated their implementation with a Bayesian neural network. We encountered problems with inference of the network weights within this framework. These problems were caused by selection of the sub-set of parameters $\mathbf{w}$ which was not a tractable sub-structure of the model. The problem is not alleviated by considering the elements of $\mathbf{w}$ as individual sub-sets. The normalisation constants of the resulting probability distributions are not analytically tractable. We therefore considered the imposition of a functional form for the variational approximation. We implemented a mixture of diagonal covariance Gaussian distributions using a continuous version of the bounds discussed in Chapter 2. Overall the approach proved promising. However, the computational penalty is again roughly linear in the number components in the approximation. Thus five components take at least five times longer than a simple diagonal covariance approximation. The toy problem we studied showed strong evidence of heavy correlation between the parameters. We mentioned that as a result we would have to utilise many components to accurately approximate the posterior distribution. Had we been using full covariance Gaussians for our mixing distributions these problems would not have occurred. Since there remain unresolved computational issues with such an approach, we didn't consider it to be currently practical.

6.4 Summary

To summarise, in this thesis, we have presented a framework for inference in intractable probabilistic models. Our approach was based on viewing such models graphically. Whilst many of the ideas we presented have been implemented in the past, we have attempted to unify them under a single approach. We have also attempted to show the utility of implementing more complex variational distributions than simple mean field theory. The more complex approaches did invariably produce better approximations and tighter bounds. There were however computational penalties associated with these approaches. If they are to be utilised for a particular model, consideration would have to be given to the likely improvement in approximations versus this computational penalty.

[1] By batch learning we mean that the parameters are updated once after all the data has been presented. The alternative approach, on-line learning, involves updating the parameters after each pattern from the data-set has been presented.

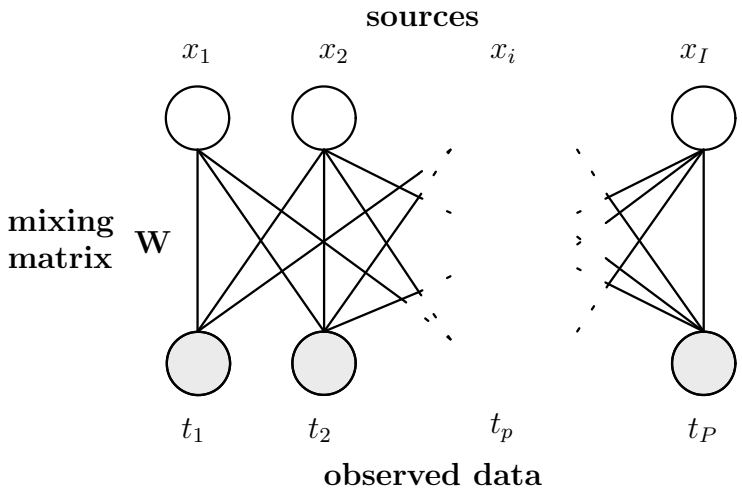


Figure 6.6: Independent component analysis model.

6.5 Further Work

6.5.1 Bayesian Independent Component Analysis

In Chapter 5 we reviewed the Bayesian approach to learning and showed how the parameters of a regression neural network could be inferred through this framework. One area of further research is a different class of models, ones which learn in an *unsupervised* fashion or in other words, models which may be applied to unlabelled data. Here we present preliminary results on the independent component analysis (ICA) model and we show how Bayesian inference may be applied in this model for the determination of structure.

6.5.2 Independent Component Analysis

The objective of independent component analysis is to find a representation for a data set which is based on probabilistically independent components. One way to achieve such a representation is to fit the data to a latent variable model where the latent variables are constrained to be independent. Such a representation is shown in figure 6.6.

We consider a model in which there are I latent dimensions, P observed dimensions and our data set contains N samples. In the ICA literature the latent dimensions are often refered to as 'sources'. Since we are seeking representations for our data in which the latent variables, $\mathbf{x}$, are generated independently, we take

$$p(\mathbf{x}_n) = \prod_{i=1}^{I} p(x_{in}) \tag{6.1}$$

for any particular data point n. The probability distribution for each instantiation of the observed variables, $\mathbf{t}_n$, may then be taken to be

$$p(\mathbf{t}_n|\mathbf{x}_n, \mathbf{W}, \beta, \boldsymbol{\mu}) = \sqrt{\frac{\beta}{2\pi}} \exp\left(\frac{\beta}{2}\|\mathbf{t}_n - \mathbf{W}\mathbf{x}_n - \boldsymbol{\mu}\|^2\right), \tag{6.2}$$

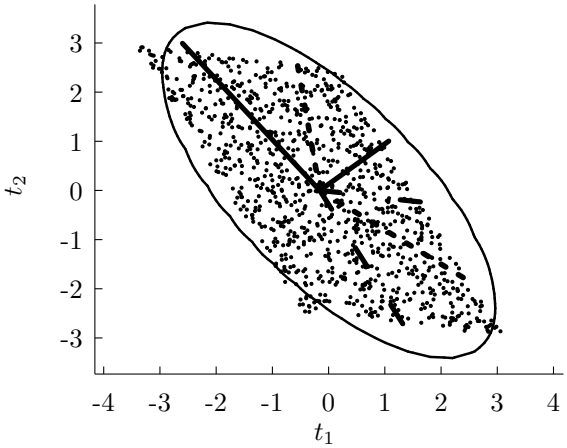


Figure 6.7: A principal component analysis (PCA) solution to non-Gaussian data. Two independent sources were uniformly sampled over (-1, 1) and a 2×2 mixing matrix was applied to get the observed data $\{t_1, t_2\}$. The ellipse is a contour from the Gaussian which underlies the PCA solution. The solid lines show the orthogonal principal component analysis solution. The dotted lines show another valid PCA solution. The true independent components are depicted by dashed lines.

where $\mathbf{W}$ is a $P \times I$ matrix of parameters, β represents an inverse noise variance and $\boldsymbol{\mu}$ a vector of means.

The Source Distribution

The choice of the latent distribution is important. Naively we may choose a convenient representation to be Gaussian,

$$p(x_i) = \mathcal{N}(m_i, \sigma_i^2), \tag{6.3}$$

with mean m_i and variance σ_i^2, unfortunately such a selection will lead to a model whose maximum likelihood solution is invariant through an 'elliptical rotation' of the parameters $\mathbf{W}$. The resulting latent variable model corresponds to a probabilistic formulation of principal component analysis (PCA) (Tipping and Bishop, 1999). This invariance is a result of the rotational invariance of the Gaussian distribution[2]. The model will determine an embedded I dimensional space known as the 'principal sub-space' within which the elliptical rotation may take place. The ICA solution we require will also be within this space and could be a valid solution for the probabilistic PCA model. In principal component analysis when the principal axes are required, the selected solution is often taken to be one where the principal axes are orthogonal(see Figure 6.7). These corresponds to the I eigenvectors of the matrix $\mathbf{W}^{\mathrm{T}}\mathbf{W}$. Fortunately, it can be shown that the Gaussian distribution is the only distribution for which this is the case.

Another consequence of this property of the Gaussian distribution is that if the sources we are trying to separate are Gaussian, we will be unable to determine the alignment of the separating axes. Non-Gaussian source distributions may be split into two classes, those with positive kurtosis or

[2]This invariance is obvious for the case of all σ_i^2 being equal and $m_i = 0$ for all i. However it can also be shown that an attempt to break the rotational symmetry by removal of these conditions will not help.

'heavy tails' and those with negative kurtosis or 'light tails'. The former are known as super-Gaussian distributions and the latter as sub-Gaussian. If our source distribution belongs in one of these two classes we may make progress. For our ICA model, we choose a flexible source distribution which may take a super or sub-Gaussian form. The resulting model will then be applicable for both eventualities. An appropriate choice is a factorised distribution in which each factor is a mixture of M Gaussians,

$$p(\mathbf{x}_n) = \prod_{i=1}^{I} \left[\sum_{m=1}^{M} \pi_m \mathcal{N}(x_{ni}|m_m, \sigma_m^2) \right], \tag{6.4}$$

where $\{\pi_m\}$ are the mixing coefficients and each component is governed by a mean m_m and a variance σ_m^2. We may now write down a likelihood which is a function of the parameters, $\mathbf{W}$, β and $\boldsymbol{\mu}$

$$p(\mathbf{t}|\mathbf{W}, \beta, \boldsymbol{\mu}) = \prod_{n=1}^{N} \int p(\mathbf{t}_n|\mathbf{W}, \beta, \mathbf{x}_n, \boldsymbol{\mu}) p(\mathbf{x}_n) d\mathbf{x}_n. \tag{6.5}$$

This function may now be maximised with respect to the parameters in order to determine the independent components. Traditionally this optimisation is performed in the limit as β tends to zero. This approach to blind source separation was introduced by Bell and Sejnowski (1995) as an information maximisation algorithm. The relationship with maximum likelihood was pointed out by various authors including Cardoso (1997) and MacKay (1996). The model as we have described it is closely related to work by Attias (1998). However Attias assumes a richer noise model.

6.5.3 A Bayesian Formalism of ICA

In this chapter we propose to once again follow the Bayesian approach of inferring the parameterisation of the model. This requires us to place priors over the model parameters. We aim to show how through a particular selection of our prior distribution $p(\mathbf{W})$ we may learn the structure of our model and automatically determine the number of sources which have produced our data. We are inspired in our approach by the Bayesian PCA of Bishop (1999) which aims to determine the dimensionality of the principal sub-space automatically.

We treat the inverse noise parameter β with a gamma prior

$$p(\beta) = \text{gam}(\beta|a_\beta, b_\beta) \tag{6.6}$$

where, as in Chapter 5, we define the gamma-distribution as

$$\text{gam}(\tau|a, b) = \frac{b^a}{\Gamma(a)} \tau^{a-1} \exp(-b\tau). \tag{6.7}$$

It is common to place a Gaussian prior over parameters

$$p(\mathbf{W}|\alpha) = \prod_{i=1}^{I} \prod_{p=1}^{P} \mathcal{N}(w_{ip}|0, \alpha^{-1}) \tag{6.8}$$

where α is a hyper-parameter determining the inverse variance of the prior. The relevance of each input may be determined through the use of the automatic relevance determination (ARD) prior

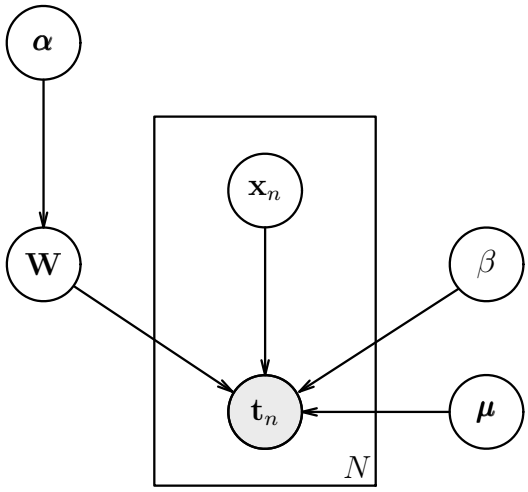


Figure 6.8: The super-graph of the proposed Bayesian formalism for ICA.

(Neal, 1996; MacKay, 1995b)

$$p(\mathbf{W}|\boldsymbol{\alpha}) = \prod_{i=1}^{I}\prod_{p=1}^{P}\mathcal{N}(w_{ip}|0,\alpha_i^{-1}) \tag{6.9}$$

where the prior is now governed by a vector of hyper-parameters, $\boldsymbol{\alpha}$, of length I. The parameters associated with each input of the network are governed by an element of the vector which determines its 'relevance'.

The hyper-parameters $\boldsymbol{\alpha}$ may again be treated with gamma priors

$$p(\boldsymbol{\alpha}) = \prod_{i=1}^{I}\mathrm{gam}(\alpha_i|a_{\alpha_i}\, b_{\alpha_i}). \tag{6.10}$$

Finally we place a Gaussian prior over the means, $\boldsymbol{\mu}$:

$$p(\boldsymbol{\mu}) = \prod_{p=1}^{P}\mathcal{N}(\boldsymbol{\mu}_p|0,\tau), \tag{6.11}$$

where τ represents the inverse variance of the prior.

We may now define our model likelihood

$$p(\mathbf{t}|\mathcal{M}) = \int p(\mathbf{t}|\mathbf{W},\beta,\mathbf{x},\boldsymbol{\mu})p(\mathbf{x})p(\mathbf{W}|\boldsymbol{\alpha})p(\boldsymbol{\alpha})p(\beta)p(\boldsymbol{\mu})d\mathbf{x}d\mathbf{W}d\boldsymbol{\alpha}d\beta d\boldsymbol{\mu}. \tag{6.12}$$

6.5.4 The Variational Approach

As it stands our model likelihood is intractable and once again we must look to approximations to make progress. Various options were presented in Chapter 5, in this thesis we are focusing on the variational approach. We make the following factorisation assumption for our variational approximation

$$q(\mathbf{x},\mathbf{W},\boldsymbol{\alpha},\beta,\boldsymbol{\mu}) = q_x(\mathbf{x})q_w(\mathbf{W})q_\alpha(\boldsymbol{\alpha})q_\beta(\beta)q_\mu(\boldsymbol{\mu}). \tag{6.13}$$

This leads to a lower bound on the model log-likelihood of the form

$$\ln p(\mathbf{t}|\mathcal{M}) \geq \langle \ln p(\mathbf{t}|\mathbf{W},\beta,\mathbf{x},\boldsymbol{\mu})p(\mathbf{x})p(\mathbf{W}|\boldsymbol{\alpha})p(\boldsymbol{\alpha})p(\beta)p(\boldsymbol{\mu})\rangle_{q_x q_w q_\alpha q_\beta q_\mu} + S(q_x) + S(q_w) + S(q_\alpha) + S(q_\beta) + S(q_\mu) \tag{6.14}$$

where as before we use the notation $\langle \cdot \rangle_q$ to denote an expectation under the distribution q. Once again the difference between the bound and the model likelihood, $\ln p(\mathbf{t}|\mathcal{M})$, is the KL-divergence between the true and approximating posterior. Recall from the introduction that for each sub-set of the parameters, $\boldsymbol{\theta}_i$, the best approximations for $q(\theta_i)$ as measured by the KL-divergence between will be

$$q(\boldsymbol{\theta}_i) \propto \exp \langle \ln p(\boldsymbol{\theta}, D) \rangle_{q(\theta_{j\neq i})}, \tag{6.15}$$

where $\langle \cdot \rangle_{q(\theta_{j\neq i})}$ indicates an expectation under the product of all the factors of the q-distribution except $q(\theta_i)$. For the Bayesian ICA model as we have described it, all the necessary expectations in Eqn 6.15 may be performed analytically giving the following results

$$q_x = \prod_{n=1}^{N} \frac{1}{Z_n} \sum_{m_1=1}^{M} \cdots \sum_{m_I=1}^{M} \tilde{\pi}_{\mathbf{m}}^{(n)} \mathcal{N}(x_n|\mathbf{f}_{\mathbf{m}}^{(n)}, \mathbf{D}_{\mathbf{m}}), \tag{6.16}$$

$$q_\mu = \mathcal{N}(\boldsymbol{\mu}|\mathbf{m}_\mu, \boldsymbol{\Sigma}_\mu) \tag{6.17}$$

$$q_w = \prod_{p=1}^{P} \mathcal{N}(\mathbf{w}_p|\mathbf{m}_w^{(p)}, \boldsymbol{\Sigma}_w) \tag{6.18}$$

$$q_\alpha = \prod_{i=1}^{I} \mathrm{gam}(\alpha_i|\tilde{a}_\alpha, \tilde{b}_\alpha^{(i)}) \tag{6.19}$$

$$q_\beta = \mathrm{gam}(\beta|\tilde{a}_\beta, \tilde{b}_\beta). \tag{6.20}$$

The parameters of the latent distribution, $\mathbf{D}_{\mathbf{m}}, \mathbf{f}_{\mathbf{m}}^{(n)}, \tilde{\pi}_{\mathbf{m}}^{(n)}$ and Z_n, are given in Appendix D. The other parameters can be found as

$$\mathbf{m}_w^{(p)} = \left(\langle \mathrm{diag}(\boldsymbol{\alpha}) \rangle + \langle \beta \rangle \sum_{n=1}^{N} \left\langle \mathbf{x}_n \mathbf{x}_n^{\mathrm{T}} \right\rangle \right)^{-1} \sum_{n=1}^{N} \langle \mathbf{x}_n \rangle (t_{np} - \mu_p) \tag{6.21}$$

$$\boldsymbol{\Sigma}_w = \left(\langle \mathrm{diag}(\boldsymbol{\alpha}) \rangle + \langle \beta \rangle \sum_{n=1}^{N} \left\langle \mathbf{x}_n \mathbf{x}_n^{\mathrm{T}} \right\rangle \right)^{-1} \tag{6.22}$$

$$\mathbf{m}_\mu^{(n)} = (\tau + N \langle \beta \rangle)^{-1} \langle \beta \rangle \sum_{n=1}^{N} (\mathbf{t}_n - \langle \mathbf{W} \rangle \langle \mathbf{x}_n \rangle) \tag{6.23}$$

$$\boldsymbol{\Sigma}_\mu = (\tau + N \langle \beta \rangle)^{-1} \tag{6.24}$$

$$\tilde{a}_\alpha = a_\alpha + \frac{P}{2} \tag{6.25}$$

$$\tilde{b}_\alpha^{(i)} = b_\alpha + \frac{\left\langle \mathbf{w}_i^{\mathrm{T}} \mathbf{w}_i \right\rangle}{2} \tag{6.26}$$

$$\tilde{a}_\beta = a_\beta + \frac{NP}{2} \tag{6.27}$$

$$\tilde{b}_\beta^{(i)} = b_\beta + \frac{1}{2} \sum_{n=1}^{N} \left\{ \mathbf{t}_n^{\mathrm{T}} \mathbf{t}_n + \left\langle \boldsymbol{\mu}^{\mathrm{T}} \boldsymbol{\mu} \right\rangle \mathrm{Tr} \left(\left\langle \mathbf{W}^{\mathrm{T}} \mathbf{W} \right\rangle \left\langle \mathbf{x}_n \mathbf{x}_n^{\mathrm{T}} \right\rangle \right) \right. \tag{6.28}$$

$$\left. + 2 \left\langle \boldsymbol{\mu}^{\mathrm{T}} \right\rangle \langle \mathbf{W} \rangle \langle \mathbf{x}_n \rangle - 2 \mathbf{t}_n^{\mathrm{T}} \langle \mathbf{W} \rangle \langle \mathbf{x}_n \rangle - 2 \mathbf{t}_n^{\mathrm{T}} \langle \boldsymbol{\mu} \rangle \right\} \tag{6.29}$$

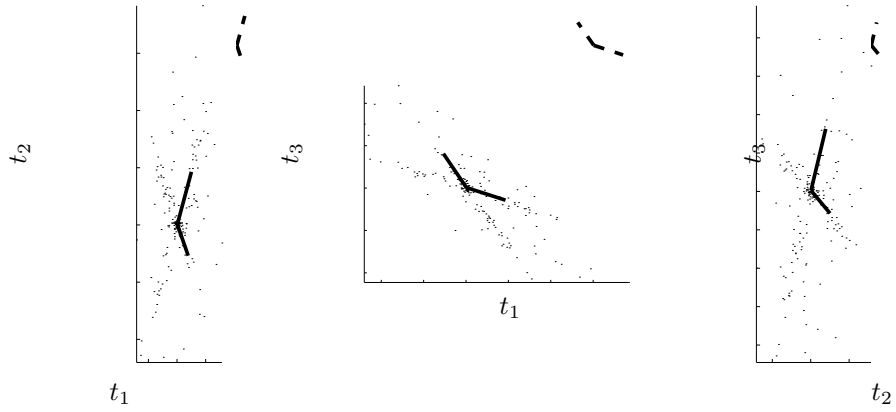


Figure 6.9: Scatter plots of samples from the model. The discovered embedded dimensions are shown on the centre of the plots while the true embedded dimensions are shown to the upper right of each plot.

where diag($\boldsymbol{\alpha}$) denotes a diagonal matrix whose diagonal elements are given by α_i. All the required expectations in these equations may be trivially calculated. Each of these parameters may thus be independently updated by applying its equation whilst keeping the other parameters fixed.

6.5.5 Preliminary Results

Toy Problem

For an evaluation of the ability of our model to determine the latent dimensionality, we sampled a toy data-set $\{\mathbf{t}_n\}$ from the model in which the two source distributions were chosen to contain two components, both with mean zero. The variances of the two components, σ_m^2, were taken to be 1 and 50. The number of observed data dimensions was taken to be three, and the inverse variance of the noise was set at 1×10^{-4}. The values of the mixing matrix $\mathbf{W}$ were randomly sampled from a Gaussian with unit variance. We obtained 200 samples from the model and then sought to determine the number of sources and the noise variance by inference in another model with three sources. The sources were of identical type to those in the generating model.

The values of $\mathbf{W}$ were initialised with a PCA solution. Optimisation of the distributions then proceeded by first updating the moments of $\boldsymbol{\mu}$ then the moments of $\mathbf{x}$ and finally the moments of $\mathbf{W}$. These moments were updated until convergence or for a maximum of 100 times. The moments of $\boldsymbol{\alpha}$ and β were then updated. This whole process was repeated fifty times.

The selected independent components are shown in Figure 6.9. Note that the true number of sources has been correctly determined. Shown in Figure 6.10 is the evolution of each $\log_{10} \alpha_i$ during learning. Note that very quickly one of the hyper-parameters becomes three orders of magnitude larger than the others effectively switching off one of the sources.

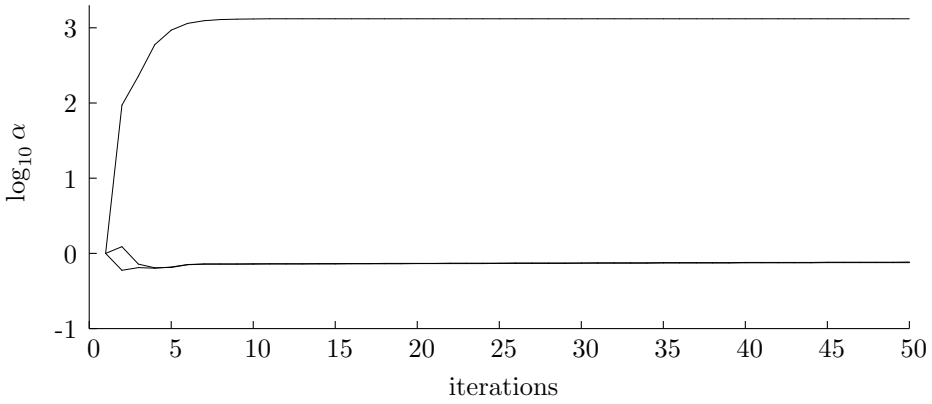


Figure 6.10: Evolution of the base 10 logarithm of the hyper-parameters during learning.

Real Data

As a further test of our model we analysed two images. The images were from The Corel Gallery 1,000,000 Collection and were averaged across their red, green and blue channels to obtain grey-scale images. The kurtosis of both images was negative, showing that both images are sub-Gaussian. We sub-sampled 1000 data-points from the images and then mixed them with the same matrix as for the toy problem. Gaussian noise was then added with a variance which was 7.8% of the source signal size. We applied two models to the data. The first contained two source distributions. The source distributions had two components both with unit variance and means of 1 and -1. The second model had three source distributions which were parameterised similarly.

The order of the update of the parameters was the same as for the toy problem and once again fifty iterations were used. Figure 6.11 shows the source images, the three mixed images with noise added and, in the bottom row, the two images recovered by the first model. The images were appropriately rescaled for presentation. Figure 6.12 and Figure 6.13 show the results for the model with three sources. This model did not remove the third source. In the first figure the true inferred sources are shown. In the second figure, the third output dimension (the one associated with the highest α_i) has been manually removed after learning and before inferring the sources to show that when the correct structure is determined reasonable results are obtained. Figure 6.14 shows scatter plots of the sub-sampled data projected along the three possible pairings of the output dimensions. Also shown in the upper right corners of each plot are the true directions of the mixing matrix, and in the centre of the plot the expected value of the mixing matrix determined by the algorithm.

6.5.6 Discussion of Bayesian ICA

In experiments on images the independent component analysis model is unable to determine the true number of sources for the real data. This problem perhaps arises because the assumed forms of the latent distributions are not matched to the true source distributions[3]. As a result the model

[3]We are not referring to the issue of sub-Gaussian versus super-Gaussian distributions, we have assumed that the sign of the kurtosis of the sources was known in these experiments and have selected our latent distributions accordingly.

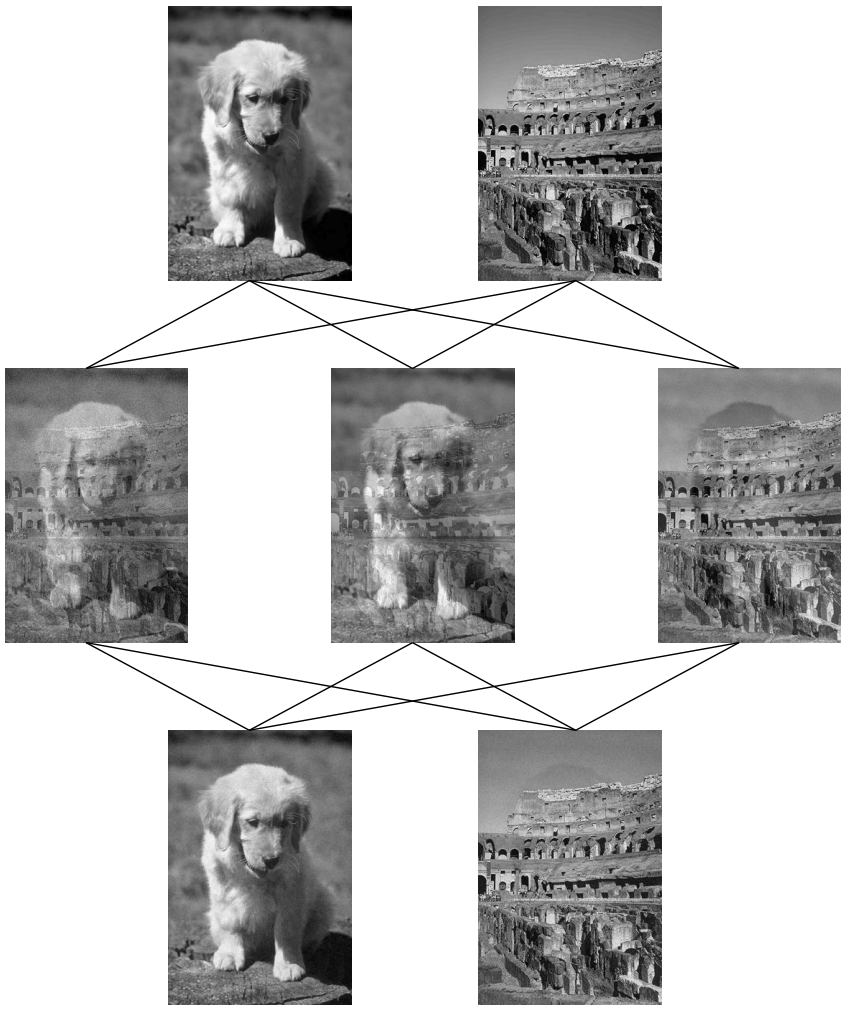


Figure 6.11: Results when the model is given the true number of latent variables. The top row of images are the original two images. The middle row are the mixed images which make up the observed data, $\mathcal{V}$. The bottom row are the inferred latent images which have been rescaled for presentation.

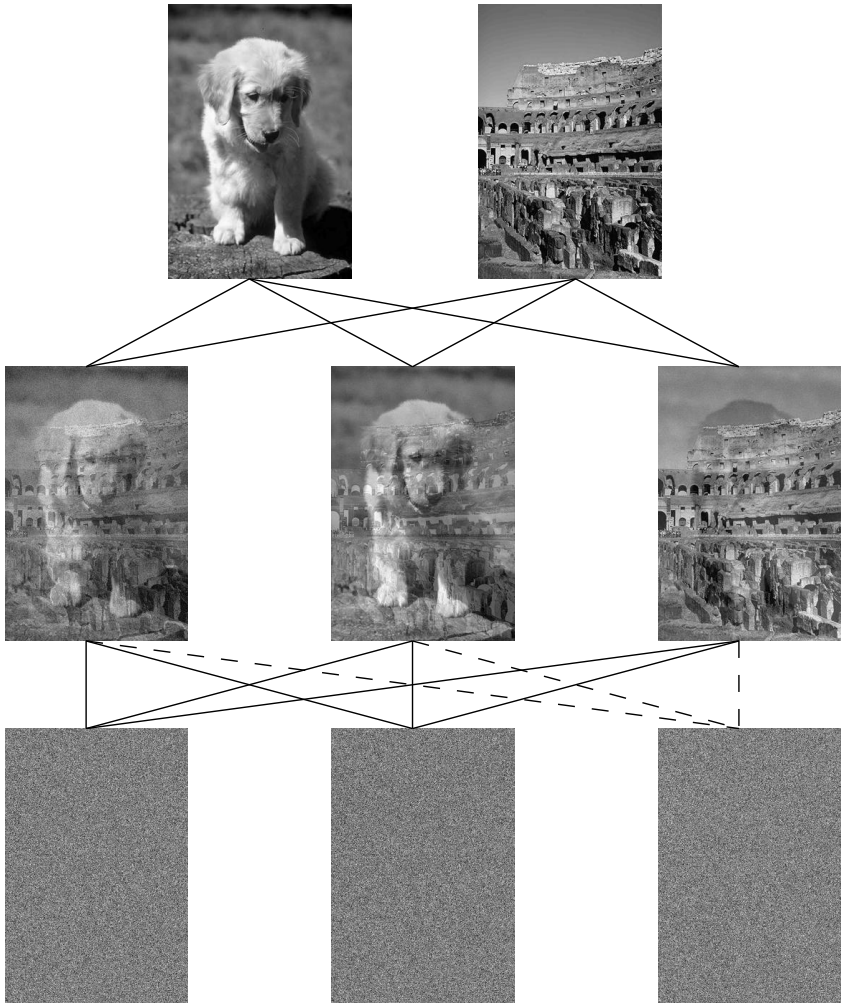


Figure 6.12: Results when the model has to determine the latent dimensionality. The recovered sources with the phantom latent dimension (indicated by dashed lines in the connections) left in.

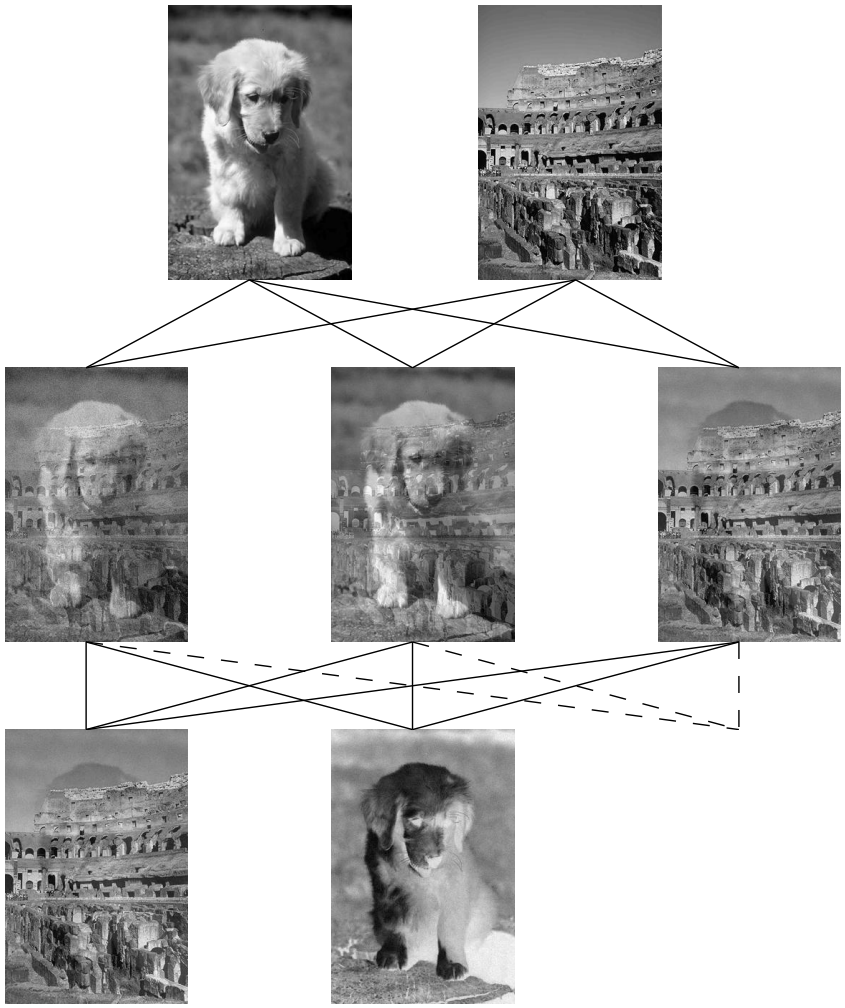


Figure 6.13: Results when the model has to determine the latent dimensionality. The recovered sources with the phantom latent dimension manually removed. Note that there is a stronger ghost of the puppy on the image of the Colosseum.

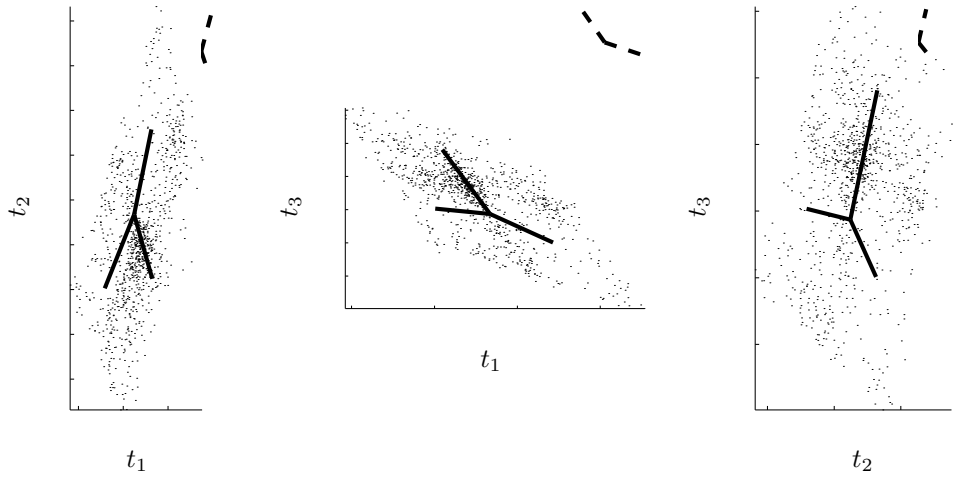


Figure 6.14: Scatter plots of sub-samples of the observed images for the experiment attempting to determine the true number of sources. The discovered embedded dimensions are shown on the centre of the plots as while the true embedded dimensions are shown to the upper right of each plot.

may always better explain the data by adding 'phantom' source distributions. This is because the embedded observation space may obtain more and more complex forms through the addition of more latent dimensions. This is not the case in PCA. In PCA the latent distributions are taken to be Gaussian and any sub-space embedded within a Gaussian distribution will also be Gaussian. As a result Bayesian PCA does not suffer from this issue.

We might take the approach of removing the source associated with the highest hyper-parameter and checking whether the lower bound on the model likelihood increases. Unfortunately, although we are maximising a lower bound on the model-likelihood for Bayesian ICA, we are unable to determine the value of this bound. This is a consequence of the posterior for the latent variables being a mixture of Gaussians. To calculate the bound, we are required to evaluate the entropy of this distribution. There is some hope for progress because while exact evaluation of this entropy is intractable, it may be lower bounded as in the previous chapter.

Another solution, and the one we would propose, would be to adaptively modify the latent distributions as part of the optimisation process. This could be achieved through modification of the latent variable parameters (see for example Attias, 1998). Handling the parameterisation of the source distributions in a Bayesian manner would be desirable, it would lead to a natural trade off between the complexity of the source distributions and the number of source distributions.

A final alternative is to use a simpler model to determine the latent dimensionality. We could apply Bayesian PCA to determine the principal sub-space and the associated latent dimensionality. Then a simple maximum likelihood ICA model could be applied within that embedded space. We prefer our suggested solution though as it would be able to handle problems where there are more sources than observed dimensions and is therefore more general.

Another and very different approach to determining the sources has been suggested by Hyvärinen and Oja (1997). This approach is not based on a likelihood model and is related to projection pursuit. Lappalainen (1999) has also proposed a variational approach to independent component analysis. However, he assumes a particular parameterised form for the q-distributions.

An additional problem with our approach is the exponential growth in the number of components as a function of the latent space dimensionality. This makes the approach impractical for models with large numbers of sources.

6.5.7 Structural Learning of Graphical models

In Chapter 5 and above we discussed Bayesian inference in the context of neural networks. In this section we suggest the use of a novel form of prior where parameters are associated with more than one hyper-parameter. We show how this prior may be used in combination with a lower bound on the model likelihood to perform optimisation of structure. As an preliminary example we will show how such techniques may be implemented in regression neural networks, like those studied in Chapter 5. We demonstrate the algorithm on a toy problem and on the real world data-sets already studied in Chapter 5.

6.5.8 Bayesian Inference

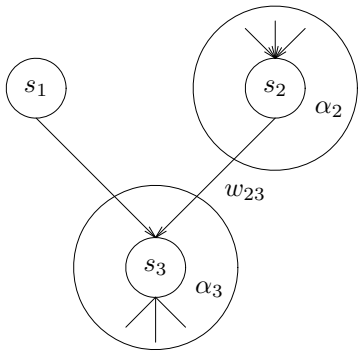
As we discussed in Chapter 1, in Bayesian inference we take our parameters, $\boldsymbol{\theta} \in \mathbb{R}$, to be random variables and we aim to determine their *posterior* distribution for use in making predictions

$$p(\boldsymbol{\theta}|D) = \frac{p(D|\boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(D|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}}. \tag{6.30}$$

A common choice of prior is a zero mean Gaussian,

$$p(\boldsymbol{\theta}|\mathbf{A}) = \mathcal{N}(\mathbf{0}, \mathbf{A}). \tag{6.31}$$

The covariance matrix, $\mathbf{A}$, is often taken to be diagonal. The prior is often constrained even further by assuming the Gaussian is spherical in which case the prior is governed by a single hyper-parameter. i.e. $\mathbf{A} = \alpha^{-1}\mathbf{I}$. We consider an alternative prior which takes the grouping of parameters a further stage. We consider a prior which places each parameter in two groups. Grouping the parameters in this manner may be of help in the optimisation of model structure. In the sigmoid belief network and Boltzmann machines, for example, a group may be associated with a variable. Each parameter is associated with two variables and therefore would belong to two groups (see Figure 6.15).



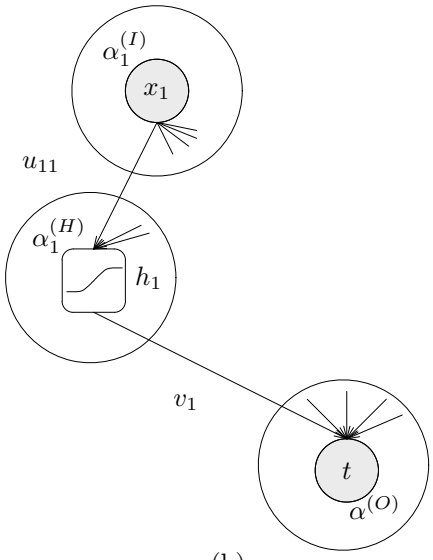
(a)

Figure 6.15: The node relevance determination prior as it would apply to a sigmoid belief network. The ith node has a hyper-parameter, α_i, associated with it. Each parameter in the connectivity matrix w_{ij} is therefore governed by two hyper-parameters: α_i and α_j.

For the case of neural networks with one hidden layer, for example, we choose to associate a hyper-parameter with each node in the network. We split the hyper-parameters into three sub-groups: $\boldsymbol{\alpha}^{(I)}, \boldsymbol{\alpha}^{(H)}, \alpha^{(O)}$, associated with the input nodes, hidden nodes and output node respectively (see Figure 6.16). Our prior then takes the form

$$\begin{aligned} p(\mathbf{w}|\boldsymbol{\alpha}^{(I)}, \boldsymbol{\alpha}^{(H)}, \alpha^{(O)}) &= \prod_{i=1}^{I}\prod_{j=1}^{H}\left(\frac{\alpha_i^{(I)}\alpha_j^{(H)}}{2\pi}\right)^{\frac{1}{2}} \exp\left\{-\frac{1}{2}\alpha_i^{(I)}\alpha_j^{(H)}u_{ij}^2\right\}\ldots \\ &\quad \prod_{i=1}^{H}\left(\frac{\alpha_i^{(H)}\alpha^{(O)}}{2\pi}\right)^{\frac{1}{2}} \exp\left\{-\frac{1}{2}\alpha_i^{(H)}\alpha^{(O)}v_i^2\right\}, \end{aligned} \tag{6.32}$$

where u_{ij} and v_i are elements of the matrix $\mathbf{u}$ and the vector $\mathbf{v}$ as defined in Eqn 5.2, the definition of the network function. We term this prior node relevance determination (NRD) as the objective is to determine the relevance of each node (or variable for graphs) in the model.



(b)

Figure 6.16: The node relevance determination prior as applied to a neural network. Here we show part of a single hidden layer network with one output. Each node in the network is associated with a hyper-parameter. A weight in the first layer of the network, u_{ij}, is governed by hyper-parameter associated with the input node, $\alpha_i^{(I)}$, and another associated with the hidden node, $\alpha_j^{(H)}$. A weight in the second layer, v_i, is governed by $\alpha_i^{(H)}$ and $\alpha^{(O)}$.

It now remains to choose a method to optimise the hyper-parameters. As in Chapter 5 we might choose to treat them in with a second level of Bayesian inference. Convenient choices of hyper-priors are products of gamma-distributions:

$$p(\boldsymbol{\alpha}^{(I)}) = \prod_{g=1}^{G} \text{gam}(\alpha_g^{(I)}|a_g, b_g) \tag{6.33}$$

$$p(\boldsymbol{\alpha}^{(H)}) = \prod_{g=1}^{G} \text{gam}(\alpha_g^{(H)}|a_g, b_g). \tag{6.34}$$

$$p(\boldsymbol{\alpha}^{(O)}) = \prod_{g=1}^{G} \text{gam}(\alpha_g^{(O)}|a_g, b_g). \tag{6.35}$$

Where we recall from Chapter 5 we define the gamma-distribution as

$$\text{gam}(\tau|a, b) = \frac{b^a}{\Gamma(a)}\tau^{a-1}\exp(-b\tau). \tag{6.36}$$

6.5.9 The Variational Approach

In Chapter 5 we reviewed the variational approach to inference in neural networks. Consider again the bound on the likelihood obtained through the introduction of a variational distribution $q(\mathbf{w}, \boldsymbol{\alpha}, \beta)$,

$$\ln p(D) \geq \int q(\mathbf{w}, \boldsymbol{\alpha}, \beta) \ln \frac{p(D, \mathbf{w}, \boldsymbol{\alpha}, \beta)}{q(\mathbf{w}, \boldsymbol{\alpha}, \beta)} \, d\mathbf{w} \, d\boldsymbol{\alpha} \, d\beta. \tag{6.37}$$

As in previous chapters we will assume that the variational distribution factorises

$$q_w(\mathbf{w}, \boldsymbol{\alpha}, \beta) = q_w(\mathbf{w}) q_\beta(\beta) q_\alpha(\boldsymbol{\alpha}). \tag{6.38}$$

As we are constraining our investigations to the treatment of the distribution, q_α, we only need consider the following result

$$q_\alpha(\boldsymbol{\alpha}) \propto \exp \left\langle \ln p(\mathbf{w}|\boldsymbol{\alpha}) p(\boldsymbol{\alpha}) \right\rangle_{q_w}, \tag{6.39}$$

where as before $\left\langle \cdot \right\rangle_q$ represents an expectation with respect to the distribution q.

When we substitute in the NRD prior from Eqn 6.32 and the hyper-priors from Eqn 6.33, Eqn 6.34 and Eqn 6.35, we must make an additional assumption of factorisability

$$q_\alpha = q_{\alpha^{(I)}}(\boldsymbol{\alpha}^{(I)}) q_{\alpha^{(H)}}(\boldsymbol{\alpha}^{(H)}) q_{\alpha^{(O)}}(\boldsymbol{\alpha}^{(O)}) \tag{6.40}$$

we then obtain

$$\begin{aligned}
q_{\alpha^{(I)}}(\boldsymbol{\alpha}^{(I)}) &= \prod_{i=1}^{I} \Gamma(\alpha_i^{(I)} | \tilde{a}_\alpha^{(I)}, \tilde{b}_{\alpha^{(I)} i}) && (6.41)\\
q_{\alpha^{(H)}}(\boldsymbol{\alpha}^{(H)}) &= \prod_{i=1}^{H} \Gamma(\alpha_i^{(H)} | \tilde{a}_\alpha^{(H)}, \tilde{b}_{\alpha^{(H)} i}) && (6.42)\\
q_{\alpha^{(O)}}(\boldsymbol{\alpha}^{(O)}) &= \Gamma(\alpha^{(O)} | \tilde{a}_\alpha^{(O)}, \tilde{b}_{\alpha^{(O)}}). && (6.43)
\end{aligned}$$

Where the parameters of the q-distributions are as follows

$$\begin{aligned}
\tilde{a}_\alpha^{(I)} &= a_\alpha^{(I)} + \frac{H}{2} && (6.44)\\
\tilde{b}_{\alpha^{(I)} i} &= b_\alpha^{(I)} + \sum_{j=1}^{H} \frac{\left\langle \alpha_j^{(H)} \right\rangle \left\langle u_{ij}^2 \right\rangle}{2} && (6.45)\\
\tilde{a}_\alpha^{(H)} &= a_\alpha^{(H)} + \frac{I+2}{2} && (6.46)\\
\tilde{b}_{\alpha^{(H)} i} &= b_\alpha^{(H)} + \sum_{j=0}^{I} \frac{\left\langle \alpha_j^{(I)} \right\rangle \left\langle u_{ji}^2 \right\rangle}{2} + \frac{\left\langle \alpha^{(O)} \right\rangle \left\langle v_i^2 \right\rangle}{2} && (6.47)\\
\tilde{a}_\alpha^{(O)} &= a_\alpha^{(O)} + \frac{H+1}{2} && (6.48)\\
\tilde{b}_{\alpha^{(O)}} &= b_\alpha^{(O)} + \sum_{j=0}^{H} \frac{\left\langle \alpha_j^{(H)} \right\rangle \left\langle v_j^2 \right\rangle}{2}. && (6.49)
\end{aligned}$$

In the case of the likelihood function we defined for the regression neural network we may also calculate bound 6.37 using the following results for the moment of $\ln x$ and the entropy under the distribution $\mathrm{gam}(x|a,b)$ (see Appendix C)

$$\begin{aligned}
\int \mathrm{gam}(x|a,b) \ln x dx &= \psi(a) - \ln b && (6.50)\\
-\int \mathrm{gam}(x|a,b) \ln \mathrm{gam}(x|a,b) dx &= -(a-1)\psi(a) - \ln b + a + \ln \Gamma(a) && (6.51)
\end{aligned}$$

where the function $\psi(a)$ is defined as

$$\psi(a) = \frac{d}{da} \ln \Gamma(a). \tag{6.52}$$

This enables us to monitor convergence of the optimisation and additionally to perform model comparison.

Expectation-Maximisation Optimisation of Structure

Optimisation of the lower bound on $p(D|\mathcal{M})$ with respect to the q-distributions can be viewed as an approximate expectation step in an expectation-maximisation algorithm. This expectation step can then be followed by a maximisation step which maximises the lower bound on $p(D|\mathcal{M})$ with respect to the structure of the model $\mathcal{M}$. This could involve the removal of individual weights, but we consider only node removal. We use the following heuristic to select a node to remove. If we wish to remove a node in the hidden layer we initially compute the lower bound 6.37. We then compute the effective number of parameters for each hidden node where

$$\gamma_h = \sum_i \frac{\alpha_h^{(H)} \alpha_i^{(I)}}{d_{ih}^2} + \frac{\alpha_h^{(H)} \alpha^{(O)}}{\sigma_h^2} \tag{6.53}$$

is defined as the effective number of parameters associated with node h and we have also defined $d_{ih}^2 = \left\langle u_{ih}^2 \right\rangle - \left\langle u_{ih} \right\rangle^2$ and $\sigma_h^2 = \left\langle v_h^2 \right\rangle - \left\langle v_h \right\rangle^2$. We remove the node with the lowest effective number of parameters and re-evaluate the bound. If it has increased we continue training with the new structure. Otherwise we replace the node. A similar process can be undertaken for the input nodes.

6.5.10 Results

Node Removal in a Toy Problem

To determine the effectiveness of the node-removal prior, we first studied a simple problem involving samples from a sine wave. We took N samples from the function $0.4\sin(2\pi x)$ uniformly over the interval[4] (0, 1) and added Gaussian noise of variance 0.0025. We then trained a regression neural network with five hidden nodes using the node relevance prior. We used a quasi-newton optimiser to determine the weights and followed this by an update of the inverse noise variance β and the hyper-parameters $\boldsymbol{\alpha}$. We then attempted to optimise structure in the manner described in the previous section. We cycled through the operations ten times. The experiment was repeated for $N = 5, 10, 25, 50, 100, 200$. Ten networks were trained for each number of data-points using different samples from the function data-points. The results are summarised in Figure 6.17 were the average number of determined hidden nodes is plotted against the number of data points.

Real Data

We next studied the two data-sets described in Chapter 5. The results are summarised in Table 6.1 and Table 6.2. The optimisations were undertaken in an identical manner to that described in Chapter 5 excepting that the networks with the NRD prior were optimised over twenty cycles of the algorithm and we attempted to remove hidden and input nodes at the end of each cycle. The extra cycles were required so that the networks would have the opportunity to remove more than the ten nodes that ten cycles would have allowed. Again five different networks were initialised and we selected with the largest lower bound to the model to the model likelihood. For the sunspot data, the NRD network

[4] This is one period of oscillation.

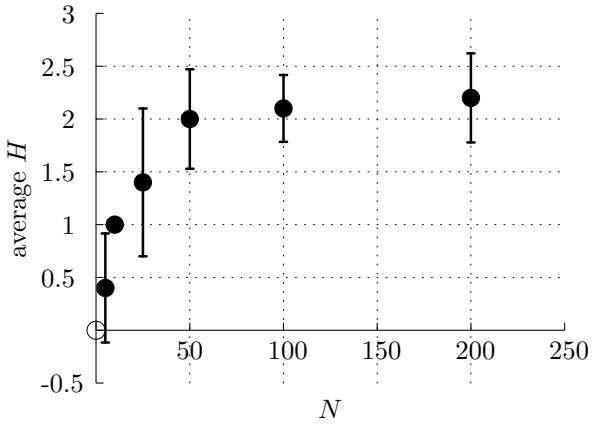


Figure 6.17: The average determined number of hidden nodes vs. number of data for the toy problem. The error bars show the standard deviation of the number of hidden nodes.

Prior type	Test error
single	0.190
grouped	0.194
ARD	0.153
NRD (3x5x1)	0.163
Hybrid Monte-Carlo grouped	0.131
Hybrid Monte-Carlo ARD	0.132

Table 6.1: Performance of different priors on the sunspot time series. The structure of the network determined using NRD is shown in the form IxHxP, where I is the number of input nodes, H is hidden nodes and P is output nodes.

with the highest model likelihood used three hidden nodes and five input nodes. The input nodes used to predict x_n were $x_{n-1}, x_{n-2}, x_{n-5}, x_{n-7}$ and x_{n-8}. Note that in time series prediction it is normal to try and select optimal windows of inputs. Normally a window of size W would include the inputs from $n-1$ to $n-W$. Our structural selection method has not chosen a window of this type. Exploring windows of a range of lengths may be possible using cross validation, however considering windows which miss out points would be impractical. Our approach to structural selection enables this.

For the Tecator data-set, the model determined a structure containing all the original input nodes but with only three hidden nodes.

Table 6.1 and Table 6.2 summarise the results obtained. The priors we tried are named 'single' which considers only one hyper-parameter; 'grouped' which groups the weights according as input-hidden layer, hidden biases, hidden-output layer and output biases; 'ARD' which further groups the input-hidden layer weights according to the input node with which they are associated and the NRD prior described above. We include in the comparison some results using hybrid Monte-Carlo Sampling.

Method	Test error
single	0.549
grouped	0.540
ARD	0.539
NRD (10x3x1)	0.532
Hybrid Monte-Carlo grouped	0.481
Hybrid Monte-Carlo ARD	0.471

Table 6.2: Performance of different priors on the Tecator data-set. Again the structure of the network determined using NRD is shown in the form IxHxP.

6.5.11 Discussion of the Node Relevance Determination Prior

We also made use of a novel form of prior to perform pruning in a neural network and indicated how such a prior could be utilised in graphical models. Pruning of nodes in a neural network is a method of controlling the capacity of the model and discovering the simplest representation for the data. Some researchers are of the opinion that a pruning approach is misguided. In particular the Gaussian process approach (Williams and Rasmussen, 1996) is to consider models in the limit as the number of units in the hidden layer goes to infinity, giving the model infinite capacity. Capacity control is then achieved through priors which incorporate knowledge from the problem domain. This is quite a persuasive argument in the application of neural networks to regression problems, however, if the task were to be knowledge discovery orientated, perhaps through structural learning of graphical models, the approach could prove useful.

Appendix A

The Sigmoid Belief Network Normalisation Factor

In this appendix we review some the approach to handling the intractable expectation which arises in Chapter 2.

For sigmoid belief networks we are interested in upper bounding

$$\left\langle \ln \left[1 + \exp \left(\sum_j w_{ij} s_j \right) \right] \right\rangle . \tag{A.1}$$

Saul *et al.*, 1996 make use of the following upper bound

$$\begin{aligned} \left\langle \ln \left[1 + \exp \left(\sum_j w_{ij} s_j \right) \right] \right\rangle \quad &\leq \quad \xi_i \sum_j w_{ij} \left\langle s_j \right\rangle \\ &\quad + \ln \left\langle \exp \left(-\xi_i \sum_j w_{ij} s_j \right) + \exp \left((1 - \xi_i) \sum_j w_{ij} s_j \right) \right\rangle \end{aligned} \tag{A.2}$$

where $\xi_i \in \mathbb{R}$. We are interested only in how this functional dependence of this bound on s_k, when the expectation is under all the approximating distributions except $Q(s_k)$ we therefore re-write the bound

$$\begin{aligned} \left\langle \ln \left[1 + \exp \left(\sum_j w_{ij} s_j \right) \right] \right\rangle_{\prod_{j \neq k} Q(s_j)} \quad &\leq \quad \xi_i w_{ik} s_k \\ &\quad + s_k \ln \Bigg[\exp \left(-w_{ik} \xi_i \right) \Bigg\langle \exp \Big(- \xi_i \sum_{j \neq k} w_{ij} s_j \Big) \\ &\quad + \exp(w_{ik}) \exp \Big((1 - \xi_i) \sum_{j \neq k} w_{ij} s_j \Big) \Bigg\rangle \Bigg] \\ &\quad + \text{const.}, \end{aligned} \tag{A.3}$$

where const represents terms which do not depend on s_k. Ignoring the constant we may rewrite the first three terms of the bound,

$$\ln \left\{ \Phi_i + \exp \left[w_{ik} \right] \Phi_i' \right\} \quad \stackrel{\text{def}}{=} \quad F(\left\langle \mathcal{S} \right\rangle, \mathbf{W}, i, k), \tag{A.4}$$

where we have made use of

$$\begin{aligned}
\Phi_i &\stackrel{\text{def}}{=} \left\langle \exp\left[-\xi_i\left(\sum_{j\neq k} w_{ij}s_j\right)\right]\right\rangle \\
&= \prod_{j\neq k}\left[\langle s_j\rangle \exp\left[-\xi_i w_{ij}\right] + (1-\langle s_j\rangle)\right] \qquad \text{(A.5)} \\
\Phi'_i &\stackrel{\text{def}}{=} \left\langle \exp\left[(1-\xi_i)\left(\sum_{j\neq k} w_{ij}s_j\right)\right]\right\rangle \qquad \text{(A.6)} \\
&= \prod_{j\neq k}\left[\langle s_j\rangle \exp\left[(1-\xi_i) w_{ij}\right] + (1-\langle s_j\rangle)\right] \qquad \text{(A.7)}
\end{aligned}$$

If we define

$$\phi_k \stackrel{\text{def}}{=} \sum_i F(\langle \mathcal{S}\rangle, \mathbf{W}, i, k) \qquad \text{(A.8)}$$

we see ϕ_k is dependent on the expectations of the parents of the children of k, or in other words the expectations of the co-parents.

Appendix B

Derivatives for the Markov Chain Q-distribution

B.1 Derivatives for Inference

We parameterise the probability μ_{ij} using $\mu_{ij} \equiv 1/(1+\exp[-\phi_{ij}])$. Thus constraining its value to be in $[0,1]$.

We therefore require the derivatives with respect to $\boldsymbol{\xi}$

$$\frac{\partial \left\langle e^{-\xi_i z_i} \right\rangle}{\partial \xi_i} = \sum_{k=0}^{\mathcal{N}} [1 \;\; 1] \left[\prod_{j=N}^{k+1} \mathbf{K}_{ij} \mathbf{A}_j\right] \begin{bmatrix} 0 & 0 \\ 0 & -w_{ik} e^{-\xi_i w_{ik}} \end{bmatrix} \mathbf{A}_k \left[\prod_{j=k-1}^{0} \mathbf{K}_{ij} \mathbf{A}_j\right] \tag{B.1}$$

$$\frac{\partial \left\langle e^{(1-\xi_i) z_i} \right\rangle}{\partial \xi_i} = \sum_{k=0}^{\mathcal{N}} [1 \;\; 1] \left[\prod_{j=\mathcal{N}}^{k+1} \tilde{\mathbf{K}}_{ij} \mathbf{A}_j\right] \begin{bmatrix} 0 & 0 \\ 0 & -w_{ik} e^{(1-\xi_i) w_{ik}} \end{bmatrix} \mathbf{A}_k \left[\prod_{j=k-1}^{0} \tilde{\mathbf{K}}_{ij} \mathbf{A}_j\right]. \tag{B.2}$$

By storing partial products in the forward and reverse directions, all of these derivatives can be computed in time that scales linearly with the connectivity.

We also require the derivatives of the likelihood with respect to the parameters of the Q distribution: To compute the derivatives of $\mathcal{L}(Q)$ with respect to ϕ_{jk} we need the derivatives of $\langle h_i \rangle$, $\left\langle e^{-\xi_i z_i} \right\rangle$ and $\left\langle e^{(1-\xi_i) z_i} \right\rangle$. If i and j are in different layers the partial derivatives of $\langle h_i \rangle$ with respect to both ϕ_{j0} and ϕ_{j1} are zero, or similarly (from Eqn 2.43) if $j > i$. Otherwise

$$\frac{\partial \langle h_i \rangle}{\phi_{j0}} = \mu_{j0}(1-\mu_{j0}) [0 \;\; 1] \left[\prod_{k=i}^{j+1} \mathbf{A}_k\right] \begin{bmatrix} -1 & 0 \\ 1 & 0 \end{bmatrix} \left[\prod_{k=j-1}^{0} \mathbf{A}_k\right] \tag{B.3}$$

$$\frac{\partial \langle h_i \rangle}{\phi_{j1}} = \mu_{j1}(1-\mu_{j1}) [0 \;\; 1] \left[\prod_{k=i}^{j+1} \mathbf{A}_k\right] \begin{bmatrix} 0 & -1 \\ 0 & 1 \end{bmatrix} \left[\prod_{k=j-1}^{0} \mathbf{A}_k\right]. \tag{B.4}$$

The derivatives of $\left\langle e^{-\xi_i z_i} \right\rangle$ and $\left\langle e^{(1-\xi_i) z_i} \right\rangle$ with respect to ϕ_{j0} and ϕ_{j1} are 0 if $j \neq i$. Otherwise

$$\begin{aligned}
\frac{\partial \left\langle e^{-\xi_i z_i} \right\rangle}{\partial \phi_{j0}} &= \mu_{j0}(1-\mu_{j0})[1 \;\; 1] \left[\prod_{k=\mathcal{N}}^{j+1} \mathbf{K}_{ik} \mathbf{A}_k \right] \mathbf{K}_{ij} \begin{bmatrix} -1 & 0 \\ 1 & 0 \end{bmatrix} \left[\prod_{k=j-1}^{0} \mathbf{K}_{ik} \mathbf{A}_k \right] \\
\frac{\partial \left\langle e^{-\xi_i z_i} \right\rangle}{\partial \phi_{j1}} &= \mu_{j1}(1-\mu_{j1})[1 \;\; 1] \left[\prod_{k=\mathcal{N}}^{j+1} \mathbf{K}_{ik} \mathbf{A}_k \right] \mathbf{K}_{ij} \begin{bmatrix} 0 & -1 \\ 0 & 1 \end{bmatrix} \left[\prod_{k=j-1}^{0} \mathbf{K}_{ik} \mathbf{A}_k \right] \\
\frac{\partial \left\langle e^{(1-\xi_i) z_i} \right\rangle}{\partial \phi_{j0}} &= \mu_{j0}(1-\mu_{j0})[1 \;\; 1] \left[\prod_{k=\mathcal{N}}^{j+1} \tilde{\mathbf{K}}_{ik} \mathbf{A}_k \right] \tilde{\mathbf{K}}_{ij} \begin{bmatrix} -1 & 0 \\ 1 & 0 \end{bmatrix} \left[\prod_{k=j-1}^{0} \tilde{\mathbf{K}}_{ik} \mathbf{A}_k \right] \\
\frac{\partial \left\langle e^{(1-\xi_i) z_i} \right\rangle}{\partial \phi_{j1}} &= \mu_{j1}(1-\mu_{j1})[1 \;\; 1] \left[\prod_{k=\mathcal{N}}^{j+1} \tilde{\mathbf{K}}_{ik} \mathbf{A}_k \right] \tilde{\mathbf{K}}_{ij} \begin{bmatrix} 0 & -1 \\ 0 & 1 \end{bmatrix} \left[\prod_{k=j-1}^{0} \tilde{\mathbf{K}}_{ik} \mathbf{A}_k \right] .
\end{aligned} \tag{B.5}$$

The time needed to evaluate these derivatives for all units in a layer scales as the product of the number of units in the layer and the number of units in the parent layer.

Having evaluated the bound and its derivatives with respect to $\{\phi_{ij}\}$ and $\{\xi_i\}$, we can use a standard gradient-based minimisation algorithm such as conjugate gradients to optimise the variational distribution.

B.2 Derivatives for Learning

Once we have optimised the variational distribution for each pattern in a training set, we can modify the network parameters, $\mathbf{W}$. To compute the derivative of the likelihood bound $\mathcal{L}$ for one input pattern, we use the following:

$$\begin{aligned}
\frac{\partial \left\langle e^{-\xi_i z_i} \right\rangle}{\partial w_{ij}} &= [1 \;\; 1] \left[\prod_{k=\mathcal{N}}^{j+1} \mathbf{K}_{ik} \mathbf{A}_k \right] \begin{bmatrix} 0 & 0 \\ 0 & -\xi_i e^{-\xi_i w_{ij}} \end{bmatrix} \mathbf{A}_j \left[\prod_{k=j-1}^{0} \mathbf{K}_{ik} \mathbf{A}_k \right] \\
\frac{\partial \left\langle e^{(1-\xi_i) z_i} \right\rangle}{\partial w_{ij}} &= [1 \;\; 1] \left[\prod_{k=\mathcal{N}}^{j+1} \tilde{\mathbf{K}}_{ik} \mathbf{A}_k \right] \begin{bmatrix} 0 & 0 \\ 0 & -\xi_i e^{(1-\xi_i) w_{ij}} \end{bmatrix} \mathbf{A}_j \left[\prod_{k=j-1}^{0} \tilde{\mathbf{K}}_{ik} \mathbf{A}_k \right] .
\end{aligned} \tag{B.6}$$

The time needed to evaluate these derivatives for all units in a layer scales as the product of the number of units in the layer and the number of units in the parent layer.

Appendix C

Derivations of Bayesian Variational Learning

C.1 Lower bound on the entropy of a mixture distribution

The entropy of a mixture distribution, $q_w(\mathbf{w}) = \sum_m Q(m) q_w(\mathbf{w}|m)$, is defined as:

$$S(q_w(\mathbf{w})) = -\sum_{m=1}^{M} Q_w(m) \int q_w(\mathbf{w}|m) \ln \left\{ \sum_{m=1}^{M} Q_w(m) q_w(\mathbf{w}|m) \right\} d\mathbf{w}. \tag{C.1}$$

By introducing the mutual information, $I(m; \mathbf{w})$, between the component label m and the variable $\mathbf{w}$ in the distribution, the entropy can be rewritten:

$$\begin{aligned} S(q_w(\mathbf{w})) &= -\sum_{m=1}^{M} Q_w(m) \int q_w(\mathbf{w}|m) \ln \left\{ q_w(\mathbf{w}|m) \frac{q_w(\mathbf{w})}{q_w(\mathbf{w}|m)} \right\} d\mathbf{w} \\ &= -\sum_{m=1}^{M} Q_w(m) \int q_w(\mathbf{w}|m) \ln q_w(\mathbf{w}|m) d\mathbf{w} \\ &\quad + \sum_{m=1}^{M} Q_w(m) \int q_w(\mathbf{w}|m) \ln \frac{q_w(\mathbf{w}|m)}{q_w(\mathbf{w})} d\mathbf{w} \\ &= \sum_{m=1}^{M} Q_w(m) S(q_w(\mathbf{w}|m)) + I(m; \mathbf{w}). \end{aligned} \tag{C.2}$$

We are still left with an average of a logarithm of a sum in the mutual information. To make progress, we look to a lower bound on the mutual information which was first introduced by Jaakkola and

Jordan (1998) for discrete systems:

$$\begin{aligned} I(m;\mathbf{w}) &= \sum_{m=1}^{M} Q_w(m) \int q_w(\mathbf{w}|m) \ln \frac{q_w(\mathbf{w}|m)}{q_w(\mathbf{w})} d\mathbf{w} \\ &= -\sum_{m=1}^{M} Q_w(m) \int q_w(\mathbf{w}|m) \ln \left\{ \frac{Q_w(m) r(\mathbf{w}|m)}{Q_w(m) r(\mathbf{w}|m)} \frac{q_w(\mathbf{w})}{q_w(\mathbf{w}|m)} \right\} d\mathbf{w} \\ &= \sum_{m=1}^{M} Q_w(m) \int q_w(\mathbf{w}|m) \ln r(\mathbf{w}|m) d\mathbf{w} - \sum_{m} Q_w(m) \ln Q_w(m) \\ &\quad + \sum_{m=1}^{M} Q_w(m) \int q_w(\mathbf{w}|m) \left\{ -\ln \frac{r(\mathbf{w}|m)}{Q_w(m)} \frac{q_w(\mathbf{w})}{q_w(\mathbf{w}|m)} \right\} d\mathbf{w}, \end{aligned} \tag{C.3}$$

where we have introduced a smoothing distribution $r(\mathbf{w}|m)$. Note that it is not necessary to normalise the smoothing distribution and we implemented it without normalisation.

In order to avoid averaging the logarithm of the sum in $q_w(\mathbf{w})$, we now make use of the convexity inequality

$$-\ln(x) \geq -\lambda x + \ln(\lambda) + 1 \tag{C.4}$$

and introduce a new variational parameter λ for each component m. We then obtain

$$\begin{aligned} I(m;\mathbf{w}) &\geq \sum_{m=1}^{M} Q_w(m) \int q_w(\mathbf{w}|m) \ln r(\mathbf{w}|m) d\mathbf{w} \\ &\quad - \sum_{m=1,m'=1}^{M} Q_w(m) \lambda_{m'} \int q_w(\mathbf{w}|m) r(\mathbf{w}|m') d\mathbf{w} \\ &\quad - \sum_{m=1}^{M} Q_w(m) \ln Q_w(m) + \sum_{m=1}^{M} Q_w(m) \ln \lambda_m + 1. \\ &\equiv \mathcal{I}(m;\mathbf{w}) \end{aligned} \tag{C.5}$$

For the case of mixtures of Gaussian distributions $q_w(\mathbf{w}|m) = \mathcal{N}(\bar{\mathbf{w}}_m, \mathbf{\Sigma}_{q_m})$ and we choose $r(\mathbf{w}|m) \propto \mathcal{N}(\bar{\mathbf{r}}_m, \mathbf{\Sigma}_{r_m})$. The second term is then the integral across two Gaussians which is itself Gaussian in form, for convenience we define

$$\Delta(m) = (\bar{\mathbf{w}}_m - \bar{\mathbf{r}}_m)^{\mathrm{T}} \mathbf{\Sigma}_{r_m}^{-1} (\bar{\mathbf{w}}_m - \bar{\mathbf{r}}_m), \tag{C.6}$$

$$\Delta(m,m') = (\bar{\mathbf{w}}_m - \bar{\mathbf{r}}_{m'})^{\mathrm{T}} (\mathbf{\Sigma}_{q_m} + \mathbf{\Sigma}_{r_{m'}})^{-1} (\bar{\mathbf{w}}_m - \bar{\mathbf{r}}_{m'}), \tag{C.7}$$

$$\mathbf{A}_{m,m'} = \mathbf{I} + \mathbf{\Sigma}_{q_m} \mathbf{\Sigma}_{r_{m'}}^{-1}, \tag{C.8}$$

giving

$$\langle r(\mathbf{w}|m') \rangle_{q_w(\mathbf{w}|m)} = |\mathbf{A}_{m,m'}|^{-\frac{1}{2}} \exp\left(-\frac{1}{2} \Delta(m,m') \right). \tag{C.9}$$

Also required is the expectation of the logarithm of the smoothing distribution, again this may be computed analytically

$$\langle \ln r(\mathbf{w}|m') \rangle_{q_w(\mathbf{w}|m)} = -\frac{1}{2} \left\{ \mathrm{tr}(\mathbf{\Sigma}_{q_m} \mathbf{\Sigma}_{r_m}^{-1}) + \Delta(m) \right\}. \tag{C.10}$$

The entropy of each component of the mixture may be written:

$$S(q_w(\mathbf{w}|m)) = \frac{1}{2}\ln|\mathbf{\Sigma}_{q_m}| + \frac{K}{2}(1+\ln 2\pi). \tag{C.11}$$

This leads us to our lower bound on the entropy of a mixture of Gaussians:

$$\begin{aligned} S(q_w(\mathbf{w})) &\geq \frac{1}{2}\sum_{m=1}^{M} Q_w(m)\ln|\mathbf{\Sigma}_{q_m}| + \frac{K}{2}(1+\ln 2\pi) \\ &\quad -\frac{1}{2}\sum_{m=1}^{M} Q_w(m)\left\{\mathrm{tr}(\mathbf{\Sigma}_{q_m}\mathbf{\Sigma}_{r_m}^{-1}) + \Delta(m)\right\} \\ &\quad -\sum_{m,n}^{M} Q_w(m)\lambda_{m'}|\mathbf{A}_{m,m'}|^{-\frac{1}{2}}\exp\left\{-\frac{1}{2}\Delta(m,m')\right\} \\ &\quad -\sum_{m=1}^{M} Q_w(m)\ln Q_w(m) + \sum_{m=1}^{M} Q_w(m)\ln\lambda_m + 1 \qquad \text{(C.12)} \\ &\equiv \hat{S}(q_w(\mathbf{w})). \qquad \text{(C.13)} \end{aligned}$$

Additionally the mutual information may be upper bounded,

$$I(m;\mathbf{w}) = S(m) - S(m|\mathbf{w}) \leq S(m) \leq \ln M. \tag{C.14}$$

C.2 Lower bound on the log-likelihood

Recall our bound 5.32,

$$\begin{aligned} \ln p(D) &\geq \sum_m Q_w(m)\int q_w(\mathbf{w}|m)q_\beta(\beta)\left\{-\beta E_D(D,\mathbf{w}) + \frac{N}{2}\ln\beta\right\}d\mathbf{w}\; d\beta \\ &\quad +\langle\ln p(\alpha)\rangle_{q_\alpha} + \langle\ln p(\beta)\rangle_{q_\beta} \\ &\quad +\sum_m Q_w(m)\int q_w(\mathbf{w}|m)q_\alpha(\alpha)\left\{-\alpha E_w(\mathbf{w}) + \frac{K}{2}\ln\alpha\right\}d\mathbf{w}\; d\alpha \\ &\quad +S(q_w(\mathbf{w})) + S(q_\alpha(\alpha)) + S(q_\beta(\beta)). \end{aligned} \tag{C.15}$$

To compute this we require the expectation of a logarithm under a gamma distribution.

$$\int_0^\infty \mathrm{gam}(x|a,b)\ln x\; dx = \int_0^\infty \ln x\frac{b^a x^{a-1}\exp(-bx)}{\Gamma(a)}dx, \tag{C.16}$$

where gam($\cdot$) represents the gamma distribution. Integrating by parts over a we may find

$$\begin{aligned} \int_0^{a'}\langle\ln x\rangle_{\mathrm{gam}(x|a,b)}\,da &= \int_0^\infty\left\{\left[\frac{b^a x^{a-1}\exp(-bx)}{\Gamma(a)}\right]_0^{a'}\right. \\ &\quad \left.-\int_0^{a'}\frac{b^a x^{a-1}\exp(-bx)}{\Gamma(a)}\left(\ln b - \frac{1}{\Gamma(a)}\frac{d\Gamma(a)}{da}\right)\right\}dx. \end{aligned} \tag{C.17}$$

Evaluating the integral over x now gives

$$\int_0^{a'}\langle\ln x\rangle_{\mathrm{gam}(x|a,b)}\,da = 1 - \int_0^{a'}\ln b - \frac{1}{\Gamma(a)}\frac{d\Gamma(a)}{da}da, \tag{C.18}$$

and finally differentiating both sides we obtain

$$\langle \ln x \rangle_{\mathrm{gam}(x|a,b)} \, da = \psi(a) - \ln b \tag{C.19}$$

where the function ψ is defined as

$$\psi(a) = \frac{d \ln \Gamma(a)}{da}. \tag{C.20}$$

We may utilise this result to obtain the entropies and cross entropies of gamma distributions and the required expectations of the hyper-parameters.

The expectation of the term derived from the prior is found as

$$\int \mathbf{w}^{\mathrm{T}} \mathbf{w} q_w(\mathbf{w}|m) d\mathbf{w} = \mathrm{tr}(\mathbf{\Sigma}_{q_m}) + \bar{\mathbf{w}}_m^{\mathrm{T}} \bar{\mathbf{w}}_m. \tag{C.21}$$

A more complex task is the computation of expectations of the network output and its square, which are required in the terms coming from the likelihood, dropping the component label m we have

$$\int q_w(\mathbf{w}) E_D(D, \mathbf{w}) d\mathbf{w} = \frac{1}{2} \sum_{n=1}^{N} \left\langle f(\mathbf{x}, \mathbf{w})^2 \right\rangle_q - \sum_{n=1}^{N} t_n \left\langle f(\mathbf{x}, \mathbf{w}) \right\rangle_q + \frac{1}{2} \sum_{n=1}^{N} t_n^2, \tag{C.22}$$

with $f(\mathbf{x}, \mathbf{w}) = \sum_h v_h g(\mathbf{u}_h^{\mathrm{T}} \mathbf{x})$. We initially follow Barber and Bishop (1998) in our treatment of the expectations,

$$\begin{aligned} q(\mathbf{u}, \mathbf{v}) \quad = \quad & \frac{1}{(2\pi)^{\frac{K}{2}} |\mathbf{\Sigma}|^{\frac{1}{2}}} \exp \Big\{ -\frac{1}{2} (\mathbf{u} - \bar{\mathbf{u}})^{\mathrm{T}} {\mathbf{\Sigma}_{\mathbf{uu}}}^{-1} (\mathbf{u} - \bar{\mathbf{u}}) \\ & -\frac{1}{2} (\mathbf{v} - \bar{\mathbf{v}})^{\mathrm{T}} {\mathbf{\Sigma}_{\mathbf{vv}}}^{-1} (\mathbf{v} - \bar{\mathbf{v}}) \Big\}, \end{aligned} \tag{C.23}$$

where we have used $\mathbf{u}$ to represent a vector formed from the vectors $\{\mathbf{u}_1 \ldots \mathbf{u}_H\}$ and we have introduced $\bar{\mathbf{u}}$ and $\bar{\mathbf{v}}$ which correspond to $\bar{\mathbf{w}}$ in the same way as $\mathbf{u}$ and $\mathbf{v}$ correspond to $\mathbf{w}$. The matrices $\mathbf{\Sigma_{vv}}$ and $\mathbf{\Sigma_{uu}}$ represent the associated covariance matrices, i.e. we have used the notation $\mathbf{\Sigma_{xy}}$ to indicate the part of the covariance matrix associated with the interactions between weights $\mathbf{x}$ and $\mathbf{y}$, both of which are taken to be diagonal in this implementation. For convenience, we use the notation $\mathbf{u}^{\mathrm{T}} \mathbf{x}_h$ instead of $\mathbf{u}_h^{\mathrm{T}} \mathbf{x}$, where $\mathbf{x}_h$ is a vector of the same dimension as the concatenated vector $\mathbf{u}$ with zero components everywhere except for those that correspond to hidden unit h. Those elements contain $\mathbf{x}$. The expectations can now be rewritten as:

$$\langle f(\mathbf{x}, \mathbf{w}) \rangle_{q_w} \quad = \quad \sum_{h=1}^{H} \left\langle v_h g(\mathbf{u}^{\mathrm{T}} \mathbf{x}_h) \right\rangle_{q(\mathbf{u},\mathbf{v})}, \tag{C.24}$$

$$\left\langle f(\mathbf{x}, \mathbf{w})^2 \right\rangle_{q_w} \quad = \quad \sum_{i=1}^{H} \sum_{j=1}^{H} \left\langle v_i v_j g(\mathbf{u}^{\mathrm{T}} \mathbf{x}_i) g(\mathbf{u}^{\mathrm{T}} \mathbf{x}_j) \right\rangle_{q(\mathbf{u},\mathbf{v})}. \tag{C.25}$$

As the components of $\mathbf{v}$ do not enter the non-linear function, we can immediately integrate over $\mathbf{v}$,

$$\langle f(\mathbf{x}, \mathbf{w}) \rangle_{q_w} \quad = \quad \sum_{h=1}^{H} \left\langle \bar{v}_h g(\mathbf{u}^{\mathrm{T}} \mathbf{x}_h) \right\rangle_{q(\mathbf{u})} \tag{C.26}$$

$$\left\langle f(\mathbf{x}, \mathbf{w})^2 \right\rangle_{q_w} \quad = \quad \sum_{i=1}^{H} \sum_{j=1}^{H} \left\langle (\bar{v}_i \bar{v}_j + \mathbf{\Sigma_{vv}}_{i,j}) g(\mathbf{u}^{\mathrm{T}} \mathbf{x}_i) g(\mathbf{u}^{\mathrm{T}} \mathbf{x}_j) \right\rangle_{q(\mathbf{u})}, \tag{C.27}$$

where $\mathbf{\Sigma_{vv}}_{i,j}$ is the element (i,j) of the matrix $\mathbf{\Sigma_{vv}}$. This leaves us with expectations of the activation functions under Gaussian distributions. These may be reduced to one dimensional integrals by firstly transforming the Gaussian to have zero mean and an isotropic covariance,

$$\left\langle g(\mathbf{u}^{\mathrm{T}}\mathbf{x}_h)\right\rangle_{q(\mathbf{u})} = \left\langle g\left(\bar{\mathbf{u}}^{\mathrm{T}}\mathbf{x}_h + \mathbf{x}_h^{\mathrm{T}}\mathbf{\Sigma_{uu}}^{\frac{1}{2}}\mathbf{s}\right)\right\rangle_{\mathcal{N}(0,\mathbf{I})} \tag{C.28}$$

$$\left\langle g(\mathbf{u}^{\mathrm{T}}\mathbf{x}_i)g(\mathbf{u}^{\mathrm{T}}\mathbf{x}_j)\right\rangle_{q(\mathbf{u})} = \left\langle g\left(\mathbf{x}_i^{\mathrm{T}}\bar{\mathbf{u}} + \mathbf{x}_i^{\mathrm{T}}\mathbf{\Sigma_{uu}}^{\frac{1}{2}}\mathbf{s}\right) g\left(\mathbf{x}_j^{\mathrm{T}}\bar{\mathbf{u}} + \mathbf{x}_j^{\mathrm{T}}\mathbf{\Sigma_{uu}}^{\frac{1}{2}}\mathbf{s}\right)\right\rangle_{\mathcal{N}(0,\mathbf{I})}, \tag{C.29}$$

where $\mathbf{s} = \mathbf{\Sigma_{uu}}^{-\frac{1}{2}}(\mathbf{u}-\bar{\mathbf{u}})$. We now rotate the co-ordinate system. For the $\langle g(\mathbf{u}^{\mathrm{T}}\mathbf{x}_h)\rangle_{q(\mathbf{u})}$ we decompose $\mathbf{s}$ into $\mathbf{s} = s_{\|}\mathbf{d} + \mathbf{s}_{\perp}$ where $\mathbf{d}$ is a unit vector parallel to $\mathbf{\Sigma_{uu}}^{\frac{1}{2}}\mathbf{x}_h$ and $\mathbf{s}_{\perp}$ is orthogonal to $\mathbf{d}$. This leads to a one-dimensional integral over $s_{\|}$

$$\left\langle g(\mathbf{u}^{\mathrm{T}}\mathbf{x}_h)\right\rangle_{q(\mathbf{u})} = \left\langle g(\sqrt{\mathbf{x}_h^{\mathrm{T}}\mathbf{\Sigma_{uu}}\mathbf{x}_h}s_{\|} + \bar{\mathbf{u}}^{\mathrm{T}}\mathbf{x}_h)\right\rangle_{\mathcal{N}(0,1)}, \tag{C.30}$$

and through application of the identity:

$$\frac{1}{\sqrt{2\pi}}\int g(ax+b)\exp\left(-\frac{1}{2}x^2\right)dx = g\left(\frac{b}{\sqrt{1+a^2}}\right), \tag{C.31}$$

we get the following result:

$$\left\langle f(\mathbf{x},\mathbf{w})\right\rangle_q = \sum_{h=1}^{H}\bar{v}_h g\left(\frac{\bar{\mathbf{u}}^{\mathrm{T}}\mathbf{x}_h}{\sqrt{1+\mathbf{x}_h^{\mathrm{T}}\mathbf{\Sigma_{uu}}\mathbf{x}_h}}\right). \tag{C.32}$$

For the expectation of the square, the approach is the same. We rotate the co-ordinate system so that $\mathbf{s} = s_1\mathbf{a} + s_2\mathbf{b} + \mathbf{s}_{\perp}$, where $\mathbf{a}$ and $\mathbf{b}$ are orthogonal. The arguments of the activation functions are independent of the component of $\mathbf{s}_{\perp}$. Barber and Bishop then treat the remaining integral by numerical means. However, if our covariance matrix is constrained as in Eqn 5.50, we may observe that $\mathbf{\Sigma_{uu}}^{\frac{1}{2}}\mathbf{x}_i$ and $\mathbf{\Sigma_{uu}}^{\frac{1}{2}}\mathbf{x}_j$ are orthogonal if $i \neq j$. This allows us not to take into account the correlations amongst the weights and to perform the integrations separately over each vector $\mathbf{u}_i$:

$$\begin{aligned}\left\langle f(\mathbf{x},\mathbf{w})^2\right\rangle_q &= \sum_{i\neq j}^{H}\left\{\bar{v}_i\bar{v}_j g\left(\frac{\bar{\mathbf{u}}^{\mathrm{T}}\mathbf{x}_i}{\sqrt{1+\mathbf{x}_i^{\mathrm{T}}\mathbf{\Sigma_{uu}}\mathbf{x}_i}}\right) g\left(\frac{\bar{\mathbf{u}}^{\mathrm{T}}\mathbf{x}_j}{\sqrt{1+\mathbf{x}_j^{\mathrm{T}}\mathbf{\Sigma_{uu}}\mathbf{x}_j}}\right)\right\} \\ &\quad + \sum_{h=1}^{H}\left(\bar{\mathbf{w}}_{\mathbf{v}_h}^2 + \mathbf{\Sigma_{v_h v_h}}\right)\left\langle g(\mathbf{u}^{\mathrm{T}}\mathbf{x}_h)^2\right\rangle_{\mathcal{N}(0,1)}\end{aligned} \tag{C.33}$$

The remaining integral $\left\langle g(\mathbf{u}_h^{\mathrm{T}}\mathbf{x})^2\right\rangle_{\mathcal{N}(0,1)}$ is still unfortunately analytically intractable. One can approximate this expectation by using lookup tables or numerical integration techniques, however a further solution, and the one preferred in our case, is to use a parametric approximation for this function. We choose to approximate $g(x)^2$ with the form $f(x) = 1 - \exp(-\frac{c}{2}x^2)$ where $c \approx 1.24$. The quality of the approximation is shown in Figure C.1. The expectation of this approximation may be calculated.

$$\left\langle g(\mathbf{u}^{\mathrm{T}}\mathbf{x}_h)^2\right\rangle_{q_u} \approx 1 - \frac{1}{\sqrt{1+c(\mathbf{x}_h^{\mathrm{T}}\mathbf{\Sigma_{uu}}\mathbf{x}_h)}}\exp\left\{\frac{-c(\bar{\mathbf{u}}^{\mathrm{T}}\mathbf{x}_h)^2}{2(1+c(\mathbf{x}_h^{\mathrm{T}}\mathbf{\Sigma_{uu}}\mathbf{x}_h))}\right\}. \tag{C.34}$$

We have now all the terms needed to compute the lower bound on the marginal log-likelihood.

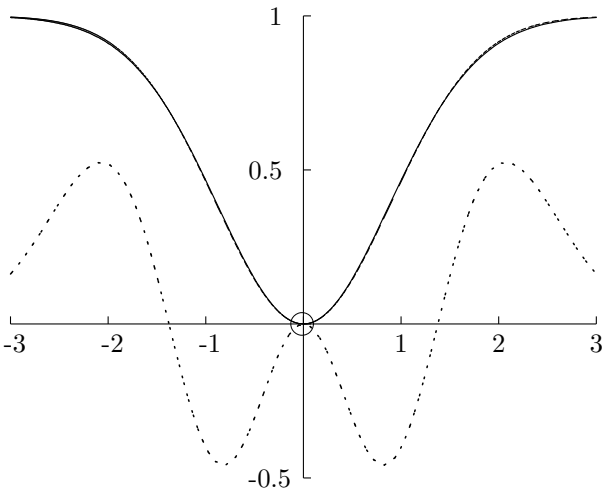


Figure C.1: The quality of the approximation to $g(x)^2$. The function $g(x)^2$ is shown as a solid line and the approximation $f(x)$ as a dashed line. The curves overlap at most points so the error is shown magnified a hundredfold as a dotted line.

C.3 Derivatives

We now give some of the required derivatives for the optimisation of our lower bound . We first present the derivatives of the lower bound C.13, for the general case of full covariance matrices

$$\begin{aligned}\frac{\partial \hat{S}(q_\alpha)}{\partial \bar{\mathbf{w}}_m} &= -Q_w(m)\boldsymbol{\Sigma}_{r_m}^{-1}(\bar{\mathbf{w}}_m - \bar{\mathbf{r}}_m) \\ &+ Q_w(m)\sum_{m'}\lambda_{m'}|\mathbf{A}_{m,m'}|^{-\frac{1}{2}}e^{-\frac{1}{2}\Delta(m,m')}(\boldsymbol{\Sigma}_{q_m}+\boldsymbol{\Sigma}_{r_{m'}})^{-1}(\bar{\mathbf{w}}_m - \bar{\mathbf{r}}_{m'}),\end{aligned} \tag{C.35}$$

$$\begin{aligned}\frac{\partial \hat{S}(q_\alpha)}{\partial \bar{\mathbf{r}}_{m'}} &= Q_w(m)\boldsymbol{\Sigma}_{r_{m'}}^{-1}(\bar{\mathbf{w}}_m - \bar{\mathbf{r}}_m) \\ &- \lambda_{m'}\sum_{m} Q_w(m)|\mathbf{A}_{m,m'}|^{-\frac{1}{2}}e^{-\frac{1}{2}\Delta(m,m')}(\boldsymbol{\Sigma}_{q_m}+\boldsymbol{\Sigma}_{r_{m'}})^{-1}(\bar{\mathbf{w}}_m - \bar{\mathbf{r}}_{m'}).\end{aligned} \tag{C.36}$$

Using some general matrix derivative rules, we get also the following equations:

$$\begin{aligned}\frac{\partial \hat{S}(q_\alpha)}{\partial \boldsymbol{\Sigma}_{q_m}} &= \frac{1}{2}Q_w(m)\Big\{\boldsymbol{\Sigma}_{q_m}^{-1} - \boldsymbol{\Sigma}_{r_m}^{-1} + \sum_{m'}\lambda_{m'}|\mathbf{A}_{m,m'}|^{-\frac{1}{2}}e^{-\frac{1}{2}\Delta(m,m')}\Big(\mathbf{A}_{m,m'}^{-1}\boldsymbol{\Sigma}_{r_{m'}}^{-1} \\ &- (\boldsymbol{\Sigma}_{q_m}+\boldsymbol{\Sigma}_{r_{m'}})^{-1}(\bar{\mathbf{w}}_m - \bar{\mathbf{r}}_{m'})^{\mathrm{T}}(\bar{\mathbf{w}}_m - \bar{\mathbf{r}}_{m'})(\boldsymbol{\Sigma}_{q_m}+\boldsymbol{\Sigma}_{r_{m'}})^{-1}\Big)\Big\}\end{aligned} \tag{C.37}$$

$$\begin{aligned}\frac{\partial \hat{S}(q_\alpha)}{\partial \boldsymbol{\Sigma}_{r_{m'}}} &= \frac{1}{2}\alpha_{m'}\Big\{\boldsymbol{\Sigma}_{r_{m'}}^{-1}\boldsymbol{\Sigma}_{q_{m'}}\boldsymbol{\Sigma}_{r_{m'}}^{-1} + \boldsymbol{\Sigma}_{r_{m'}}^{-1}(\bar{\mathbf{w}}_{m'} - \bar{\mathbf{r}}_{m'})^{\mathrm{T}}(\bar{\mathbf{w}}_{m'} - \bar{\mathbf{r}}_{m'})\boldsymbol{\Sigma}_{r_{m'}}^{-1}\Big\} \\ &- \frac{1}{2}\lambda_{m'}\Big\{\sum_{m} Q_w(m)|\mathbf{A}_{m,m'}|^{-\frac{1}{2}}e^{-\frac{1}{2}\Delta(m,m')}\Big(\mathbf{A}_{m,m'}^{-1}\boldsymbol{\Sigma}_{r_{m'}}^{-1}\boldsymbol{\Sigma}_{q_m}\boldsymbol{\Sigma}_{r_{m'}}^{-1} \\ &+ (\boldsymbol{\Sigma}_{q_m}+\boldsymbol{\Sigma}_{r_{m'}})^{-1}(\bar{\mathbf{w}}_m - \bar{\mathbf{r}}_{m'})^{\mathrm{T}}(\bar{\mathbf{w}}_m - \bar{\mathbf{r}}_{m'})(\boldsymbol{\Sigma}_{q_m}+\boldsymbol{\Sigma}_{r_{m'}})^{-1}\Big)\Big\}.\end{aligned} \tag{C.38}$$

Secondly we consider the derivatives of the output function expectations for the case of diagonal covariance Gaussian. The derivatives of $\langle g(\mathbf{u}^{\mathrm{T}}\mathbf{x}_h)^2\rangle_{q(\mathbf{u})}$ are similar and are omitted.

$$\begin{aligned}\frac{\partial\langle g(\mathbf{u}^{\mathrm{T}}\mathbf{x}_h)\rangle_{q(\mathbf{u})}}{\partial\bar{\mathbf{w}}_{\mathbf{u}_h}} &= \sqrt{\frac{2}{\pi}}\exp\left\{\frac{-(\bar{\mathbf{u}}^{\mathrm{T}}\mathbf{x}_h)^2}{2(1+\mathbf{x}_h^{\mathrm{T}}\boldsymbol{\Sigma}_{\mathbf{uu}}\mathbf{x}_h)}\right\}\cdot\frac{x_h}{(1+\mathbf{x}_h\boldsymbol{\Sigma}_{\mathbf{uu}}\mathbf{x}_h)^{\frac{1}{2}}} \\ \frac{\partial\langle g(\mathbf{u}^{\mathrm{T}}\mathbf{x}_h)\rangle_{q(\mathbf{u})}}{\partial\boldsymbol{\Sigma}_{\mathbf{u}_h}} &= \sqrt{\frac{2}{\pi}}\exp\left\{\frac{-(\bar{\mathbf{u}}^{\mathrm{T}}\mathbf{x}_h)^2}{2(1+\mathbf{x}_h^{\mathrm{T}}\boldsymbol{\Sigma}_{\mathbf{uu}}\mathbf{x}_h)}\right]\cdot\frac{(\bar{\mathbf{u}}^{\mathrm{T}}\mathbf{x})(-x_i^2\boldsymbol{\Sigma}_{\mathbf{u}_h})}{(1+\mathbf{x}_h^{\mathrm{T}}\boldsymbol{\Sigma}_{\mathbf{uu}}\mathbf{x}_h)^{\frac{3}{2}}}\end{aligned} \tag{C.39}$$

We have also omitted the equations relative to the derivatives with respect to the mixture coefficients $Q_w(m)$ and λ_m. These derivatives are straightforward to obtain (see Jaakkola and Jordan, 1998, Bishop *et al.*, 1998 and Lawrence *et al.*, 1998) and may be used to form fixed point update equations for these parameters. In our simulations we utilised fixed point equations for the mixing coefficients, but chose to perform gradient based optimisations in the λ_m parameters.

With the lower bound on the model likelihood, these derivatives may be used in a non-linear optimisation algorithm to find the parameters $\bar{\mathbf{w}}_m$, $\boldsymbol{\Sigma}_{q_m}$, $Q_\alpha(m)$, $\bar{\mathbf{r}}_{m'}$, $\boldsymbol{\Sigma}_{r_{m'}}$ and λ_m that represent the best fit for the mixture of Gaussians.

Appendix D

Source q-distribution for Bayesian ICA

First of all we note that the joint distribution factorises across the data-points and we may therefore take

$$q_x(\mathbf{X}) = \prod_{n=1}^{N} q_x^{(n)}(\mathbf{x}_n). \tag{D.1}$$

In Eqn 6.15 we need only take expectations of the components of $\ln p(D, \mathbf{W})$ which are in the Markov blanket of $\mathbf{x}_n$.

$$\ln q_x^{(n)}(\mathbf{x}) = \text{const} - \left\langle \frac{\beta}{2} \left\| \mathbf{t}_n - \mathbf{W}\mathbf{x}_n - \boldsymbol{\mu} \right\|^2 \right\rangle_{q_w q_\beta q_\mu} + \sum_{i=1}^{I} \ln(p(\mathbf{x}_n)) \tag{D.2}$$

where $p(\mathbf{x})$ is the latent variable model which we took to be a factorised distribution in which each factor is a mixture of M Gaussians

$$p(\mathbf{x}_n) = \prod_{i=1}^{I} \left[\sum_{m=1}^{M} \pi_m \mathcal{N}(x_{ni} | m_m, \sigma_m^2) \right]. \tag{D.3}$$

The second term of Eqn D.2 may be expanded and rewritten in terms of the expectations of each parameter

$$-\frac{\langle\beta\rangle}{2} \mathbf{x}_n^{\mathrm{T}} \left\langle \mathbf{W}^{\mathrm{T}}\mathbf{W} \right\rangle \mathbf{x}_n + \langle\beta\rangle \mathbf{x}_n^{\mathrm{T}} \langle \mathbf{W} \rangle \left(\mathbf{t}_n - \langle\boldsymbol{\mu}\rangle \right)^{\mathrm{T}} \equiv A(\mathbf{x}_n). \tag{D.4}$$

Exponentiating both sides we obtain

$$q_x^{(n)}(\mathbf{x}_n) = \text{const} \,.\, \exp[A(\mathbf{x}_n)] \prod_{i=1}^{I} \left[\sum_{m=1}^{M} \pi_m \mathcal{N}(x_{ni} | m_m, \sigma_m^2) \right]. \tag{D.5}$$

The product of the sums of Gaussians may be rewritten in terms of sums of products of Gaussians

$$\prod_{i=1}^{I} \left[\sum_{m=1}^{M} \pi_m \mathcal{N}_{im} \right] = \sum_{m_1=1}^{M} \cdots \sum_{m_I=1}^{M} \left[\prod_{i=1}^{I} \pi_{m_i} \mathcal{N}_{im_i} \right]. \tag{D.6}$$

This means $q_x^{(n)}(\mathbf{x}_n)$ is a mixture of Gaussians with M^I terms

$$\begin{aligned} q_x^{(n)}(\mathbf{x}_n) &= \text{const} \cdot \exp\{A(\mathbf{x}_n)\} \left[\sum_{m_1=1}^{M} \cdots \sum_{m_I=1}^{M} \left(\prod_{i=1}^{I} \pi_{m_i} \mathcal{N}_{im_i} \right) \right] \\ &= \text{const} \cdot \sum_{m_1=1}^{M} \cdots \sum_{m_I=1}^{M} \pi_{\mathbf{m}} \exp\left\{ A(\mathbf{x}_n) - \sum_{i=1}^{I} \frac{1}{2\sigma_{m_i}^2} (x_{ni} - m_{m_i})^2 \right\} \end{aligned} \tag{D.7}$$

where $\pi_{\mathbf{m}} = \prod_{i=1}^{I} \pi_{m_i}$. The argument of the exponential may be rewritten as

$$-E_{q_x} \equiv A(\mathbf{x}_n) - \frac{1}{2}(\mathbf{x}_n - \mathbf{m}_{\mathbf{m}})^{\mathrm{T}} \mathbf{B}_{\mathbf{m}}^{-1} (\mathbf{x}_n - \mathbf{m}_{\mathbf{m}}) \tag{D.8}$$

where $\mathbf{B}_{\mathbf{m}} = \text{diag}(\sigma_{m_i}^2)$ and $\mathbf{m}_{\mathbf{m}} = \{m_{m_i}\}^{\mathrm{T}}$. Again we rewrite E_{q_x}

$$\begin{aligned} E_{q_x} &= \frac{1}{2}(\mathbf{x}_n - \mathbf{f}_{\mathbf{m}}^{(n)})^{\mathrm{T}} \mathbf{D}_{\mathbf{m}}^{-1} (\mathbf{x}_n - \mathbf{f}_{\mathbf{m}}^{(n)}) \\ &\quad + \frac{1}{2} \mathbf{m}_{\mathbf{m}}^{\mathrm{T}} \mathbf{B}_{\mathbf{m}}^{-1} \mathbf{m}_{\mathbf{m}} - \frac{1}{2} \mathbf{f}_{\mathbf{m}}^{(n)^{\mathrm{T}}} \mathbf{D}_{\mathbf{m}}^{-1} \mathbf{f}_{\mathbf{m}}^{(n)} \end{aligned} \tag{D.9}$$

where $\mathbf{D}_{\mathbf{m}}$ and $\mathbf{f}_{\mathbf{m}}$ are defined as

$$\mathbf{D}_{\mathbf{m}} = \left(\langle \beta \rangle \left\langle \mathbf{W}^{\mathrm{T}} \mathbf{W} \right\rangle + \mathbf{B}_{\mathbf{m}}^{-1} \right)^{-1} \tag{D.10}$$

$$\mathbf{f}_{\mathbf{m}}^{(n)} = \left(\langle \beta \rangle \left\langle \mathbf{W}^{\mathrm{T}} \mathbf{W} \right\rangle + \mathbf{B}_{\mathbf{m}}^{-1} \right)^{-1} \left[\langle \beta \rangle \left\langle \mathbf{W}^{\mathrm{T}} \right\rangle (\mathbf{t}_n - \langle \boldsymbol{\mu} \rangle) + \mathbf{B}_{\mathbf{m}}^{-1} \mathbf{m}_{\mathbf{m}} \right]. \tag{D.11}$$

Thus each factor of the variational distribution is:

$$q_x^{(n)}(\mathbf{x}_n) = \frac{1}{Z_n} \sum_{m_1=1}^{M} \cdots \sum_{m_I=1}^{M} \tilde{\pi}_{\mathbf{m}}^{(n)} \mathcal{N}(x_n | \mathbf{f}_{\mathbf{m}}^{(n)}, \mathbf{D}_{\mathbf{m}}), \tag{D.12}$$

where

$$\tilde{\pi}_{\mathbf{m}}^{(n)} = \frac{\pi_{\mathbf{m}} |\mathbf{D}_{\mathbf{m}}|^{1/2} e^{-\frac{1}{2} \mathbf{m}_{\mathbf{m}}^{\mathrm{T}} \mathbf{B}_{\mathbf{m}}^{-1} \mathbf{m}_{\mathbf{m}}} e^{\frac{1}{2} \mathbf{f}_{\mathbf{m}}^{(n)^{\mathrm{T}}} \mathbf{D}_{\mathbf{m}}^{-1} \mathbf{f}_{\mathbf{m}}^{(n)}}}{|\mathbf{B}_{\mathbf{m}}|^{\frac{1}{2}}} \tag{D.13}$$

$$Z_n = \sum_{m_1=1}^{M} \cdots \sum_{m_I=1}^{M} \tilde{\pi}_{\mathbf{m}}^{(n)}. \tag{D.14}$$

The required moments are evaluated to give

$$\langle \mathbf{x}_n \rangle = \frac{1}{Z_n} \sum_{m_1=1}^{M} \cdots \sum_{m_I=1}^{M} \tilde{\pi}_{\mathbf{m}}^{(n)} \mathbf{f}_{\mathbf{m}}^{(n)} \tag{D.15}$$

$$\left\langle \mathbf{x}_n \mathbf{x}_n^{\mathrm{T}} \right\rangle = \frac{1}{Z_n} \sum_{m_1=1}^{M} \cdots \sum_{m_I=1}^{M} \tilde{\pi}_{\mathbf{m}}^{(n)} \left(\mathbf{D}_{\mathbf{m}} + \mathbf{f}_{\mathbf{m}}^{(n)} \mathbf{f}_{\mathbf{m}}^{(n)\mathrm{T}} \right). \tag{D.16}$$

References

Ackley, D., G. E. Hinton, and T. J. Sejnowski (1985). A learning algorithm for Boltzmann machines. *Cognitive Science* **9**, 147–169.

Anderson, J. A. and E. Rosenfeld (Eds.) (1988). *Neurocomputing: Foundations of Research*, Cambridge, MA. MIT Press.

Appel, K. and W. Haken (1977). Every planar map is four colorable. Part I. Discharging. *Illinois Journal of Mathematics* **21**, 429–490.

Appel, K., W. Haken, and J. Koch (1977). Every planar map is four colorable. Part II. Reducibility. *Illinois Journal of Mathematics* **21**, 491–567.

Attias, H. (1998). Independent factor analysis. *Neural Computation* **11**, 803–851.

Barber, D. and C. M. Bishop (1998). Ensemble learning in Bayesian neural networks. In C. M. Bishop (Ed.), *Neural Networks and Machine Learning*, Volume 168 of *Series F: Computer and Systems Sciences*, pp. 215–237. Berlin: Springer-Verlag.

Barber, D. and C. K. I. Williams (1997). Gaussian processes for Bayesian classification via hybrid Monte Carlo. In M. C. Mozer, M. I. Jordan, and T. Petsche (Eds.), *Advances in Neural Information Processing Systems*, Volume 9. Cambridge, MA: MIT Press.

Bell, A. J. and T. J. Sejnowski (1995). An information maximization approach to blind separation and blind deconvolution. *Neural Computation* **7** (6), 1129–1159.

Bhadeshia, H. K. D. H., D. J. C. MacKay, and L. E. Svensson (1995). Impact toughness of C-MN steel arc welds - Bayesian neural network analysis. *Materials Science and Technology* **11** (10), 1046–1051.

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

Bishop, C. M. (1999). Bayesian PCA. In M. J. Kearns, S. A. Solla, and D. A. Cohn (Eds.), *Advances in Neural Information Processing Systems*, Volume 11, pp. 482–388. Cambridge, MA: MIT Press.

Bishop, C. M., N. D. Lawrence, T. S. Jaakkola, and M. I. Jordan (1998). Approximating posterior distributions in belief networks using mixtures. In M. I. Jordan, M. J. Kearns, and S. A. Solla (Eds.), *Advances in Neural Information Processing Systems*, Volume 10, pp. 416–422. Cambridge, MA: MIT Press.

Burges, C. J. C. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery* **2** (2), 121–167.

Cardoso, J.-F. (1997). Infomax and maximum likelihood for blind source separation. *IEEE Letters on Signal Processing* **4** (4), 112–114.

Cayley, A. (1879). On the colourings of maps. *Proceedings of the Royal Geographical Society* **1**, 259–261.

Cheeseman, P. and J. Stutz (1996). Bayesian classification (AutoClass): Theory and results. In U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy (Eds.), *Advances in Knowledge Discovery and Data Mining*, pp. 153–180. Cambridge, MA: MIT Press.

Chickering, D. M., D. Geiger, and D. Heckerman (1994). Learning Bayesian networks is NP-hard. Technical Report MSR-TR-94-17, Microsoft Research. Available from `http://research.microsoft.com/~heckerman/`.

Chickering, D. M. and D. Heckerman (1996). Efficient approximations for the marginal likelihood of Bayesian networks with hidden variables. In E. Horvitz and F. V. Jensen (Eds.), *Uncertainty in Artificial Intelligence*, Volume 12, pp. 158–168. San Fransisco, CA: Morgan Kauffman.

Cool, T., H. K. D. H. Bhadeshia, and D. J. C. MacKay (1997). Modelling the mechanical properties in the HAZ of power plant steels i: Bayesian neural network analysis of proof strength. In H. Cerjak (Ed.), *Mathematical Modelling of Weld Phenomena 3*, Materials Modelling Series, pp. 403–441. London: Institute of Materials.

Cooper, G. F. (1990). The computational complexity of using probabilistic inference using Bayesian belief networks. *Artificial Intelligence* **42**, 393–405.

Cooper, G. F. and S. Moral (Eds.) (1998). *Uncertainty in Artificial Intelligence*, Volume 14, San Fransisco, CA. Morgan Kauffman.

Cowell, R. (1998). Introduction to inference for Bayesian networks. See Jordan (1998), pp. 9–26.

Cox, R. T. (1946). Probability, frequency and reasonable expectation. *American Journal of Physics* **14** (1), 1–13.

Crevier, D. (1993). *AI. The Tumultuous History of the Search for Artificial Intelligence*. New York: Basic Books.

Dagum, P. and M. Luby (1993). Approximating probabilistic inference in Bayesian belief networks is NP-hard. *Artificial Intelligence* **60**, 141–153.

Dempster, A. P., N. M. Laird, and D. B. Rubin (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, B* **39** (1), 1–38.

Duhaime, L. (1998). Legal dictionary. `http://www.fifthdistrictcourt.com/dictionary/legal.htm`.

Euler, L. (1736). Solutio problematis ad geometriam situs pertinentis. *Commentarii Academiae Scientiarum Imperialis Petropolitanae* **8**, 128–140.

Frey, B. J., N. D. Lawrence, and C. M. Bishop (1998). Markovian inference in belief networks. Technical report, The Beckman Institute, University of Illinois at Urbana-Champaign, 405 North Mathews Avenue, Urbana, IL 61801, USA. Available from `http://www.cs.toronto.edu/~frey/fg/fg.html`.

Fujii, H., D. J. C. MacKay, and H. K. D. H. Bhadeshia (1996). Bayesian neural network analysis of fatigue crack growth rate in nickel base superalloys. *ISIJ International* **36** (11), 1373–1382.

Galland, C. C. (1993). The limitations of deterministic Boltzmann machines. *Network: Computation in Neural Systems* **4** (3), 355–379.

Gavard, L., H. K. D. H. Bhadeshia, D. J. C. MacKay, and S. Suzuki (1996). Bayesian neural network model for Austentite formation in steels. *Materials Science and Technology* **12**, 453–463.

Geman, S., E. Bienenstock, and R. Doursat (1992). Neural networks and the bias/variance dilema. *Neural Computation* **4** (1), 1–58.

Geman, S. and D. Geman (1984). Stochastic relaxation, Gibbs distributions, and Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **6** (6), 721–741.

General Assembly of United Nations (1948). Universal declaration of human rights. `http://www.un.org/Overview/rights.html`.

Ghahramani, Z. and M. I. Jordan (1997). Factorial hidden markov models. *Machine Learning* **29**, 245–273.

Gorry, G. A. and G. O. Barnett (1968). Experience with a model of sequential diagnosis. *Computers and Biomedical Research* **1**, 490–507.

Gull, S. F. (1988). Bayesian inductive inference and maximum entropy. In G. J. Erickson and C. R. Smith (Eds.), *Maximum-Entropy and Bayesian Methods in Science and Engineering*, Volume 1: Foundations, pp. 53–74. Dordrecht, The Netherlands: Kluwer.

Haft, M., R. Hoffmann, and V. Tresp (1999). Model-independent mean field theory as a local method for approximate propagation of information. *Network: Computation in Neural Systems* **10**, 93–105.

Haykin, S. (1999). *Neural Networks: A Comprehensive Foundation* (2nd ed.). Upper Saddle River, NJ: Prentice Hall.

Heckerman, D. (1986). Probabilistic interpretation for MYCIN's certainty factors. In L. N. Kanal and J. F. Lemmer (Eds.), *Uncertainty in Artificial Intelligence*, pp. 167–196. North-Holland, New York: Elsevier Science Publishers.

Heckerman, D. and E. Horvitz (1998). Inferring informational goals from free-text queries. See Cooper and Moral (1998).

Heisenberg, W. (1927). Uber den anschaulichen Inhalt der quantentheoretischen Kinematik und Mechanik. *Zeitschrift für Physik* **43**, 172–198. Reprinted in Wheeler and Zurek (1983).

Hinton, G. E. (1989). Deterministic Boltzmann machine performs steepest descent in weight-space. *Neural Computation* **1** (1), 143–150.

Hinton, G. E. (1999). Products of experts. In *ICANN 99: Ninth international conference on artificial neural networks*, Volume 1, pp. 1–6. IEE Press.

Hinton, G. E., P. Dayan, B. J. Frey, and R. M. Neal (1995). The wake-sleep algorithm for unsupervised neural networks. *Science* **268**, 1158–1161.

Hinton, G. E. and T. J. Sejnowski (1983). Optimal perceptual inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (Washington, 1983)*, pp. 448–453. New York: IEEE Press.

Hinton, G. E. and D. van Camp (1993). Keeping neural networks simple by minimizing the description length of the weights. In *Proceedings of the Sixth Anuual Conference on Computational Learning Theory*, pp. 5–13.

Hopcroft, J. and U. Ullman (1979). *Introduction to Automata Theory, Languages and Computation*. Reading, MA: Addison-Wesley.

Horvitz, E., J. Breese, D. Heckerman, D. Hovel, and K. Rommelse (1998). The lumiere project: Bayesian user modeling for inferring the goals and needs of software users. See Cooper and Moral (1998).

Hyvärinen, A. and E. Oja (1997). A fast fixed-point algorithm for independent component analysis. *Neural Computation* **9**, 1483–1492.

Jaakkola, T. S. (1997). *Variational Methods for Inference and Estimation in Graphical Models*. Ph.D. thesis, MIT.

Jaakkola, T. S. and M. I. Jordan (1998). Improving the mean field approximation via the use of mixture distributions. See Jordan (1998), pp. 163–174.

Jensen, F. V. (1996). *An Introduction to Bayesian Networks*. London: UCL Press.

Jordan, M. I. (Ed.) (1998). *Learning in Graphical Models*, Volume 89 of *Series D: Behavioural and Social Sciences*, Dordrecht, The Netherlands. Kluwer.

Jordan, M. I., Z. Ghahramani, T. S. Jaakkola, and L. K. Saul (1998). An introduction to variational methods for graphical models. See Jordan (1998), pp. 105–162.

Kappen, H. J. and F. B. Rodriguez (1998). Efficient learning in Boltzmann machines using linear response theory. Technical report, Department of Biophysics, University of Nijmegen.

König, D. (1936). *Theorie der Endlichen und Unendlichen Graphen*. Leipzig: Teubner. Reprinted by Chelsea in 1965.

Kullback, S. (1959). *Information Theory and Statistics*. New York: Dover Publications.

Kullback, S. and R. A. Leibler (1951). On information and sufficiency. *Annals of Mathematical Statistics* **22**, 79–86.

Lappalainen, H. (1999). Ensemble learning for independent component analysis. In *Proceedings of the First International Workshop on Independent Component Analysis and Blind Signal Separation*, pp. 7–12.

Lawrence, N. D. and M. Azzouzi (1999). A variational Bayesian committee of neural networks. Submitted to *Neural Networks*.

Lawrence, N. D., C. M. Bishop, and M. I. Jordan (1998). Mixture representations for inference and learning in Boltzmann machines. See Cooper and Moral (1998), pp. 320–327.

Lawrence, N. D. and B. Lerner (2000). The use of implicit solutions for approximating posteriors in directed acyclic graphs. Work in progress.

Legal Lexicon's Lyceum (1999). The legal lexicon's lyceum. `http://www.lectlaw.com/def.htm`.

Lerner, B. and N. D. Lawrence (1999). A comparison of state-of-the-art classification techniques with application to cytogenetics. Submitted to *Neural Computing and Applications*.

Lindley, D. V. (1980). Approximate Bayesian methods. In J. M. Bernardo, M. H. DeGroot, D. V. Lindley, and A. F. M. Smith (Eds.), *Bayesian Statistics*, pp. 223–237. Valencia: Valencia University Press.

MacKay, D. J. C. (1992). A practical Bayesian framework for back-propagation networks. *Neural Computation* **4** (3), 448–472.

MacKay, D. J. C. (1995a). Developments in probabilistic modelling with neural networks—ensemble learning. In *Neural Networks: Artificial Intelligence and Industrial Applications. Proceedings of the 3rd Annual Symposium on Neural Networks, Nijmegen, Netherlands, 14-15 September 1995*, pp. 191–198. Berlin: Springer.

MacKay, D. J. C. (1995b). Probable networks and plausible predictions – a review of practical Bayesian methods for supervised neural networks. *Network: Computation in Neural Systems* **6** (3), 469–505.

MacKay, D. J. C. (1996). Maximum likelihood and covariant algorithms for independent component analysis. Unpublished manuscript, available from `http://wol.ra.phy.cam.ac.uk/mackay/homepage.html`.

MacKay, D. J. C. (1997). Ensemble learning for hidden Markov models. Unpublished manuscript, available from `http://wol.ra.phy.cam.ac.uk/mackay/homepage.html`.

MacKay, D. J. C. (1998). Introduction to Monte Carlo methods. See Jordan (1998), pp. 175–204.

McCulloch, W. S. and W. Pitts (1943). A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics* **5**, 115–133. Reprinted in Anderson and Rosenfeld (1988).

McDermott, J. (1982). R1: A rule based configurer of computer systems. *Artificial Intelligence* **19**, 39–88.

Metropolis, N., A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller (1953). Equation of state calculations by fast computing machines. *Journal of Chemical Physics* **21** (6), 1087–1092.

Minsky, M. L. and S. A. Papert (1969). *Perceptrons*. Cambridge, MA: MIT Press. Expanded Edition 1990, partially reprinted in Anderson and Rosenfeld (1988).

Muggleton, S. and L. de Raedt (1994). Inductive logic programming: Theory and methods. *Journal of Logic Programming* **19,20**, 629–679.

Nabney, I. T. (1998). NETLAB neural network software. Available from `http://www.ncrg.aston.ac.uk/netlab/`.

Neal, R. M. (1992). Connectionist learning of belief networks. *Artificial Intelligence* **56**, 71–113.

Neal, R. M. (1993). Probabilistic inference using Markov chain Monte Carlo methods. Technical Report CRG-TR-93-1, Department of Computer Science, University of Toronto, Cananda.

Neal, R. M. (1996). *Bayesian Learning for Neural Networks*. Springer. Lecture Notes in Statistics 118.

Neal, R. M. (1999). Software for flexible Bayesian modeling and Markov chain sampling. Available from `http://www.cs.utoronto.ca/~radford/fbm.software.html`.

of Ockham, W. (1496). *Expositio aurea super totam artem veterem*. written c. 1320, first printed in Bologna 1496. Commentaries on Aristotle's *Elenchus* and Porphyry's *Isagoge*.

Parisi, G. (1988). *Statistical Field Theory*. Redwood City, CA: Addison-Wesley.

Pearl, J. (1986). Fusion, propagation, and structuring in belief networks. *Artificial Intelligence* **29**, 241–288.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems*. San Francisco: Morgan Kaufmann.

Pearl, J. (1995). From Bayesian networks to causal networks. In A. Gammerman (Ed.), *Probabilistic Reasoning and Bayesian Belief Networks*, pp. 1–31. Alfred Waller.

Peterson, C. and J. R. Anderson (1987). A mean field learning algorithm for neural networks. *Complex Systems* **1**, 995–1019.

Platt, J. C. (1998). Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C. J. C. Burges, and A. J. Smola (Eds.), *Advances in Kernel Methods: Support Vector Learning*, pp. 185–208. Cambridge, MA: MIT Press.

Raftery, A. E. (1996). *Hypothesis Testing and Model Selection*, Chapter 10, pp. 164–187. Chapman and Hall.

Ripley, B. D. (1994). Flexible non-linear approaches to classification. In V. Cherkassky, J. H. Friedman, and H. Wechsler (Eds.), *From Statistics to Neural Networks. Theory and Pattern Recognition Applications*, Series F: Computer and Systems Sciences, pp. 105–126. Springer-Verlag.

Ripley, B. D. (1996). *Pattern Recognition and Neural Networks*. Cambridge, U.K.: Cambridge University Press.

Rosenblatt, F. (1962). *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. Spartan.

Rumelhart, D. E., G. E. Hinton, and R. J. Williams (1986). Learning internal representations by error propagation. In *Parallel Distributed Programming: Explorations in the Microstructure of Cognition*, Volume 1: Foundations, pp. 318–362. Cambridge, MA: MIT Press. Reprinted in Anderson and Rosenfeld (1988).

Sage, A. P. (Ed.) (1990). *Concise Encyclopedia of Information Processing in Systems and Organizations*. New York, NY: Pergamon.

Sahami, M., S. Dumais, D. Heckerman, and E. Horvitz (1998, July). A Bayesian approach to filtering junk e-mail. In *AAAI Workshop on Learning for Text Categorization*. Available from `http://research.microsoft.com/~heckerman/`.

Saul, L. K., T. S. Jaakkola, and M. I. Jordan (1996). Mean field theory for sigmoid belief networks. *Journal of Artificial Intelligence Research* **4**, 61–76.

Shortcliffe, E. H. (1976). *Computer-Based Medical Consultations: MYCIN*. North-Holland, New York: Elsevier Science Publishers.

Spiegelhalter, D. J., A. Thomas, N. G. Best, and W. R. Gilks (1996). Bugs: Bayesian inference using Gibbs sampling, version 0.5, (version ii). Available from `http://www.mrc-bsu.cam.ac.uk/bugs/`.

Stone, M. (1974). Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society, B* **36** (1), 111–147.

Stone, M. (1978). Cross-validation: A review. *Math. Operationsforsch. Statist. Ser. Statistics* **9** (1), 127–139.

Thodberg, H. H. (1996). A review of Bayesian neural networks with an application to near infrared spectroscopy. *IEEE Transactions on Neural Networks* **7** (1), 56–72.

Tipping, M. E. and C. M. Bishop (1999). Probabilistic principal component analysis. *Journal of the Royal Statistical Society, Series B* **6** (3), 611–622.

Tong, H. (1995). *Non-linear Time Series: a Dynamical System Approach*, Volume 6 of *Oxford Statistical Science Series*. Oxford: Clarendon Press.

Vapnik, V. N. (1982). *Estimation of Dependences Based on Empirical Data*. New York: Springer-Verlag.

Vapnik, V. N. and A. Y. Chervonenkis (1971). On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications* **16** (2), 264–280.

Wahba, G. and S. Wold (1975). A completely automatic French curve: fitting spline functions by cross-validation. *Communications in Statistics, Series A* **4** (1), 1–17.

Waugh, F. R., C. M. Marcus, and R. M. Westervelt (1990). Fixed-point attractors in analog neural computation. *Physical Review Letters* **64** (16), 1986–1989.

Weigend, A. S., B. A. Huberman, and D. E. Rumelhart (1990). Predicting sunspots and exchange rates with connectionist networks. In S. Eubank and M. Casdagli (Eds.), *Proceedings of the 1990 NATO Workshop on Nonlinear Modeling and Forcasting, Santa Fe, New Mexico*. Addison-Wesley.

Wheeler, J. A. and W. H. Zurek (Eds.) (1983). *Quantum Theory and Measurement*, Princeton, NJ. Princeton University Press.

Williams, C. K. I. (1997). Regression with Gaussian processes. In S. W. Ellacott, J. C. Mason, and I. J. Anderson (Eds.), *Mathematics of Neural Networks: Models, Algorithms and Applications*. Dordrecht, The Netherlands: Kluwer. Paper presented at the *Mathematics of Neural Networks and Applications conference*, Oxford, UK, July 1995.

Williams, C. K. I. and C. E. Rasmussen (1996). Gaussian processes for regression. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo (Eds.), *Advances in Neural Information Processing Systems*, Volume 8, pp. 514–520. Cambridge, MA: MIT Press.