Bayesian Methods for Neural Networks

Christopher M. Bishop
Neural Computing Research Group
Dept. of Computer Science and Applied Mathematics
Aston University, Birmingham, B4 7ET, U.K.
C.M.Bishop@aston.ac.uk

Technical Report: NCRG/95/009
Available from: http://www.ncrg.aston.ac.uk/

Abstract

Bayesian techniques have been developed over many years in a range of different fields, but have only recently been applied to the problem of learning in neural networks. As well as providing a consistent framework for statistical pattern recognition, the Bayesian approach offers a number of practical advantages including a potential solution to the problem of over-fitting. This chapter aims to provide an introductory overview of the application of Bayesian methods to neural networks. It assumes the reader is familiar with standard feed-forward network models and how to train them using conventional techniques.

1 Introduction

Conventional approaches to network training are based on the minimization of an error function, which itself might be derived from some underlying principle such as maximum likelihood (see Ripley (1994, 1995) and also the chapter by Ripley in this book). Such approaches can suffer from a number of deficiencies, for example the problem of determining the appropriate level of model complexity. More complex models (e.g. ones with more hidden units or with smaller values of regularization parameters) give better fits to the training data, but if the model is too complex it may give poor generalization (the phenomenon of over-fitting). The usual approach is to set aside data to form a validation set and to optimize the model complexity to give the best validation set performance.

The Bayesian viewpoint provides a general and consistent framework for statistical pattern recognition and data analysis. In the context of neural networks, a Bayesian approach offers several important features including the following:

- The conventional approach to networks training, based on the minimization of an error function, can be seen as a specific approximation to a full Bayesian treatment.
- Similarly, the technique of regularization arises in a natural way in the Bayesian framework.
- For regression problems, error bars, or confidence intervals, can be assigned to the predictions generated by a network.

- For classification problems, the tendency of conventional approaches to make overconfident predictions in regions of sparse data can be avoided.
- Bayesian methods allow the values of regularization coefficients to be selected using only the training data, without the need to set data aside in a validation set. Thus the Bayesian approach avoids the problem of over-fitting which occurs in conventional approaches to network training. Furthermore, it allows relatively large numbers of regularization coefficients to be used, which would be computationally prohibitive if their values had to be optimized using cross-validation.
- Similarly, the Bayesian approach allows different models (e.g. networks with different numbers of hidden units, or different network types such as multi-layer perceptrons and radial basis function networks) to be compared using only the training data. More generally, it provides an objective and principled framework for dealing with the issue of model complexity, and avoids many of the problems which arise when using maximum likelihood.

In this chapter we give an introductory account of Bayesian methods and their application to neural networks. The focus here is on underlying principles rather than mathematical details. A more comprehensive introduction to the Bayesian treatment of neural networks can be found in Chapter 10 of Bishop (1995).

1.1 Bayes' Theorem

We are quite used to the idea of dealing with uncertainty in our everyday lives. For example, we might believe that it is unlikely to rain tomorrow if the last few days have been sunny. However, if we then discover that a cold front is about to arrive, we might revise our views and decide that it is in fact quite likely to rain. Here we are discussing subjective beliefs, and the way they are modified when we obtain more information. We might seek to put such reasoning on a more formal footing, and to quantify our uncertainty by encoding the degrees of belief as real numbers. In a key paper, Cox (1946) showed that, provided we impose some simple consistency requirements, then these numbers obey the rules of conventional probability theory. In other words, if we use a value of 1 to denote complete certainty that an event will occur, and 0 to denote complete certainty that the event will not occur (with intermediate values representing corresponding degrees of belief), then these real values behave exactly like conventional probabilities.

Once our beliefs have been represented as probabilities they can be manipulated using two simple rules. Consider a pair of random variables A and B each of which can take on a number of discrete values. We denote by $P(a, b)$ the *joint* probability that $A = a$ *and* $B = b$. Using the *product rule* this joint probability can be expressed in the form

$$P(a, b) = P(b|a)P(a) \tag{0.1}$$

Here $P(b|a)$ denotes the *conditional* probability, in other words the probability that $B = b$ *given* that $A = a$. We can similarly consider a conditional probability of the form $P(a|b)$. The quantity $P(a)$ in (0.1) denotes the *marginal* probability, in other words the probability that $A = a$ irrespective of the value of B. The second relation between probabilities that we need to consider is the *sum rule* given by

$$\sum_b P(a, b) = P(a) \tag{0.2}$$

where the sum is over all possible values of b. From the product rule we obtain the following relation

$$P(a|b) = \frac{P(b|a)P(a)}{P(b)} \tag{0.3}$$

which is known as *Bayes' theorem*. Using the sum rule, we see that the denominator in (0.3) is given by

$$P(b) = \sum_a P(b|a)P(a) \tag{0.4}$$

and plays the role of a normalizing factor, ensuring that the probabilities on the left hand side of (0.3) sum to one. For continuous rather than discrete variables, the probabilities are replaced by probability density functions, and summations are replaced by integrations.

We can consider $P(a)$ to be the *prior* probability of $A = a$ before we observe the value of B, and $P(a|b)$ to be the corresponding *posterior probability* after we have observed the value of B. Posterior probabilities play a central role in pattern recognition, and Bayes' theorem allows them to be re-expressed in terms of quantities which may be more easily calculated.

As we shall see, we can treat the problem of learning in neural networks from a Bayesian perspective simply by application of the above rules of probability. This leads to a unique formalism which is principle simple to apply, and which can lead to some very powerful results. We shall also see, however, that the application of Bayesian inference to realistic problems presents many difficulties which require careful analytical approximations or sophisticated numerical approaches to resolve.

1.2 Model Comparison

As we have already indicated, a Bayesian approach allows the model complexity issue to be tackled using only the training data. To gain some insight into how this comes about, consider a hypothetical example of three different models, $\mathcal{H}_1$, $\mathcal{H}_2$ and $\mathcal{H}_3$, which we suppose have steadily increasing flexibility, corresponding for instance to a steadily increasing number of hidden units. Thus, each model consists of a specification of the network architecture (number of units, type of activation function, etc.) and is governed by a number of adaptive parameters. By varying the values of these parameters, each model can represent a range of input–output functions. The more complex models, with a greater number of hidden units for instance, can represent a greater variety of such functions. We can find the relative probabilities of the different models, given a data set D,

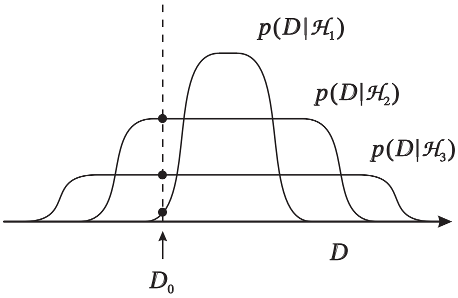

FIG. 1. Schematic example of three models, $\mathcal{H}_1$, $\mathcal{H}_2$ and $\mathcal{H}_3$, which have successively greater complexity, showing the probability (known as the *evidence*) of different data sets D given each model $\mathcal{H}_i$. We see that more complex models can describe a greater range of data sets. Note, however, that the distributions are normalized. Thus, when a particular data set D_0 is observed, the model $\mathcal{H}_2$ has a greater evidence than either the simpler model $\mathcal{H}_1$ or the more complex model $\mathcal{H}_3$.

using Bayes' theorem in the form

$$p(\mathcal{H}_i|D) = \frac{p(D|\mathcal{H}_i)p(\mathcal{H}_i)}{p(D)} \tag{0.5}$$

The quantity $p(\mathcal{H}_i)$ represents a prior probability for model $\mathcal{H}_i$. If we have no particular reason to prefer one model over another, then we would assign equal priors to all of the models. Since the denominator $p(D)$ does not depend on the model, we see that different models can be compared by evaluating $p(D|\mathcal{H}_i)$, which is called the *evidence* for the model $\mathcal{H}_i$ (MacKay, 1992a). This is illustrated schematically in Figure 1, where we see that the evidence favours models which are neither too simple nor too complex.

1.3 Marginalization

An important concept in Bayesian inference is that of *marginalization*, which involves integrating out unwanted variables. Suppose we are discussing a model with two variables w and t. Then the most complete description of these variables is in terms of the joint distribution $p(t, w)$. If we are interested only in the distribution of t then we should integrate out w as follows:

$$\begin{aligned} p(t) &= \int p(t, w)\, dw \\ &= \int p(t|w)p(w)\, dw \end{aligned} \tag{0.6}$$

Thus the predictive distribution for t is obtained by averaging the conditional distribution $p(t|w)$ with a weighting factor given by the distribution $p(w)$. We shall encounter several examples of marginalization later in this chapter.

2 Regression Problems

In this section we discuss the application of Bayesian methods to 'regression', i.e. the prediction of the values of continuous output variables given the values of some input variables. The application of Bayesian methods to classification problems is described in MacKay (1992b). We consider a feed-forward network model (for example a multi-layer perceptron) which maps an input vector $\mathbf{x}$ to an output value[1] y, and which is governed by a vector $\mathbf{w}$ of adaptive parameters (weights and biases). The observed dataset D consists of N input vectors $\mathbf{x}^n$ and corresponding targets t^n, where $n = 1, \ldots, N$.

2.1 Distribution of Weights

We begin by finding the distribution function $p(\mathbf{w}|D)$ of the weight vector $\mathbf{w}$ once we have observed the dataset D. Note that this description of our state of knowledge of the weights, in terms of a probability distribution, is in contrast to the conventional approach in which the weights in a trained network take specific values. We shall see shortly that this conventional description corresponds to a particular approximation to the Bayesian description.

We can find the posterior distribution of weights through the use of Bayes' theorem in the form

$$p(\mathbf{w}|D) = \frac{p(D|\mathbf{w})p(\mathbf{w})}{p(D)} \tag{0.1}$$

The conditional distribution of the data $p(D|\mathbf{w})$ can be regarded as a function of $\mathbf{w}$ in which case it is called the *likelihood.* We shall encounter a specific example shortly. The conventional approach to network training involves seeking a single weight vector $\mathbf{w}^*$ which maximizes the likelihood function.

The picture of learning provided by the Bayesian formalism is as follows. We start with some prior distribution over the weights given by $p(\mathbf{w})$. Since we generally have little idea at this stage of what the weight values should be, the prior might express some rather general properties such as smoothness of the network function, but will otherwise leave the weight values fairly unconstrained. The prior will therefore typically be a rather broad distribution, as indicated schematically in Figure 2. Once we have observed the data, this prior distribution can be converted to a posterior distribution using Bayes' theorem in the form (0.1). This posterior distribution will be more compact, as indicated in Figure 2, expressing the fact that we have learned something about the extent to which different weight values are consistent with the observed data.

In order to evaluate the posterior distribution we need to provide expressions for the prior distribution $p(\mathbf{w})$ and for the likelihood function $p(D|\mathbf{w})$. One of the simplest choices for the prior is to assume that it is a zero-mean Gaussian function of the weights, of the form

$$p(\mathbf{w}) = \frac{1}{Z_W(\alpha)} \exp\left(-\frac{\alpha}{2}||\mathbf{w}||^2\right) \tag{0.2}$$

[1] The extension to multiple output variables is straightforward.

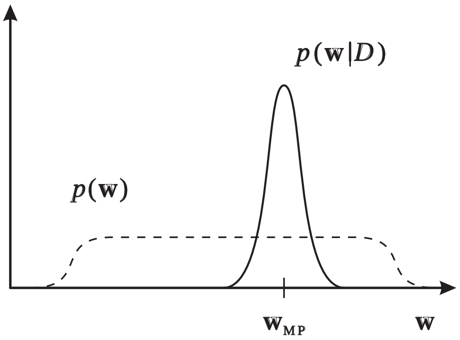

FIG. 2. Schematic plot of the prior distribution of weights $p(\mathbf{w})$ and the posterior distribution $p(\mathbf{w}|D)$ which arise in the Bayesian inference of network parameters. The most probable weight vector $\mathbf{w}_{\mathrm{MP}}$ corresponds to the maximum of the posterior distribution. In practice the posterior distribution will typically have a complex structure with many local maxima.

in which the normalization factor $Z_W(\alpha)$ is given by

$$Z_W(\alpha) = \left(\frac{2\pi}{\alpha}\right)^{W/2} \tag{0.3}$$

where W represents the total number of weight parameters. Since inverse variance α of the Gaussian controls the distribution of other parameters (weights and biases), it is called a *hyperparameter*. For the moment we shall assume that an appropriate value for α is known, and we shall return to the problem of how to determine α in Section 2.3. In practice, more complicated priors are often used, which may contain multiple hyperparameters.

Next we turn to the choice of likelihood function. This can be written down once we have specified a model for the distribution of target values for a given input vector. Again we consider a very simple example, namely a Gaussian with mean given by the output $y(\mathbf{x};\mathbf{w})$ of the network, and variance governed by a parameter β^{-1} so that

$$p(t|\mathbf{x},\mathbf{w}) = \left(\frac{\beta}{2\pi}\right)^{1/2} \exp\left(-\frac{\beta}{2}\{y(\mathbf{x};\mathbf{w}) - t\}^2\right) \tag{0.4}$$

Again we will assume for the moment that the value of β is known. For the data set D we assume that the patterns are drawn independently from this distribution, and hence that the probabilities are multiplicative, so that

$$\begin{aligned} p(D|\mathbf{w}) &= \prod_{n=1}^{N} p(t^n|\mathbf{x}^n,\mathbf{w}) \\ &= \frac{1}{Z_D(\beta)} \exp\left(-\frac{\beta}{2}\sum_{n=1}^{N}\{y(\mathbf{x}^n;\mathbf{w}) - t^n\}^2\right) \end{aligned} \tag{0.5}$$

where the normalization factor $Z_D(\beta)$ is given by

$$Z_D(\beta) = \left(\frac{2\pi}{\beta}\right)^{N/2} \tag{0.6}$$

Our main interest in using neural networks is to predict the values of the output variable for new values of the input variables. From the discussion of Section 1.3, we see that such predictions should be made by integrating over the weight variables, so that

$$p(t|\mathbf{x}, D) = \int p(t|\mathbf{x}, \mathbf{w}) p(\mathbf{w}|D)\, d\mathbf{w} \tag{0.7}$$

If the posterior distribution $p(\mathbf{w}|D)$ is sharply peaked about a maximum $\mathbf{w}_{\mathrm{MP}}$, as indicated schematically in Figure 2, then we can approximate the integral on the right hand side of (0.7) by $p(t|\mathbf{x}, \mathbf{w}_{\mathrm{MP}})$. This corresponds to a conventional approach in which predictions are made with the weight vector set to a specific value. To make use of this in practice we need to determine the most probable weight vector $\mathbf{w}_{\mathrm{MP}}$. Instead of finding a maximum of the posterior probability, it is usually more convenient to seek instead a minimum of the negative logarithm of the probability which is generally called an error function (these two procedures are equivalent since the negative logarithm is a monotonic function). For the particular prior distribution (0.2) and likelihood function (0.5) we see that, neglecting additive constant terms, the negative log probability is given by

$$E(\mathbf{w}) = \frac{\beta}{2} \sum_{n=1}^{N} \{y(\mathbf{x}^n; \mathbf{w}) - t^n\}^2 + \frac{\alpha}{2} \|\mathbf{w}\|^2 \tag{0.8}$$

Up to an overall multiplicative factor, this is just the usual sum-of-squares error function with a 'weight decay' regularization term.

2.2 Error Bars

As we have indicated, the Bayesian approach should take account not just of the most probable weight vector, but of the complete posterior distribution of weight vectors. In practice, the required integration over $\mathbf{w}$ is analytically intractable, and so we either need to use numerical techniques, as discussed in Section 4, or to make approximations. Here we consider an approach based on assuming that the posterior can be represented as a Gaussian centred on $\mathbf{w}_{\mathrm{MP}}$ (MacKay, 1992d). This will allow us to predict not only the most probable value of the output vector, but also to assign error bars to this prediction.

We can make a Gaussian approximation to the posterior distribution by representing the error function $E(\mathbf{w})$ in (0.8) by a Taylor expansion around $\mathbf{w}_{\mathrm{MP}}$ and keeping terms up to second order, so that

$$E(\mathbf{w}) = E(\mathbf{w}_{\mathrm{MP}}) + \frac{1}{2}(\mathbf{w} - \mathbf{w}_{\mathrm{MP}})^{\mathrm{T}}\, \mathbf{A}\, (\mathbf{w} - \mathbf{w}_{\mathrm{MP}}) \tag{0.9}$$

where $\mathbf{A}$ is called the *Hessian* matrix and consists of the second derivatives of the error function with respect to the weights. Note that the first derivative term is absent from (0.9) since we are expanding around a local minimum of $E(\mathbf{w})$. The Hessian matrix can be evaluated using an extension of the back-propagation procedure (Bishop, 1992).

Some partial justification for this approximation comes from the result of Walker (1969), which says that, under very general circumstances, a posterior distribution will tend to a Gaussian in the limit where the number of data points goes to infinity. For very large data sets we might then expect the Gaussian approximation to be a good one. However, the primary motivation for the Gaussian approximation is that it greatly simplifies the analysis.

Even with this Gaussian approximation we still cannot evaluate (0.7) analytically. We therefore assume that the posterior distribution is relatively narrow and hence that the network function does not vary too much over the region of significant probability density. This allows us to approximate the network function $y(\mathbf{x};\mathbf{w})$ by its linear expansion around $\mathbf{w}_{\mathrm{MP}}$

$$y(\mathbf{x};\mathbf{w}) = y(\mathbf{x};\mathbf{w}_{\mathrm{MP}}) + \mathbf{g}^{\mathrm{T}}\Delta\mathbf{w} \tag{0.10}$$

where $\Delta\mathbf{w} = \mathbf{w} - \mathbf{w}_{\mathrm{MP}}$ and

$$\mathbf{g} \equiv \nabla_{\mathbf{w}} y|_{\mathbf{w}_{\mathrm{MP}}} \tag{0.11}$$

The integration in (0.7) now becomes Gaussian and can be evaluated analytically with the result

$$p(t|\mathbf{x}, D) = \frac{1}{(2\pi\sigma_t^2)^{1/2}} \exp\left(-\frac{(t-y_{\mathrm{MP}})^2}{2\sigma_t^2}\right) \tag{0.12}$$

where we have restored the normalization factor explicitly. This distribution has a mean given by $y_{\mathrm{MP}} \equiv y(\mathbf{x};\mathbf{w}_{\mathrm{MP}})$, and a variance given by

$$\sigma_t^2 = \frac{1}{\beta} + \mathbf{g}^{\mathrm{T}}\mathbf{A}^{-1}\mathbf{g} \tag{0.13}$$

The first term in (0.13) arises from the intrinsic noise on the data, and is governed by the parameter β. The second term arises from the uncertainty in the weights, and would go to zero in the limit of an infinite data set. We can use (0.13) to assign error bars to network predictions, as illustrated for a toy problem in Figure 3. It can be seen from Figure 3 that the error bars are larger in regions where there is little data (Williams *et al.*, 1995), as we might expect.

2.3 Hyperparameters

So far we have assumed that the hyperparameters α and β are known[2]. In practice we will generally have little idea of what values these parameters should take. The treatment of hyperparameters is not trivial since a straightforward maximum likelihood approach would give over-fitted models which have poor

[2] Note that we will refer to β as a hyperparameter since, although it does not control the distribution of other parameters in the way that α does, it can be treated by similar techniques.

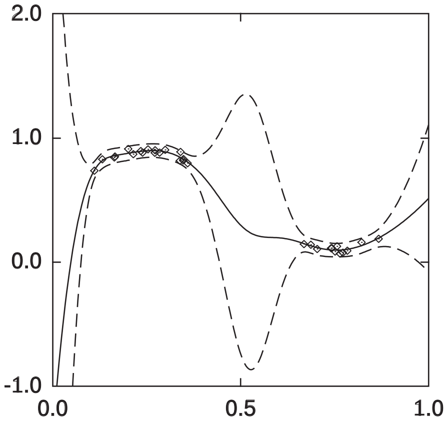

FIG. 3. A simple example of the application of Bayesian methods to a 'regression' problem. Here 30 data points have been generated by sampling the function $h(x) = 0.5 + 0.4\sin(2\pi x)$, and the network consists of a multi-layer perceptron with four hidden units having 'tanh' activation functions, and one linear output unit. The solid curve shows the network function with the weight vector set to $\mathbf{w}_{\mathrm{MP}}$ corresponding to the maximum of the posterior distribution, and the dashed curves represent the $\pm 2\sigma_t$ error bars calculated using (0.13).

generalization. For example, the best fit to the data is obtained with a very small value of α allowing the network function to give over-fitted solutions.

As we have discussed already, the correct Bayesian treatment for parameters such as α and β, whose values are unknown, is to integrate them out of any predictions. For example, the posterior distribution of network weights is given by

$$\begin{aligned} p(\mathbf{w}|D) &= \iint p(\mathbf{w}, \alpha, \beta|D)\, d\alpha\, d\beta \\ &= \iint p(\mathbf{w}|\alpha, \beta, D) p(\alpha, \beta|D)\, d\alpha\, d\beta \qquad (0.14) \end{aligned}$$

Note that we have extended our notation to include dependencies on α and β explicitly in the various probability densities.

In general the integrations required by (0.14) will be analytically intractable. Two practical analytically-based approaches to the treatment of hyperparameters have been discussed in the literature. The first of these is called the *evidence approximation* (MacKay, 1992a; MacKay, 1992d), and is based on techniques developed by Gull (1988, 1989) and Skilling (1991). It is computationally equivalent to the *type II maximum likelihood* (ML-II) method of conventional statistics (Berger, 1985).

Let us suppose that the posterior probability distribution $p(\alpha, \beta|D)$ for the hyperparameters in (0.14) is sharply peaked around their most probable values

α_{MP} and β_{MP}. Then (0.14) can be written

$$\begin{aligned} p(\mathbf{w}|D) &\simeq p(\mathbf{w}|\alpha_{\mathrm{MP}}, \beta_{\mathrm{MP}}, D) \int\!\!\int p(\alpha, \beta|D)\, d\alpha\, d\beta && (0.15) \\ &= p(\mathbf{w}|\alpha_{\mathrm{MP}}, \beta_{\mathrm{MP}}, D). && (0.16) \end{aligned}$$

This says that we should find the values of the hyperparameters which maximize the posterior probability, and then perform the remaining calculations with the hyperparameters set to these values.

In order to find α_{MP} and β_{MP}, we need to evaluated the posterior distribution of α and β. This is given by

$$p(\alpha, \beta|D) = \frac{p(D|\alpha, \beta) p(\alpha, \beta)}{p(D)} \tag{0.17}$$

which requires a choice for the prior $p(\alpha, \beta)$. Since this represents a prior over the hyperparameters, it is sometimes called a *hyperprior*. We see that the distribution of weight parameters, for example, is governed by a hyperparameter α which itself is described by a distribution. Schemes such as this are called *hierarchical models* and can be extended to any number of levels. If we have no idea of what would be suitable values for α and β, then we should choose a prior which in some sense gives equal weight to all possible values. Such priors are called *non-informative* and are discussed at length in Berger (1985). Since the denominator in (0.17) is independent of α and β, we see that the maximum-posterior values for these hyperparameters are found by maximizing the likelihood term $p(D|\alpha, \beta)$. This term is called the *evidence* for α and β.

If we make the dependences on α and β explicit, then we can write the evidence in the form

$$\begin{aligned} p(D|\alpha, \beta) &= \int p(D|\mathbf{w}, \alpha, \beta) p(\mathbf{w}|\alpha, \beta)\, d\mathbf{w} && (0.18) \\ &= \int p(D|\mathbf{w}, \beta) p(\mathbf{w}|\alpha)\, d\mathbf{w} && (0.19) \end{aligned}$$

where we have made use of the fact that the prior is independent of β and the likelihood function is independent of α. Using the exponential forms (0.2) and (0.5) for the prior and likelihood distributions, together with (0.8), we can write this in the form

$$p(D|\alpha, \beta) = \frac{1}{Z_D(\beta)} \frac{1}{Z_W(\alpha)} \int \exp\left(-E(\mathbf{w})\right) d\mathbf{w} \tag{0.20}$$

If we now make use of the Taylor expansion (0.9) the integration over $\mathbf{w}$ becomes a Gaussian integral and can be evaluated analytically. We omit the details here. The resultant expression can then be maximized with respect to α and β. This gives expressions for α_{MP} and β_{MP} in terms of the eigenvalues of the Hessian matrix $\mathbf{A}$. Thus, the problem of finding the most probable values for the hyperparameters requires little additional calculation beyond that needed to

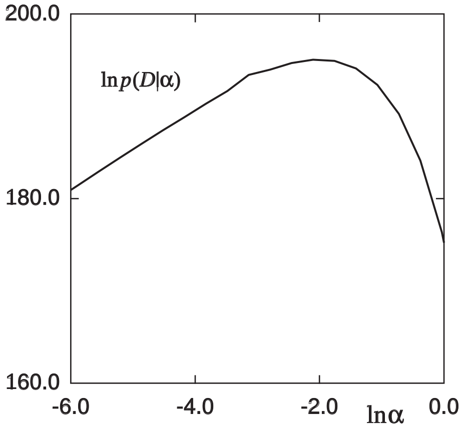

FIG. 4. This shows a plot of the log evidence for α versus $\ln \alpha$, corresponding to the example of Figure 3. The noise parameter β has been set to its true value. We see that small values of α as well as large values are less probable than some intermediate value.

evaluate the error bars. Figure 4 shows a plot of the log evidence versus α for the toy problem used in Figure 3.

For suitable choices of the hyperprior, it is possible to perform the integrations in (0.14) analytically (Buntine and Weigend, 1991; Wolpert, 1993; MacKay, 1994b; Williams, 1995). Since integration is formally the correct procedure, we might expect that this would be superior to the evidence approximation discussed above. However, MacKay (1994b) has argued that in practice the evidence approximation will often be expected to give superior results. The reason that this could in principle be the case, even though formally we should integrate over the hyperparameters, is that in practice with exact integration the remainder of the Bayesian analysis cannot be carried through without introducing further approximations, and these subsequent approximations can lead to greater inaccuracies than the evidence approach. Typically, these approximations would involve finding the maximum posterior weight vector $\mathbf{w}_{\mathrm{MP}}$ by a standard nonlinear optimization algorithm, and then fitting a Gaussian approximation around this value (Buntine and Weigend, 1991). Clearly the integration approach is capable of finding a true value for $\mathbf{w}_{\mathrm{MP}}$, and so the value found within the evidence approximation must be in error (to the extent that the two approaches differ). However, MacKay (1994b) has argued that the Gaussian approximation found by the evidence approach finds a better representation for most of the *volume* of the posterior probability distribution.

3 Model Comparison Revisited

As discussed in Section 1.2, the Bayesian approach automatically penalizes highly complex models and so is able to pick out an optimal model without resorting to the use of independent data as in methods such as cross-validation. Models can be compared using the evidence (the probability of the observed data set under the given model), which can be evaluated by using

$$p(D|\mathcal{H}_i) = \int\int p(D|\alpha, \beta, \mathcal{H}_i)p(\alpha, \beta|\mathcal{H}_i)\, d\alpha\, d\beta. \tag{0.1}$$

The quantity $p(D|\alpha, \beta, \mathcal{H}_i)$ is just the evidence for α and β which we considered earlier (with the dependence on the model again made explicit). The integration in (0.1) can be performed by making a Gaussian approximation for $p(D|\alpha, \beta, \mathcal{H}_i)$ as a function of α and β around their most probable values. This leads to an expression for the model evidence which involves the determinant of the Hessian matrix.

In practice the accurate evaluation of the evidence can prove to be very difficult. One of the reasons for this is that the determinant of the Hessian is given by the product of the eigenvalues and so is very sensitive to errors in the small eigenvalues. This was not the case for the evaluation $\alpha_{\rm MP}$ and $\beta_{\rm MP}$ since these depend on the sum of the eigenvalues.

Since the Bayesian approach to model comparison incorporates a mechanism for penalizing over-complex models, we might expect that the model with the largest evidence would give the best results on unseen data, in other words that it would have the best generalization properties. MacKay (1992d) and Thodberg (1993) both report observing empirical (anti) correlation between model evidence and generalization error. However, this correlation is far from perfect. Although we expect some correlation between a model having high evidence and the model generalizing well, the evidence is not measuring the same thing as generalization performance. In particular, we can identify several distinctions between these quantities:

- The test error is measured on a finite data set and so is a noisy quantity.
- The evidence provides a quantitative measure of the relative probabilities of different models. Although one particular model may have the highest probability, there may be other models for which the probability is still significant. Thus the model with the highest evidence will not necessarily give the best performance. We shall return to this point shortly when we discuss committees of networks.
- If we had two different models which happened to give rise to the same most-probable interpolant, then they would necessarily have the same generalization performance, but the more complex model would have have a smaller evidence. Thus, for two models which make the same predictions, the Bayesian approach favours the simpler model.
- The generalization error, in the form considered above, is measured using a network with weights set to the maximum of the posterior distribution. The evidence, however, takes account of the complete posterior distribution around the most probable value.
- The Bayesian analysis implicitly assumes that the set of models under consideration contains the 'truth' as a particular case. If all of the models

> are poorly matched to the problem then the relative evidences of different models may be misleading. MacKay (1992d) argues that a poor correlation between evidence and generalization error can be used to infer the presence of limitations in the models.

An additional reason why the correlation between evidence and test error may be poor is that there will be inaccuracies in evaluating the evidence. These arise from the use of a Gaussian approximation to the posterior distribution, and are particularly important if the Hessian matrix has one or more very small eigenvalues, as discussed above.

Further insight into the issue of model complexity in the Bayesian framework has been provided by Neal (1994) who has argued that, provided the complete Bayesian analysis is performed without approximation, there is no need to limit the complexity of a model even when there is relatively little training data available. Many real-world applications of neural networks (for example the recognition of handwritten characters) involve a multitude of complications and we do not expect them to be accurately solved by a simple network having a few hidden units. Neal (1994) was therefore led to consider the behaviour of priors over weights in the limit as the number of hidden units tends to infinity. He showed that, provided the parameters governing the priors are scaled appropriately with the number of units, the resulting prior distributions over network functions are well behaved in this limit. Such priors could in principle permit the use of very large networks. In practice, however, we may wish to limit the complexity in order to ensure that Gaussian assumptions are valid, or that Monte Carlo techniques (discussed in Section 4) can produce acceptable answers in reasonable computational time.

In our discussions of regression problems in Section 2, we approximated the posterior distribution of weights by a single Gaussian centred on a maximum of the distribution. In practice, we know that the posterior distribution will be multi-modal and that there will often be many local maxima (corresponding to local minima of the error function). To take account of this we can consider an approximation consisting of a Gaussian centred on each of the local maxima found by our optimization algorithm. The posterior distribution of the weights can be represented as

$$\begin{aligned} p(\mathbf{w}|D) &= \sum_i p(m_i, \mathbf{w}|D) \\ &= \sum_i p(\mathbf{w}|m_i, D) P(m_i|D) \end{aligned} \tag{0.2}$$

where m_i denotes one of the local maxima. This distribution is used to determine other quantities by integration over the whole of weight space. For instance, the mean network output is given by

$$\overline{y} = \int y(\mathbf{x}; \mathbf{w}) p(\mathbf{w}|D)\, d\mathbf{w}$$

$$
\begin{aligned}
&= \sum_i P(m_i|D) \int_{\Gamma_i} y(\mathbf{x};\mathbf{w}) p(\mathbf{w}|m_i, D)\, d\mathbf{w} \\
&= \sum_i P(m_i|D) \overline{y}_i \qquad (0.3)
\end{aligned}
$$

where Γ_i denotes the region of weight space surrounding the ith local maximum, and $\overline{y}_i$ is the corresponding network prediction averaged over this region. Thus we see that the overall prediction is given by a linear combination of the predictions of each of the local solutions separately. Such combinations of multiple trained networks are known as *committees*. In practice, the coefficients are difficult to evaluate accurately (since they correspond to model evidences) and so the committee coefficients may be simply set equal to $1/L$ where L is the total number of minima, or may be chosen using cross-validation.

4 Monte Carlo Methods

In the conventional (maximum likelihood) approach to network training, the bulk of the computational effort is concerned with *optimization*, in order to find the minimum of an error function. By contrast, in the Bayesian approach, the central operations require *integration* over multi-dimensional spaces. For example, the evaluation of the distribution of network outputs involves an integral over weight space given by (0.7). Similarly, the evaluation of the evidence for the hyperparameters also involves an integral over weight space given by (0.19). So far in this chapter, we have concentrated on the use of a Gaussian approximation for the posterior distribution of the weights, which allows these integrals to be performed analytically. This also allows the problem of integration to be replaced again with one of optimization (needed to find the mean of the Gaussian). If we wish to avoid the Gaussian approximation then we might seek numerical techniques for evaluating the required integrals directly.

Many standard numerical integration techniques, which can be used successfully for integrations over a small number of variables, are totally unsuitable for integrals of the kind we are considering, which involve integration over spaces of hundreds or thousands of weight parameters. For instance, if we try to sample weight space on some regular grid then, since the number of grid points grows exponentially with the dimensionality, the computational effort would be prohibitive. We resort instead to various forms of random sampling of points in weight space. Such methods are called *Monte Carlo* techniques.

The integrals we wish to evaluate take the form

$$
I = \int F(\mathbf{w}) p(\mathbf{w}|D)\, d\mathbf{w} \qquad (0.1)
$$

where $p(\mathbf{w}|D)$ represents posterior distribution of the weights, and $F(\mathbf{w})$ is some integrand. The basic idea is to approximate (0.1) with the finite sum

$$I \simeq \frac{1}{L} \sum_{i=1}^{L} F(\mathbf{w}_i) \tag{0.2}$$

where $\{\mathbf{w}_i\}$ represents a sample of weight vectors generated from the distribution $p(\mathbf{w}|D)$.

We must therefore face the task of generating a sample of vectors $\mathbf{w}$ representative of the distribution $p(\mathbf{w}|D)$, which in general will not be easy. To do it effectively, we must search through weight space to find regions where $p(\mathbf{w}|D)$ is reasonably large. This can be done by considering a sequence of vectors, where each successive vector depends on the previous vector as well as having a random component. Such techniques are called *Markov chain Monte Carlo* methods, and are reviewed in Neal (1993). The simplest example is a *random walk* in which at successive steps we have

$$\mathbf{w}_{\text{new}} = \mathbf{w}_{\text{old}} + \boldsymbol{\epsilon} \tag{0.3}$$

where $\boldsymbol{\epsilon}$ is some small random vector, chosen for instance from a spherical Gaussian distribution having a small variance parameter. Note that successive vectors generated in this way will no longer be independent. As a result of this dependence, the number of vectors needed to achieve a given accuracy in approximating an integral using (0.2) may be much larger than if the vectors had been independent. We can arrange for the distribution of weight vectors to correspond to $p(\mathbf{w}|D)$ by making use of the *Metropolis* algorithm (Metropolis *et al.*, 1953), which was developed to study the statistical mechanics of physical systems. The idea is to make candidate steps of the form (0.3), but to reject a proportion of the steps which lead to a reduction in the value of $p(\mathbf{w}|D)$. This must be done with great care, however, in order to ensure that resulting sample of weight vectors represents the required distribution. In the Metropolis algorithm this is achieved by using the following criterion:

$$\begin{array}{lll} \text{if} \quad p(\mathbf{w}_{\text{new}}|D) > p(\mathbf{w}_{\text{old}}|D) & \text{accept} & \\ \text{if} \quad p(\mathbf{w}_{\text{new}}|D) < p(\mathbf{w}_{\text{old}}|D) & \text{accept with probability} & \dfrac{p(\mathbf{w}_{\text{new}}|D)}{p(\mathbf{w}_{\text{old}}|D)} \end{array} \tag{0.4}$$

In the case of the Bayesian integrals needed for neural networks, however, this approach can still prove to be deficient due to the strong correlations in the posterior distribution, as illustrated in Figure 5.

This problem can be tackled by taking account of information concerning the gradient of $p(\mathbf{w}|D)$ and using this to choose search directions which favour regions of high posterior probability. For neural networks, the gradient information is easily obtained using standard back-propagation (recall that $-\ln p(\mathbf{w}|D)$ is an error function). Great care must be taken to ensure that the gradient information is used in such a way that the distribution of weight vectors which is generated corresponds to the required distribution. A procedure for achieving this, known as *hybrid Monte Carlo*, was developed by Duane *et al.* (1987), and

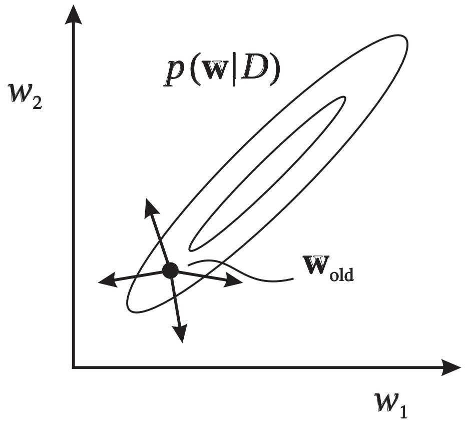

FIG. 5. When the standard Metropolis algorithm is applied to the evaluation of integrals in the Bayesian treatment of neural networks, a large proportion of the candidate steps are rejected due to the high correlations in the posterior distribution. Starting from the point $\mathbf{w}_{\text{old}}$, almost all potential steps (shown by the arrows) will lead to a decrease in $p(\mathbf{w}|D)$. This problem becomes more severe in spaces of higher dimensionality.

was applied to the Bayesian treatment of neural networks by Neal (1992, 1994).

By using the hybrid Monte Carlo algorithm it is possible to generate a suitable sample of weight vectors $\mathbf{w}_i$ for practical applications of neural networks in reasonable computational time. For a given test input vector $\mathbf{x}$, the corresponding network predictions $y(\mathbf{x}; \mathbf{w}_i)$ represent a sample from the distribution $p(y|\mathbf{x}, D)$. This allows the uncertainties on the network outputs, associated with a new input vector, to be assessed. Hyperparameters can be integrated over at the same time as the weights, using the technique of *Gibbs sampling.*

The estimation of the model evidence, however, remains a difficult problem. Another significant problem with Monte Carlo methods is the difficulty in defining a suitable termination criterion. Despite these drawbacks, Monte Carlo techniques offer a promising approach to Bayesian inference in the context of neural networks.

5 Additional Topics

There are many aspects of Bayesian methods which we have not had space to discuss at length here. For example, Bayesian methods allow choices to be made about where in input space new data should be collected in order that it be the most informative (MacKay, 1992c). Such use of the model itself to guide the collection of data during training is known as *active learning.* Also, the relative importance of different inputs can be determined using the Bayesian technique of *automatic relevance determination* (MacKay, 1994a, 1995; Neal, 1994), based on the use of a separate hyperparameters for each input. If a particular hyperparameter acquires a large value, this indicates that the corresponding input is irrelevant and can be eliminated. For examples of the application of Bayesian methods to real problems see (MacKay, 1995) and (Thodberg, 1993).

References

Berger, J. O. (1985). *Statistical Decision Theory and Bayesian Analysis* (Second ed.). New York: Springer-Verlag.

Bishop, C. M. (1992). Exact calculation of the Hessian matrix for the multilayer perceptron. *Neural Computation* **4** (4), 494–501.

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

Buntine, W. L. and A. S. Weigend (1991). Bayesian back-propagation. *Complex Systems* **5**, 603–643.

Cox, R. T. (1946). Probability, frequency and reasonable expectation. *American Journal of Physics* **14** (1), 1–13.

Duane, S., A. D. Kennedy, B. J. Pendleton, and D. Roweth (1987). Hybrid Monte Carlo. *Physics Letters B* **195** (2), 216–222.

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, Vol. 1: Foundations*, pp. 53–74. Dordrecht: Kluwer.

Gull, S. F. (1989). Developments in maximum entropy data analysis. In J. Skilling (Ed.), *Maximum Entropy and Bayesian Methods, Cambridge, 1988*, pp. 53–71. Dordrecht: Kluwer.

MacKay, D. J. C. (1992a). Bayesian interpolation. *Neural Computation* **4** (3), 415–447.

MacKay, D. J. C. (1992b). The evidence framework applied to classification networks. *Neural Computation* **4** (5), 720–736.

MacKay, D. J. C. (1992c). Information-based objective functions for active data selection. *Neural Computation* **4** (4), 590–604.

MacKay, D. J. C. (1992d). A practical Bayesian framework for back-propagation networks. *Neural Computation* **4** (3), 448–472.

MacKay, D. J. C. (1994a). Bayesian methods for backpropagation networks. In E. Domany, J. L. van Hemmen, and K. Schulten (Eds.), *Models of Neural Networks III*, Chapter 6. New York: Springer-Verlag.

MacKay, D. J. C. (1994b). Hyperparameters: optimise or integrate outΓ In G. Heidbreder (Ed.), *Maximum Entropy and Bayesian Methods, Santa Barbara 1993.* Dordrecht: Kluwer.

MacKay, D. J. C. (1995). Bayesian non-linear modelling for the 1993 energy prediction competition. In G. Heidbreder (Ed.), *Maximum Entropy and Bayesian Methods, Santa Barbara 1993.* Dordrecht: Kluwer.

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.

Neal, R. M. (1992). Bayesian training of backpropagation networks by the hybrid Monte Carlo method. Technical Report CRG-TR-92-1, Department of

Computer Science, University of Toronto, Canada.

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. (1994). *Bayesian Learning for Neural Networks.* Ph.D. thesis, University of Toronto, Canada.

Ripley, B. D. (1994). Neural networks and related methods for classification. *Journal of the Royal Statistical Society, B* **56** (3), 409–456.

Ripley, B. D. (1995). *Pattern Recognition and Neural Networks.* Cambridge: Cambridge University Press.

Skilling, J. (1991). On parameter estimation and quantified MaxEnt. In W. T. Grandy and L. H. Schick (Eds.), *Maximum Entropy and Bayesian Methods, Laramie, 1990*, pp. 267–273. Dordrecht: Kluwer.

Thodberg, H. H. (1993). Ace of Bayes: application of neural networks with pruning. Technical Report 1132E, The Danish Meat Research Institute, Maglegaardsvej 2, DK-4000 Roskilde, Denmark.

Walker, A. M. (1969). On the asymptotic behaviour of posterior distributions. *Journal of the Royal Statistical Society, B* **31** (1), 80–88.

Williams, C. K. I., C. Qazaz, C. M. Bishop, and H. Zhu (1995). On the relationship between Bayesian error bars and the input data density. In *Proceedings Fourth IEE International Conference on Artificial Neural Networks*, pp. 160–165. Cambridge, UK: IEE.

Williams, P. M. (1995). Bayesian regularization and pruning using a Laplace prior. *Neural Computation* **7** (1), 117–143.

Wolpert, D. H. (1993). On the use of evidence in neural networks. In S. J. Hanson, J. D. Cowan, and C. L. Giles (Eds.), *Advances in Neural Information Processing Systems*, Volume 5, pp. 539–546. San Mateo, CA: Morgan Kaufmann.