Bayesian Learning in Feed Forward Neural Networks

Carl Edward Rasmussen
email: carl@cs.toronto.edu

September 29, 1994

Abstract

Bayesian methods are applicable to complex modeling tasks. In this review, the principles of Bayesian inference are presented and applied to neural network models. Several approximate implementations are discussed, and their advantages over conventional frequentist model training and selection are outlined. It is argued that Bayesian methods are preferable to traditional approaches, although empirical evidence for this is still sparse.

Contents

1 Introduction

Bayesian methods provide a rigorous framework for data modeling, which has recently been applied to learning in feed forward neural networks. The focus of this review is the principles of Bayesian techniques for selecting and training networks. Emphasis is placed on practical methods that can be managed computationally with fairly complex models.

Bayesian methods potentially have several advantages over the traditional frequentist approach. These include the ability to explicitly incorporate prior knowledge, the ability to automatically set regularisation constants and the ability to compare different networks. Unfortunately there are two main difficulties involved in implementing Bayesian methods. Firstly, prior knowledge is usually available in terms of the functions the networks might implement and this must be translated into a prior over model parameters. Secondly, the computations involved in a full Bayesian procedure are infeasibly complex even for fairly simple models. This means that approximations must be made.

No rigorous investigation has yet been performed into whether the available practical implementations of the Bayesian framework for neural networks give a substantial improvement over frequentist approaches. This issue is complicated by the many different approximations to the Bayesian methods, and the lack of experience as to which approximations are most reliable.

In the following section the Bayesian viewpoint on model training and model selection is described, and the Bayesian framework is adapted to the training of neural networks. In section 3 computational methods for dealing with the involved computations are presented and discussed. Some further issues and comparisons with other methods are mentioned in section 4 and conclusions are drawn in section 5.

2 Bayesian Approach

In this section the basic concepts of the Bayesian approach are presented. Special attention is paid to those aspects of learning for which the Bayesian method differs most greatly from the frequentist approach. The general principles of Bayesian inference are discussed first, without focus on neural networks. The framework is then adapted to the case of training a neural network. Computational issues are discussed in the next chapter.

2.1 The problem

We wish to infer a function mapping inputs to outputs. We do not model the input distribution, but assume that the inputs as always given. The training data base consists of n noisy examples drawn from the function which we wish to learn. In general both the input and the output can be vectors, but for convenience only scalar output will be considered here (although generalization to vector outputs is fairly straight forward). In this review I consider regression problems, where the outputs are real numbers. Classification tasks instead have outputs that are class labels, or probabilities of classes. The training data is written in terms of sets of inputs and the corresponding outputs

$$I = \{\mathbf{x}^{(i)} : i = 1 \ldots n\}, \qquad O = \{y^{(i)} : i = 1 \ldots n\}, \qquad D = \{(\mathbf{x}^{(i)}, y^{(i)}) : i = 1 \ldots n\}. \quad (1)$$

The central problem is to make a prediction y based on a novel input $\mathbf{x}$. The prediction has some uncertainty, which is conveniently expressed by a probability density $p(y|\mathbf{x})$.

So, the task is to make a good approximation to $p(y|\mathbf{x})$ based on the training data D. There are both statistical and computational reasons to why we have to be content with an approximation. We shall call this approximation $p(y|\mathbf{x}, D)$. It is sometimes useful to be able to summarise the predictive distribution for example by a point estimate and associated error bars.

Only the outputs are considered to be noisy; the framework can however also be applied when the inputs are noisy, by treating the noise as if it were added at the output, although this strategy might not work well when modelling extremely nonlinear functions. I do not consider the case of noise free data, since the treatment of this case is very different from the noise corrupted case (since the likelihood function in this case becomes discontinuous, and gradient descent methods are no longer applicable).

2.2 Inference using Bayes rule

In the Bayesian framework, inference is performed at several levels. At the first level we are concerned with the model parameters, assuming that the model we are given is true. This level is analogous to the parameter estimation procedure in the frequentist approach. At the subsequent level(s), higher order properties such as noise levels and regularisation parameters are inferred. The two main differences between the two approaches at the first level are that a Bayesian has an explicit way of incorporating prior knowledge and that the result of the inference is a posterior distribution, and not a single estimated weight value.

2.2.1 Inferring model parameters

At the first level of inference, a certain model, $\mathcal{H}$, parameterised by the m dimensional vector, $\mathbf{w}$, is given, and we wish to infer a distribution over the values of the parameters of the model. The model $\mathcal{H}$ specifies the function $f(\mathbf{x}, \mathbf{w})$. Bayesian inference starts from prior knowledge, which is then updated in the light of the training data, giving rise to the posterior distribution. The prior is intended to capture our expectations about the model parameters, before we have seen any training data. The specification of the prior is controversial, since it requires that the concept of probability be expanded to include degrees of belief. This philosophical issue is beyond the scope of this paper.

The prior knowledge is formulated as a probability distribution over the quantities with which the inference is concerned. The prior distribution for the parameters is $p(\mathbf{w}|\mathcal{H})$. It can be useful to specify priors as functions of variables, α, called hyperparameters. In this case the prior is a density conditioned on the hyperparameters $p(\mathbf{w}|\alpha, \mathcal{H})$. In the following, α will be used to designate either a single hyperparameter, or a vector of these. The value of the hyperparameter itself is also to be inferred from the data at a higher level of inference. At this higher level we employ a hyperprior, which states our prior expectations about the value of the α, however vague these may be.

The data dependent term is also given in probabilistic terms, namely by the likelihood. In order to write down the likelihood we have to make some assumptions about the errors that we expect our model to make. A popular error model assumes independent, additive, zero mean Gaussian noise with constant variance, σ^2. Since the noise variance typically is unknown, we treat it as a hyperparameter. The likelihood is given by $p(O|I, \mathbf{w}, \sigma, \mathcal{H})$.

Using Bayes' Rule, we get the posterior probability density, conditioned on the hyperparameters

$$p(\mathbf{w}|\alpha, \sigma, D, \mathcal{H}) = \frac{p(O|I, \mathbf{w}, \sigma, \mathcal{H}) p(\mathbf{w}|\alpha, \mathcal{H})}{p(O|I, \alpha, \sigma, \mathcal{H})}. \tag{2}$$

In order to get rid of the conditioning on the hyperparameters (whos values are typically unknown), we integrate with respect to the posterior distribution of the hyperparameters.

$$p(\mathbf{w}|D, \mathcal{H}) = \int p(\mathbf{w}|\alpha, \sigma, D, \mathcal{H}) p(\alpha, \sigma|D, \mathcal{H}) d\alpha d\sigma. \tag{3}$$

The posterior for the hyperparameters is given by Bayes' Rule

$$p(\alpha, \sigma|D, \mathcal{H}) = \frac{P(O|I, \alpha, \sigma, \mathcal{H}) p(\alpha, \sigma|\mathcal{H})}{p(O|I, \mathcal{H})} = \frac{P(O|I, \alpha, \sigma, \mathcal{H}) p(\alpha|\mathcal{H}) p(\sigma)}{p(O|I, \mathcal{H})}, \tag{4}$$

where the likelihood for the hyperparameters is given by

$$p(O|I, \alpha, \sigma, \mathcal{H}) = \int p(O|I, \mathbf{w}, \sigma, \mathcal{H}) p(\mathbf{w}|\alpha, \mathcal{H}) d^m\mathbf{w}. \tag{5}$$

Note that at this stage, all the quantities are conditioned on $\mathcal{H}$, ie, at this level of inference it is assumed that the model is correct.

2.2.2 Model predictions

The Bayesian method does not give a particular value for the model parameters (in contrast to the maximum likelihood approach), but rather a probability distribution over the parameter space. To use the model to make predictions, it is necessary to integrate over the entire posterior distribution, eq. (3). The predictive distribution is given by

$$p(y|\mathbf{x}, D, \mathcal{H}) = \int p(y|\mathbf{x}, \mathbf{w}, D, \mathcal{H}) p(\mathbf{w}|D, \mathcal{H}) d^m\mathbf{w}, \tag{6}$$

where the conditional predictive distribution $p(y|\mathbf{x}, \mathbf{w}, D, \mathcal{H})$ is given by the assumed error model. In many cases it is impractical to work with the entire predictive distribution and we wish to summarise it with a few parameters or even more commonly by a point estimate. The *loss* function gives the penalty that is incurred by making a certain error in a prediction. For example, if the loss function is quadratic, then the best estimate $\hat{y}$ given a novel input $\mathbf{x}$ is given by minimizing

$$\min_{\hat{y}} \left[\int (y - \hat{y})^2 p(y|\mathbf{x}, D, \mathcal{H}) dy \right] \tag{7}$$

with the solution

$$\hat{y} = \int y p(y|\mathbf{x}, D, \mathcal{H}) dy, \tag{8}$$

which is simply the mean of y. Other loss functions give different estimates. For example, for absolute error one should predict the median rather than the mean. We are also able to compute the variance (squared error bars) on the estimated $\hat{y}$ value, namely

$$e^2 = \int (y - \hat{y})^2 p(y|\mathbf{x}, D, \mathcal{H}) dy. \tag{9}$$

In general this expression will consist of two terms. One term is due to the inherent noise in the examples (commonly assumed to be Gaussian) and the second term due to the uncertainly in the estimated value for the weights given by $p(\mathbf{w}|D, \mathcal{H})$ in eq. (6).

2.2.3 Model comparison

At the next level of inference, we calculate the posterior probability of the model. The prior over models is expressed by $p(\mathcal{H}_i)$, and the likelihood is now $p(O|I, \mathcal{H}_i)$, which can be calculated from the normalisation term $p(O|I, \alpha, \sigma, \mathcal{H})$ of eq. (2). This term is independent of the parameters $\mathbf{w}$, and it can be evaluated by the normalisation condition implied by eq. (2), when integrating over the weights

$$p(O|I, \alpha, \sigma, \mathcal{H}) = \int p(O|I, \alpha, \sigma, \mathbf{w}, \mathcal{H}) p(\mathbf{w}|\alpha, \mathcal{H}) d^m\mathbf{w}. \tag{10}$$

The likelihood for the model can be computed by integrating out the hyperparameters from eq. (10)

$$p(O|I, \mathcal{H}) = \int p(O|I, \alpha, \sigma, \mathcal{H}) p(\alpha|\mathcal{H}) p(\sigma) d\alpha d\sigma. \tag{11}$$

Bayes rule gives

$$p(\mathcal{H}_i|D) = \frac{p(O|I,\mathcal{H}_i)p(\mathcal{H}_i)}{p(O|I)} \propto p(O|I,\mathcal{H}_i)p(\mathcal{H}_i). \tag{12}$$

We are usually concerned only about comparing different models trained on the same training data, so the normalisation by $p(O|I)$ can be omitted.

The posterior for the models $p(\mathcal{H}_i|D)$ can be used to select the best model, or following the true Bayesian spirit, to weight the predictions of individual models, when averaging (since only a finite number of models have been proposed) over models

$$p(y|\mathbf{x},D) = \sum_i p(y|\mathbf{x},D,\mathcal{H}_i)p(\mathcal{H}_i|D). \tag{13}$$

2.3 Applying Bayes to a neural network

In this section the Bayesian framework is adapted to training of a neural network. As a concrete example, look at the feed forward network defined by

$$\begin{aligned} f(\mathbf{x},\mathbf{w}) &= \sum_{i=1}^{H} w_i h_i + b & (14)\\ h_i &= \tanh(\sum_{j=1}^{n} w_{ij}x_j + b_j), & (15) \end{aligned}$$

where w (the weights) and b (the biases) are the parameters of the model, collectively referred to by the m dimensional vector $\mathbf{w}$. The n dimensional input vector is denoted $\mathbf{x}$. There are H hyperbolic tangent hidden units. This set of equations defines a network with one layer of nonlinear hidden units and a single linear output unit. Networks of this type are popular, since it can be shown [Funahashi 1989] that in the limit of a large number of hidden units, the net is capable of implementing any continuous mapping (from a compact input space).

2.3.1 The Meaning of the Prior

In a typical modeling task the prior knowledge is over the space of functions to which the data might conform. For example, it could be that the function is expected to exhibit some kind of smoothness. In the case of a neural network, this prior on functions has to be converted to a prior on the parameters of the model, ie, the weights in the network.

In order to set appropriate priors it is essential to understand the relation between the value of parameters in the network, and the functions which the network implements. Some work has been done [Neal 1994] on understanding the effect of different priors and their scaling with network size. This work is based on sampling from the prior distributions, and on analytical results derived in the limit of infinitely large networks.

A thorough treatment of this question is in general fairly complicated; here I will list a few considerations that will help give a feel for the type of knowledge that can typically be incorporated in the prior. The size of the output weights to a large degree determines the overall magnitude of the output, constrained by the degree of saturation and correlation in the hidden layer. This is because the hidden units have outputs that are constrained (by the tanh function) to lie in the ± 1 region. The input weights together with the biases of the hidden units determine the degree of nonlinearity of the function implemented. This is due to the roughly linear shape of the tanh function near the origin.

These considerations suggest that one useful prior would be to specify that the input weights are expected to be small. Following MacKay [1992a] we employ a prior which specifies that we expect the weights to have a zero mean Gaussian distribution. The typical size of the weights under this prior is given by the standard deviation of the Gaussian. This prior is

$$p(\mathbf{w}|\alpha, \mathcal{H}) = \prod_{i=1}^{M} \sqrt{\frac{\alpha}{\pi}} \exp(-\alpha w_i^2), \tag{16}$$

where α is a regularisation parameter, which is to be treated as a hyperparameter. MacKay presumably chose a Gaussian prior for analytical convenience, but other priors can also be used [Williams 1994; Neal 1994; Nowlan and Hinton 1992]. The hyperparameter α has an associated hyperprior, which MacKay chose to be flat (in log space, since α is a scale parameter).

Hyperparameters are a useful tool for specifying priors, but they are not essential for the modeling. To see this, note that we can calculate the unconditional prior over weights by integrating out the hyperparameter

$$p(\mathbf{w}|\mathcal{H}) = \int p(\mathbf{w}|\alpha, \mathcal{H}) p(\alpha|\mathcal{H}) d\alpha. \tag{17}$$

The hyperparameter is therefore mostly just a convenient way of specifying dependence between weights. The prior specifies that we expect all the weights to be of the same scale (although this requirement will later be slackened, by allowing multiple groups of weights governed by different hyperparameters). Later we will see that the use of hyperparameters may also lead to a more efficient implementation of an approximate Bayesian method.

The traditional frequentist approach does not involve explicit setting of priors. This leads to problems that are most easily seen in a model for fitting a curve to data points (in two dimensions). Although there are, of course, an infinite number of functions that fit the data well, we will usually have a strong feeling that many of these functions (eg. functions with huge leaps between nearby data points) are very unlikely to be close to the function which we are hoping to approximate. Using a prior for the weights gives us the possibility of incorporating precisely this knowledge in our model.

2.3.2 The error model

Finally we need to specify a model for which kind of errors we expect our model to make. A very common choice is independent, zero-mean Gaussian noise with constant variance

$$p(O|I, \mathrm{w}, \sigma, \mathcal{H}) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^{(i)} - f(\mathbf{x}^{(i)}))^2}{2\sigma^2}\right). \tag{18}$$

Other error models may also be used. For example the Gaussian error model does not take *outliers* into account. These are data points that for some reason don't conform very well to the trend in the data. In this case, the model might be distorted by trying to account for the outlying point. Other error distributions, such as the Laplacian, may be much more robust to outliers. The key point is that the error model must be good in order to obtain good predictions.

Another possibility is to allow the noise variance to vary with the input $\mathbf{x}$. This could be realised using two networks, one predicting the mean of y, the other predicting the variance. This idea was mentioned by Buntine and Weigend [1991], but so far there are no published results on applying this in method to neural network models.

3 Computational aspects

In the previous section we have discussed what quantities we would ideally wish to compute in order to make inferences using our models. However, we have not yet discussed how to compute these quantities, and indeed this is not a simple task. For general nonlinear models such as neural networks, the distributions which we must integrate over in order to obtain the posterior distributions are complex and high dimensional, and the integrals cannot be handled analytically. This section discusses several approaches to evaluating the integrals or approximations to them.

3.1 The Evidence Framework

The *evidence* framework was developed by Gull [1989] as an approximate method for doing Bayesian inference. In this paper the Bayesian method is used to justify the use of Maximum Entropy priors in inference. MacKay [1992a] reviewed this framework, and in a companion paper [MacKay 1992b] showed how it can be practically applied to the task of training neural networks. Thodberg [1993] has successfully used the evidence framework on a practical task.

The evidence framework relies on two approximations. Firstly, the hyperparameters are *optimised* rather than integrated over, which is a deviation from the Bayesian approach. Secondly, the posterior distribution for the weights is approximated by a Gaussian, centered on a (local) maximum of the distribution. These approximations make analytical derivations possible.

Whereas the Bayesian method produces a posterior distribution for the weights (or a predictive distribution for the outputs) which can be very complicated, the evidence framework produces a multivariate Gaussian approximation to the weight posterior, so the predictions can be summarised by a point estimate (based on the mode of the Gaussian) and error bars (calculated from the covariance matrix of the Gaussian). From a practical viewpoint it is often advantageous to work with such compact representations of the model predictions. In many cases the unimodal Gaussian may be inadequate for approximating the weight posterior, and this may lead to poor model predictions. This problem can be alleviated by making predictions based on ensembles of networks, corresponding to a mixture of Gaussians model of the weight posterior.

The final goal of the evidence framework is model comparison. Models trained from different initial weights or with different architectures can be compared by evaluating the evidence for the models $p(O|I, \mathcal{H})$. This ranking can be used to select a good model, or alternatively, to weight different models in predictions.

3.1.1 The algorithm for the evidence approximation

In the evidence framework, we optimize the hyperparameters instead of integrating over their posterior distribution. As an approximation to the true Bayesian approach, we are thus using penalized maximum likelihood (or plain maximum likelihood if our hyperpriors are flat) at the level of the hyperparameters. This will in many cases turn out to be a good approximation, since there are usually only a few hyperparameters, and these are well determined by the data. The computations follow from eq. (4) and (5). Since we are only interested in maximising $p(\alpha, \sigma|D, \mathcal{H})$, we omit the normalising constant $p(D|\mathcal{H})$ (which is independent of α and σ), and we thus want to find $\hat{\alpha}$ and $\hat{\sigma}$ which maximises

$$p(\alpha, \sigma|D, \mathcal{H}) \propto p(\alpha|\mathcal{H})p(\sigma) \int p(D|\mathbf{w}, \sigma, \mathcal{H})p(\mathbf{w}|\alpha, \mathcal{H})d^m\mathbf{w} \quad (19)$$

In order to perform this maximisation, we must evaluate the integral in eq. (19). MacKay uses a Gaussian approximation to the integrand, around a point, $\hat{\mathbf{w}}$. The value of $\hat{\mathbf{w}}$ is determined by freezing the hyperparameters at their current values and then searching for a (local) maximum of the integrand over weight space. The hyperparameters are then re-evaluated by maximising the Gaussian approximation to eq. (19).

Once the hyperparameters have been assigned values, these are used at the lower levels. The central approximation here comes when introducing the 'optimal' values for the hyperparameters in the posterior for the weights, eq. (3)

$$
\begin{aligned}
p(\mathbf{w}|D,\mathcal{H}) &= \int p(\mathbf{w}|\alpha,\sigma,D,\mathcal{H})p(\alpha,\sigma|D,\mathcal{H})d\alpha d\sigma && (20)\\
&\simeq p(\mathbf{w}|\hat{\alpha},\hat{\sigma},D,\mathcal{H}). && (21)
\end{aligned}
$$

This approximation is good if the posterior for the hyper parameters has a dominant peak at $\hat{\alpha}$ and $\hat{\sigma}$, and the conditional posterior for the weights $p(\mathbf{w}|\alpha,\sigma,D,\mathcal{H})$ does not change rapidly with α and σ near $\hat{\alpha}$ and $\hat{\sigma}$.

The algorithm that MacKay [1992b] suggests relies on iteratively updating the weights and the hyperparameters, starting at some initial random values, perhaps guided by experience in earlier simulations. The weights are updated to find a maximum (local or global) of the posterior weight distribution, conditioned on the hyperparameters. For fixed values of the hyperparameters, this is equivalent to minimising the following cost function

$$
E = -\log p(\mathbf{w}|\hat{\alpha},\hat{\sigma},D,\mathcal{H}) + c = \frac{1}{2\hat{\sigma}^2}\sum_{i=1}^{n}(y^{(i)} - f(\mathbf{x}^{(i)},\mathbf{w}))^2 + \hat{\alpha}\sum_{i=1}^{m} w_i^2. \qquad (22)
$$

where c is a constant, which does not depend on the weights. It is interesting to note that a similar cost function is used in the penalized maximum likelihood approach, using the standard sum of squared weights regulariser. However this method assumes that the weight decay parameter has been set by some other means (eg. cross validation).

The hyperparameters are updated by maximising eq. (19) using the Gaussian approximation to the integrand that has been outlined. It is probably computationally inefficient to try to find a very good approximation to the extrema in each iteration, instead iterations could be viewed as successive refinements of previous approximations. It is hoped that this procedure will converge, and that the solution won't be critically dependent upon the initial random values. MacKay [1992b] presents a numerical verification of this approach.

In the evidence framework, we thus make predictions based on a single weight vector, $\hat{\mathbf{w}}$, and it's associated error bars. In practise we often see point estimates based solely on $\hat{\mathbf{w}}$. An illuminating interpretation of this custom is to consider the prediction made by adapting eq. (8), using a linear approximation to the network function, the Gaussian approximation to the posterior weight distribution, and a squared error loss function

$$\hat{y} = \int f(\mathbf{x}, \mathbf{w}) p(\mathbf{w}|D, \mathcal{H}) d^m \mathbf{w} \simeq f(\mathbf{x}, \hat{\mathbf{w}}). \tag{23}$$

MacKay [1992b] found that it is beneficial to employ several groups of weights, each associated with its own weight decay parameter. This finding makes intuitive sense, since there is no reason to believe that the input weights and the output weights should have the same scale, since an arbitrary rescaling of the inputs in the data base would affect the scale of the input weights, but not of the output weights. Thodberg [1993] has given the explicit derivation of the equations for multiple weight decays. A related idea was proposed by Nowlan and Hinton [1992], who employed a mixture of Gaussians prior for the weights. Here the weights are 'softly' tied to one of the Gaussians, but can move from one group to another during learning. The groups are not restricted to have zero mean; the mean instead adapts with the weights as learning proceeds. It is important to note that the evidence framework may be less suitable for dealing with these priors, since there are many hyperparameters, which might not all be well determined.

To evaluate the evidence for a model we return to eq. (5) and (11) which taken together give

$$p(O|I, \mathcal{H}) = \int p(O|I, \mathbf{w}, \sigma, \mathcal{H}) p(\mathbf{w}|\alpha, \mathcal{H}) p(\alpha|\mathcal{H}) p(\sigma) d^m \mathbf{w} \, d\alpha \, d\sigma, \tag{24}$$

where we assume that the likelihood and the weight priors are Gaussian, that there is zero covariance between weights and hyperparameters (only the covariances between the weights themselves are considered), and that the hyperpriors are flat. This simplifies the integral sufficiently that it can be done analytically.

3.1.2 Multiple Minima in the Cost Function

So far, we have (implicitly) assumed that the posterior distribution for the weights is unimodal, so that the Gaussian approximation around the mode is meaningful. Unfortunately this is not the case when the model is a neural network. Multiple modes arise first because the type of network we are considering has symmetries of two types: the hidden units can be permuted, and the signs of incoming and outgoing weights can be interchanged without altering the function. Further modes arise from local minima in the cost function.

The existence of multiple symmetric solutions does not influence the predictions of the network, but does affect the correct expression for the posterior model probability. The symmetric solutions can be accounted for by incorporating an adjustment in the expression for the evidence. It is harder to take proper account of local minima.

MacKay [1992b] simply trained a number of different networks, and then chose the one which had the highest value of the evidence, thereby ignoring local minima. A more Bayesian approach was used by Thodberg [1993] for predictions of fat content in meat. He has extended the framework to incorporate the idea of a committee, or ensemble [Hansen and Salamon 1990]. The predictions are made from an ensemble of networks, ideally weighted by the evidence for the different networks, eq. (13). Thodberg found however that this weighting did not perform successfully, since the evidence of a few networks was vastly greater than that of the majority, although most of the networks had good performance. This result was probably due to the lack of accurateness (due to the approximations) in the evaluation of the evidence.

Thodberg instead used a heuristic approach, where he selected a fraction of networks with high evidence, and weighted them evenly in the final predictions. This procedure reduced the error on the predictions, and also gave a better estimate of performance. With this model, the error bars on a prediction have three parts: the assumed noise level σ^2, a noise contribution from the uncertainty in the weights, and an uncertainty across committee members. The two last uncertainties are dependent on the network input x, that is, the accuracy of a prediction may be different for different inputs.

3.1.3 Incorporating pruning

Pruning is a popular method used in a frequentist framework to combat overfitting and to obtain compact networks. Various forms of pruning have been suggested. For example the Optimal Brain Damage [Le Cun et al. 1990] and Optimal Brain Surgeon [Hassibi and Stork 1993] methods prune based on an approximation to the increase in training error that would be incurred by removing a particular weight.

Pruning does not have any fundamental role in the Bayesian framework. From a Bayesian view, even a very poorly determined parameter should be kept in the network. The use of a regulariser, and the posterior integration over weights, should prevent overfitting. However, very poorly determined parameters tend to make the expansion around the mode of the posterior weight distribution numerically troublesome, since the covariance matrix becomes badly conditioned. Since the accuracy of the prediction is not very sensitive to the value of these severely ill-determined parameters, it might be useful to remove them, from a purely computational point of view. Thodberg [1993] uses pruning, stopping pruning when the evidence reaches a minimum. The pruning of weights can be seen as a change of the prior for this weight (to a delta function in zero). From a theoretical point of view, it is not clear that this should lead to improved generalisation, but this is nevertheless seen in the simulations.

Within the evidence framework, it is the possible for the hyperparameters to go to infinity, effectively pruning part of the network. This effect is due to the approximations in the evidence framework, since if proper hyperpriors were used, there would be a bound on the size of the hyperparameter. It might nevertheless be a useful effect. An idea known as 'automatic relevance determination' (ARD) due to MacKay and Neal (unpublished) uses separate hyperparameters for the weights from each input. If a particular input is not useful for the network, the hyperparameter controlling this group will go to infinity, effectively eliminating the input from the problem. This offers the possibility of adapting the architecture during learning, and sidesteps the combinatorial problem of deciding which inputs are relevant (and the dependencies between inputs, ie, input A might only be important in absence of input B, etc.).

Pruning is also reported by Williams [1994], who uses a Laplacian prior for the weights. He argues that one should use the distribution with highest possible entropy, thereby limiting the number of assumptions that are introduced into the learning task by the prior. This line of reasoning seems to be questionable when applied to neural networks. As [Neal 1994] has shown, the functional form of the prior distribution governs the form of the functions that are likely to be implemented by the network. For example, Gaussian priors produce smooth functions, whereas distributions with heavier tails, produce less smooth functions. It is unclear to me that the maximum entropy prior introduces fewer assumptions on the network function form than other priors, and it is also not clear to me why it is advantageous to introduce few assumptions. The prior should simply be chosen to reflect our prior belief about the function. A difficulty related to the Laplacian prior is that the derivative of the prior has a discontinuity at zero, which complicates the use of gradient descent methods.

3.2 The MAP method

Wolpert [1993] has argued that the evidence approximation is inaccurate and unnecessary. For the distributions considered by MacKay [1992b], it is possible to analytically integrate out the hyperparameters, as the Bayesian method prescribes [Buntine and Weigend 1991]. If $p(\mathbf{w}|\alpha, \mathcal{H})$ is a Gaussian and $p(\alpha|\mathcal{H}) = 1/\alpha$ (the standard non-informative prior), we can analytically find the prior for the weights by

$$
\begin{aligned}
p(\mathbf{w}|\mathcal{H}) &= \int p(\mathbf{w}|\alpha, \mathcal{H}) p(\alpha|\mathcal{H}) d\alpha && (25) \\
&= \int_0^\infty \left(\frac{\alpha}{\pi}\right)^{m/2} \exp -(\alpha \sum_i w_i^2) \frac{1}{\alpha} d\alpha \;=\; (\pi \sum_i w_i^2)^{-m/2}, && (26)
\end{aligned}
$$

and a similar approach can be applied to the noise parameter σ. Using this method, called the 'maximum a posteriori' (MAP) method, it is possible to sidestep the need to optimise the hyperparameters as in the evidence framework. We are able to find the posterior distribution of the weights, but we are really seeking a particular weight value, which is representative of the posterior. In the MAP method we chose the $\mathbf{w}$ that maximises the posterior. This is of course again a deviation from the Bayesian approach, where we should integrate over the predictions made from the whole ensemble of networks defined by the posterior. However, this integral is analytically intractable when using the prior given by eq. (26).

So essentially, the difference between the evidence and the MAP method is, that the evidence framework optimises over the hyperparameters (based on an approximation derived by optimising over the model parameters) and integrates over the model parameters (weights), and the MAP methods does the opposite. MacKay [1993] has shown that in the case where there are a significant number of ill-determined parameters, the evidence approximation is superior to the MAP estimate. In a numerical example he showns how the introduction of vacuous information (poorly determined parameters) leads to over-regularisation in the MAP method. This is caused by the fact that the mode of the posterior is not representative of the distribution. If there are ill-determined parameters, the posterior has a skew peak, and the mode of the distribution lies far from its centre of mass.

Unfortunately it is hard to devise a small and simple illustrative example where the evidence and the MAP method can be compared to the proper Bayesian approach (eg. implemented by sampling methods, see section 3.4). The MAP method and the evidence framework can be compared by comparing their $\mathbf{w}$, as in [MacKay 1993], but since the true Bayesian approach integrates over the posterior to make predictions, and does not assign particular values to the weights, no comparison can be made to the ideal approach at this level. It will only be possible to see which of the two approximate procedures give results that are closest to the Bayesian method by comparing their predictions with predictions obtained using a sampling approach. Also, comparison is complicated by the fact that the MAP method (as well as the sampling approach) must have a proper prior in order to avoid the singularity in eq. (26) at $\mathbf{w} = 0$, and will thus not be directly comparable with the evidence framework as commonly practised.

3.3 The Kullback-Leibler method

In the evidence framework, the posterior weight distribution is approximated by a Gaussian around a (local) maximum. However, it is not clear that expansions around these modes yield the 'best' approximations; perhaps the posterior has many high peaks with a small associated volume, so that approximations around points of maximum density will be poor. This suggests that it might be useful to be able to use Gaussian approximations which are not necessarily centered on points of maximum density.

One way to measure the distance between two density functions is the Kullback–Leibler difference. Using this measure, the distance between an approximating density $q(\mathbf{w}, \theta)$ and the posterior for the weights is

$$F(\theta) = \int q(\mathbf{w}, \theta) \log \frac{q(\mathbf{w}, \theta)}{p(\mathbf{w}|D, \mathcal{H})}, \tag{27}$$

where the density $q(\mathbf{w}, \theta)$ is a function of $\mathbf{w}$ parameterised by θ, eg. a Gaussian. The task is now to minimise $F(\theta)$ over θ to find a good approximation to the posterior. This is in itself not an easy task.

In a paper on the Minimum Description Length (MDL) principle (which can be shown to be identical to a Bayesian framework) Hinton and van Camp [1993] has shown that, for networks with a single hidden layer and linear output units, it is possible to compute the exact gradient of a upper bound of $F(\theta)$, provided that the covariance matrix of $q(\mathbf{w}, \theta)$ is diagonal. It is unclear whether this approximation, with an adaptable mean but diagonal covariance matrix, will be superior to the evidence approximation. However, the evidence framework is certainly more flexible, since there is no constraint on the architecture of the net, and it also avoids the huge table lookups involved in the MDL inspired method.

3.4 Monte Carlo approach

The Gaussian approximations that were employed in the evidence framework were to a large degree motivated by analytical tractability of the resulting integrals. However there may be situations where these Gaussian assumptions are inaccurate. In this case techniques based on sampling can be employed to evaluate the integrals. This approach has been explored by [Neal 1993].

When a sampling method is used to evaluate the predictive distribution, eq. (6), the integral is typically of a very high dimensionality. In this case it is infeasible to sample the integrand on a grid, or use rejection sampling. Instead, a method that actively searches for regions with high probability must be utilised.

The idea behind the sampling method is to approximate an integral over a function times a probability function by the mean of the function values, when sampled from the probability distribution

$$\int f(\mathbf{x}) p(\mathbf{x}) d\mathbf{x} \simeq \frac{1}{n} \sum_{i=1}^{n} f(\mathbf{x}), \tag{28}$$

where the $\mathbf{x}$ are drawn from the distribution $p(\mathbf{x})$. In the limit of $n \to \infty$ this result is of course exact.

The approach of Neal relies on constructing a Markov chain, that has the distribution which we wish to integrate over (in this case the posterior weight distribution) as a stationary distribution. He uses the Hybrid Monte Carlo method, which combines the Metropolis algorithm with dynamic simulation, for efficiently sampling the posterior. The dynamical simulations allows the random walk behavior to be avoided, which speeds up the sampling.

One of the main problems with the sampling approach is that it is difficult to know when the stationary distributions have been reached, and when they have been sampled sufficiently to give good results. Neal has shown however that the methods can be applied to practical problems, and has obtained marginally better results than MacKay using the evidence approximation on a small toy problem. In this example, however, the amount of computation required is much greater for the sampling method (19 hours) than for the evidence approximation (about 60 mins in total to train 10 networks and select one), but it is possible that shorter runs would still yield comparable results. Even if it turns out that the sampling methods are too slow to be of practical importance, they will still be useful as a reference for comparing approximate methods.

An additional problem with the sampling approach arises when trying to evaluate the evidence for a model, eq. (11). Here, producing samples from the prior, and collecting values of the likelihood for the weights, we encounter the problem that the likelihood function is extremely peaked in certain very small regions of weight space, whereas the prior typically is a rather flat function. This means that a sampling approach to evaluating this integral is an extremely difficult task, ie, the sampling would have to extend for an astronomical length of time before converging on the correct result. So it does not at this point seem feasible to employ sampling methods for evaluating the evidence for different models, although there exist more sophisticated methods (related to free energy estimation) that perhaps are feasible.

4 Discussion

4.1 Evidence versus Generalisation

Our real goal is to make models that generalise well. A potentially worrying aspect of the Bayesian framework is that the exact relationship between the evidence (or likelihood) for a model and its generalisation performance is not entirely clear. It does not seem to be possible, for example, to prove that a network with high evidence makes more accurate predictions than a network with lower evidence, on average, or with a high probability. Partly, this is because a Bayesian relies on priors, which may live up to the domain with various degrees of accuracy. However the experimental work that has been presented by various workers indicates that the evidence does indeed correlate quite well with the generalisation performance.

4.2 The Occam hill

In [MacKay 1992a] there is a plot of the evidence as a function of the network size. The relationship has a shape that MacKay calls an 'Occam Hill', with a steep decline in evidence for small networks and a shallow decline for large networks. Ideally, this curve should be steep on the side of too few parameters, but then approximately constant for larger models, ie, we should not encounter the phenomenon of overfitting (assuming that our priors are set up properly).

There are two reasons why MacKay might have found this Occam Hill behavior. Firstly, as we have already discussed, the numerical evaluations of the covariance matrix of the Gaussian approximation to the weight posterior become inaccurate and unreliable (the covariance matrix becomes ill-conditioned) for models with a large number of parameters (compared to the number of training examples). The second reason is that the prior used by MacKay becomes too vague when training huge models on very little data. I will discuss this topic in a broader perspective below.

In the frequentist approach to learning, the adaptation of model complexity to the amount of available training data is governed by the Bias Variance Dilemma. Simple models make predictions with small variance but large bias, whereas complex models have the opposite effect; see [Geman et al. 1992] for a discussion in the context of neural networks. To overcome the bias variance dilemma it is necessary to find a good balance, by adapting the model according to the amount of available data. This adaptation of the model complexity is avoided in the Bayesian framework through the specification of the prior. So it seems as though we can just turn the crank on the Bayesian approach, and this will solve all our problems.

This is not exactly true however. The problem lies in the specification of the prior. Suppose we have a problem where our prior knowledge is limited to some vague notion of smoothness, as is the case considered in [MacKay 1992b]. A very vague prior is used: a Gaussian, with a variance which has a flat (improper) hyperprior. For small models, the value of the hyperparameter is well determined by the data, but for large models with less data, the posterior for the hyperparameter becomes dominated by it's prior. Since the hyperprior allows for any value (it is flat), the procedure breaks down when there is very little data because of the non-normalisable prior, and the posterior weights collapse to zero. In order to avoid this scenario we must use a proper hyperprior, forcing us to include more knowledge in the model. This knowledge will set bounds on the complexity of the model, thus addressing the bias variance dilemma directly. Of course there may be situations where we are unable to specify good hyperpriors, and in such cases we are left with the bias variance dilemma. Obviously, this is not the fault of the Bayesian method for inference though, and in typical situations quite large models will be compatible with very broad priors, see [Gull 1989].

4.3 Other Approaches

A number of alternatives to the Bayesian approach will briefly be mentioned below.

One of these methods, which is mostly used for classification problems, relies on the Vapnik-Chervonenkis (VC) dimensionality of a model, see [Abu-Mostafa 1989] for a review. The VC dimension is related to the complexity of the network, and there exists a fundamental relationship between the VC dimension and the number of examples required to bound the probability of unexpectedly bad generalisation.

While this kind of bound may be interesting, it is important to realise that the Bayesian scheme estimates the average behavior, and not a bound. In fact there may be many situations where this average and the VC bound are very far from each other, and in these cases the average estimate might be a lot more informative than a worst case estimate.

A different method which is frequently employed in frequentist settings in order to deal with the bias variance dilemma, is cross validation. The idea here is to hold back a part of the training data and use this validation set to monitor the prediction performance. The disadvantage of this approach is that an accurate estimate of performance requires a large validation set, which reduces the number of examples available for training. This can to some extent be avoided by n-fold cross validation, were the small validation sets are picked and then rotated through the entire training set. This method however is computationally very demanding, since many models have to be trained to get an estimate of performance, and schemes relying on cross validation become hopelessly intractable if the architecture of the network as well as several weight decay parameters are being optimised.

A more promising scheme was introduced by Akaike [1969]. This approach estimates a penalty term, which is paid for fitting the parameters of the model. The more parameters in a model, the higher is the accuracy of its fit to the training data, but it also carries a higher penalty. This Final Prediction Error (FPE) estimate is computed using a linear expansion of the network function and examining the relation between testing and training error in the limit of many examples. This estimator has been extended to take the use of regularisation (weight decay) into account [Hansen et al. 1994]. The General Prediction Error (GPE) of [Moody 1991], is an estimator along the same lines as FPE. Both estimators give point estimates of the output target values, and no error bars on predictions are available.

The FPE and GPE estimators address the same issues as the Bayesian approach that has been presented here, but they are not couched in a rigorous theoretical setting as the Bayesian methods. Most importantly, none of these methods require the use of validation sets, which are wasteful of data, or very computer-intensive to implement. The hope is that the estimates of performance given by these methods are sufficiently reliable to be widely applicable. The risk is, of course, that if some of the assumptions in the model are inaccurate the performance estimates may be bad. It is a strength of the Bayesian approach that all assumptions are made explicitly, making it easier to check their plausibility.

5 Conclusions

The Bayesian view of model training and selection has been presented. It has been shown how this approach can be applied to training of neural networks. Several practical approximate implementations of the Bayesian methods for training neural networks have been discussed. A few examples of application of these methods to practical modeling tasks have been mentioned.

The main strength of the Bayesian framework is that it is a consistent, theoretically well founded set of methods. All prior knowledge and assumptions can be stated explicitly, which improves the understanding of the modeling process and task. This contrasts with many of the frequentist approaches to these problems. Also, the predictions of the Bayesian models include error bars on predicted values, and a measure of relative plausibility between models.

The main difficulties of the Bayesian methods are in specifying suitable priors and carrying out the prescribed computations. The performance of a model (such as a neural network) might depend heavily upon whether it is possible to devise suitable priors for the parameters involved.

Among several approximate methods for implementing Bayesian methods, the evidence framework seems especially promising. It also compares well with the the commonly used cross validation, since it is less computationally demanding.

Thus, Bayesian methods seem appropriate for practical modelling tasks. It has yet to be demonstrated that Bayesian methods can outperform models that are trained in more conventional ways, but I think this is likely to be the case for many complicated modelling tasks.

References

Abu-Mostafa, Y. (1989). The Vapnik-Chervonenkis dimension: Information versus complexity in learning. *Neural Computation*, 1(3):312–317.

Akaike, H. (1969). Fitting autoregressive models for prediction. *Ann. Inst. Stat. Mat.*, 21:243–247.

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

Funahashi, K. I. (1989). On the approximate realization of continuous mappings by neural networks. *Neural Networks*, 2:183–192.

Geman, S., Bienenstock, E., and Doursat, R. (1992). Neural networks and the bias/variance dilemma. *Neural Computation*, 4(1):1–58.

Gull, S. F. (1989). Developments in maximum entropy data analysis. In Skilling, J., editor, *Maximum Entropy and Bayesian Methods in Science*, pages 53–71, Cambrige, 1988. Kluwer, Dordrecht.

Hansen, L. K., Rasmussen, C. E., Svarer, C., and Larsen, J. (1994). Adaptive regularization. In *IEEE Workshop on Neural Networks for Signal Processing*.

Hansen, L. K. and Salamon, P. (1990). Neural network ensembles. *IEEE Transactions on Patters Analysis and Machine Intelligence*, 12:993–1001.

Hassibi, B. and Stork, D. G. (1993). Second order derivatives for network pruning: Optimal brain surgeon. In Hanson, S. J., Cowan, J. D., and Giles, C. L., editors, *Advances in Neural Information Processing Systems*, volume 5, pages 164–171. Morgan Kaufmann.

Hinton, G. E. and van Camp, D. (1993). Keeping neural networks simple by minimizing the description length of the weights. In *Sixth ACM conference on Computational Learning Theory*, Santa Cruz.

Le Cun, Y., Denker, J. S., and Solla, S. A. (1990). Optimal brain damage. In Touretzky, D., editor, *Advances in Neural Information Processing Systems*, volume 2, pages 598–605, San Mateo. (Denver 1989), Morgan Kaufmann.

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

MacKay, D. (1992b). A practical bayesian framework for backpropagation networks. *Neural Computation*, 4(3):448–472.

MacKay, D. J. C. (1993). Hyperparameters: optimise, or integrate out? *Draft 6 – June 21*, pages 1–16.

Moody, J. (1991). Note on generalization, regularization and architecture selection in nonlinear systems. In Kung, S., Juang, B., and Kamm, C., editors, *IEEE International Conference on Neural Networks*, pages 1–10. IEEE.

Neal, R. M. (1993). Bayesian learning via stochastic dynamics. In Giles, C., Hanson, S., and Cowan, J., editors, *Neural Information Processing Systems 5*, San Mateo. Morgan Kaufmann.

Neal, R. M. (1994). Priors for infinite networks. Technical Report CRG-TR-94-1, Department of Computer Science, University of Toronto.

Nowlan, S. J. and Hinton, G. E. (1992). Adaptive soft weight tying using gaussian mixtures. In Moody, J. E., Hanson, S. J., and Lippmann, R. P., editors, *Neural Information Processing Systems 4*, pages 993–1000, San Mateo. Morgan Kaufmann.

Thodberg, H. H. (1993). Ace of bayes: Application of neural networks with pruning. *preprint*.

Williams, P. M. (1994). Bayesian regularisation and pruning using a Laplace prior. *preprint*.

Wolpert, D. H. (1993). On the use of evidence in neural networks. In Hanson, S. J., Cowan, J. D., and Giles, C. L., editors, *Neural Information Processing Systems 5*, pages 539–546. Morgan Kaufmann.