# The Structure of Bayesian Neural Network Posteriors

**Neil D. Lawrence**
Microsoft Research,
St. George House,
1 Guildhall Street,
Cambridge, CB2 3NH, U.K.
neil@thelawrences.net

**Mehdi Azzouzi**
11 rue Antoine Lumiere
69008 Lyon
France
azzouzim@hotmail.com

## Abstract

Exact inference in Bayesian neural networks is non analytic to compute and as a result approximate approaches such as the evidence procedure, Monte Carlo sampling and variational inference have been proposed. In this paper we explore the structure of the posterior distributions in a Bayesian neural network through these approximations and a new variational approximating distribution based on *mixtures* of Gaussian distributions.

## 1 Introduction

In Bayesian learning for neural networks, rather than one set of weights to represent the data, we are interested in inferring the posterior distribution of the weights given the data. Unfortunately the integrals we require to do this are non-analytic to compute and we must look to approximations. Since the introduction of the Bayesian methodology, several competing approaches have been developed: the Laplace approximation, sampling techniques and variational methods. In this paper we introduce a new variational approximation, based on mixtures and utilise it, along with traditional approaches, to explore the nature of the posterior distributions found in neural networks.

## 2 Regression Networks

We will consider a two-layer feed-forward regression neural network with $I$ input nodes, $H$ hidden nodes and a single output node. The resulting network function may then be written as: $f(\mathbf{x}, \mathbf{w}) = \sum_{h=1}^{H} v_h g(\mathbf{u}_h^{\mathrm{T}}\mathbf{x} + u_{0h}) + v_0$, where

$\mathbf{w} = \{\mathbf{u}_1 \ldots \mathbf{u}_H, \mathbf{v}\}$ is a vector representing the parameters or 'weights' of the network. The input-hidden weights are represented by $H$ vectors $\mathbf{u}_h$, each vector being the weights that 'fan-in' to hidden unit $h$. $\mathbf{v}$ is the vector of the hidden-output weights, consisting of $H$ elements $v_h$. The hidden layer biases are accounted for by a vector $\mathbf{u}_0$ and the output node bias by $v_0$. The activation function, $g$, is often taken to be a hyperbolic tangent. For reasons of tractability though, we follow Barber and Bishop [1] in our use of the cumulative Gaussian distribution activation function which has a similar form to the hyperbolic tangent.

Given a data set, $D = \{(\mathbf{x}_n, t_n\}_{n=1}^N$, we consider a likelihood function of the form

$$p(D|\mathbf{w}, \beta) = \prod_{n=1}^{N} \left(\frac{\beta}{2\pi}\right)^{1/2} \exp\left(-\frac{\beta}{2}(t_n - f(\mathbf{x}_n, \mathbf{w}))^2\right), \tag{1}$$

where $\beta$ is a parameter governing the inverse noise variance. To perform Bayesian inference we typically define a Gaussian prior distribution $p(\mathbf{w}|\alpha) = \left(\frac{\alpha}{2\pi}\right)^{K/2} \exp(-\frac{\alpha}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w})$, where $\alpha$ is a hyper parameter governing the prior and $K = H(I+1)$ is the number of weights in the model. More complex Gaussian priors can also be implemented in which the weights grouped according to their connectivity.

Bayesian inference involves determining the weight posterior

$$p(\mathbf{w}|D, \alpha, \beta) \quad = \quad \frac{1}{Z(\alpha, \beta)} \exp\left(-\frac{\beta}{2}\sum_{n=1}^{N}(t_n - f(\mathbf{x}_n, \mathbf{w}))^2 - \frac{\alpha}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w}\right) \tag{2}$$

where $Z$ is a normalisation constant. Network predictions on unseen data are obtained by looking at the expected value of the output under the posterior distribution, $\langle f(\mathbf{x}, \mathbf{w})\rangle_{p(\mathbf{w}|D)} = \int f(\mathbf{x}, \mathbf{w}) p(\mathbf{w}|D)\ d\mathbf{w}$. The variance of predictions, $\langle f(\mathbf{x}, \mathbf{w})^2\rangle - \langle f(\mathbf{x}, \mathbf{w})\rangle^2$ may also be obtained and used to place error-bars on the predictions.

Unfortunately computation of the normalisation constant $Z$, which as well as being necessary for computation of the posterior is needed for used for model selection as the evidence, involves a high dimensional non-linear integral. We must therefore look to approximations to the posterior to make progress. For Bayesian neural networks three principal strategies exist.

**Markov chain Monte-Carlo sampling.** Sampling approximations consist of approximating an integral, $\hat{I} = \int f(\mathbf{w}) p(\mathbf{w}|D) d\mathbf{w}$, with a finite sum, $\hat{I} \approx \frac{1}{S}\sum_{s=1}^{S} f(\mathbf{w}_n)$, where $\{\mathbf{w}_s\}_{s=1}^S$ are samples from the posterior distribution $p(\mathbf{w}|D)$. The approximation becomes exact in the limit $S \to \infty$. The major difficulty relies on finding a representative set of samples. A Markov chain Monte-Carlo algorithm creates a Markov chain which converges to the desired distribution. Radford Neal's implementation of a hybrid Monte-Carlo technique has proved highly effective in neural networks [9].

**The Laplace approximation.** The evidence procedure [7] involves making the Laplace approximation to the posterior. This involves finding a single local optimal point $\mathbf{w}^*$ in weight space and constructing a full covariance Gaussian approximation to the posterior distribution around this point.

$$p(\mathbf{w}|D, \alpha, \beta) \approx \frac{|\mathbf{H}|^{\frac{1}{2}}}{(2\pi)^{\frac{K}{2}}} \exp\left\{-\frac{1}{2}(\mathbf{w} - \mathbf{w}^*)^{\mathrm{T}}\mathbf{H}(\mathbf{w} - \mathbf{w}^*)\right\}, \tag{3}$$

where the matrix $\mathbf{H}$ is the Hessian matrix at $\mathbf{w}^*$. While the Laplace approximation provides a quick and easy method of approximating the posterior distribution, the

approximation is only locally valid, it is concentrated at a mode of the posterior. We will show later that the resulting approximation can be unrepresentative of the mass of the posterior distribution, even in the mode which the approximation finds.

**Ensemble Learning.** Ensemble learning was first applied in the context of neural networks by Hinton and van Camp [2], where it was introduced through the implementation of the minimum description length principle. The approximation to the posterior in this approach is more responsive to the mass of the distribution than that of the evidence procedure and has the additional advantage of providing a rigourous lower bound on the model likelihood.

Ensemble learning can be viewed as a special case of the general framework of variational techniques for inference in probabilistic models [5]. Consider the following marginal likelihood for a Bayesian neural network

$$p(D|\alpha,\beta) \quad = \quad \int p(D|\mathbf{w},\beta)p(\mathbf{w}|\alpha)\ d\mathbf{w}. \tag{4}$$

This likelihood may be bounded from below through the introduction of a variational distribution $q(\mathbf{w})$,

$$\ln p(D|\alpha,\beta) \geq \int q(\mathbf{w}) \ln p(D,\mathbf{w}|\alpha,\beta) + \mathcal{H}\left(q(\mathbf{w})\right)\ d\mathbf{w}, \tag{5}$$

where $\mathcal{H}(\cdot)$ is the entropy of the distribution. The difference between this bound and the true likelihood can easily be shown to be the Kullback-Leibler (KL) divergence between the approximating distribution $q(\mathbf{w})$ and the true posterior, $p(\mathbf{w}|D)$. The bound (5) is maximised when the KL-divergence is minimised.

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

Barber and Bishop used a full covariance Gaussian in their implementation and their approximation is thus able to account for correlations in the posterior. However, for such an approximation, some integrations are not tractable and the use of lookup tables or numerical integrals is required. The efficient implementation of this approach remains an unresolved issue.

In order to more effectively model the posterior distribution, we propose to implement a mixture distribution, $q_w(\mathbf{w}) = \sum_{m=1}^{M} Q_w(m) q_w(\mathbf{w}|m)$, where each component, $q_w(\mathbf{w}|m) = \mathcal{N}(\bar{\mathbf{w}}_m, \mathbf{\Sigma}_m)$, is a diagonal covariance Gaussian. This representation is able to capture correlations between the weights and would also be able to represent skewness or kurtosis in the posterior. In practice however, strong correlations between weights may require many components for a good approximation to the posterior distribution and it may not be practical to optimise such large numbers of components. It is possible to extend our approach for mixtures of full covariance Gaussians, however here we constrain ourselves to implementation of diagonal covariance Gaussians.

The use of the mixture distribution does not complicate the calculation of the first term in bound (5) beyond what has been tackled in previous works. However, the entropy of the mixture distribution presents other problems, namely we are required to evaluate the expectation of the logarithm of a sum. We make progress by considering the mutual information, $\mathcal{I}(m;\mathbf{w})$, between the component label $m$

of the mixture distribution and an observed weight vector. Whilst we may not efficiently compute this term, a lower bound, first introduced in the context of discrete systems by Jaakkola and Jordan [3] and applied to probabilistic graphical models by Lawrence *et al.* [6], on the likelihood may be computed. For details of the implementation of this bound for mixtures of Gaussians see [6].

We can utilise the lower bound on the mixtures entropy to formulate a rigorous lower bound on the model log-likelihood. For diagonal covariance Gaussian distributions and gamma hyper-priors, all but one of the resulting bound's expectations prove to be analytically tractable. The intractable expectation concerns the square of the output function $\langle f(\mathbf{x}, \mathbf{w})^2 \rangle_{q_w}$ for which an approximation is required [6].

## 3 Structure of the Posterior

Having given a brief outline of the approximating techniques utilised we now make use of a simple toy problem to investigate the structure of the weight posterior. We compare hybrid Monte Carlo sampling, the evidence procedure, a diagonal covariance variational approximation and our mixture approximation to explore the nature of the posterior. To aid comparison of the different Bayesian approaches, all the models were implemented with the prior which places weights within four groups: input-hidden layer weights, hidden biases, hidden-output layer weights and output biases. A hyper parameter, $\alpha_\gamma$, was then associated with each of the groups giving a a vector of hyper parameters, $\boldsymbol{\alpha}$.

For our data set, we considered samples from the function $h(x) = 0.5 + 0.4\sin(2\pi x)$ with additive Gaussian noise of standard deviation 0.05. We generated thirty data points. The $x$ position of the data points was determined by sampling fifteen points from a uniform distribution between 0.125 and 0.375 and a further fifteen points from a uniform distribution between 0.625 and 0.875. We used networks containing cumulative Gaussian activation functions. The networks contained five hidden nodes.

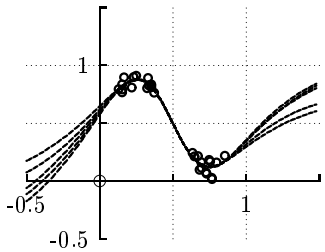


Figure 1: The expected output under each component of the mixture approximation.

We first trained a network using the *evidence procedure*. Each hyper parameter, $\alpha_\gamma$, was initialised as 0.1 and the parameter governing the inverse noise variance, $\beta$, was initialised as 50. The weight parameters, $\mathbf{w}$, were then optimised for a maximum of 500 iterations in a scaled conjugate gradient optimiser followed by an update of $\boldsymbol{\alpha}$ and $\beta$ [7]. This process was repeated ten times. Normally, the quality of the resulting approximation can be estimated by considering the expected output of the network function under the posterior distribution and comparing it to the data. However, practitioners of the evidence procedure utilise a further approximation to obtain this expectation, $\langle f(\mathbf{x}, \mathbf{w}) \rangle \approx f(\mathbf{x}, \mathbf{w}^*)$. Since the point estimate $\mathbf{w}^*$ was obtained as a MAP solution for the network this approximation generally matches the data well. However, $f(\mathbf{x}, \mathbf{w}^*)$ is often an extremely poor approximation to the true expectation of the network function. Figure 2(a) shows $f(\mathbf{x}, \hat{\mathbf{w}}_s)$ for different weight vectors, $\hat{\mathbf{w}}_s$, sampled from the approximating distribution. The mean of these samples, which is in effect a sample based approximation to $\langle f(\mathbf{x}, \mathbf{w}) \rangle$, is also shown, but is off scale for most of the plot. For two layer regression neural networks this expectation can also be calculated [6] and is plotted, but indistinguishable from the sample based approximation.

We next initialised a *diagonal covariance Gaussian* variational approximation with a mean equal to the mean of the Laplace approximation and variances of $1 \times 10^{-6}$. We trained the resulting model for 500 iterations of a scaled conjugate gradient optimiser. We fixed the expected values of $\boldsymbol{\alpha}$ and $\beta$ to those learnt in the evidence procedure. In Figure 2(b) we show $\langle f(\mathbf{x}, \mathbf{w}) \rangle_{q(\mathbf{w})}$ as well as the expected standard deviation of the network function. Samples were also taken form this posterior to illustrate how well they match the data.

For the *mixture approximation* we used five components. Each component's mean was initialised with that learnt from the diagonal covariance Gaussian approximation, with Gaussian noise added. The standard deviation of the added noise was taken to 1% of the absolute value of the mean. If the noise was not added the components of the mixture model still separated but took longer to do so. Again fixing the hyper parameters to the values determined using the evidence procedure, we optimised the models in the following way: 1) Maximise the lower bound on the entropy of the mixture distribution. 2) Perform fifty iterations of scaled conjugate gradient optimisation with respect to the means and variances of the mixture approximation. 3) Optimise the mixing coefficients using a fixed point equation. This procedure was repeated ten times. The resulting expectation of the network function under each component, $\langle f(\mathbf{x}, \mathbf{w}) \rangle_{q(\mathbf{w}|m)}$, of the approximating distribution is shown in Figure 1. Note how the components have separated in regions of low data density. Figure 2(c) shows the overall expectation and the expected standard deviation as well as samples from the approximating distribution.

Finally we applied *hybrid Monte Carlo* (HMC) sampling. We initialised the sampler at the mode found in the Laplace approximation and used the values for the hyper parameters found by the evidence procedure. We then obtained samples from the posterior distribution using hybrid Monte Carlo sampling without persistence. We obtained 100 samples using leapfrogs of 100. Figure 2(d) shows the samples from the true posterior using hybrid Monte Carlo techniques (faint dotted lines).

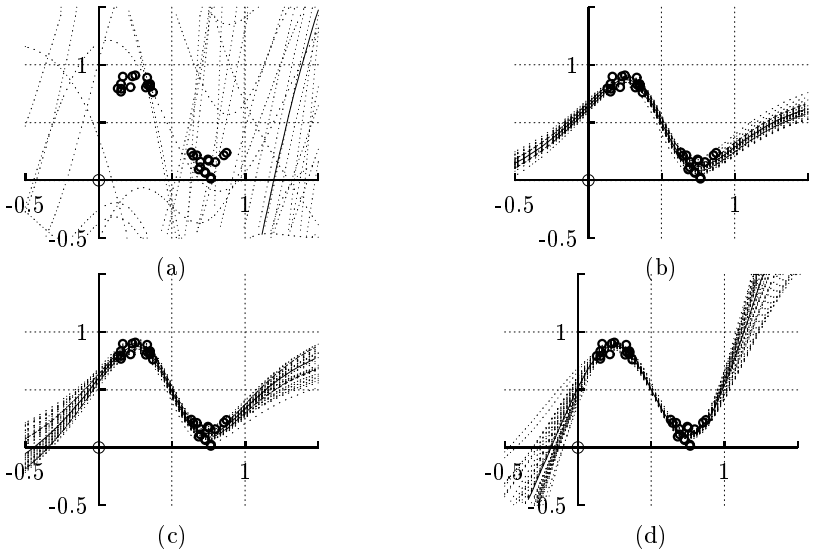


Figure 2: Samples from the approximations to the posterior (light dotted lines) and the expected value of the network function (solid lines) for (a) Laplace approximation, (b) variational diagonal covariance Gaussian, (c) variational mixture approximation and (d) HMC sampling.

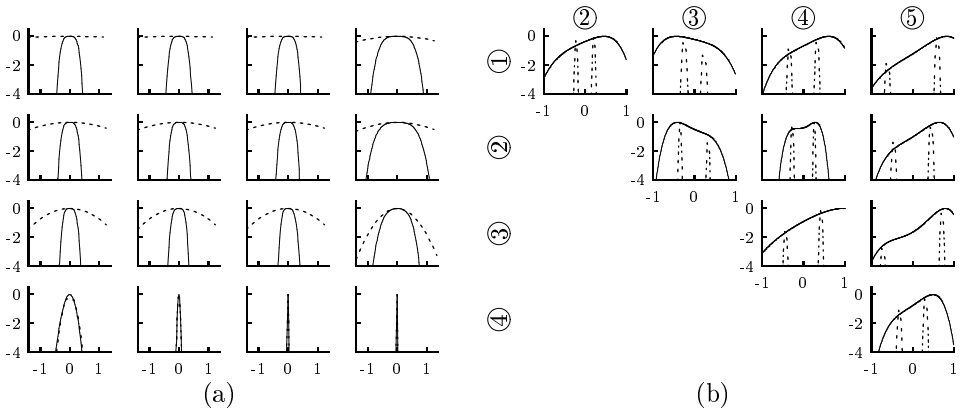


Figure 3: (a) The logarithm of the Laplace approximation (dotted line) plotted alongside the logarithm of the true posterior (up to a constant offset) for a regression neural network (solid lines). The plots show the logarithms plotted along the eigenvalues of the covariance matrix for the Laplace approximation. The plots are displayed in the order of highest eigenvalue first (top left), lowest last (bottom right). Each plot is centred at the mode, $\mathbf{w}^*$. In the last four plots the approximation overlays the true value. The curves are shown with offsets so that they overlay. (b) The logarithm of the variational mixture approximation (dotted line) plotted alongside the logarithm of the true posterior (again up to a constant offset) for a regression neural network (solid line). The curves are plotted between each possible combination of two components. The circled numbers across the top and along the left of the plots specify the components between which the logarithms are plotted. The plots are centred at the mid point between each pairing of the approximation's components and once again offsets have been applied to the curves.

**The Poor Quality of the Laplace Approximation Samples.** We investigated the Laplace approximation further by plotting the logarithm of the approximation along the eigenvectors of its covariance matrix. We compared these values to the true logarithm of the posterior[1]. The resulting plots are shown in Figure 3. We note that in the directions associated with the covariance's larger eigenvalues the approximation is very poor. To better represent the mass of the distribution these eigenvalues should be smaller. This explains the poor samples we observe in Figure 2(c).

**The Posterior in the Region of the Mixture Approximation.** To contrast the nature of the Laplace approximation and our mixture based approximation, we have also plotted, in Figure 3, the logarithm of the posterior in the region of the mixture approximation. The plots are made along lines between each possible pairing of the components of the mixture distribution. Note in particular the narrow widths associated with the components of the mixture. This is the same effect as that visible in Figure 4, which is a contour plot of the minimum KL divergence solution between a highly correlated Gaussian and a mixture of two diagonal covariance Gaussians. This is a result of the expectations being taken under the approximating distribution as opposed to under the distribution to be approximated. The fact that

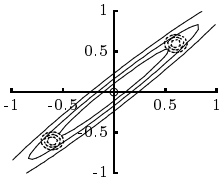

Figure 4: A minimum KL divergence fit of a mixture of diagonal Gaussians to a highly correlated Gaussian.

[1]The true logarithm of the posterior may be plotted subject to a constant offset which arises from the intractable normalisation term.

the components are all fairly narrow in Figure 3 indicates
that the posterior is narrow in the directions which are out of the plane of the paper, this prevents each component from better filling the posterior. This in turn suggests strong correlations in the posterior and as a result we would require many more components to represent the posterior accurately.

## 4 Discussion

In this paper we have reviewed three Bayesian framework for regression in neural networks and utilised them to explore the nature of the weight posterior in these models. We have shown that the posterior can be highly correlated. This means that it is difficult to represent the mass of the distribution accurately with a mixture approximation based on diagonal covariance Gaussians. We also showed that the Laplace approximation over-estimates the length of the posterior modes in the directions associated with the larger eigenvalues of the covariance matrix. As a result samples from the Laplace approximation are not representative of the true posterior distribution. Note that this does not affect the approximation that the evidence procedure utilises to make network predictions in this framework; this approximation does not utilise the covariance of the Laplace approximation. However, the over-estimation of the covariance will have a knock on effect in the estimation of the hyper parameters, $\boldsymbol{\alpha}$, which govern the weight prior. As a result of the over-estimation the hyper parameters will be under-estimated. This will lead to less regularisation of the network function and hence to the possibility of over-fitting. This may be one of the reasons that, for neural networks, Markov chain Monte Carlo methods often out-perform the Laplace approximation.

## Acknowledgements

The experiments utilised portions of code from Ian Nabney's NETLAB package [8].

## References

[1] D. Barber and C. M. Bishop. Ensemble learning in Bayesian neural networks. In C. M. Bishop, editor, *Neural Networks and Machine Learning*, volume 168 of *Series F: Computer and Systems Sciences*, pages 215–237, Berlin, 1998. Springer-Verlag.

[2] G. E. Hinton and D. van Camp. Keeping neural networks simple by minimizing the description length of the weights. In *Proceedings of the Sixth Anuual Conference on Computational Learning Theory*, pages 5–13, 1993.

[3] T. S. Jaakkola and M. I. Jordan. Improving the mean field approximation via the use of mixture distributions. In Jordan [4], pages 163–174.

[4] M. I. Jordan, editor. *Learning in Graphical Models*, volume 89 of *Series D: Behavioural and Social Sciences*, Dordrecht, The Netherlands, 1998. Kluwer.

[5] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. In Jordan [4], pages 105–162.

[6] N. D. Lawrence and M. Azzouzi. A variational Bayesian committee of neural networks. Available from `http://www.thelawrences.net/neil`, 1999.

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

[8] I. T. Nabney. *Netlab: Algorithms for Pattern Recognition*. Advances in Pattern Recognition. Springer, 2001. Code available from `http://www.ncrg.aston.ac.uk/netlab/`.

[9] R. M. Neal. *Bayesian Learning for Neural Networks*. PhD thesis, University of Toronto, Canada, 1994.