Ace of Bayes: Application of Neural Networks with Pruning

Hans Henrik Thodberg
The Danish Meat Research Institute
Maglegaardsvej 2, DK-4000 Roskilde
e-mail `thodberg@nn.meatre.dk`

Manuscript 1132E, May 19, 1993

Abstract

MacKay's Bayesian framework for backpropagation is a practical and powerful means of improving the generalisation ability of neural networks. The framework is reviewed and extended in a pedagogical way. The notation is simplified using the ordinary weight decay parameter, and the noise parameter β is shown to be nothing more than an overall scale. A detailed and explicit procedure for adjusting several weight decay parameters is given.

Pruning is incorporated into the Bayesian framework. Appropriate symmetry factors on sparse architectures are deduced. Bayesian weight decay is demonstrated using artificial data generated by a sparsely connected network. Pruning yields computational advantages: by removing unimportant weights the posterior weight distribution becomes Gaussian, and pruning removes zero-modes of the Hessian and redundant hidden units. In addition, pruning improves generalisation. The Bayesian evidence is used as a stop criterion for pruning.

Bayesian backprop is applied in the prediction of fat content in minced meat from near infrared spectra. It outperforms "early stopping" as well as quadratic regression. The evidence of a committee of differently trained networks is computed and the corresponding improved generalisation is verified. The error bars on the predictions of the fat content are computed. There are three contributors: The random noise, the uncertainty in the weights, and the deviation among the committee members. Finally the Bayesian framework is compared to Moody's GPE.

1 Introduction

1.1 Highlights

Bayesian neural networks or simply Bayesian backprop is the application of MacKay's Bayesian framework to feedforward networks trained by supervised learning, i.e. by the backpropagation algorithm. The main benefits are:

- Bayesian backprop adjusts the weight decay parameters automatically to their near-optimal values yielding the best generalisation. The adjustment is done during training, so the tedious and computer-intensive search for weight decay parameters is no longer needed.

- The instrument of Bayesian analysis is the training data and the network itself. No separate validation set is used. Hence all available data can be used for training, giving better models.

- The Bayesian formalism estimates the *evidence* for each model. The evidence measures how probable the model is, given the data. The evidence correlates well with the generalisation error and is used as a quality measure to select the best model. Networks with different initial weights and different numbers of nodes, layers and inputs can be compared and ranked according to the evidence. The evidence can also be used as a stop criterion for network growing or pruning.

1.2 The Proper Treatment of Bayesian Backprop

Bayesian backprop was introduced by MacKay in 1991 [1, 2, 3, 4] as a radically different approach to the problem of overfitting and model comparison. The Bayesian framework for backprop has its origin in the field of Maximum Entropy [5] which develops better models for the analysis of empirical data from chemistry, geophysics, astronomy, physics, economics etc., and the approach is both theoretical and applied. However, the maximum entropy principle plays a minor if any role in Bayesian backprop.

Bayesian backprop is based on the Bayesian school of statistics [6, 7]. This is distinct from the main-stream "sampling theory" statistics, where the concept of probability must be attached to frequencies of samples drawn from a distribution. In contrast, Bayesians also use probabilities to describe *degrees of belief* in parameter values or models. With this interpretation Bayes' formula acquires a deeper meaning since it is now a way to compute beliefs.

The Bayesian analysis computes the evidence which expresses our posterior belief in each model. Empirically the evidence (anti)correlates well with the generalisation error of the model, so the evidence can be used as a quality measure for models. At present it is an "incomplete" framework, because the relation between evidence and test error cannot be proved within a closed framework. It is conceivable that it is fundamentally impossible to prove this relationship. There may be a complementarity (in the sense used by Niels Bohr) between the depth of a relation and the possibility of proving it within a closed mathematical framework, analogous with Gödel's Theorem. Supporting this conjecture is the simple fact that the evidence contains some subjective prior beliefs, and it seems contradictory to prove a relation to the test error which is objective.

That a formalism cannot be deduced does not mean that it isn't true, relevant, useful or scientific. It can be regarded as a metaprinciple and this implies that the field must always consider the following two aspects in conjunction:

- The theoretical Bayesian framework must be presented in a convincing and consistent manner.

- The relation between the test error and the evidence must be studied for a wide range of problems.

The first aspect gives the evidence a truth of its own, while the second brings it into contact with reality. If the applied aspect reveals inconsistencies, it can lead to a revision of the theoretical framework.

MacKay's papers display a good balance between these two aspects. Other workers in the field have dealt exclusively with the second aspect and presented alternative Bayesian approaches to neural networks [8, 9]. Wolpert [9] argues on purely theoretical grounds that his alternative treatment of the weight priors is more correct. However, he presents no simulations, so irrespective of the correctness of his theory, it is unclear what kind of problems it is suitable for [20].

There is no doubt that MacKay's aim was that the Bayesian techniques should be used in real-life applications. However, this has not yet happened in the case of neural networks. This paper is intended to bridge the gap between academic theory and its industrial application. The proper balance between the theoretical and applied aspects is emphasised. MacKay's formalism is recapitulated using a simpler choice of parameters which makes contact with ordinary weight decay. The mechanism of multiple weight decay parameters and the estimation of the Hessian are spelled out at the implementation level. The formalism is illustrated on artificial as well as real-life data. While MacKay studied learning problems with one or two input units, we consider problems with 8-10 inputs. This is a more appropriate domain of application for backprop.

The use of many input units leads naturally to the issue of limited receptive fields of the hidden units. This is explored by marrying Bayesian backprop with pruning. Pruning not only cures some of the computational problems in the formalism but can also lead to improved generalisation. With pruning, the number of different architectures is vastly increased and the need for an objective quality measure is accentuated. The Bayesian evidence meets this need adequately.

1.3 Ockham's Razor

Bayesian model comparison addresses a question which has been on the agenda for centuries. William of Ockham (1285-1349) was already analysing and comparing theories of Nature and God. At the time, philosophers were keen on unifying Aristotelian philosophy with Catholicism. William did not like over-complicated thinking systems – why mix God into an explanation of a physical phenomenon? He systematically applied a principle of economy in explanations, which was later given the name Ockham's Razor:

> *If several theories account for a phenomenon we should prefer the simplest which describes the data sufficiently well.*

The principle states that a model has two virtues: simplicity and goodness of fit. But William did not specify the meaning of "sufficiently well" - the optimal trade-off between the two virtues was left to the intuition of the scientist. People are in fact rather good at assessing this trade-off in situations related to daily life. However, in other matters, like abstract scientific theories and neural network models, it is useful to formalise this trade-off.

Bayesian model comparison does just that. We express our belief in a model as its probability given the data, and use Bayes' formula:

$$P(\mathcal{H} \,|\, D) = \frac{P(D \,|\, \mathcal{H}) P(\mathcal{H})}{P(D)} \tag{1}$$

We assume that the prior belief $P(\mathcal{H})$ is the same for all models, so we can compare models by comparing $P(D \,|\, \mathcal{H})$ which is called *the evidence for* $\mathcal{H}$, and acts as a quality measure in Bayesian model comparison. To compute $P(D \,|\, \mathcal{H})$ we integrate over the tunable parameters $\mathbf{w}$ of the model:

$$P(D \,|\, \mathcal{H}) = \int P(D \,|\, \mathbf{w}, \mathcal{H}) P(\mathbf{w} \,|\, \mathcal{H}) d\mathbf{w}$$

The integrand is assumed to possess a pronounced maximum at $\mathbf{w}_{\mathrm{MP}}$ of width $\Delta\mathbf{w}_{\mathrm{posterior}}$, and writing the prior probability for $\mathbf{w}$ as $P(\mathbf{w} \mid \mathcal{H}) = 1/\Delta\mathbf{w}_{\mathrm{prior}}$ we get:

$$\begin{aligned} P(D \,|\, \mathcal{H}) &= P(D \,|\, \mathbf{w}_{\mathrm{MP}}, \mathcal{H}) \frac{\Delta\mathbf{w}_{\mathrm{posterior}}}{\Delta\mathbf{w}_{\mathrm{prior}}} \\ \mathrm{Evidence} &= \mathrm{Likelihood} \times \mathrm{OckhamFactor} \\ \mathrm{ModelQuality} &= \mathrm{DataFit} \times \mathrm{Simplicity} \end{aligned}$$

We have split the evidence for the model into two factors:

- The best fit likelihood, i.e. the probability of the data given the model and the tuned parameters. It measures how well the *tuned* model fits the data.

- The integrated probability of the tuned model parameters with their uncertainties, i.e. the collapse of the available parameter space when the data is taken into account. This factor is small when the model has many parameters or when some parameters must be tuned very accurately to fit the data. This measures the simplicity of the model and is therefore called *the Ockham Factor*.

Thus the Bayesian framework is essentially a mathematical formulation of Ockham's Razor.

1.4 Bayesian Model Comparison and Headache

Let's illustrate Bayesian model comparison with an example from daily life. A man has frequent headaches. He decides to record the dates of his headaches in his calendar. After one year he analyses the pattern of dates. At first sight the headaches seem to occur randomly on average every 4 days.

The man is influenced by the Danish astronomer Tycho Brahe (1546-1601) who calculated 32 days of the year where accidents are more likely to occur. A day where more things than usual go wrong is still today called a Tycho Brahe day.

The man looks for regularities in the dates of the headaches and discovers a periodic phenomenon: Among other days, he had a headache every 72 days starting on a particular date. He regards these five periodic days of headache as a separate phenomenon. However, his wife tells him to stop having pity with himself. She rejects that he is a victim of some supernatural agent and attributes the five periodic headaches to a coincidence.

Since women have a better intuition than men he embarks on a Bayesian comparison of the two alternative explanations of the phenomenon $D = \{$the five periodic days of headache$\}$. The two explanations are:

Random Model. All headache days occur randomly and independently with a probability $\frac{1}{4}$ which is assumed to be known prior to the modelling from the statistics on the whole year. This model does not regard the five periodic days as a separate phenomenon. They are of the same origin as the rest of the headache days. The model has no tunable parameters so the evidence is simply the likelihood: $P(D \,|\, \mathcal{H}_{\text{random}}) = (\frac{1}{4})^5 = \frac{1}{1024}$.

Tycho Brahe Model. Some intangible mechanism with a bearing on the Tycho Brahe days gives periodic headaches. The model has two tunable integer parameters: the period and the starting date. We limit the scope of the model to encompass periods from 20 to 100 days with equal prior probability $\frac{1}{80}$. To tune the model to the data we fix the period at 72 days which gives[1] an Ockham factor $\frac{1}{80}$. Then we fix the starting day to one of the 72 possibilities which gives an Ockham factor $\frac{1}{72}$. The likelihood at the tuned parameters is 1, so the evidence equals the Ockham Factor: $P(D \,|\, \mathcal{H}_{\text{Tycho}}) = \frac{1}{80}\frac{1}{72} = \frac{1}{5760}$.

Thus the evidence favours the random model. This reasoning is based on equal prior probabilities of the models. You are allowed to assign lower odds to the Tycho Brahe model, but the evidence itself has sufficient power to favour the random model.

With many parameters and observations the evidences for different models span several orders of magnitude so that the subjective priors of the models play a negligible role.

2 Elements of Bayesian Backprop

2.1 Probabilistic Interpretation of Neural Networks

The Likelihood

The training set D consists of N cases of the form $(\mathbf{x}, t)$ assumed to be independently drawn from a distribution $P(\mathbf{x}, t)$. We model t as a function of $\mathbf{x}$,

[1] A more consistent Bayesian treatment considers the period s to be a scale parameter (see section 2.2) with a uniform prior probability of $\log s$ over an interval of width $\log 5$. The prior becomes $P(s) = \frac{1}{s \log 5}$, i.e. the Ockham factor becomes $\frac{1}{72 \log 5}$

$t = y(\mathbf{x}) + \nu$, assuming the noise ν to be Gaussian with a mean of zero. (Often there is noise $\mathbf{n}$ on the input data so that $t = y(\mathbf{x} + \mathbf{n}) + \nu$. We then approximate $t = y(\mathbf{x}) + \partial y/\partial\mathbf{x} \cdot \mathbf{n} + \nu$ and assume that the noise term $\partial y/\partial\mathbf{x} \cdot \mathbf{n} + \nu$ is Gaussian and $\mathbf{x}$-independent. Then the same formalism applies.)

A neural network $\mathcal{H}$ with weights $\mathbf{w}$ is used to implement $y(\mathbf{x})$ which is an estimate of $E(t\,|\,\mathbf{x})$, the *regression* of t on $\mathbf{x}$. The noise scale is a free parameter $\beta = 1/\sigma_\nu^2$. The probability of a single datum under this model is:

$$P(t\,|\,\mathbf{x}, \mathbf{w}, \beta, \mathcal{H}) = \sqrt{\frac{\beta}{2\pi}} \exp(-\tfrac{1}{2}\beta(y-t)^2)$$

The data error is defined as

$$E_D \equiv \tfrac{1}{2}\sum(y-t)^2 \tag{2}$$

where the sum extends over the N cases. The formalism also applies to models with several outputs with equal noise level. Then the sum in (2) extends over outputs as well as cases and N is the number of terms.

The probability of the data, called *the likelihood*, is

$$\begin{aligned} P(D\,|\,\mathbf{w}, \beta, \mathcal{H}) &= \frac{1}{Z_D(\beta)} \exp(-\beta E_D) \\ Z_D(\beta) &\equiv (\frac{2\pi}{\beta})^{N/2} \end{aligned}$$

Here $P(D \mid \ldots)$ denotes the probability of the t values. The $\mathbf{x}$ values are understood to be part of the condition, but are suppressed in the notation.

The Prior Weight Distribution

So far we have described the likelihood of ordinary statistics. We now introduce the genuinely Bayesian features, starting with the prior distribution of the model parameters. The model contains k adjustable parameters $\mathbf{w}$, called weights, which are in general split into G groups $\{\mathcal{G}_g, g = 1, \ldots, G\}$ with k_g weights in group g. A weight is assigned a prior Gaussian distribution:

$$P(w\,|\,\beta, \xi_g, \mathcal{H}) = \sqrt{\frac{\beta\xi_g}{2\pi}} \exp(-\tfrac{1}{2}\beta\xi_g w^2) \text{ for } w \in \mathcal{G}_g$$

where ξ_g will turn out to be *the weight decay parameter* for group g.

This prior is based on our experience that positive and negative weights are equally frequent, that smaller weights are more frequent than larger ones, and that very large weights are very unlikely. The Gaussian prior is a natural description of such a compact distribution. It excludes network solutions with a few extremely large weights. It is a matter of experience that we do fine without these as long as the regression is reasonably smooth.

We call β and the ξ_g's *the scales* of the model. The scales are free parameters determined by the data but on a higher level than the weights. Defining what we metaphorically call the "weight error"

$$E_{Wg} \equiv \tfrac{1}{2} \sum_{w \in \mathcal{G}_g} w^2$$

the prior of all the weights $\mathbf{w}$ is

$$
\begin{aligned}
P(\mathbf{w} \mid \beta, \Xi, \mathcal{H}) &= \frac{1}{Z_W(\beta,\Xi)} \exp(-\sum_{g=1}^{G} \beta \xi_g E_{W_g}) \\
Z_W(\beta,\Xi) &\equiv \prod_{g=1}^{G} (\frac{2\pi}{\beta \xi_g})^{k_g/2}
\end{aligned}
$$

where $\Xi = \{\xi_1, \ldots, \xi_G\}$.

Separate weight decay parameters for the layers of a network are needed for several reasons. The simplest reason is that the inputs and outputs have an arbitrary scale. If all inputs are multiplied by 2, the weights of the first layer must be divided by 2 to keep the mapping unchanged and hence the weight decay parameter must be multiplied by 4 to maintain its strength. Usually we do our best to scale the data properly. We center the inputs and scale them to a standard deviation of $\frac{1}{4}$. This is done to match the scale of the hidden unit outputs, which are in the interval $(-\frac{1}{2}, \frac{1}{2})$. Similarly, in regression (where we apply linear output units) the target t is centred and scaled to unit standard deviation to match a typical net input to a hidden unit. But even after these preparations, the optimal weight decay parameters typically differ from each other by factors up to 10.

The Posterior Weight Distribution

We want to adjust the weights to their most probable values given the data, some values of the scales (to be determined later) and the model. We compute the posterior weight distribution using Bayes' rule (the *swapping rule*) in its general form $P(A \mid B, F) = P(B \mid A, F)P(A \mid F)/P(B \mid F)$:

$$
\begin{aligned}
P(\mathbf{w} \mid D, \beta, \Xi, \mathcal{H}) &= \frac{P(D \mid \mathbf{w}, \beta, \Xi, \mathcal{H}) P(\mathbf{w} \mid \beta, \Xi, \mathcal{H})}{P(D \mid \beta, \Xi, \mathcal{H})} \\
&= \frac{\exp(-\beta C)}{Z_C(\beta, \Xi, \mathcal{H})} \qquad (3)
\end{aligned}
$$

with[2]

$$C \equiv E_D + \sum_g \xi_g E_{W_g} \qquad (4)$$

$$Z_C(\beta, \Xi, \mathcal{H}) \equiv \int \exp(-\beta C) d\mathbf{w} \qquad (5)$$

C is recognised as the well-known cost function for backprop with weight decay. So *the most probable – or maximum posterior – weights according to the probabilistic interpretation are identical to the weights obtained by minimising the cost function C*. Assuming the maximum posterior value to be a good approximation to the average of the posterior distribution we have a nice Bayesian account for the success of weight decay. It has been used for linear models in statistics for many years under the name ridge regression [10].

[2]MacKay uses $\alpha_g = \beta \xi_g$ and β as independent parameters rather than (β, ξ_g), but the answers are the same.

2.2 Determination of Scales

The Posterior Probability of the Scales

Above we determined the most probable (or maximum posterior, MP) weights given some values of the scales, which we haven't determined yet. We now determine the MP scales given the model and the data. To this end we use the MP weights found earlier. In fact we are setting up a loop, where we alternately determine

- the MP weights given the MP scales, the data and model
- the MP scales given the data and the model.

This converges unless the model is completely wrong (see section 7.3). Section 4.3 discusses these two levels of inference in perspective.

To evaluate the posterior for a generic scale s, we swap the scale and the data:

$$P(\log s \,|\, D, \mathcal{H}) = \frac{P(D \,|\, s, \mathcal{H}) P(\log s \,|\, \mathcal{H})}{P(D \,|\, \mathcal{H})}$$

We write $P(\log s \mid \ldots)$, to indicate a probability density for $\log s$, which is related to the density for s by $P(s \,|\, \ldots) ds = P(\log s \,|\, \ldots) d \log s$. (When s is in the condition it makes of course no difference whether we use s or $\log s$.) A scale describes the ratio of two quantities in the model. A weight decay parameters is the ratio between two kinds of error, and β is the ratio between the noise level and the arbitrary unit of the target values t. It should make no difference whether we model the ratio or its inverse, i.e. the prior should have the same form. This implies that the prior and the posterior are naturally expressed in terms of the logarithm of the scale. We assume that the prior $P(\log s \,|\, \mathcal{H})$ is a parameter in an interval containing $\log s_{\mathrm{MP}}$. Then s_{MP} maximises $P(D \,|\, s, \mathcal{H})$.

We call $P(D \,|\, \mathbf{w}, s, \mathcal{H})$ *the likelihood of the* $\mathbf{w}$*-parameters*, following standard terminology. We denote $P(D \mid s, \mathcal{H})$ *the evidence for* $(s, \mathcal{H})$ because it is used as a quality measure for the model $(s, \mathcal{H})$.

To compute the evidence we use *the rule of divide and conquer* $P(A) = \sum_i P(A \,|\, B_i) P(B_i)$ where $\{B_i\}$ is a division of the whole:

$$\begin{aligned} P(D \,|\, \beta, \Xi, H) &= \int P(D \,|\, \mathbf{w}, \beta, \Xi, \mathcal{H}) P(\mathbf{w} \,|\, \beta, \Xi, \mathcal{H}) d\mathbf{w} \\ &= \int \frac{\exp(-\beta E_D)}{Z_D(\beta)} \frac{\exp(-\sum_{g=1}^{G} \beta \xi_g E_{W g})}{Z_W(\beta, \Xi)} d\mathbf{w} \\ &= \frac{Z_C(\beta, \Xi, \mathcal{H})}{Z_D(\beta) Z_W(\beta, \Xi)} \end{aligned} \tag{6}$$

This is the evidence for $(\beta, \Xi, \mathcal{H})$.

The Gaussian Approximation to the Evidence

To evaluate Z_C we approximate the cost function by the quadratic form $C(\mathbf{w}) = C(\mathbf{w}_{\mathrm{MP}}) + \frac{1}{2}(\mathbf{w} - \mathbf{w}_{\mathrm{MP}})^{\mathrm{T}}(\mathbf{B} + \sum_g \xi_g \mathbf{I}_g)(\mathbf{w} - \mathbf{w}_{\mathrm{MP}})$. $\mathbf{B}$ is the Hessian matrix of the data error, $\mathbf{B} \equiv \nabla\nabla E_D$, or

$$B_{ij} \equiv \frac{\partial^2 E_D(\mathbf{w})}{\partial w_i \partial w_j}$$

evaluated at $\mathbf{w}_{\mathrm{MP}}$ and $\mathbf{I}_g = \nabla\nabla E_{W_g}$ is a diagonal matrix which is zero except that $(\mathbf{I}_g)_{ii} = 1$ for $w_i \in \mathcal{G}_g$.

Then Z_C of equation (5) is a Gaussian integral[3]

$$Z_C(\beta, \Xi, \mathcal{H}) = \exp(-\beta C) \frac{(2\pi)^{k/2}}{\sqrt{\det \beta(\mathbf{B} + \sum_g \xi_g \mathbf{I}_g)}} \tag{7}$$

(where C is evaluated at $\mathbf{w}_{\mathrm{MP}}$). Now from (6) we get

$$\begin{aligned} \log P(D \,|\, \beta, \Xi, \mathcal{H}) = & - \beta(E_D + \sum_g \xi_g E_{W_g}) \\ & - \frac{N}{2} \log \frac{2\pi}{\beta} - \sum_g \frac{k_g}{2} \log \frac{1}{\xi_g} \\ & - \tfrac{1}{2} \mathrm{Tr} \log(\mathbf{B} + \sum_g \xi_g \mathbf{I}_g) \end{aligned} \tag{8}$$

Differentiation of the Evidence

The MP noise scale is determined by differentiation:

$$\frac{\partial}{\partial \beta} \log P(D \,|\, \beta, \Xi, \mathcal{H}) = -(E_D + \sum_g \xi_g E_W^g) + \frac{N}{2\beta} \tag{9}$$

Setting this to zero gives:

$$\beta_{\mathrm{MP}} = \frac{N}{2C} \tag{10}$$

We also want the MP values of the weight decay parameters, so we differentiate (8) with respect to ξ_g:

$$\frac{\partial}{\partial \xi_g} \log P(D \,|\, \beta, \Xi, \mathcal{H}) = \frac{k_g}{2\xi_g} - \beta E_{W_g} - \tfrac{1}{2} \mathrm{Tr} \frac{1}{\mathbf{B} + \sum_g \xi_g \mathbf{I}_g} \mathbf{I}_g \tag{11}$$

Here we assumed $\partial \mathbf{B}/\partial \xi_g = 0$[4].

[3] In the following we use some algebra for symmetric, positive definite matrices: $\log\det\mathbf{B} = \mathrm{Tr}\log\mathbf{B}$ and $(d/ds)\mathrm{Tr}\log\mathbf{B} = \mathrm{Tr}\mathbf{B}^{-1}(d/ds)\mathbf{B}$ and $\int \exp(-\frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{B}\mathbf{w}) d^k\mathbf{w} = (2\pi)^{k/2} \frac{1}{\sqrt{\det \mathbf{B}}}$.

[4] This approximation is not always good. When ξ is increased, $\mathbf{w}_{\mathrm{MP}}$ changes. The Hessian $\nabla\nabla E_D$ is evaluated at $\mathbf{w} = \mathbf{w}_{\mathrm{MP}}$ and depends on $\mathbf{w}$ except when E_D is a quadratic form as in a linear network. The approximation $\partial B/\partial \xi_g = 0$ is at the heart of MacKay's formula (16) for ξ_{MP}. The approximation can be tested in the following way (in the case of one weight group or fixed weight scales): Determine the evidence (23) for $\log\xi = \log\xi_{\mathrm{MP}}$ given by by (16). Then evaluate the evidences at $\log\xi = \log\xi_{\mathrm{MP}} \pm 4\sigma_{\log\xi}$, remembering that changing ξ requires a new value of $\mathbf{w}_{\mathrm{MP}}$, i.e. the network must be retrained to the new minimum of C. Then fit the evidence to a quadratic function of $\log\xi$ and check whether the maximum occurs close to ξ_{MP}.

Diagonalising the Hessian

To evaluate (8) and (11) we diagonalise the total Hessian $\mathbf{B}+\sum_g \xi_g \mathbf{I}_g$. We want to do this via diagonalising the Hessian of the data error $\mathbf{B}$. This is possible if all weight decay parameters are equal. We therefore reparametrise the network by changing the weights and the ξ_g's:

$$\begin{aligned} w &\rightarrow w/s_g \text{ for } w \in \mathcal{G}_g \\ \xi_g &\rightarrow \xi \end{aligned}$$

where the *weight scales* are defined

$$s_g \equiv \sqrt{\xi/\xi_g} \tag{12}$$

and $\xi \equiv \xi_1$. Quantities like $\xi_g E_{Wg}$, E_D, β and $\log P(D \mid \beta, \Xi, \mathcal{H})$ are invariant under this reparametrisation.

We diagonalise $\mathbf{B}$ of the reparametrised network. Let the eigenvalues be $\{\lambda_i, i = 1, \ldots, k\}$ with corresponding eigenvectors $\{\mathbf{s}_i\}$. We introduce the matrices $\mathbf{\Lambda} \equiv \mathrm{diag}(\lambda_i)+\xi\mathbf{I}$, and $\mathbf{S} \equiv (\mathbf{s}_1, \ldots, \mathbf{s}_k)$, with $\mathbf{S}^T\mathbf{S} = \mathbf{I}$. From $(\mathbf{B}+\xi\mathbf{I})\mathbf{S} = \mathbf{S}\mathbf{\Lambda}$ we get $\frac{1}{\mathbf{B}+\xi\mathbf{I}} = \mathbf{S}\mathbf{\Lambda}^{-1}\mathbf{S}^T$ so

$$\begin{aligned} \mathrm{Tr}\frac{1}{\mathbf{B}+\xi\mathbf{I}}\mathbf{I}_g &= \mathrm{Tr}(\frac{1}{\mathbf{\Lambda}}\mathbf{S}^T\mathbf{I}_g\mathbf{S}) \\ &= \sum_{i=1}^{k} \frac{1}{\lambda_i+\xi}(\mathbf{S}^T\mathbf{I}_g\mathbf{S})_{ii} \end{aligned}$$

The quantity $\pi_g^i \equiv (\mathbf{S}^T\mathbf{I}_g\mathbf{S})_{ii} = \sum_j S_{ji}(I_g)_{jj}S_{ji} = \sum_{j\in\mathcal{G}_g} S_{ji}^2$ is the square of the projection of eigenvector $\mathbf{s}_i$ on the subspace of weight group g.

Determination of ξ

Setting (11) to zero determines the MP values of ξ_g:

$$2\xi_g\beta E_{Wg} = k_g - \xi_g \mathrm{Tr}\frac{1}{\mathbf{B}+\sum_g \xi_g \mathbf{I}_g}\mathbf{I}_g$$

The first two terms in this equation are invariant under the reparametrisation, therefore the last term must be as well, so we can evaluate it in reparametrised form:

$$\begin{aligned} \xi_g \mathrm{Tr}\frac{1}{\mathbf{B}+\sum_g \xi_g \mathbf{I}_g}\mathbf{I}_g &= \xi \mathrm{Tr}\frac{1}{\mathbf{B}+\xi\mathbf{I}}\mathbf{I}_g \\ &= \sum_i \frac{\xi}{\lambda_i+\xi}\pi_g^i \end{aligned}$$

Using $k_g = \sum_i \pi_g^i$ we now get for the MP value of ξ_g, $\xi_{g\mathrm{MP}}$:

$$2\beta\xi_{g\mathrm{MP}} E_{Wg} = \gamma_g \tag{13}$$

where we have introduced the *number of well-determined parameters in weight group g*

$$\gamma_g \equiv \sum_i \frac{\lambda_i}{\lambda_i + \xi_{\mathrm{MP}}} \pi_g^i \tag{14}$$

Negative λ_i's are omitted from this sum. We also introduce the *certainty of weight group g*

$$\omega_g \equiv \frac{\gamma_g}{k_g} \tag{15}$$

From (10) and (13) we get

$$\frac{\xi_{g\mathrm{MP}} E_{Wg}}{C} = \frac{\gamma_g}{N} \tag{16}$$

for $g = 1, \ldots, G$. This is a main result by MacKay. It states that *the proportion of the cost due to the weight error should be the number of well-determined weights per training example.* To reestimate $\xi_{g\mathrm{MP}}$ during training we recast these expressions into

$$\xi_{g\mathrm{MP}} = \frac{E_D}{E_{Wg}} \frac{\gamma_g}{N - \sum_h \gamma_h} \tag{17}$$

2.3 The Ockham Factors for the Scales

For a generic scale parameter s, the logarithm of the evidence for $(s, \mathcal{H})$ is assumed to be a quadratic form in $\log s$, the natural scale variable:

$$\log P(D \,|\, s, \mathcal{H}) = \log P(D \,|\, s_{\mathrm{MP}}, \mathcal{H}) - \tfrac{1}{2} \frac{(\log s - \log s_{\mathrm{MP}})^2}{\sigma^2_{\log s}}$$

where $(\sigma_{\log s})^{-2} = -s^2 \frac{\partial^2}{\partial s^2} \log P(D \,|\, s, \mathcal{H})$ at $s = s_{\mathrm{MP}}$, so

$$\begin{aligned} P(D \,|\, \mathcal{H}) &= \int P(D \,|\, s, \mathcal{H}) P(\log s \,|\, \mathcal{H}) d\log s \\ &= P(D \,|\, s_{\mathrm{MP}}, \mathcal{H}) \int \exp(-\tfrac{1}{2} \frac{(\log s - \log s_{\mathrm{MP}})^2}{\sigma^2_{\log s}}) \frac{1}{\log \Omega} d\log s \end{aligned}$$

which gives

$$P(D \,|\, \mathcal{H}) = P(D \,|\, s_{\mathrm{MP}}, \mathcal{H}) \mathrm{Ock}(s) \tag{18}$$

$$\mathrm{Ock}(s) = \frac{\sqrt{2\pi} \sigma_{\log s}}{\log \Omega} \tag{19}$$

We assumed that the prior of the scale is uniform on the logarithmic scale within an interval of length $\log \Omega$ containing the MP value. The parameter Ω is set to 10^3, which is a subjective estimate of the prior ignorance of a scale.

Ock(p) denotes the *Ockham Factor* for a parameter p. It accounts for the uncertainty of the parameter value after tuning it to the data relative to the prior uncertainty:

$$\mathrm{Ock}(p) = \frac{\sigma_{\mathrm{posterior}}(p)}{\sigma_{\mathrm{prior}}(p)}$$

It costs an Ockham Factor to *marginalise* (i.e. "integrate away") a parameter from the evidence. A parameter which is determined well by the data gives a small Ockham Factor and hence a large penalty in the evidence. It may seem unfair to punish a model for the accuracy of its parameters when these are estimated from the data. But, in fact, it agrees with common sense. We don't like explanations which are accurately tuned to the phenomena in question. We regard them as improbable or contrived.

From (9) and (11) we find[5]

$$\sigma_{\log\beta} = \sqrt{\frac{2}{N-\sum_g \gamma_g}} \tag{20}$$

$$\sigma_{\log\xi_g} = \sqrt{2/\gamma_g} \tag{21}$$

2.4 Symmetries of the Network

When evaluating Z_C in equation (7) we did not take into account that a minimum of C at $\mathbf{w}$ can be obtained with other $\mathbf{w}$'s related to $\mathbf{w}$ by two kinds of symmetries:

- By changing the sign of all incoming and outgoing weights of a hidden unit, we get an identical mapping (we are using a hidden unit activation function which is symmetric about zero). With h hidden units this gives 2^h realisations of the solution.

- The hidden units can be permuted according to the symmetry group of the network topology. For a fully connected network with h hidden units in one hidden layer this gives $h!$ times as many realisations of the solution. For a general network topology we have $S_{\mathcal{H}}$ realisations, where $S_{\mathcal{H}}$ is the order of $\mathcal{H}$'s symmetry group.

The Gaussian integral in Z_C measures the posterior $\mathbf{w}$ volume while the Gaussian integral Z_W measures the prior $\mathbf{w}$ volume. The ratio of these volumes is the Ockham Factor for $\mathbf{w}$. All solutions are covered by Z_W so if we disregard the symmetric solutions in Z_C, the Ockham Factor is underestimated — we punish the model too much. The prior volume includes all the symmetric solutions, so the posterior volume must as well. Therefore Z_C must be multiplied by the factor

$$2^h S_{\mathcal{H}} \tag{22}$$

The solution itself possesses a symmetry if two hidden units have identical weights to and from the unit, or in general whenever the posterior distribution for the weights for two units overlap to a significant degree. Then the factor (22) must be reduced accordingly. This complicates the computation of the evidence and may even deteriorate the generalisation ability. Pruning can be used to eliminate such symmetries.

---

[5]Using just $\partial^2/(\partial\beta)^2 \log P(D \mid \beta, \Xi, \mathcal{H})$ gives $\sigma_{\log\beta} = \sqrt{2/N}$. But the quantity $\partial^2/(\partial\beta\partial\xi_g) \log P(D \mid \beta, \Xi, \mathcal{H})$ is non–negligible. The correct posterior volume for the scales obtained from the determinant of the Jacobian $\partial^2/\partial(\beta,\xi_g) \log P(D \mid \beta, \Xi, \mathcal{H})$ is equivalent to (20,21) in agreement with [1]. Equation (21) is not exact; in the Gaussian approximation $\sigma_{\log\xi_g}$ lies between $\sqrt{2/k_g}$ and $\sqrt{2/\gamma_g}$

2.5 Marginalisation of Scales

From (8) we infer that the term $-\sum_g \frac{k_g}{2}\log\frac{1}{\xi_g} - \frac{1}{2}\mathrm{Tr}\log(\mathbf{B}+\sum_g \xi_g \mathbf{I}_g)$ is reparametrisation invariant. We evaluate it in reparametrised form:

$$
\begin{aligned}
-\sum_g \frac{k_g}{2}\log\frac{1}{\xi_g} - \tfrac{1}{2}\mathrm{Tr}\log(\mathbf{B}+\sum_g \xi_g\mathbf{I}_g) &= \frac{k}{2}\log\frac{1}{\xi} - \tfrac{1}{2}\mathrm{Tr}\log(\mathbf{B}+\xi\mathbf{I}) \\
&= \tfrac{1}{2}\sum_i \log\frac{\xi}{\lambda_i+\xi}
\end{aligned}
$$

The evidence for $(\Xi, \mathcal{H})$ is obtained from the evidence for $(\beta, \Xi, \mathcal{H})$ (8) by marginalisation of β. We apply the general rule (18) with (20) and (10):

$$
\begin{aligned}
\log P(D\,|\,\Xi,\mathcal{H}) &= -\frac{N-\gamma}{2} - \frac{N}{2}\log\frac{4\pi C}{N} \\
&\quad + \log \mathrm{Ock}(\mathbf{w}) + \log \mathrm{Ock}(\beta) && (23) \\
\log \mathrm{Ock}(\mathbf{w}) &= \tfrac{1}{2}\sum_{i=1}^{k}\log\frac{\xi}{\xi+\lambda_i} - \frac{\gamma}{2} \\
&\quad + \log S_{\mathcal{H}} + h\log 2 && (24) \\
\mathrm{Ock}(\beta) &= \frac{\sqrt{4\pi/(N-\gamma)}}{\log\Omega} && (25) \\
\gamma &\equiv \sum_h \gamma_h && (26)
\end{aligned}
$$

The first line in (23) is the log likelihood[6]. The Ockham Factor for the weights $\mathrm{Ock}(\mathbf{w})$ shows that large eigenvalues λ_i, corresponding to well-determined weights, contribute a small Ockham Factor, i.e. a low evidence for the model.

An eigenvalue $\lambda_i = -\xi$ corresponds to a zero eigenvalue of $\nabla\nabla C$. This occurs for instance when the cost in weight space is like a ball in a Mexican hat potential: it can move around in the valley without changing the cost.

Eigenvalues close to or smaller than $-\xi$ are called zero-modes and they render the Ockham Factor for $\mathbf{w}$ divergent. Therefore a cut-off Z is introduced: If a factor $\sqrt{\frac{\xi}{\xi+\lambda_i}}$ exceeds Z it is truncated to Z. The simulations use $Z = 1$. This is consistent with the neglect of negative eigenvalues in (14)[7].

Finally the ξ_g's are marginalised according to (16)

$$
\begin{aligned}
\log P(D\,|\,\mathcal{H}) &= \log P(D\,|\,\Xi_{\mathrm{MP}},\mathcal{H}) + \sum_g \log \mathrm{Ock}(\xi_g) && (27) \\
\mathrm{Ock}(\xi_g) &= \frac{\sqrt{4\pi/\gamma_g}}{\log\Omega} && (28)
\end{aligned}
$$

[6] This employs (16) so this interpretation is valid only at the MP value of Ξ.

[7] The uncertainty of the evidence from the zero-modes is estimated by evaluating the evidence $\mathrm{Ev}(Z)$ for different values of Z: Uncertainty $= \frac{1}{2}(\log \mathrm{Ev}(2Z) - \log \mathrm{Ev}(Z/2))$. The zero-modes are partly due to numerical problems, e.g. computing the Hessian from finite differences of gradients. But the zero-modes are also real, indicating wide posterior weight distributions. We still want to cut these off at some point because they lead to overlapping posterior distributions for two units in conflict with the factor (22).

2.6 The Trade-off in Bayesian Weight Decay

Let us illustrate how Bayes deduces the optimal value (16) of the ξ's. We simplify matters by assuming a single weight group. We focus on the factors in the evidence involving ξ and get from (23) and (24):

$$P(D\,|\,\xi, \mathcal{H}) \propto \frac{\prod_i \sqrt{\frac{\xi}{\xi+\lambda_i}}}{(E_D + \xi E_W)^{\frac{N}{2}}}$$

This is the ratio between the Ockham Factor and the cost to some positive power. The ξ is chosen to maximise the above expression, or minimise:

$$P(D\,|\,\xi, \mathcal{H})^{-\frac{2}{N}} \propto (E_D + \xi E_W) \prod_i (1 + \frac{\lambda_i}{\xi})^{\frac{1}{N}}$$

Here $E_D + \xi E_W$ is the cost at $\mathbf{w}_{\mathrm{MP}}$, where $\mathbf{w}_{\mathrm{MP}}$ depends on ξ. Thus ξ is determined by minimising the MP cost function multiplied by a factor that penalises many weights and large eigenvalues. When ξ is increased, the MP cost increases while the penalty factor decreases. MacKay's formula (16) locates the value of ξ where the loss in the first factor balances the gain in the second.

3 Sparse Architecture and Pruning

The real strength of neural networks is their ability to handle a high-dimensional input. A fully connected network using all the inputs is often fairly efficient, and it is certainly easy to simulate on an ordinary computer. But under some circumstances it is important to optimise the architecture along the lines of Ockham's Razor. If the processing can be done sufficiently well using instead a sparsely connected network this should improve generalisation. This includes eliminating some inputs all together.

In this section we extend the Bayesian framework to a consistent treatment of sparsely connected networks. We discuss how weight decay copes with networks with too many connections. Then we present the benefits of taking Ockham's "Razor" literally and pruning insignificant connections of a fully connected network. The evidence is monitored during pruning and used to select the network which implements the optimal trade-off between data fit and model simplicity. There are several examples in the literature demonstrating that sparse networks can generalise better than fully connected networks [11, 12, 13, 14, 15, 16].

3.1 The Prior for Sparsely Connected Networks

There is a problem with the factor $S_{\mathcal{H}}$ in the Ockham Factor for the weights (24). To see this, consider a fully connected network $\mathcal{H}$. Assume that E_D is independent of a certain weight w, i.e. $w_{\mathrm{MP}} = 0$, $\frac{\partial E_D}{\partial w} = 0$. Then w corresponds to an eigenvector of $\nabla\nabla E_D$ with zero eigenvalue.

Now we remove the connection governed by w. This gives a new model $\mathcal{H}'$ with a different architecture. To compute the evidence for $\mathcal{H}'$ we note that C

and $\sum_{i=1}^{k} \log \frac{\xi}{\xi+\lambda_i}$ are the same for $\mathcal{H}$ and $\mathcal{H}'$. But $\mathcal{H}'$ is not invariant under permutations involving the hidden unit attached to the pruned connection, so $S_{\mathcal{H}'} = S_{\mathcal{H}}/h$, and the evidence has decreased:

$$P(D\,|\,\mathcal{H}') = \frac{1}{h} P(D\,|\,\mathcal{H})$$

although the two models are effectively the same. This is not a fair quality measure for model comparison!

To resolve this inconsistency we return to equation (1) and consider $P(\mathcal{H}\,|\,D)$ which is more fundamental than the evidence. If we require $P(\mathcal{H}\,|\,D) = P(\mathcal{H}'\,|\,D)$ we infer that the priors are not equal:

$$P(\mathcal{H}') = hP(\mathcal{H})$$

In general this consistency argument leads to the following relation between the priors of networks with one hidden layer with h units

$$P(\mathcal{H}_h) = \frac{h!}{S_{\mathcal{H}_h}} P(\mathcal{H}_{h,\text{fully}}) \tag{29}$$

where $\mathcal{H}_{h,\text{fully}}$ is the fully connected network.

We can also account for these model priors in a more "deductive" way. Consider networks with 2 input units and 3 hidden units and 1 output unit. The sparsely connected network in figure 1 is shown in its 6 different realisations, while the fully connected network in figure 2 has only one realisation. In general $\mathcal{H}_h$ has $h!/S_{\mathcal{H}_h}$ realisations. We now appeal to the principle that all models have equal prior probabilities. This principle was used already in section (1.3). By interpreting each realisation as a separate model we confirm (29)[8]. Finally, the simulations in section 5.3 also support (29).

Model comparison should include these priors. To simplify our notation we absorb these priors into a modified evidence, which we distinguish from the strict evidence with the notation

$$\text{Ev}(\mathcal{H}) \equiv \frac{h!}{S_{\mathcal{H}_n}} P(D\,|\,\mathcal{H}) \tag{30}$$

but still call *the evidence for* $\mathcal{H}$. For fully connected networks $\text{Ev}(\mathcal{H}) = P(D\,|\,\mathcal{H})$[9].

[8]These symmetry factors have an analogy in physics. The Feynmann diagrams of quantum field theory, which are used to compute probabilities for elementary processes, possess symmetry factors $1/S_{\mathcal{H}}$. They penalise symmetric diagrams relative to less symmetric in the same way as for neural networks.

[9]The requirement of consistency in model comparison can be taken further by asking: What are the relative priors of fully connected networks with different numbers of hidden units? In the definition of the "consistent evidence" (30) it was tacitly assumed that

$$P(\mathcal{H}_{h,\text{fully}}) = P(\mathcal{H}_0) \tag{31}$$

where $\mathcal{H}_0$ denotes a network with no hidden units. Arguments similar to the ones used to arrive at (30) lead to the alternative prior $P(\mathcal{H}_{h,\text{fully}}) = \frac{1}{h!} P(\mathcal{H}_0)$. MacKay used the prior (31) in [2] and his data (in particular Fig. 10) support (31). We have repeated these simulations and the correlation between test error and evidence is significantly better with (31) than with the alternative prior. Therefore we use the prior (31).

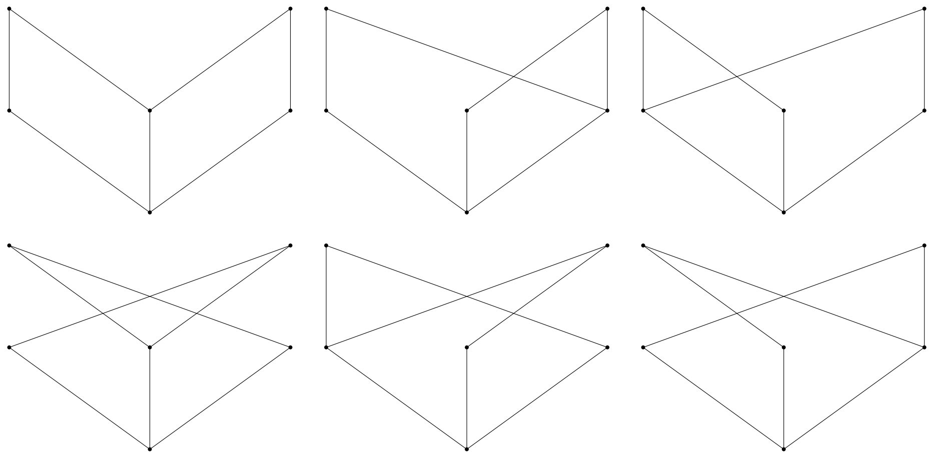

Figure 1: *The 6 different realisations of the same sparsely connected network architecture (the input units are on top and the output unit on the bottom). Consequently the prior probability is 6 times larger than for the network in figure 2.*

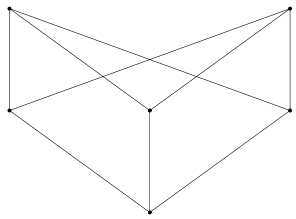

Figure 2: *A fully connected network like this is symmetric with respect to permutations of the hidden units and hence has just a single realisation*

We have now demonstrated our pragmatic attitude to Bayesian modelling. Priors are not considered absolute quantities but are subject to change when experience or consistency demands it. We adhere to the saying of the former Danish Prime Minister J.O.Krag (1914-1978): *"You have a standpoint until you take another"*.

3.2 Benefits of Bayesian Weight Decay

Before describing the effect of pruning it is worthwhile to consider the effect of weight decay for networks which have too many connections. Depending on the learning problem, two scenarios can emerge:

- Many irrelevant weights are driven to near-zero values. This happens in the learning problem generated by a sparsely connected network described in section 5.3.
- The hidden units become redundant, i.e. they tend to form the same internal representations. This happens in MacKay's robot arm problem [2].

The simulations presented later demonstrate that Bayesian weight decay to a large extent can rescue over-capacity networks, i.e. make them generalise almost optimally.

3.3 Benefits of Bayesian Pruning

Pruning can be used in conjunction with weight decay and gives several benefits. There are three computational benefits:

- Weight decay can produce a posterior weight distribution composed of two Gaussians, a narrow for the near-zero weights and a wide for the rest. This conflicts with the assumption of a single Gaussian prior and leads to an underestimation of the evidence (demonstrated in the simulations). By pruning near-zero connections, a single Gaussian posterior emerges and the evidence is reliable[10].
- Pruning reduces the number of zero-modes of the Hessian. The zero-modes tend to invalidate the Gaussian approximation. Such a problem is indicated by a strong dependence of the evidence on the cut-off Z (section 2.5). Zero-modes also cause computational problems with error bars on predictions (section 4.4).
- Pruning reduces the number of redundant hidden units and in general reduces the overlap of the posterior distributions of the weights of the units. These create symmetries of the network that conflict with the explicit symmetry factors $S_{\mathcal{H}}$ in (24). So redundant hidden units render the evaluation of the evidence less reliable.

Other benefits of pruning are

- Pruning improves generalisation beyond the capability of weight decay (see the simulations).
- Pruning gives insight into the nature of the problem by displaying which inputs are used and in what combinations.

So Bayesian backprop benefits a lot from pruning. On the other hand, pruning benefits from the Bayesian method which provides the evidence as a stop criterion for pruning.

3.4 Alleviating the Bias/Variance Dilemma

Geman *et al* [17] introduced the *Bias/Variance Dilemma* for neural networks. They argue as follows: Neural networks are nearly *un-biased*, they can realise any mapping, but the price is a large number of parameters. With a limited training set these models become poorly estimated, and this gives a large so-called *variance* contribution to the generalisation error. They conclude that

[10] Soft Weight Sharing [14] models the weight distribution as a mixture of several components with a dynamic membership. Typically, one component contains the "pruned" connections. The Ockham Factor of this group of weights is 1 because these weights are equally close to zero before and after the model is tuned. Hence the evidence can be estimated correctly.

neural networks are not useful in practice, since it always pays off to include domain-specific knowledge in the data representation and the model.

Pruning alleviates this dilemma by introducing *levels of learning* or *hierarchical learning* into the learning machine. The levels are:

1. The search for architecture.

2. The estimation of parameters.

First the data are used to discover the optimal architectural bias. This can include elimination of some inputs from the network. The search in the space of architectures is appropriately performed with a pruning algorithm. Secondly, the data is used to tune the remaining parameters.

The hierarchical learning alleviates the Bias/Variance Dilemma by discovering the bias on the first level, and tuning the biased model on the second. In many application areas – like nonlinear regression in chemistry – the procedure finds the optimal bias. This is a matter of experience and can be attributed to the sigmoid nonlinearity being more common than e.g. polynomial relationships. Once the model is correctly biased, it yields low variance error (and high evidence) on the second level.

4 Further topics of Bayesian Backprop

4.1 The Evidence for a Committee of Networks

For a given data set we usually train several networks with different numbers of hidden units and different initial weights. Several of these networks have evidence near or at the maximal value, but the networks differ in their predictions. This is a source of distress to some: How can neural networks be useful if they don't agree between themselves? However, the probabilistic (or Bayesian) interpretation naturally turns this ambiguity into an advantage. The different solutions are interpreted as components of the posterior distribution and the correct Bayesian answer is obtained by summing over the solutions weighted by their posterior probabilities, i.e. their evidences. However, this is not practical because the evidence is not accurately determined, due to various approximations. Instead a so-called *committee* is defined as the set of networks with maximal evidence within the uncertainty of the evidence. Section 7.1 describes how this uncertainty can be estimated.

The committee gives two advantages.

- Predictions using the average of the committee gives a better generalisation than the average network in the committee [18].

- The degree of dissent within the committee contributes to the uncertainty of the predictions. By including this we get more reliable error bars on the predictions. This is treated in section 4.4 [8].

Here we shed light on the first item by computing the evidence of the committee. Assume for the sake of simplicity that all networks in the committee $\mathcal{C}$ are of the same architecture. We estimate the number $N_{\mathcal{C}}$ of truly different solutions in the committee. Of course we count symmetric realisations only once. The posterior volume, i.e. the Ockhams Factor, for the weights is now $N_{\mathcal{C}}$ times larger. This renders the committee more probable – it has a larger evidence:

$$\log \mathrm{Ev}(\mathcal{C}) = \log N_{\mathcal{C}} + \log \mathrm{Ev}(\mathcal{H}) \tag{32}$$

where $\log \mathrm{Ev}(\mathcal{H})$ denotes the average log evidence of the members.

Since the evidence is highly correlated with the generalisation error, we expect the committee to generalise better than the committee members. This is confirmed by the simulations in section 5.5.

4.2 The Meaning of the Evidence

Empirically there is a high correlation between the evidence and the generalisation error. According to common sense, it is not surprising that the most probable model of the data is the best at making predictions in the same domain and indeed the correlation is usually higher at the upper end of the evidence scale. But we have not deduced this relation.

In practice we alter the set of models to maximise the correlation between evidence and test error (especially at high evidence), but there are several reasons for this correlation to be imperfect. They are presented here together with some means to examine or overcome them:

- If the test set is small, it gives an uncertain estimate of the true test error. This effect can be eliminated by using a test set which is much larger than the training set, as is done in the study of artificial data in section 5.3.

- Often there are multiple solutions with maximum evidence which have somewhat different test errors. The solution is to use a committee.

- Some models contain assumptions which are in conflict with the data, e.g. the weights or the residual errors are not Gaussian. The evidence of such models is underestimated. This situation calls for a modification of the models. It is a symptom that the best model may not have been found yet, due to wrong assumptions in the modelling. However, in the end we select the models with the highest evidence, and it is fortunate that this type of error shifts models towards lower value of evidence. In other words, it is acceptable to have some inconsistent models. MacKay [2] has emphasised that the evidence formalism is an efficient analysis tool for discovering inconsistencies in the models so that better models can be constructed.

- Often the training set is limited so that the model parameters and the evidence are poorly estimated. This is a general condition of inference from small samples. The evidence is particularly well suited to the problem of model selection under these circumstances, because it encompasses a universal and consistent treatment of the overfitting problem.

- There are approximations in the evidence which may be more or less accurate, in particular approximating E_D as a quadratic form near $\mathbf{w}_{\mathrm{MP}}$. In addition the MP solution may not be representative of the posterior distribution, i.e. the value of a predicted quantity at the mode (the maximum) is not always a good approximation to the integral over the posterior distribution. To overcome these problems Neal has developed a Monte Carlo technique [19].

- The evidence as a quality measure could reflect a mixture of virtues, of which the generalisation error is just one. Other virtues could be correct architecture, explanatory power and other concepts related to *truth*. The simulations provide an example of this.

4.3 The Four Levels of Bayesian Inference

A good overview of the Bayesian framework for backpropagation networks is obtained by outlining the four levels of inference following MacKay [1]:

Level 1 Make predictions including error bars for new input data.

Level 2 Estimate the weight parameters and their uncertainties.

Level 3 Estimate the scale parameters and their uncertainties

Level 4 Select the $\mathbf{w}$-minimum and the network architecture. Optionally select a committee to reflect the uncertainty on this level.

A key concept of Bayesian inference is the posterior uncertainty. Large uncertainties on level 2–4 are reflected by large Ockham factors. Large uncertainty on level 1, i.e. large $1/\beta$, gives small likelihood. Since the evidence is the product of the Ockham factors and the likelihood, we draw the remarkable conclusion that large uncertainties on level 2–4 gives *large* evidence while large uncertainty on level 1 gives *small* evidence.

We start the description on the highest level, level 4. When we talk about a model $\mathcal{H}$ we mean a particular architecture and the environment of a particular minimum $\mathbf{w}_{\mathrm{MP}}$. We expand the concept of a model by the *committee* $\mathcal{C}$, as described in section 4.1. The simplest kind of committee is a particular architecture and the environments of the best minima. The committee is a natural Bayesian construct since it introduces a measure of uncertainty on level 4.

Level 3 deals with the inference of the the scale parameters β and ξ_g. It was found that the posterior distribution of the scales has a narrow peak if there are many observations and parameters (equations (21) and (20)).

Level 2 determines the weight parameters. The Bayesian weights are given by the posterior distribution $P(\mathbf{w} \mid D, \mathcal{H})$. This is a complicated entity and we are seeking a compact description in terms of the centre of gravity of the relevant part of it, and some uncertainty. MacKay has clarified [20] that the true maximum posterior value is a poor approximation to the centre of gravity. The desired description is instead obtained by introducing the extra level 3 of inference where the MP scales are determined, and then on level 2 defining $\mathbf{w}_{\mathrm{MP}}$ as the maximum of $P(\mathbf{w} \mid \beta_{\mathrm{MP}}, \Xi_{\mathrm{MP}}, D, \mathcal{H})$. The uncertainty is given by the Hessian evaluated at $\mathbf{w}_{\mathrm{MP}}$.

Level 1 deals with the predictions of the model on new data which is the typical purpose of the modelling. For a given value of the weights $\mathbf{w}$ the network implements a mapping of input $\mathbf{x}$ onto a prediction $y(\mathbf{x};\mathbf{w})$. This mapping is an estimate of $E(t|\mathbf{x})$, the *regression* of t on $\mathbf{x}$. As described in section 2.1 our modelling assumes that the actual output t is obtained by adding to $y(\mathbf{x};\mathbf{w})$ an independent Gaussian noise term ν with variance $1/\beta_{\mathrm{MP}} = \sigma_\nu^2$, which describes the uncertainty due to the presumed noise process.

4.4 Bayesian Error Bars on Predictions

In Bayesian inference the model is not just given by the MP weight $\mathbf{w}_{\mathrm{MP}}$. We also take into account the posterior distribution of weights given in (3):

$$\begin{aligned} P_{\text{posterior}}(\mathbf{w}) &\propto \exp(-\beta_{\mathrm{MP}} C) \\ &= \exp(-\tfrac{1}{2}(\mathbf{w}-\mathbf{w}_{\mathrm{MP}})^{\mathrm{T}}\beta_{\mathrm{MP}}(\mathbf{B}+\xi_{\mathrm{MP}}\mathbf{I})(\mathbf{w}-\mathbf{w}_{\mathrm{MP}})) \end{aligned} \quad (33)$$

where $\mathbf{B} = \nabla\nabla E_D$ is the Hessian of the data error and the weights are scaled to make all weight decay parameters equal ξ_{MP}. Eqn.(33) is a k-dimensional normal distribution with mean $\mathbf{w}_{\mathrm{MP}}$ and covariance matrix $\mathbf{D} = (\beta_{\mathrm{MP}}(\mathbf{B}+\xi_{\mathrm{MP}}\mathbf{I}))^{-1}$. The mean over the posterior weight distribution of the network prediction is denoted $y(\mathbf{x})$ and the variance of the network prediction due to the weight uncertainty (WU) is denoted $\sigma_{\mathrm{WU}}(\mathbf{x})^2$. These are given by:

$$\begin{aligned} y(\mathbf{x}) &= \int y(\mathbf{x};\mathbf{w}) P_{\text{posterior}}(\mathbf{w}) d\mathbf{w} \\ \sigma_{\mathrm{WU}}(\mathbf{x})^2 &= \int (y(\mathbf{x};\mathbf{w}) - y(\mathbf{x}))^2 P_{\text{posterior}}(\mathbf{w}) d\mathbf{w} \end{aligned}$$

To compute these integrals we assume that the network predictions depend *linearly on* $\mathbf{w}$ within the bulk of the posterior $\mathbf{w}$ distribution:

$$y(\mathbf{x};\mathbf{w}) = y(\mathbf{x};\mathbf{w}_{\mathrm{MP}}) + \mathbf{g}(\mathbf{x}) \cdot (\mathbf{w} - \mathbf{w}_{\mathrm{MP}})$$

where $\mathbf{g}(\mathbf{x}) = \frac{\partial y(\mathbf{x};\mathbf{w}=\mathbf{w}_{\mathrm{MP}})}{\partial \mathbf{w}}$. Then the average prediction equals the prediction at $\mathbf{w}_{\mathrm{MP}}$: $y(\mathbf{x}) = y(\mathbf{x};\mathbf{w}_{\mathrm{MP}})$.

The WU variance of the prediction is computed by introducing the variables $\mathbf{u} \equiv \mathbf{w} - \mathbf{w}_{\mathrm{MP}}$ and the normalisation constant $a = \sqrt{(2\pi)^k \det \mathbf{D}}$:

$$\begin{aligned} \sigma_{\mathrm{WU}}(\mathbf{x})^2 &= \int (\mathbf{g}(\mathbf{x})\cdot\mathbf{u})^2 \frac{1}{a} \exp(-\tfrac{1}{2}\mathbf{u}^{\mathrm{T}}\frac{1}{\mathbf{D}}\mathbf{u}) d\mathbf{u} \\ &= \sum_{i=1}^{k}\sum_{j=1}^{k} g_i(\mathbf{x}) g_j(\mathbf{x}) \int u_i u_j \frac{1}{a} \exp(-\tfrac{1}{2}\mathbf{u}^{\mathrm{T}}\frac{1}{\mathbf{D}}\mathbf{u}) d\mathbf{u} \\ &= \sum_{i=1}^{k}\sum_{j=1}^{k} g_i(\mathbf{x}) g_j(\mathbf{x}) D_{ij} \\ &= \mathbf{g}(\mathbf{x}) \cdot \mathbf{D}\mathbf{g}(\mathbf{x}) \end{aligned} \quad (34)$$

The gradient $\mathbf{g}(\mathbf{x}) = \frac{\partial y(\mathbf{x};\mathbf{w}=\mathbf{w}_{\mathrm{MP}})}{\partial \mathbf{w}}$ is well-known as a building block of the gradient of the cost function computed with backpropagation. To compute

$\sigma_{\mathrm{WU}}(\mathbf{x})^2$ we need $\mathbf{D}$, the inverse of $\mathbf{B}+\xi_{\mathrm{MP}}\mathbf{I}$. In order for this to be well-defined we set negative eigenvalues of $\mathbf{B}$ to zero corresponding to the zero-mode cut-off in section 2.5. The explicit formula is given in the recipe in section 5.1.

Thus the WU variance is obtained by propagating the uncertainty on level 2 to level 1. The uncertainty on level 3 is believed to have negligible influence on level 1. Finally let us propagate the uncertainty on level 4 to level 1. The committee prediction is the average of the member predictions. For a committee of networks of the same architecture the members are defined by their respective MP weights: $\mathbf{w}^j_{\mathrm{MP}}, j = 1, \ldots, N_{\mathcal{C}}$ so the average prediction is

$$y(\mathbf{x}) = \frac{1}{N_{\mathcal{C}}} \sum_{j=1}^{N_{\mathcal{C}}} y(\mathbf{x}, \mathbf{w}^j_{\mathrm{MP}}) \tag{35}$$

The committee members disagree to a certain extent and this committee uncertainty (CU) gives the following contribution to the prediction variance:

$$\sigma_{\mathrm{CU}}(\mathbf{x})^2 = \frac{1}{N_{\mathcal{C}} - 1} \sum_{j=1}^{N_{\mathcal{C}}} (y(\mathbf{x}) - y(\mathbf{x}, \mathbf{w}^j_{\mathrm{MP}}))^2 \tag{36}$$

The total prediction variance is

$$\sigma_{\mathrm{total}}(\mathbf{x})^2 = \sigma_\nu^2 + \sigma_{\mathrm{WU}}(\mathbf{x})^2 + \sigma_{\mathrm{CU}}(\mathbf{x})^2 \tag{37}$$

where σ_ν^2 and $\sigma_{\mathrm{WU}}(\mathbf{x})^2$ are understood to be averaged over the committee. The significance of these contributions is illustrated with the application (see figure 10).

An apparent problem with the WU error bars is the zero-mode cut-off. This could lead to an underestimate of the error bars. However, this potential loss is remedied by the CU error bars according to the following argument: A zero-mode indicates a poorly defined minimum. When several networks are trained it is likely that the committee members populate different parts of such a flat "basin of attraction". Limiting the extent through the zero-mode cut-off is therefore allowed *if* the CU contribution to the error bar is included. In fact some cut-off must be performed in order not to double-count the "flat basin" uncertainty.

5 Simulations

Here we give a recipe suitable for Bayesian backprop for networks of up to 1000 connections. It requires a minimum of programming. The recipe was used for all the simulations in this work which required 3 days' computing time on a Sun SPARCstation 2 performing 1 Mflop per second.

We give the error of a network as the *standard error*, i.e. the square root of the mean squared error.

5.1 Recipe for Bayesian Weight Decay

1. Centre each of the inputs and scale them to standard deviation $\frac{1}{4}$.

2. Centre each of the outputs and scale them to unit standard deviation. Alternatively use the output unit activation function $y(h) = ah + b$ with a and b set to the empirical mean and standard deviation of the targets.

3. Set the *error scale* E equal to the expected standard training error $\sqrt{\frac{2E_D}{N}}$ based for instance on a previous simulation or a linear model.

4. Set the initial weight decay parameter $\xi_{\rm MP} = E^2/4$ and initialise all scales $s_g = 1$. Initialise the weights to random values in the interval $(-\frac{1}{2}, \frac{1}{2})$.

5. Select a total number of training epochs T and train for $T/6$ epochs (an epoch is one traversal of the training set. We typically use $T = 10.000$. The learning rate was set dynamically as in [11]).

6. Initialise the certainty of weight group g: $\omega_g = \frac{9}{10}$.

7. Train the network for T/6 epochs, reestimating the weight decay parameters according to (17) after each epoch:

$$\xi_{g\rm MP} = \frac{E_D}{E_{W_g}} \frac{\omega_g k_g}{N - \sum_h \omega_h k_h}$$

8. Compute the Hessian in the scaled weight by finite differences of gradients. Reset the scales

$$s_g = \sqrt{\xi_{\rm MP}/\xi_{g\rm MP}}$$

where $\xi_{\rm MP} \equiv \xi_{1\rm MP}$. Let $\mathbf{q}(\mathbf{w}) = \frac{\partial C(\mathbf{w})}{\partial \mathbf{w}}$ be the gradient of the cost function at $\mathbf{w}$, and $\mathbf{e}_i$ a unit vector in the i'th weight direction and let $s_{g(i)}$ be the weight scale of weight i, which is in group $g(i)$. The ij'th element of the Hessian is computed as:

$$B_{ij} = \frac{q_j(\mathbf{w} + \epsilon s_{g(i)}\mathbf{e}_i) - q_j(\mathbf{w} - \epsilon s_{g(i)}\mathbf{e}_i)}{2\epsilon} s_{g(j)} \tag{38}$$

It takes $2k$ epochs to compute the Hessian in this way. If the step length ϵ is set too high we run into deviations from the quadratic behaviour of C, and if it is too small the numerical precision becomes critical, so the calculations were performed with double precision. An efficient way to check the precision of the matrix is to monitor the asymmetry. Finally the Hessian is symmetrised and diagonalised[11].

9. Compute the number of well-determined parameters in group g[12]:

$$\begin{aligned}
\pi_g^i &= \sum_{j \in \mathcal{G}_g} S_{ji}^2 \\
\gamma_g &= \sum_i \frac{\lambda_i}{\lambda_i + \xi_{\rm MP}} \pi_g^i \\
\omega_g &= \frac{\gamma_g}{k_g}
\end{aligned}$$

[11] The Hessian can be evaluated analytically using an extension to backprop developed by Bishop[21]. This involves some elaborate programming but the Hessian is evaluated in just h epochs.

[12] The equations for $\xi_{g\rm MP}$ and γ_g are coupled and it may seem natural to solve these iteratively, but this turns out to increases the convergence time.

where S_{ij} is the j'th component of the eigenvector for λ_i. Negative eigenvalues are left out in this and the next step.

10. Compute the evidence from (23-28):

$$\begin{aligned}\log \mathrm{Ev}(\mathcal{H}) &= -\frac{N}{2} - \frac{N}{2}\log\frac{4\pi C}{N} \\ &+\tfrac{1}{2}\sum_{i=1}^{k}\log\frac{\xi_{\mathrm{MP}}}{\xi_{\mathrm{MP}}+\lambda_i} \\ &+\log h! + h\log 2 \\ &+\log\frac{\sqrt{4\pi/(N-\sum_h \gamma_h)}}{\log\Omega} + \sum_g \log\frac{\sqrt{4\pi/\gamma_g}}{\log\Omega} \qquad (39)\end{aligned}$$

where $\Omega = 10^3$ and h is the number of hidden units. With more than one hidden layer there is a term $\log h! + h\log 2$ for each hidden layer.

11. Stop here or repeat from 7.

We always start with small weights. This improves the convergence of the network. The fixed weight decay parameter defined in step 4 is guessed using (16) and corresponds to "mature" weights, $\sigma(w/s_g) = 2$. This allows the weights to "grow up" during the first $T/6$ epochs.

A quick and dirty Bayesian weight decay procedure is obtained by skipping the Hessian calculation, step 8–10

The noise parameter β_{MP} does not appear in the recipe but can be computed by $\sigma_\nu^2 = 1/\beta_{\mathrm{MP}} = 2C/N$.

The WU error bar on the regression can be computed for an arbitrary $\mathbf{x}$. The gradient with respect to the unscaled weights $\mathbf{g}(\mathbf{x}) = \frac{\partial y(\mathbf{x};\mathbf{w}=\mathbf{w}_{\mathrm{MP}})}{\partial \mathbf{w}}$ is computed with backpropagation. The gradient with respect to the *scaled* weights $g'_j(\mathbf{x}) = s_{g(j)}g_j(\mathbf{x})$ is projected onto the eigenvectors $\mathbf{s}_i$ to obtain the WU error bar according to (34):

$$\sigma_{\mathrm{WU}}(\mathbf{x})^2 = \sigma_\nu^2 \sum_{i=1}^{k} \frac{(\mathbf{s}_i\cdot\mathbf{g}'(\mathbf{x}))^2}{\lambda_i+\xi_{\mathrm{MP}}} \qquad (40)$$

Negative λ_i's are set to zero.

The Hessian can be computed approximately by neglecting the nonlinearities of the sigmoid functions:

$$\mathbf{B} \approx \sum_{i=1}^{N} \mathbf{g}'(\mathbf{x}_i)\mathbf{g}'(\mathbf{x}_i)^{\mathrm{T}} \qquad (41)$$

where $\mathbf{x}_i$ is the input for training case i. This is sufficiently accurate for computing error bars and the number of well-determined parameters.

For a committee the CU error bars are computed as the variance of the committee predictions, see (36) and (37).

Pass Number	R	A
1	20	0.5
2	30	1
3	40	2
4	50	4
5	60	8
6	70	16
7	80	32
8	90	64
9	100	128

Table 1: *The pruning is performed in 9 passes over the connections. The passes use an increasing number of retraining epochs R and an increasing allowable error increase A (expressed in percent of $\frac{1}{2}NE^2$)*

5.2 The Pruning Scheme

Pruning is incorporated into the basic Bayesian recipe of the previous section in the following manner: first the network is trained according to the basic recipe. Then steps 7–10 are repeated a number of times called *passes* but before returning to step 7 the following pruning step is performed:

Go through all the connections in the network (excluding the biases; a bias is eliminated only if the corresponding unit looses all its connections). For each connection do the following substeps:

a Save the network and the cost C.

b Remove the connection.

c Retrain the network for R epochs.

d Compute the new cost C'. If the network has not retrained sufficiently well, i.e. if $C' > C + (A/100)NE^2/2$, the saved network is retrieved, i.e. the connection is not pruned.

e Continue to the next connection.

When the training step 7 is invoked, the number of connections k_g in group g has decreased, and this affects the weight decay parameter. During pruning the ω_g's typically change by less than 10%, while the $\xi_{g\mathrm{MP}}$'s change by orders of magnitude.

The passes are defined with increasing values of the number of retraining epochs R and the allowable error increase A. In the simulations we use the series of passes in table 1. This gives a series of 10 networks pruned to an ever higher degree. For each network in the series we have the evidence and we select the network with the highest evidence. This is our Bayesian stop criterion for pruning.

Other pruning schemes can be used instead. For instance it is possible to speed up the pruning with more programming effort using "Optimal Brain Surgeon" [16].

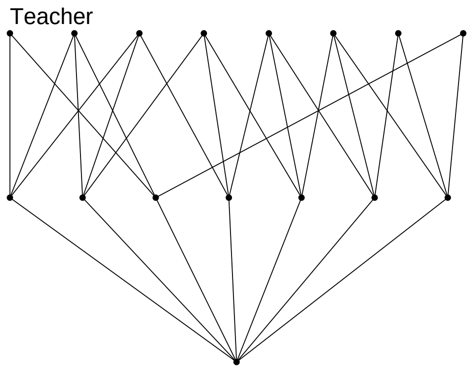

Figure 3: *The network used to generate the artificial training data. There are 8 input units on top and 7 hidden units with a receptive field of 3 adjacent input units (the input field wraps around).*

5.3 Bayesian Weight Decay and Artificial Data

We now present numerical simulations of Bayesian neural networks. First we consider artificial data sets generated by the sparsely connected network in figure 3.

The weights of this "teacher network" are divided into two weight groups. In each group the weights are generated with a Gaussian distribution. The first group consists of the weights and biases of the hidden units and these have $\sigma = 2$. The weights and biases of the output unit form the second weight group with $\sigma = 2.5$. The activation function of the output unit is $f(h) = h$. The input data are Gaussian random numbers with zero mean and $\sigma = 0.5$. Random Gaussian noise $\sigma = 0.05$ is added to the output. The training set contains 200 cases, and the test set contains 2000 cases.

To *learn* the regression from the examples generated by the teacher we apply networks with the same number of hidden units as the teacher. We use 6 different architectures with receptive field sizes ranging from 3 to 8, where 8 corresponds to a fully connected network.

Two weight groups are used as in the teacher network. 20 networks are trained with each architecture, and the standard test error, also called *the standard error of prediction, SEP*, is shown versus log evidence in figure 4. Some of the networks have log evidence as low as -200, corresponding to local minima of the cost function, but we show only networks with log evidence larger than 0. (The natural zero-point for log evidence is $\log P(D)$, which we don't bother to compute, since the evidence is used only to compare different models of the same data set D. Notice also that the evidence $P(D \mid \mathcal{H})$ is a probability density in the N-dimensional target space so the evidence depends on the unit in target space). The highest evidence and the lowest test error are obtained with receptive field 3. The overall correlation between test error and evidence is not high, but if each architecture is considered separately the correlation is quite good. The log evidence is shifted by a constant amount 20 ± 2 when the size of the receptive field is incresad by one. This also holds for the shift between a receptive field of 7 and the fully connected network.

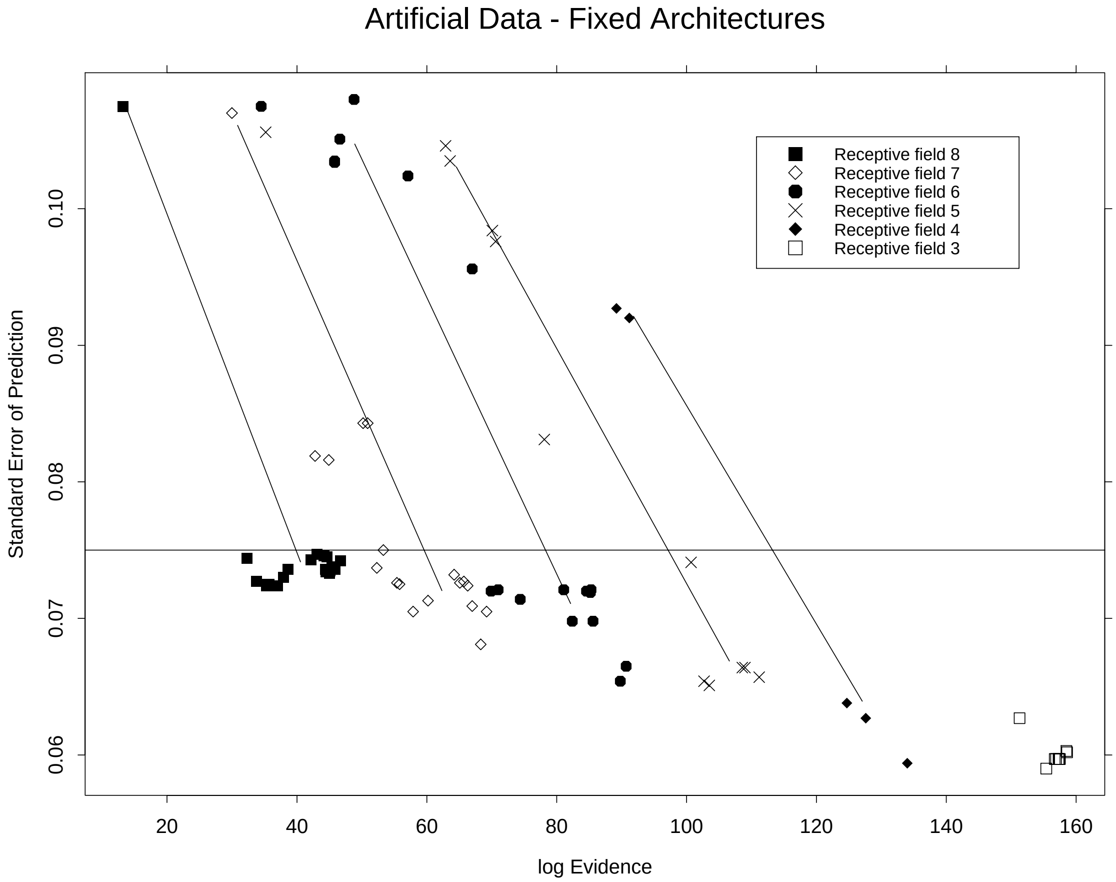

Figure 4: *The standard test error (also called the standard error of prediction, SEP) as a function of the log evidence for networks with different receptive fields. Lines are drawn to guide the eye to the trend of the test errors for each architecture. The log evidence at SEP=0.075 progresses evenly as the receptive field shrinks from fully to sparser connected networks.*

This supports the prior $\log P(\mathcal{H}) = \log 7 = 8.5$ which is included in all networks except the fully connected. Without this prior the shift of log evidence to the fully connected network would be 11.5 instead of 20[13].

The networks with larger receptive fields have a larger capacity than needed for the problem. They have less architectural bias and hence give larger variance contribution to the test error.

The shift in log evidence between the different architectures can also be interpreted as support for the conjecture that the evidence reflects other virtues than the generalisation error: For fixed test error, the log evidence is higher for the networks with a more correct architecture.

The distributions of the weights and biases to the hidden layer for some networks are shown in figure 5. For the larger receptive fields the histograms

[13]A source of uncertainty in the evidence is the dependence on the cut-off Z mentioned in section 2.5. However, the increase in log evidence when Z is doubled is less than 1.5 for all the networks in this figure.

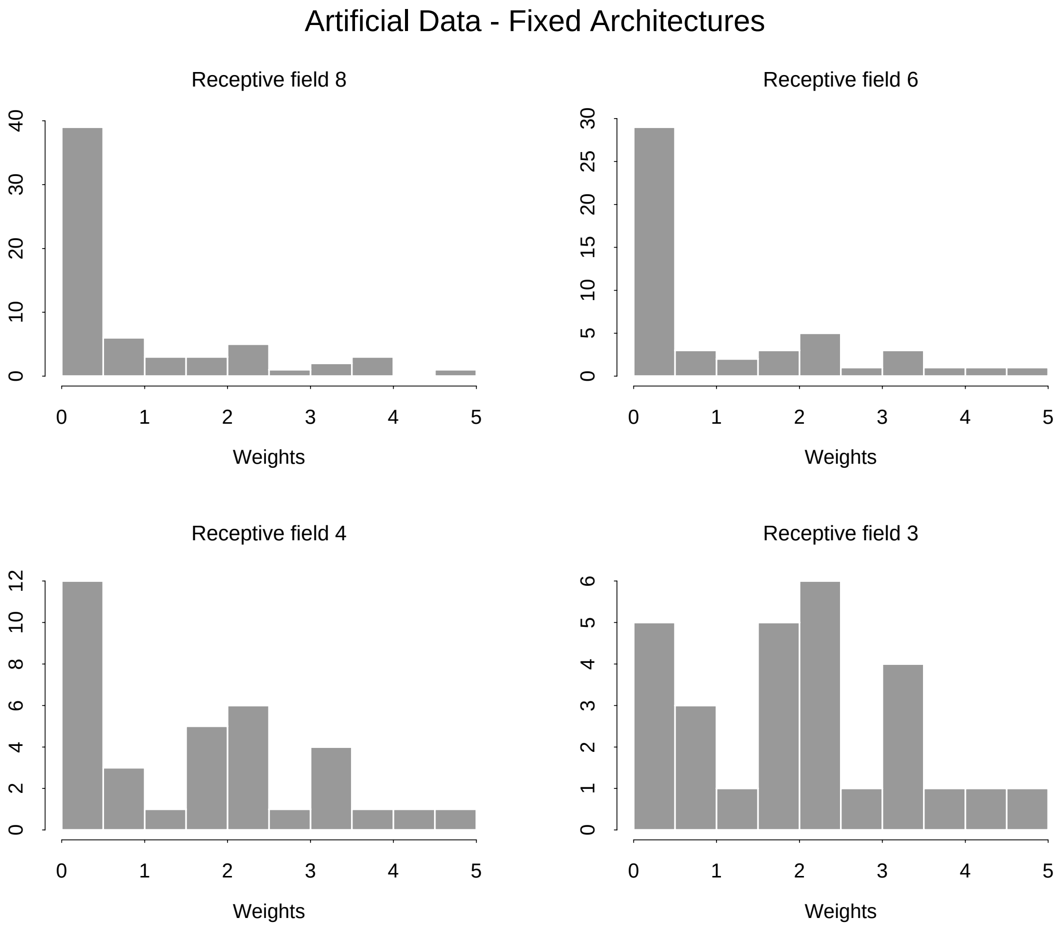

Figure 5: *The distribution of the size of the weights from the input to the hidden layer for networks with different receptive fields. (Note the different vertical scales of the histograms.) The weight decay has driven the spurious weights to values near zero.*

reveal an excess near zero. We can account for this from the number of useless weights which is $7(r-3)$ for a receptive field r, i.e. 35 for the fully connected network.

5.4 Bayesian Pruning and Artificial Data

Now let us be completely ignorant of the true architecture and start with a fully connected network and investigate whether Bayesian pruning can *discover* the teacher architecture. We also ignore our knowledge of the number of hidden units of the teacher and start with a fully connected network with 10 hidden units (the simulations were also performed with 14 hidden units with very similar results). With the large number of hidden units the networks have no trouble learning the problem. No networks get caught in the low evidence minima which were experienced with the networks with 7 hidden units. The networks are pruned according to table 1 with $E = 0.05$. Figure 6 shows the test error for all twenty pruning series of networks. As the pruning progresses

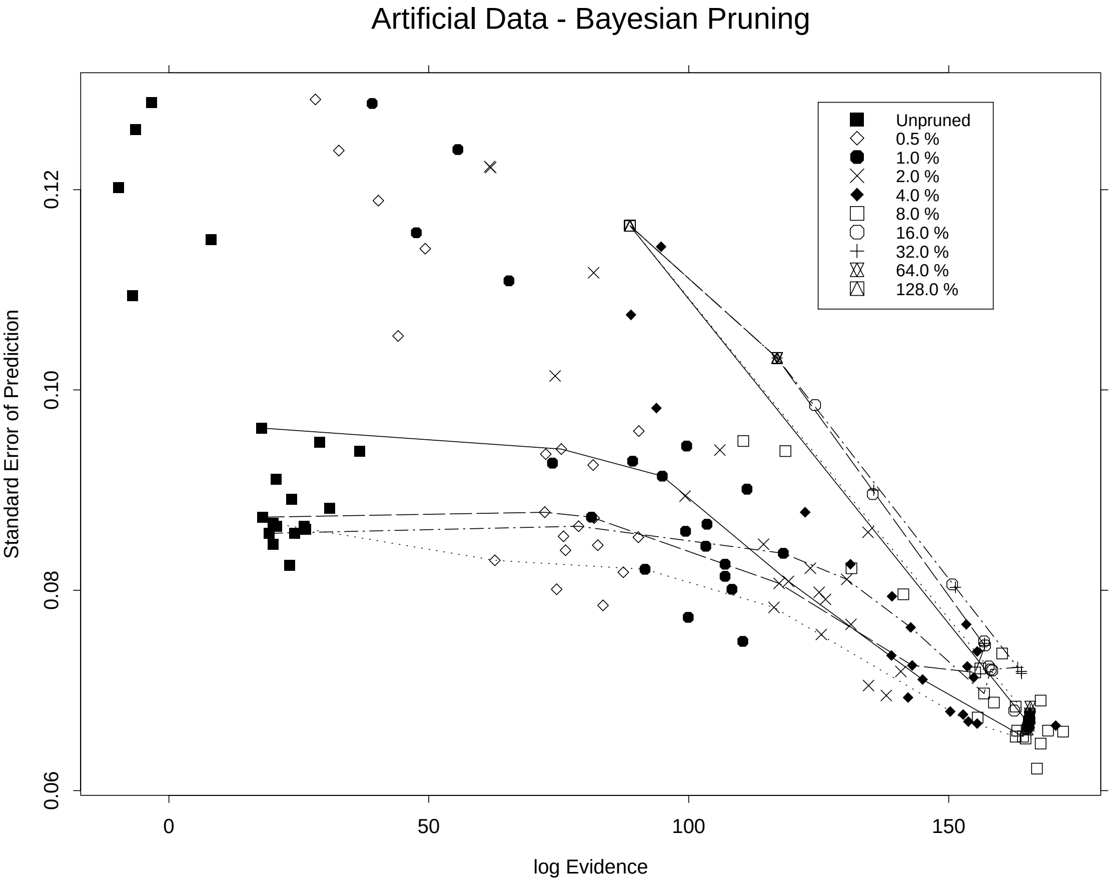

Figure 6: *The test error as a function of the log evidence for networks pruned to a larger and larger extent. Four series of networks are connected with broken lines. The symbols indicate the allowable error increase of the pruning.*

the evidence reaches a maximum. It is pleasing to see that this coincides well with the minimum test error. The minimum SEP attained is only 10% higher than the SEP obtained with the teacher architecture (see figure 4).

The networks of a typical pruning series are shown in figure 7. The architecture favoured by the evidence deviates slightly from the teacher architecture. The first hidden unit is missing and a connection from the third input is missing. It turns out that the weight from the missing hidden teacher unit to the output is 0.69, the smallest of that layer. And the missing teacher weight from input unit 3 is 0.62, the third smallest weight of the first layer. Hence pruning removes not only irrelevant weights but also non–zero weights which are not significantly different from zero, given the limited training set and the noise.

The distributions of the weights in the first layer changes as pruning progresses as shown in figure 8. The excess of weights near zero disappears and eventually becomes a depletion[14].

[14]Given that we prune our networks maximally, the depletion is not surprising. There is no such thing as a zero weight in a minimal architecture! This hole in the posterior Gaussian distribution amounts to a slight inconsistency: to be more consistent we should incorporate a

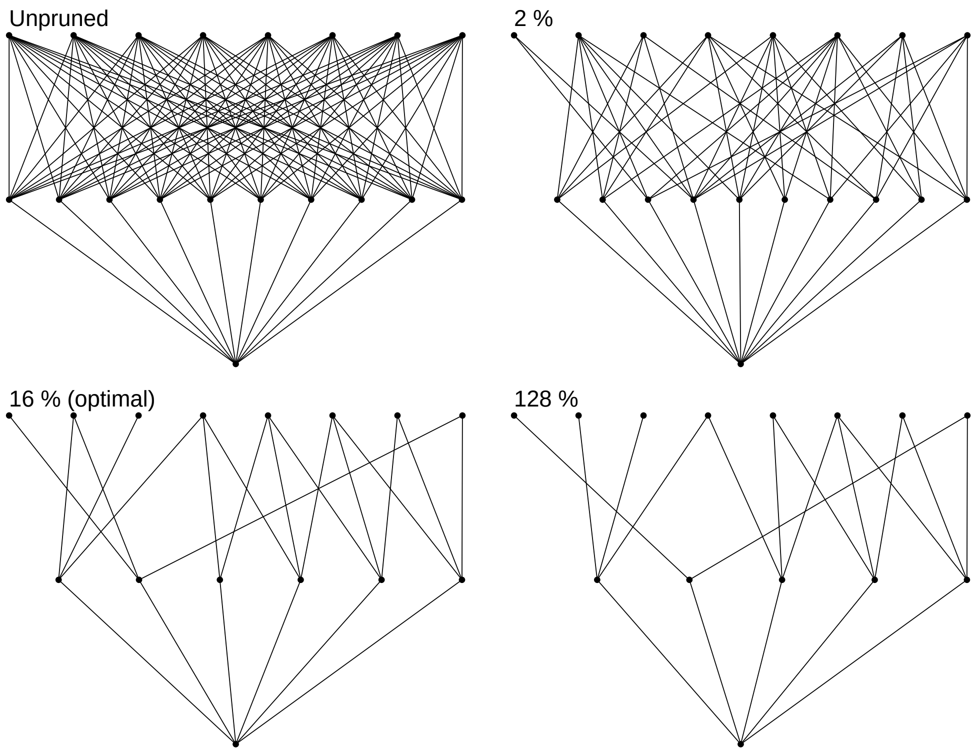

Figure 7: *The history of a network as it is pruned to an ever higher degree. The allowable error increase ($A/100$) used for each network is indicated. The highest evidence and smallest test error are obtained with the network with $A/100 = 16\%$; compare this to the teacher in figure 3. The evidences and test errors of this series is shown in figure 6 with a solid broken line.*

5.5 Application to Spectroscopic Data

We now apply Bayesian Backprop to a real-life application from the meat industry. The data were recorded by a Tecator near-infrared spectrometer which measured the spectrum of light transmitted through a sample of minced pork meat [22]. The spectrum gives the absorbance at 100 wavelengths in the region 850-1050 nm. We want to calibrate the spectrometer to determine the fat content from the spectrum. The target values of the fat content are determined by wet-chemistry (the Soxhlet method) and ranges from 2 to 59%. The spectra are preprocessed using principal component analysis. The first 10 principal components are used as input to a neural network.

In an earlier work [22] we have analysed these data with a linear model and with neural networks. The data were divided according to table 2. The data available for model estimation were split into a set C used for training and a "monitoring" set M. The SEP on M was monitored during training and the hitherto best network saved. Several networks were trained and finally the network with the best SEP on M was selected. This training technique, denoted

depletion in the prior weight distribution. This would reduce the denominator of the Ockham Factor, i.e. increase the evidence.

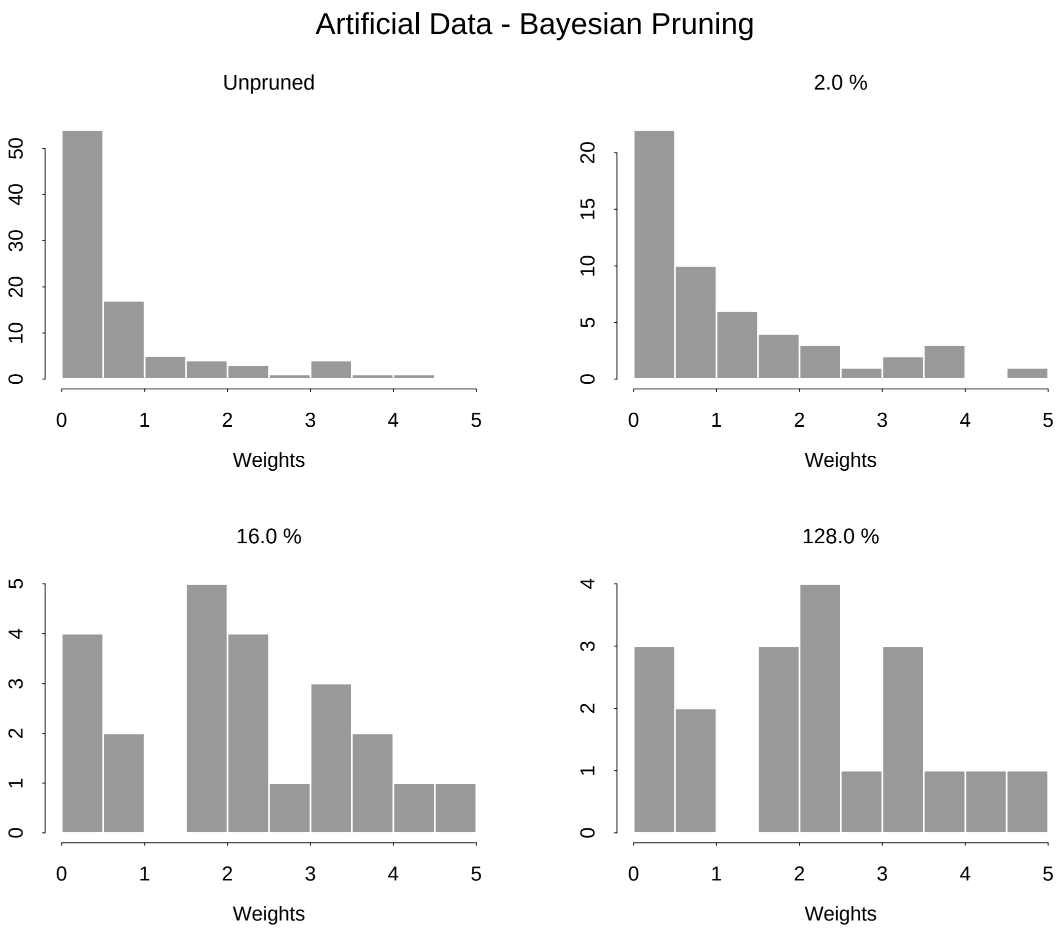

Figure 8: *The distributions of the size of the weights of the hidden layer of a network pruned to an ever higher degree. The corresponding architectures are shown in figure 7. Pruning eliminates the excess of near-zero weights.*

Data set	Use	Number of cases
C	Training	129
M	Monitoring	43
T	Testing	43
E1	Extrapolation, Water	17
E2	Extrapolation, Fat	8

Table 2: *The spectroscopic data sets and their use when training networks with early stopping*

Model	Model selection	SEP
Linear regression	F-test	2.84
Quadratic regression	F-test	0.80
Neural network	Early Stopping	0.65
Neural network	GPE	0.60
Neural network	Evidence	0.55
Neural network committee	Evidence	0.52

Table 3: *The Standard Error of Prediction on the data set T.*

"early stopping", is rather successful for many applications, as it is an easy way to cope with the problem of overfitting [15]. The optimal architecture had six hidden units and direct connections from input to output.

In this section we demonstrate that the Bayesian method outperforms early stopping and that a committee of networks improves the model even further. We compare with standard quadratic regression and we investigate how to predict the generalisation error. Table 3 summarises the results.

For the Bayesian analysis we combine the data sets C and M for the training. Three separate weight decay parameters are used: one for the weights and biases of the hidden layer, one for the connections from the hidden to the output layer, and one for the direct connections from the inputs to the output as well as the output bias. Like earlier, T is used only to compute SEP in table 3.

The relation between test error and log evidence is shown in figure 9[15]. We see that networks with 3 hidden units are most frequent among the networks with the highest evidence, so we select the 12 networks with 3 hidden units and evidence larger than -270 for further investigation. The average SEP is 0.55, a 15% reduction relative to the method of early stopping (see table 3). There are several reasons for this improvement:

- The optimisation of three separate weight decay parameters gives a more careful regularisation of the complexity than early stopping.
- The use of C+M for training rather than C gives a larger training set.
- The evidence is a less noisy model selection criterion than the error on the monitoring set [1].
- The longer training times give a more extensive search for good networks than the shorter training times used with early stopping.

To improve the model further we form a committee of the 12 networks with 3 hidden units which are favoured by the evidence. This committee gives a further 6% reduction of the test error to SEP = 0.52.

Quadratic regression is a standard statistical method used for this kind of data. All products of pairs of the 10 original variables are included as 55

[15] One source of uncertainty in the evidence is the dependence on the cut-off Z mentioned in section 2.5. The increase of log evidence when Z is doubled is 1.5 for a network with 3 hidden units and 8 for a network with 6 hidden units. Thus the evidences of the larger networks are poorly defined so they should be avoided

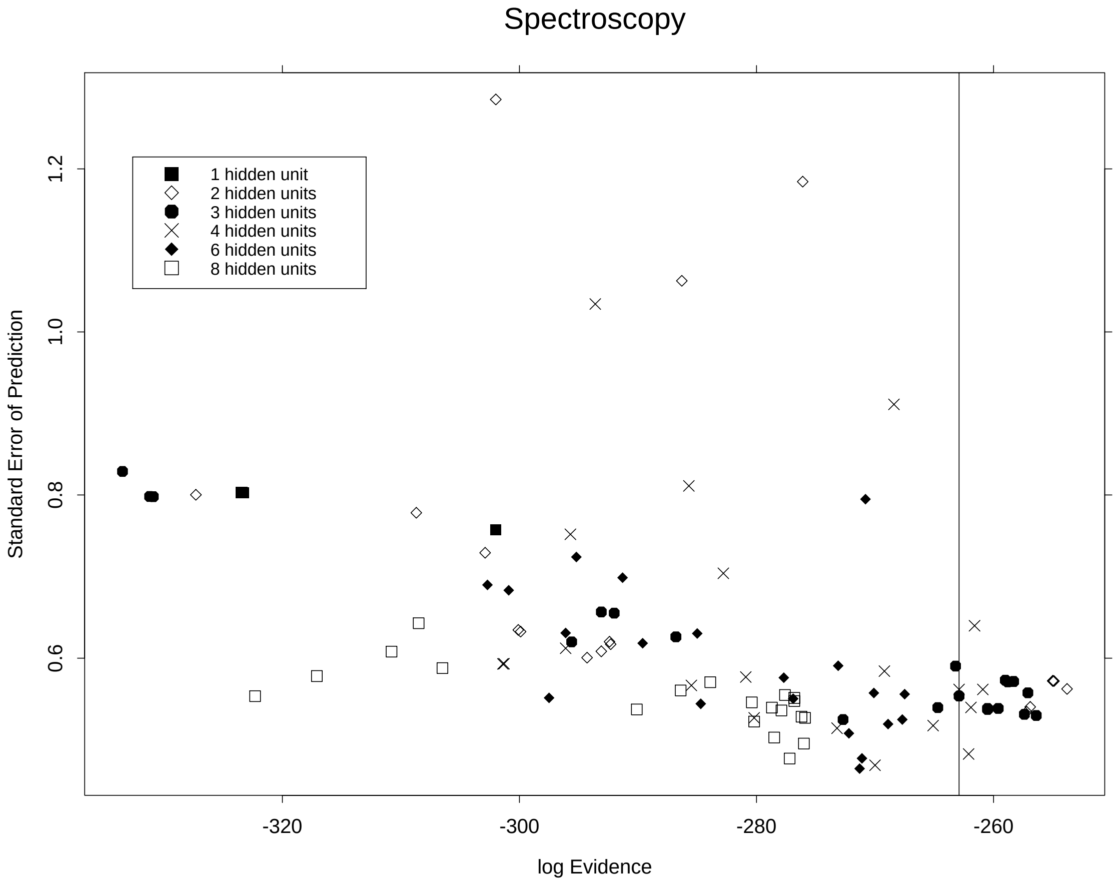

Figure 9: *The test error as a function of the log evidence for networks trained on the spectroscopic data. Networks with 2 and 3 hidden units attain the highest evidence. The 20 networks with highest evidence are delimited by the vertical line and have on average SEP=0.55.*

new input variables. The 65 variables are used as independent variables in a stepwise regression (see e.g. [10]) using the data C+M. Model selection is performed with the F-test which tests the significance of adding or removing a variable in the model. This selects a model with 42 variables and SEP = 0.80. Hence quadratic regression is considerably less accurate than neural networks.

5.6 Error Bars for the Spectroscopic Data

The Bayesian error bars are illustrated for the spectroscopic data in figure 10. We study the model predictions on the line through input space defined by the second principal component axis, i.e the second input is varied and all other inputs are set at zero. The principal component is scaled to unit variance so the training data has standard deviation 1 in this projection. The predictions within $(-1, +1)$ can be considered interpolations and as we go further away from 0 we are extrapolating to a larger and larger degree.

Figure 10 shows that the error increases dramatically when the model is

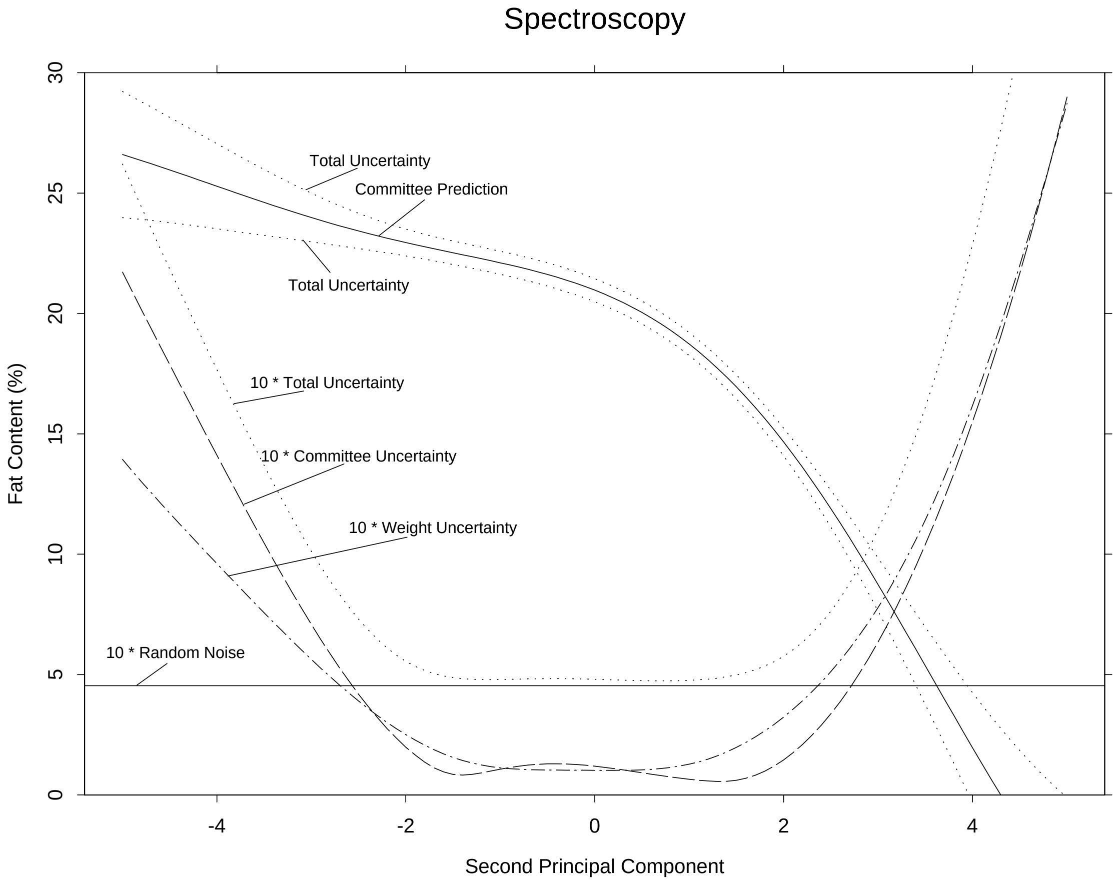

Figure 10: *Prediction of the fat content as a function of the second principal component p_2 of the NIR spectrum. 95% of the training data has $|p_2| < 2$. A committee of 12 networks with 3 hidden units was used. The total standard error bars are indicated by a "1 sigma" band with the dotted lines. The total standard errors and the standard errors of its contributions are shown separately, multiplied by a factor of 10.*

used outside the range of the training data. On the same figure are shown the total error amplified by a factor of 10 and its break-down into the three components, see eqn. (37). The random noise is constant, while the committee and weight uncertainties vary with input and are of the same order of magnitude. The committee uncertainty typically dominates for wild extrapolations, but in general the contributions vary in a complicated manner. The general CU/WU proportion was found to be rather insensitive to the number of hidden units.

We now compare the computed error bars with the actual errors for test data. In addition to the test set T drawn from the same population as the training set C+M, there are two extrapolation data sets (see table 2). E1 contains samples with larger fat content than the training set. Similarly E2 consists of samples with larger water content. The fat predictions were computed with error bars for the 68 samples in T, E1 and E2 using the committee of networks with three hidden units. It was checked that the residuals are independent of

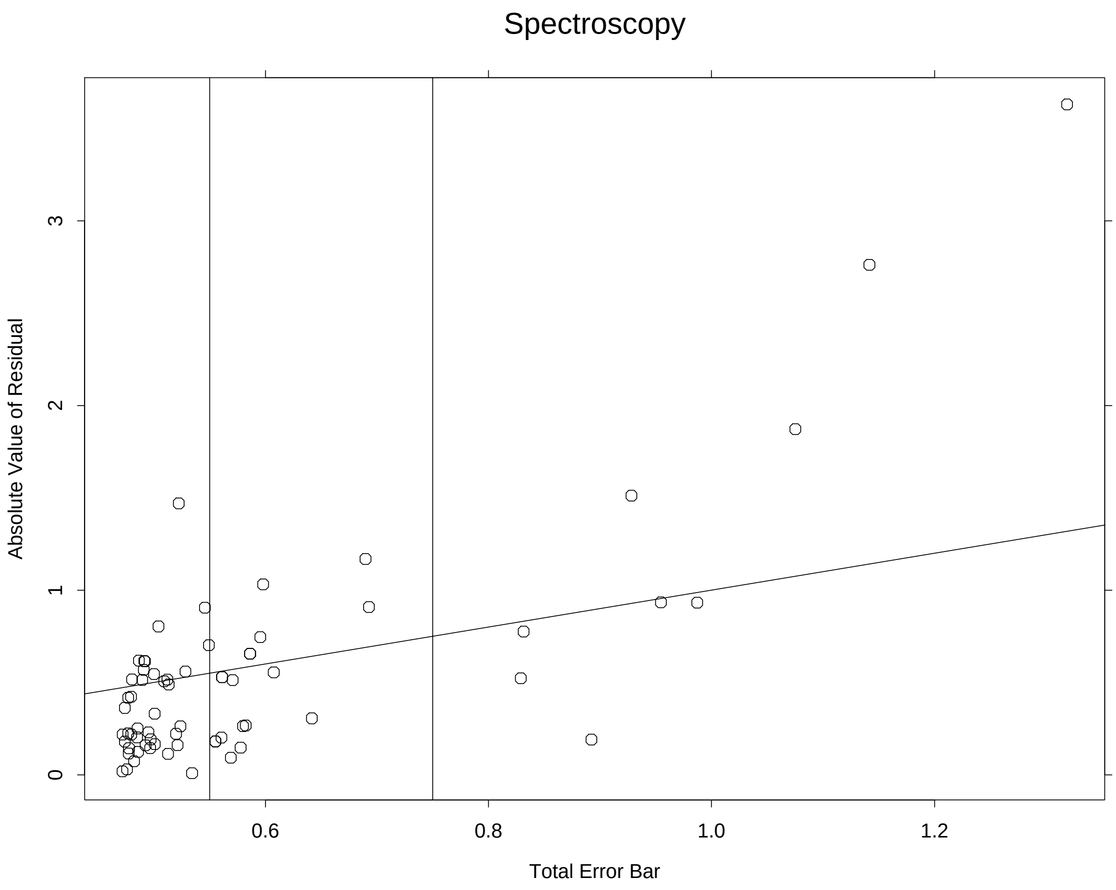

Figure 11: *The observed residuals of the predictions of fat on test samples against the Bayesian standard error bars. The line (Residual=Error bar) is drawn. The Bayesian error bars account well for the residuals except for the 3 outmost extrapolations. The test data are divided into three groups indicated by the vertical lines. The observed SEP in each group is compared to the Bayesian estimate in table 4.*

the predicted fat value on C, M and T [22]. The absolute values of the observed residuals are shown versus the total error bars in figure 11.

The observed SEPs are consistent with the Bayesian error bars when these are below 1.0. The three cases above this point all have positive residuals and are extrapolations to the largest fat values, so the model has a bias here. This bias is of course not described by the error bars which assume that the model is correct.

The error bar can be used as a measure of the degree of extrapolation. A large error bar is a warning that the predictions may be biased, so that the error bars underestimate the actual error. It is natural that the model becomes biased in regions where different laws of nature apply, laws that we have no chance of learning from the training region.

σ_{total}	Predicted SEP	Observed SEP
0.00–0.55	0.50	0.46±0.05
0.55–0.75	0.59	0.59±0.10
0.75–∞	1.01	1.80±0.42

Table 4: *The predicted and observed SEP (Standard Error of Prediction) on test samples grouped according to the predicted error bars (see figure 11). The observed SEPs are given with standard errors computed from χ^2-statistics.*

6 Moody's GPE

Moody [23] proposed an estimator of the generalisation error for neural networks. It is a generalisation of Akaike's Final Prediction Error and is called the Generalised Prediction Error. GPE predicts the test error from the training error E_D and the "effective number of parameters" for which we substitute the number of well-determined parameters, $\gamma = \sum_g \gamma_g$:

$$\text{GPE} = \frac{2E_D}{N}\frac{N+\gamma}{N-\gamma} \tag{42}$$

GPE is compared to the actual test error in figure 12. A comparison with figure 9 shows that the evidence is superior to the GPE when it comes to selecting the models with the best generalisation.

To summarise, here follows a comparison of GPE and the evidence:

Foundations. The GPE is a conventional statistical estimator of the generalisation error. Its properties are proved under specific assumptions. It is not clear whether these assumptions are fulfilled for neural networks. The evidence has a different status. It is not proved to be an estimator but is a quality measure based on metaprinciples.

Performance. For the spectroscopic data GPE is a poorer selector of good networks than is the evidence. However, for the robot arm problem in [24] and the artificial data of section 5.3 the GPE is as good as the evidence. This suggests that the GPE performs *no better* than the evidence in model comparison.

Properties. GPE has a simpler structure than the evidence. It does not involve the sometimes ill-defined determinant of the Hessian. The GPE scales differently from the evidence with γ and N (see [24])

Use in Regularisation. We used GPE for networks which were already regularised with the Bayesian method. It has not yet been demonstrated that GPE - like the Bayesian framework - can be used to set several weight decay parameters during training.

Generality. The Bayesian framework is more general than the GPE. It can be used to determine error bars on each prediction and it applies to classification problems and active learning [3, 4].

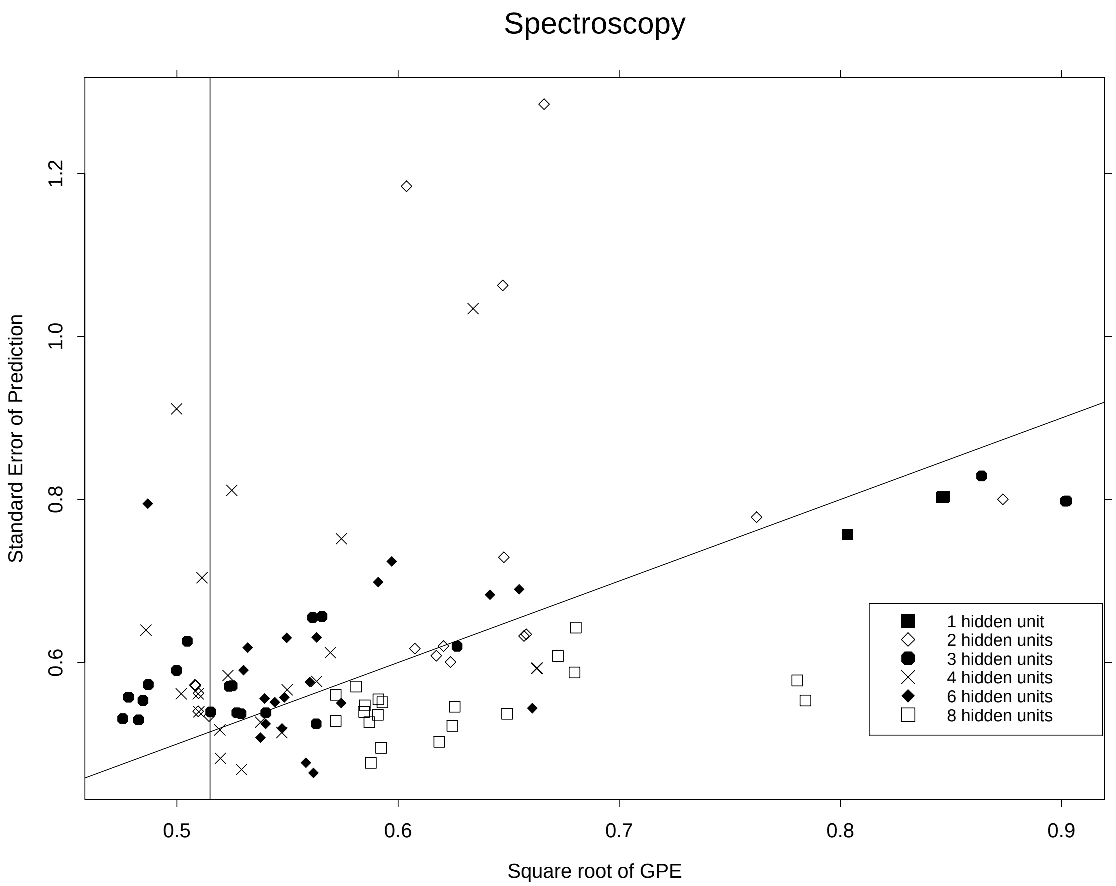

Figure 12: *The test error versus the square root of Moody's GPE (Generalised Prediction Error) for networks trained on the spectroscopic data. (The diagonal of the plot is drawn). The 20 networks with smallest GPE are delimited by the vertical line and have on average SEP=0.60. Compare with figure 9.*

7 Practical Comments

This section discusses details pertaining to the practical application of Bayesian backprop.

7.1 Use of Validation

A critical issue in the application of the evidence for selection of models is its high correlation with the test error. It is recommended to check this relation in all applications. This is done by splitting the available data into a training and a test set. Then the Bayesian analysis is performed on the training set and validated on the test set. This can reveal inconsistencies in the models and help cure them. The goal is to obtain a high correlation between the evidence and the test error. When this has been accomplished the data sets are merged to a single training set and the final set of models is developed. Now the best model – or a committee of the best models – can be selected with confidence according to the evidence.

In the initial phase, the spread of evidence for networks with the same generalisation error is recorded. This is useful later when the committee of networks is to be defined as the networks with maximum evidence within this evidence spread.

7.2 Checking the Model

It is important to check the distribution of the weights and the residuals. The Bayesian framework requires that these be Gaussian. If the weights of a group are not normally distributed, the group should be subdivided into groups with separate weight decay parameters. If the residual errors are not Gaussian distributed it can be due to two causes:

- Some of the training examples are outliers, for instance measurement or typing errors. These are removed from the training data. This is a tricky point since an outlier is always in the eye of the beholder (see [6] for a thorough Bayesian treatment).

- The noise level depends on the output or input. A possible solution is to transform the output variable or to introduce a noise level which depends on the output. A treatment of input dependent noise level is given in [25].

7.3 Testing for Non-linearities

Neural networks are well suited to test whether a data set defines a linear or a nonlinear regression. Linear models and neural networks are trained and the evidences are compared. The universal approximation capability of neural networks minimises the risk of overlooking a nonlinearity.

When neural networks are used to model a relation which is not significantly nonlinear they degenerate in a sensible way: in a network with direct connections from input to output the connections of the first hidden layer are driven to zero by the Bayesian weight decay, i.e. the weight decay parameter diverges. In a network without direct input-output connections the weights of the first layer are driven to small finite values so that the sigmoid functions of the hidden layer are used in the linear region near zero.

8 Conclusions

Bayesian backprop was presented as a consistent framework in the tradition of Ockham's Razor. No attempt was made to prove the essential relation between the evidence and the test error. Instead it was argued that the relation is a metaprinciple founded in experience and common sense. The evidence framework brings us something we cannot get from classical statistics.

A detailed recipe for Bayesian backprop was given. With this it is easy to upgrade existing simulators.

With Bayesian weight decay and pruning, neural networks have matured. The unsatisfactory "early stopping" technique has finally been superseded by a consistent and efficient treatment of the overfitting problem.

For a wide range of applications Bayesian backprop is likely to be the most efficient method of nonlinear regression.

Acknowledgements

I am grateful to David MacKay for extensive discussions of many details of Bayesian Backprop. Also thanks to Radford Neal, Sara Solla and Carl Edward Rasmussen for valuable corrections to the manuscript and to Lars K.Hansen for ideas on generalisation.

References

[1] D.J.C.MacKay, "Bayesian Interpolation" *Neural Comp.* **4** (1992) 415-447.

[2] D.J.C.MacKay, "A Practical Bayesian Framework for Backpropagation Networks" *Neural Comp.* **4** (1992) 448-472.

[3] D.J.C.MacKay, "Information-Based Objective Functions for Active Data Selection" *Neural Comp.* **4** (1992) 590-604.

[4] D.J.C.MacKay, "The Evidence Framework Applied to Classification Networks" *Neural Comp.* **4** (1992) 720-736.

[5] S.F.Gull "Bayesian inductive inference and maximum entropy", in *Maximum Entropy and Bayesian Methods in science and engineering, vol 1: Foundations*, eds. G.J.Ericson and C.R.Smith, Kluwer 1988.

[6] G.E.P.Box and G.C.Tiao, *Bayesian Inference in statistical analysis*, Addison-Wesley 1973.

[7] J.O.Berger, *Statistical Decision Theory and Bayesian Analysis*, Springer 1985.

[8] W.L.Buntine and A.S.Weigend, "Bayesian Back-Propagation", *Complex Systems* **5**, (1991) 603-643.

[9] D.H.Wolpert, "Use of Evidence Neural Networks", *Neural Information Processing Systems, Vol.5* ed. C.L.Giles, S.J.Hanson and J.D.Cowan (Morgan Kaufmann, San Mateo, 1993).

[10] N.R.Draper and H.Smith, *Applied Regression Analysis*, second ed., Wiley 1981.

[11] H.H.Thodberg, "Improving Generalization of Neural Networks through Pruning", *Int.J.Neural Systems* **1** (1991) 317-325.

[12] Y.Le Cun, J.S. Denker and S.A Solla, "Optimal Brain Damage", *Neural Information Processing Systems, Vol.2* ed. D.Touretzky (Morgan Kaufmann, San Mateo, 1989) 598-605.

[13] Y.Le Cun, B.Boser, J.S.Denker, D.Henderson, R.E.Howard and W.Jakel, "Handwritten Digit Recognition with a Backpropagation Network", *Neural Information Processing Systems, Vol.2* ed. D.Touretzky (Morgan Kaufmann, San Mateo, 1990) 396-404.

[14] S.Nowlan and G.Hinton, "Simplifying Neural Networks by Soft Weight-Sharing", *Neural Comp.* **4** (1992) 473-493.

[15] A.S.Weigend, B.A.Huberman and D.E.Rumelhart, "Predicting the future: a connectionist approach", *Int.J.Neural Systems* **1** (1990) 193-209.

[16] B.Hassibi and D.G.Stork, "Second order derivatives for Network Pruning: Optimal Brain Surgeon", *Neural Information Processing Systems, Vol.5* ed. C.L.Giles, S.J.Hanson and J.D.Cowan (Morgan Kaufmann, San Mateo, 1993).

[17] S.Geman, E.Bienenstock and R.Doursat, "Neural Networks and the Bias/Variance Dilemma" *Neural Comp.* **4** (1992) 1-58.

[18] L.K.Hansen and P.Salamon, "Neural Network Ensembles", *IEEE Trans. PAMI* **12** (1990) 993-1001.

[19] R.M.Neal, "Bayesian Learning via Stochastic Dynamics", *Neural Information Processing Systems, Vol.5* ed. C.L.Giles, S.J.Hanson and J.D.Cowan (Morgan Kaufmann, San Mateo, 1993), and "Bayesian training of backpropagation networks by the hybrid Monte Carlo Method", CRG-TR-92-1, Dept.of Computer Science, Univ. of Toronto (1992).

[20] D.J.C.MacKay, paper on Bayesian hyperparameters, in preparation 1993.

[21] C.Bishop, "Exact Calculation of the Hessian Matrix for the Multilayer Perceptron" *Neural Comp.* **4** (1992) 494-501.

[22] C.Borggaard and H.H.Thodberg, "Optimal Minimal Neural Interpretation of Spectra", *Analytic Chemistry* **64** (1992) 545-551.

[23] J.Moody, "The Effective Number of Parameters: An Analysis of Generalization and Regularization in Nonlinear Learning Systems", *Neural Information Processing Systems, Vol.4* ed. J.E.Moody et al (Morgan Kaufmann, San Mateo, 1992).

[24] D.J.C.MacKay, "Bayesian Model Comparison and Backprop Nets", *Neural Information Processing Systems, Vol.4* ed. J.E.Moody et al (Morgan Kaufmann, San Mateo, 1992) 839-846.

[25] D.J.C.MacKay, *Bayesian Methods for Adaptive Models*, PhD thesis, California Inst.Tech. Pasadena 1991, chapter 6.