parameters such as σ and Δ, that is $Pr(\boldsymbol{x}|w,\sigma) = Pr(\boldsymbol{x})$. Posterior probabilities of network weights are as follows. For regression with Gaussian error and unknown σ,

$$Pr(w \mid \boldsymbol{x}, \boldsymbol{y}) \;\propto\; Pr(w, \boldsymbol{y} \mid \boldsymbol{x}) \;=\; Pr(w) \frac{\Gamma(\frac{N-1}{2})}{(N\pi)^{\frac{N-1}{2}} \sqrt{N} \left(\sum_{i=1,\dots,N} (y_i - o(x_i, w))^2 \right)^{\frac{N-1}{2}}} \,,$$

for regression with Laplacian error and unknown Δ,

$$Pr(w \mid \boldsymbol{x}, \boldsymbol{y}) \;\propto\; Pr(w, \boldsymbol{y} \mid \boldsymbol{x}) \;=\; Pr(w) \frac{\Gamma(N)}{2^N \left(\sum_{i=1,\dots,N} |y_i - o(x_i, w)| \right)^N} \,,$$

and for classification

$$Pr(w \mid \boldsymbol{x}, \boldsymbol{y}) \;\propto\; Pr(w, \boldsymbol{y} \mid \boldsymbol{x}) \;=\; Pr(w) \prod_{i=1,\dots,N} o_{y_i}(x_i, w) \;.$$

These posterior probabilities define the Bayesian solution to the problem of training networks. With a Gaussian error model, high posterior weights w trade-off the prior $Pr(w)$ and the mean-square error. With a Laplacian error model, the trade-off is with the average absolute deviation. In all cases, as the sample size N gets large, the prior term will become negilible and can be assumed constant.

6 Analyzing weights

The section describes how we can use the posterior analysis just given to assist basic tasks done during network training. In Section 6.1 we describes how we derive "cost functions" for a given set of weights from the posterior formulae and Section 6.2 describes how we can evaluate a set of weights. The cost function allows us to find a locally optimum set of weights and the evaluation allows us to compare the quality of one local optimum with another.

6.1 Cost functions

The posterior probabilities just given represent functions that should be maximized. Calculation is usually done in logarithms to prevent arithmetic underflow and to turn products into sums. Hence when searching for a high posterior set of weights we would like to minimize the "cost function" $-\log Pr(w \mid \boldsymbol{x}, \boldsymbol{y})$. Notice that

$$-\log Pr(w \mid \boldsymbol{x}, \boldsymbol{y}) \;=\; -\log Pr(w, \boldsymbol{y} \mid \boldsymbol{x}) + \log Pr(\boldsymbol{y} \mid \boldsymbol{x}) \;,$$

and since the term $\log Pr(y \mid \boldsymbol{x})$ is constant in w and difficult to calculate in general, we can instead minimize the cost function

$$\mathrm{Cost}(w) \;=\; -\log Pr(w, \boldsymbol{y} \mid \boldsymbol{x}) \;,$$

which can be directly computed using Lemma 5.1.

These cost functions differ from maximum likelihood methods for network training [BW87, EJM90] in that they introduce a prior term and sometimes have "nuisance" parameters eliminated. For instance, σ is termed a "nuisance" parameter when trying to determine a good set of weights and have no concern for the typical error. The prior term corresponds to the regularizing term found in smoothing methods [HT90] and used to good success in techniques such as weight-elimination [WRH91]. A similar Bayesian formulation is given in [Mac91].

Using the prior discussed in the previous section, with maximum *a posterior* analysis we would minimize in the regression case with Gaussian error and unknown σ

$$\begin{aligned} & -\log Pr(w, \boldsymbol{y} \mid \boldsymbol{x}) \\ = & \quad \frac{N-1}{2} \log \sum_{i=1,\ldots,N} (y_i - o(x_i, w))^2 - \log Pr(w) + \text{constant} \ , \end{aligned} \tag{4}$$

where the logarithm of the prior is determined as before. If the value of σ is known the cost to be minimized is

$$\frac{1}{2\sigma^2} \sum_{i=1,\ldots,N} (y_i - o(x_i, w))^2 - \log Pr(w) + \text{constant} \ .$$

Notice the difference between these above two cost functions. In the second case, the cost function is proportional to the mean squared error plus a "regularizing" term, but in the first case the logarithm of the mean squared error is taken. If the value of σ is determined from the inputs as well, as $o'(x_i, r)$, the cost to be minimized is now $-\log Pr(w, r, \boldsymbol{y} \mid \boldsymbol{x})$ given by

$$\sum_{i=1,\ldots,N} \frac{1}{2o'(x_i, r)^2} (y_i - o(x_i, w))^2 + \sum_{i=1,\ldots,N} \log o'(x_i, r) - \log Pr(r, w) + \text{constant} \ .$$

Minimization has to be done for r and w concurrently. A prior for r was given at the end of Section 5.3.

In the classification case, the overall cost is given by

$$-\log Pr(w, \boldsymbol{y} \mid \boldsymbol{x}) \ = \ - \sum_{i=1,\ldots,N} \log o_{y_i}(x_i, w) - \log Pr(w) + \text{constant} \ . \tag{5}$$

If the output variable is boolean and represented as 1 for true and 0 otherwise, then the first sum expands to

$$- \sum_{i=1,\ldots,N} \left(y_i \log o_1(x_i, w) \ + \ (1 - y_i) \log(1 - o_1(x_i, w)) \right) \quad .$$

Notice this is the familiar cross entropy error for the sample.

While this gives the cost function to minimize, we do not consider here the computational problem of how the minimization should be done. Various gradient descent, conjugate gradient or approximate second-order methods [BL88, EJM90] can be tried to solve the minimization problem. See, for instance, [PFTV88, Chap. 14] for a tutorial discussion of methods for solving the minimization problems for linear and non-linear regression with Gaussian and other more robust error models.

Notice the formulation given here assumes learning is done in batch (or epoch) mode because the posterior needs to be calculated on the full sample. When initializing search, it may help to use smaller batches to get somewhere near a local minimum, and perhaps do batch learning on the full sample when refining an already reasonable solution. This corresponds to the stochastic gradient descent strategy that is common in back-propagation implementations. We do not consider which strategy is better here.

6.2 Weight evaluation

To obtain a measure of the quality of each local maximum *a posterior* estimate $\widehat{w}$ found during search, an estimate of the local area under the posterior around $\widehat{w}$ is usually done. The actual posterior value itself is not the best measure of quality because some peaks may be thinner than others so they contain much less of the posterior probability in their vicinity. For instance, consider an idealized learning problem where the scalar parameter a is being learnt. Suppose a has posterior given in Figure 2. Notice the left peak is higher but

¡hand-drawn and not on line¿

Figure 2: Posterior for a

much thinner. The expected value of a, $\mathbf{E}_a(a)$, and other functions of a would therefore come more from the fatter peak on the right. This becomes more pronounced in higher dimensions, or when comparing networks of different dimension.

This local area estimate of quality lets different local maximum *a posterior* estimates be compared on an equal footing, for instance networks with different numbers of layers or connections. This local area estimate and successively coarser approximations to it corresponds to the various encoding measures such as MML [WF87] and MDL [Ris87], as discussed in the next section.

To make this estimate, we approximate the posterior at the local maximum $\widehat{w}$ by a function of the form

$$\begin{aligned} & Pr(w \mid \boldsymbol{x}, \boldsymbol{y}) \\ \approx \quad & Pr(\widehat{w} \mid \boldsymbol{x}, \boldsymbol{y}) \exp\left(-\frac{1}{2}(w-\widehat{w})^T I(\widehat{w})(w-\widehat{w}) + O((w-\widehat{w})^3)\right) , \end{aligned} \tag{6}$$

where

$$\begin{aligned} I(w) &= -\frac{\mathrm{d}^2 \log Pr(w \mid \boldsymbol{x}, \boldsymbol{y})}{\mathrm{d}w\mathrm{d}w} = -\frac{\mathrm{d}^2 \log Pr(w, \boldsymbol{y} \mid \boldsymbol{x})}{\mathrm{d}w\mathrm{d}w} , \\ &= \frac{\mathrm{d}^2 \mathrm{Cost}(w)}{\mathrm{d}w\mathrm{d}w} \end{aligned}$$

Notice the differentials are $\mathrm{d}w$ rather than ∂w because the derivative represents the full matrix of second derivatives w.r.t. the different weights $w_{n,m}$. Roughly, this approximation is valid for large sample size N because the posterior is found by normalizing for w a formula of the form $g(w)^N$ (see for instance [Ber85, p224]). The second derivative $I(\widehat{w})$ is taken of the cost function, the log posteriors such as in Equation (4) or (5). Generally, the sample size N should be at least some factor of the number of weights in the network.

When using a uniform prior, $I(\widehat{w})$ is referred to as the observed Fisher information matrix and its determinant is referred to as the Fisher information. Both play a central part in statistical theory [Ame85, Ber85]. In some simple non-network models, such as distributions from the exponential family [Ber85], class probability trees and Bayesian networks on discrete variables [Bun90], the posterior can be dealt with exactly so the approximation using $I(\widehat{w})$ is not necessary. Because the constants in Equations (4) and (5) are independent of w, they can be ignored when evaluating the derivative. The local posterior area for w near $\widehat{w}$ is found by integrating out w using the multivariate Gaussian approximation, and is given by

$$Pr(near\ \widehat{w} \mid \boldsymbol{x}, \boldsymbol{y}) \;\approx\; Pr(\widehat{w} \mid \boldsymbol{x}, \boldsymbol{y}) \frac{(2\pi)^{|w|/2}}{\det(I(\widehat{w}))^{1/2}} \;,$$

where $|w|$ is the dimensionality of the variable set w and $\det(\cdot)$ is the matrix determinant.

The desired quality measure is now given by the negative logarithm of the local area (and adding in the constant $-\log Pr(\boldsymbol{y} \mid \boldsymbol{x})$)

$$\begin{aligned} \mathrm{Eval}(\widehat{w}) &= -\log Pr(near\ \widehat{w}, \boldsymbol{y} \mid \boldsymbol{x}) \\ &= -\log Pr(\boldsymbol{y} \mid \widehat{w}, \boldsymbol{x}) - \log Pr(\widehat{w}) - \frac{|w|}{2}\log(2\pi) + \frac{1}{2}\log\det(I(\widehat{w})) \;, \end{aligned} \tag{7}$$

which wants to be minimized. Notice the first and second terms together give $\mathrm{Cost}(\widehat{w})$. We separate them out here because they correspond to the likelihood and prior components respectively.

Notice also, that from the functional form of Equation (6), the matrix inverse of $I(\widehat{w})$, $I(\widehat{w})^{-1}$, represents an approximate posterior variance-covariance matrix for w, [Ber85, p224], and is at least approximately correct in the neighborhood of $\widehat{w}$. This means, for instance, that the posterior variance of the weight $w_{n,m}$ in the neighborhood of $\widehat{w}$, denoted $\mathbf{v}_{w|\mathrm{x},\mathrm{y},near\ \widehat{w}}(w_{n,m})$, is given by the diagonal entry for $w_{n,m}$ in the matrix $I(\widehat{w})^{-1}$. That is, if $\mathbf{v}_{w|\mathrm{x},\mathrm{y},near\ \widehat{w}}(w_{n,m})$ is large, then the weight $w_{n,m}$ is poorly determined by the data and perhaps should be ignored.

The determinant $\det(I(\widehat{w}))$ could be approximated by looking only at the diagonals or block diagonals (block corresponds to one node which is square in the number of weights for the node). Individual second derivatives can be determined using a number of methods [BW91]. Fortunately, this only has to be done once for each local minima $\widehat{w}$ found, instead of repeatedly during search, so a fast approximation is not essential. Fast packages exist, such as the Fortran MATLIB, that can find determinants and inverses of matrices of row/column size several hundred in a few minutes.

6.3 Minimum encoding methods

Quality measures similar to $\mathrm{Eval}(\widehat{w})$ can be derived using various quantitization, encoding and approximation methods. We discuss these briefly here to draw the strong connections between $\mathrm{Eval}(\widehat{w})$ and the often discussed encoding measures given in [WF87, BC91, Ris87]. We view these methods as potentially useful approximations but do not consider their application here.

Rissanen's MDL

Rissanen develops his MDL measure [Ris87] as an upper bound on $-\log Pr(w, \boldsymbol{y} \mid \boldsymbol{x})$. For exposition purposes, we develop a related bound here. $I(\widehat{w})$ is a positive indefinite symmetric matrix (since $\widehat{w}$ is a local maxima), so[1]

$$\det I(\widehat{w}) \;\leq\; \left(\frac{\mathrm{trace} I(\widehat{w})}{|w|}\right)^{|w|} ,$$

where trace$A$ is the sum of the diagonal entries in the matrix A. Now for large N, trace$I(\widehat{w})/N$ is a mean so is approximately Gaussian, with mean $O(|w|)$ and standard deviation $O(|w|/\sqrt{N})$. Therefore with probability approaching 1 as N gets larger,

$$\frac{1}{2}\log\det I(\widehat{w}) \;=\; \frac{|w|}{2}\log N + O(|w|) .$$

We get that

$$\mathrm{Eval}(\widehat{w}) \;=\; -\log Pr(\boldsymbol{y} \mid \widehat{w}, \boldsymbol{x}) - \log Pr(\widehat{w}) + \frac{|w|}{2}\log N + O(|w|) .$$

This is an approximation developed by Schwarz and others [BC91]. The corresponding MDL form ignores the term $\log Pr(\widehat{w})$ which is probably $O(|w|)$ anyway and has an additional term $\frac{|w|}{2}\log|w|$. In our case, $O(|w|)$ is quite large because we are dealing with networks with many weights. It is not unusual in practice for the number of weights to be a similar order of magnitude to sample size N, so ignoring terms of order $O(|w|)$, as this approximation does, is probably unwise. We conclude this approximation is informative but perhaps too crude.

Wallace *et al.* and Barron *et al.*'s measure

Wallace and Freeman [WF87] and Barron and Cover [BC91] interpret a form like Eval$(\widehat{w})$ as the cost of encoding y and $\widehat{w}$ given $\boldsymbol{x}$. Wallace *et al.* refer to this as the minimum message length (MML). Of course, $\boldsymbol{y}$ and $\widehat{w}$ can only be encoded to finite precision (in classification, $\boldsymbol{y}$ is finite already). The precision of $\boldsymbol{y}$ is implicit in the supplied data, so if the volume of the precision for the data vector $\boldsymbol{y}$ is δ then the cost of encoding $\boldsymbol{y}$ given $\widehat{w}$ and $\boldsymbol{x}$ is

$$-\log Pr(\boldsymbol{y} \mid \widehat{w}, \boldsymbol{x}) - \log\delta .$$

Because $\boldsymbol{y}$ is encoded given $\widehat{w}$, we must encode $\widehat{w}$ without knowledge of $\boldsymbol{y}$. The second, third and fourth terms of Equation (7) are not an appropriate code length, however, because $I(\widehat{w})$ is dependent on $\boldsymbol{y}$.

The third and fourth terms of Equation (7) essentially correspond to the "precision" of $\widehat{w}$ because if the continuous space for w is quantified then the cell in the neighborhood of $\widehat{w}$ has prior probability given approximately by $Pr(\widehat{w})$Area$(\widehat{w})$ where Area$(\widehat{w})$ is the area of the cell, so $-\log \mathrm{Area}(\widehat{w})$ is the precision to which $\widehat{w}$ is specified. To interpret this precision component, notice that if the weights w happened to be *a posterior* independent (an unlikely event), then

$$-\frac{|w|}{2}\log(2\pi) + \frac{1}{2}\log\det(I(\widehat{w})) \;\approx\; -\sum_{n,m}\log\left(2.51\cdot\sqrt{\mathbf{v}_{w|\mathrm{x},\mathrm{y},near\ \widehat{w}}(w_{n,m})}\right) .$$

[1] The determinant is a product of eigenvalues, whereas the trace is a sum of eigenvalues, which in this case are all non-negative. The inequality follows by maximizing $\prod_i \lambda_i$ given a fixed value for $\sum_i \lambda_i$.

The recommended precision for encoding each weight is therefore approximately 2.5 times the posterior standard deviation of the weight.

Wallace *et al.* and Barron *et al.* make the fourth term in Equation (7) independent of y using an involved quantification argument. We can interpret this by noting

$$\frac{1}{2}\log\det(I(\widehat{w})) \;=\; \frac{|w|}{2}\log N + \frac{1}{2}\log\det\left(\frac{1}{N}I(\widehat{w})\right) \;.$$

The term $\frac{1}{N}I(\widehat{w})$ is now in the form of an average since in general $I(\widehat{w})$ is a sum across the N points in the sample plus a fixed prior term. We now replace this "average" observed Fisher information matrix by what is called the expected Fisher information matrix:

$$\begin{aligned}\text{Coding}(\boldsymbol{y},\widehat{w}) \;&=\; -\log Pr(\boldsymbol{y}\mid\widehat{w},\boldsymbol{x}) - \log\delta - \log Pr(\widehat{w}) - \frac{|w|}{2}\log(2\pi) - \frac{|w|}{2} \\ &\quad + \frac{|w|}{2}\log N + \frac{1}{2}\log\det(\overline{I}(\widehat{w}))\;,\end{aligned}$$

where $\overline{I}(w)$ is the expected Fisher information matrix (the expected value of the observed Fisher information for a single pattern) given by[2]

$$\begin{aligned}\overline{I}(w) \;&=\; \mathbf{E}_{x,y|w}\left(-\frac{\mathrm{d}^2\log Pr(x,y\mid w)}{\mathrm{d}w\mathrm{d}w}\right)\quad, \\ &=\; \mathbf{E}_{x}\left(\mathbf{E}_{y|w,x}\left(-\frac{\mathrm{d}^2\log Pr(y\mid w,x)}{\mathrm{d}w\mathrm{d}w}\right)\right)\quad, \\ &\approx\; \frac{1}{N}\sum_{i=1,\ldots,N}\mathbf{E}_{y|w,x_i}\left(-\frac{\mathrm{d}^2\log Pr(y\mid w,x_i)}{\mathrm{d}w\mathrm{d}w}\right)\quad,\end{aligned}$$

where (x,y) in these formula denotes a single pattern rather than the training sample $(\boldsymbol{x},\boldsymbol{y})$. Notice that this last formula is an estimate of $\frac{1}{N}I(\widehat{w})$ involving knowledge of the sample inputs $\boldsymbol{x}$ but not the sample outputs y. It is therefore a valid inclusion in the code-length of w.

7 Applications to network training

This section describes some applications of the tools just developed.

7.1 Weight variance and elimination

The Gaussian approximation to the likelihood near a local maximum *a posterior* also gives a statistically sound method for pruning of networks [LDS90]. The matrix inverse of $I(\widehat{w})$, $I(\widehat{w})^{-1}$, represents an approximate posterior variance-covariance matrix for w, so we can use this to test if a weight is significantly different from zero. Those that are not can be set to zero.

[2] There is actually some slight of hand here. Wallace *et al.* and Barron *et al.* develop their framework to deal with likelihoods of the form $p(x|w)$ for data x and parameters w whereas we are considering conditional likelihoods such as $p(y|w,x)$. Here we give our interpretation of their methods in this different context.

If we use the diagonal approximation to this matrix, then we get an estimate of the posterior variance of the weight $w_{n,m}$ in the neighborhood of $\widehat{w}$

$$\mathbf{v}_{w|\mathrm{x,y}}\left(w_{n,m}\right) \;\approx\; \mathbf{v}_{w|\mathrm{x,y},near\ \widehat{w}}\left(w_{n,m}\right) \;\approx\; -1 \Big/ \left.\frac{\partial \log Pr(w \mid \boldsymbol{x}, \boldsymbol{y})}{\partial w_{n,m} \partial w_{n,m}}\right|_{w=\widehat{w}} .$$

Because $\widehat{w}$ is a local maximum for the posterior, the second derivative must be non-positive so the variance estimate non-negative. A zero variance estimate means some other estimate will have to be made. A large variance means the sample gave us little idea as to what value the weight should be. So if the magnitude of a weight is within a standard deviation (square root of the variance), the weight can be safely set to zero, which is similar to the suggestion in [Ish90]. If the standard deviation of a weight is small, and the weight itself $w_{n,m}$ is smaller than the standard deviation, then the weight can be safely set to zero. Alternatively, if the weight's standard deviation is large but the magnitude of the weight larger, then the variance gives no support for zeroing the weight.

A more thorough test goes as follows. The quadratic term $(w-\widehat{w})^T I(\widehat{w})(w-\widehat{w})$ appears in the multivariate Gaussian approximation for the posterior for the weights w. For large N, we therefore get that this quadratic term is approximately χ-squared with $|w|$ degrees of freedom. The $X\%$ confidence region (for instance, $X = 99.0$) for the weights w is the set of weights within the upper $X\%$ point of the $\chi^2_{|w|}$ distribution, $\chi^2_{|w|,X}$, giving

$$w \quad : \quad (w-\widehat{w})^T I(\widehat{w})(w-\widehat{w}) \leq \chi^2_{|w|,X} \quad .$$

For instance, the upper 99% point for 30 weights is given from standard χ-squared tables as 50.89. We therefore proceed as follows. We repeatedly set weights to zero while the resultant weights remain in the $X\%$ confidence region. Notice this simple test can be done directly from $I(\widehat{w})$ without having to first calculate a determinant or inverse, so might also be done at stages during the back-propagation cycle.

7.2 Prediction and generalization error

Once search has located some local minima $\widehat{w}$ to the cost function, they can be used in inference on new patterns. A naive approximation is to say that w must now be $\widehat{w}$ and to make inference about y and σ^2, etc., based entirely on this local minima $\widehat{w}$ using $y = o(x', \widehat{w})$, etc. The reason this is naive is that in reality we cannot be sure that the "true" or "optimum" w is $\widehat{w}$. For instance, with a sample twice the size we might change our estimate of w quite dramatically. A full Bayesian approach says to take into account the many other values of w near $\widehat{w}$ or even some other local minima because these other values just might happen to be the "true" one. Only when we have a really large sample (for instance, when uniform convergence bounds tell us the sample is large enough [Hau91]) can we be sure that $\widehat{w}$ is a sufficiently good estimate so that no other need be considered.

In principle what we would like to calculate for a new pattern x' is $\mathbf{E}_{w|\mathrm{x,y}}\left(o(x', w)\right)$ and $\mathbf{E}_{w,\sigma|\mathrm{x,y}}\left(\sigma^2\right)$, etc. These values give us the posterior average of the quantities $o(x', w)$ and σ^2 that give better predictions for these quantities on average than those calculated for $w = \widehat{w}$. The problem here is that we cannot calculate the posterior $Pr(w \mid \boldsymbol{x}, \boldsymbol{y})$ in reasonable time, and only have the Gaussian approximation to the posterior described above.

This section describes how these predictions can be approximated using the posterior approximations available. These approximations all make use of the second derivatives of the cost function and the approximate weight variance discussed above. While it is hard to predict how accurate these approximations will be in practice, their form at least gives some idea of the interaction between weight variances and the behavior of the network output for w in the vicinity of $\widehat{w}$.

Predictions are given for three quantities. The first is an approximation to $\mathbf{E}_{w|\mathrm{x,y}}\left(o(x', w)\right)$. In classification this can be used to get an estimate of the posterior distribution for y given x', and can be used to predict generalization error, and in regression this gives an estimate of the posterior mean of y. This can also be used to give generalization error for classification. The second prediction is for $\mathbf{v}_{w|\mathrm{x,y}}\left(o(x', w)\right)$ which measures our uncertainty in the first prediction. Notice as the sample size gets large this uncertainty will shrink to zero. We give two versions of this uncertainty depending on which approximation to $\mathbf{E}_{w|\mathrm{x,y}}\left(o(x', w)\right)$ is used. The third prediction is for $\mathbf{E}_{w,\sigma|\mathrm{x,y}}\left(\sigma^2\right)$ which predicts generalization error for regression with unknown standard deviation.

Predictions for a local minima

In what follows, we use the notation $\mathbf{E}_{w|\mathrm{x,y},near\ \widehat{w}}\left(U(w)\right)$ to denote the expectation of the quantity $U(w)$ when averaged according to the approximate Gaussian posterior for w in the vicinity of $\widehat{w}$. That is

$$\mathbf{E}_{w|\mathrm{x,y},near\ \widehat{w}}\left(U(w)\right) \;=\; \int_w U(w)\, \frac{\det(I(\widehat{w}))^{\frac{1}{2}}}{(2\pi)^{|w|/2}} \exp\left(-\frac{1}{2}(w-\widehat{w})^T I(\widehat{w})(w-\widehat{w})\right) dw\ .$$

We evaluate these integrals by using standard moments of the multivariate Gaussian [Ber85], and approximating $U(w)$ in the vicinity of $\widehat{w}$ using the second-order Taylor expansion which in vector notation is as follows:

$$\begin{aligned} U(w) \;\approx\; & U(\widehat{w}) \;+\; (w-\widehat{w})^T \cdot \left.\frac{\mathrm{d}U(w)}{\mathrm{d}w}\right|_{w=\widehat{w}} \\ & +\; \frac{1}{2}\;(w-\widehat{w})^T \cdot \left.\frac{\mathrm{d}^2U(w)}{\mathrm{d}w\mathrm{d}w}\right|_{w=\widehat{w}} \cdot (w-\widehat{w})\ . \end{aligned}$$

Again, $\frac{\mathrm{d}o(x',w)}{\mathrm{d}w}$ denotes a vector of first derivatives and the second derivative a matrix. This gives

$$\mathbf{E}_{w|\mathrm{x,y},near\ \widehat{w}}\left(U(w)\right) \;\approx\; U(\widehat{w}) \;+\; \frac{1}{2}\mathrm{trace}\left(I(\widehat{w})^{-1} \cdot \left.\frac{\mathrm{d}^2U(w)}{\mathrm{d}w\mathrm{d}w}\right|_{w=\widehat{w}}\right) \quad ,$$

where, again, trace$A$ denotes the sum of the diagonal entries of the matrix A. If any of the approximate weight variances are large, then the approximation of $\mathbf{E}_{w|\mathrm{x,y},near\ \widehat{w}}\left(U(w)\right)$ will require accurate estimates of $U(w)$ for w far from $\widehat{w}$. In this case the Taylor expansion will be a poor approximation and our approximate predictions will be poor. Notice we could also use the diagonal approximation for the matrices $I(\widehat{w})$ and the second derivatives of $U(w)$, however, the estimates may then become very poor and only be useful to indicate where potential problems lie. These and other approximations are discussed in [Pre89].

$\mathbf{E}_{w|\mathrm{x,y},near\ \widehat{w}}\left(o(x', w)\right)$: the posterior expected output of the network given input x', averaged over weights in the vicinity of $\widehat{w}$. This is approximated by

$$o(x', \widehat{w}) + \frac{1}{2}\mathrm{trace}\left(I(\widehat{w})^{-1}\, \left.\frac{\mathrm{d}^2 o(x', w)}{\mathrm{d}w\mathrm{d}w}\right|_{w=\widehat{w}}\right) \quad .$$

This modifies the output $o(x', \widehat{w})$ to account for how the output $o(x', w)$ varies in the neighborhood of $\widehat{w}$, tempered by the posterior variance-covariance of w. In the case of classification, this gives an estimate of out-of-sample accuracy for the network given by

$$
\begin{aligned}
&\frac{1}{N}\sum_{i=1}^{N} \mathbf{E}_{w|\mathrm{x},\mathrm{y},near\ \widehat{w}}\left(o_{y_i}(x_i, w)\right) \\
&\approx \quad \frac{1}{N}\sum_{i=1}^{N} o_{y_i}(x_i, \widehat{w}) + \frac{1}{2N}\mathrm{trace}\left(I(\widehat{w})^{-1}\, \frac{\mathrm{d}^2 \sum_{i=1}^{N} o_{y_i}(x_i, w)}{\mathrm{d}w \mathrm{d}w}\bigg|_{w=\widehat{w}} \right) \quad . \qquad (8)
\end{aligned}
$$

$\mathbf{E}_{w|\mathrm{x},\mathrm{y},near\ \widehat{w}}\left((o(x', w) - o(x', \widehat{w}))^2\right)$: when using $o(x', \widehat{w})$ to estimate $o(x', w)$, the expected variance of the output in the neighborhood of $\widehat{w}$, which gives a measure of variance for our posterior uncertainty when using $o(x', \widehat{w})$ to estimate $o(x', w)$. For instance, in the regression case $o(x', \widehat{w})$ corresponds to an estimate of the mean value of y conditioned on x'. The variance is approximated by

$$
\frac{\mathrm{d}o(x', w)}{\mathrm{d}w}\bigg|_{w=\widehat{w}}^{T} \cdot I(\widehat{w})^{-1} \cdot \frac{\mathrm{d}o(x', w)}{\mathrm{d}w}\bigg|_{w=\widehat{w}} \qquad (9)
$$

$\mathbf{v}_{w|\mathrm{x},\mathrm{y},near\ \widehat{w}}(o(x', w))$: the variance of the estimate $\mathbf{E}_{w|\mathrm{x},\mathrm{y},near\ \widehat{w}}(o(x', w))$ in the neighborhood of $\widehat{w}$. This is approximated by

$$
\frac{\mathrm{d}o(x', w)}{\mathrm{d}w}\bigg|_{w=\widehat{w}}^{T} \cdot I(\widehat{w})^{-1} \cdot \frac{\mathrm{d}o(x', w)}{\mathrm{d}w}\bigg|_{w=\widehat{w}} - \frac{1}{4}\mathrm{trace}\left(I(\widehat{w})^{-1} \cdot \frac{\mathrm{d}^2 o(x', w)}{\mathrm{d}w \mathrm{d}w}\bigg|_{w=\widehat{w}} \right) \quad .
$$

Notice this is lower than the previous variance estimate of Formula (9) due to the second term which corrects for the fact that we are now using a better estimate for $o(x', w)$.

$\mathbf{E}_{w,\sigma|\mathrm{x},\mathrm{y},near\ \widehat{w}}\left(\sigma^2\right)$: the posterior expected value of σ^2 in the regression case with Gaussian error and unknown σ. This is sometimes termed the generalization error. This is approximated using Equation (3) by

$$
\frac{1}{N-1}\sum_{i=1,\ldots,N} (\mathbf{E}_{w|\mathrm{x},\mathrm{y},near\ \widehat{w}}(o(x_i, w)) - y_i)^2 + \frac{1}{N-1}\sum_{i=1,\ldots,N} \mathbf{v}_{w|\mathrm{x},\mathrm{y},near\ \widehat{w}}(o(x_i, w)) \ ,
$$

where the variances in the second sum were just given. This simplifies to

$$
\frac{N}{N-1}\ s^2\big|_{w=\widehat{w}} + \frac{N}{2(N-1)} trace\left(I(\widehat{w})^{-1} \cdot \frac{\mathrm{d}^2 s^2}{\mathrm{d}w \mathrm{d}w}\bigg|_{w=\widehat{w}} \right) \quad . \qquad (10)
$$

Notice the first term is the observed variance, and the second term increases this because the estimate $\mathbf{E}_{w|\mathrm{x},\mathrm{y},near\ \widehat{w}}(o(x_i, w))$ is only approximate and has a variance of its own.

Multiple networks

If multiple local minima $\widehat{w}$ of similar quality have been found during restarts in the search for weights, then the expected values above can be averaged over the different local minima $\widehat{w}$ to produce a pooled estimate. This same "multiple models" approach can produce significant improvement in out-of-sample prediction when learning classification trees [Bun90]. It corresponds to a Monte Carlo estimation of the full posterior for w, rather than just an approximation about $\widehat{w}$. Suppose a small set $\widehat{W}$ of local minima are found, and for each we have a proportion $p(\widehat{w} \mid \boldsymbol{x}, \boldsymbol{y})$ to be used in the Monte Carlo estimation below, where $\sum_{\widehat{w} \in \widehat{W}} p(\widehat{w} \mid \boldsymbol{x}, \boldsymbol{y}) = 1$. For instance, these proportions could be constructed proportional to $Pr(near\ \widehat{w}, \boldsymbol{y} \mid \boldsymbol{x})$, although equal proportions are sometimes used in Monte Carlo estimation. The estimate of the posterior for the weights w is now a weighted some of the Gaussians in the neighborhood of each $\widehat{w}$:

$$Pr(w \mid \boldsymbol{x}, \boldsymbol{y}) \approx \sum_{\widehat{w} \in \widehat{W}} p(\widehat{w} \mid \boldsymbol{x}, \boldsymbol{y}) Pr(w \mid \boldsymbol{x}, \boldsymbol{y}, near\ \widehat{w}),$$

where the Gaussian approximation described at the beginning of this section is used for each $Pr(w \mid \boldsymbol{x}, \boldsymbol{y}, near\ \widehat{w})$.

This leads to the following corrections for the previous predictions. To find the expected output of the network for different weights, we pool the expected outputs for the individual local minima in $\widehat{W}$:

$$\mathbf{E}_{w|\mathrm{x,y}}\left(o(x', w)\right) \approx \sum_{\widehat{w} \in \widehat{W}} p(\widehat{w} \mid \boldsymbol{x}, \boldsymbol{y}) \mathbf{E}_{w|\mathrm{x,y},near\ \widehat{w}}\left(o(x', w)\right) \quad .$$

To find the variance of this value, we pool the variances for individual local minima together with a measure of how much the output varies from minima to minima:

$$\begin{aligned} \mathbf{v}_{w|\mathrm{x,y}}\left(o(x', w)\right) \quad \approx \quad & \sum_{\widehat{w} \in \widehat{W}} p(\widehat{w} \mid \boldsymbol{x}, \boldsymbol{y}) \mathbf{v}_{w|\mathrm{x,y},near\ \widehat{w}}\left(o(x', w)\right) \\ & + \ \mathbf{v}_{\widehat{w}|\mathrm{x,y}}\left(\mathbf{E}_{w|\mathrm{x,y},near\ \widehat{w}}\left(o(x', w)\right)\right) \quad . \end{aligned}$$

To estimate the standard error σ in regression with a Gaussian error model, we pool the individual estimates of σ.

$$\mathbf{E}_{w,\sigma|\mathrm{x,y}}\left(\sigma^2\right) \approx \sum_{\widehat{w} \in \widehat{W}} p(\widehat{w} \mid \boldsymbol{x}, \boldsymbol{y}) \mathbf{E}_{w,\sigma|x,\boldsymbol{y},near\ \widehat{w}}\left(\sigma^2\right) \ .$$

7.3 Adjustments for missing values

In many practical problems, training patterns are supplied with some values of variables missing, unknown or undetermined. Of course if a pattern has its output value missing the pattern can just be ignored. If an input variable has a value missing, however, we need to deal with it. Approximate methods for dealing with this kind of problem exist when learning tree classifiers [Qui89]:

- One can ignore the training pattern with missing values. Stochastic learning can proceed initially using only the patterns without missing values.

- One can replace the missing value by some simple estimate such as the modal or mean value. For instance, if the boolean input variable x_i is either 0 (false) or 1 (true) and if it is unknown, set it to be about 0.5.
- One can complete the pattern in various ways (to fill in missing values) and treat each of these completions as a partial pattern (so the sum of fractional patterns adds to 1). We explain this idea in the context of feed-forward networks below. Tree learning algorithms do this completion in an efficient demand driven manner (i.e. they complete a missing value only when the value is asked for by the algorithm) that unfortunately is not available with feed-forward networks.

The second approach works quite well and is the simplest to implement, however the third approach gives the best performance in terms of out-of-sample prediction.

In this section we develop a modified algorithm for handling missing values related to the third approach above. It also approximates the Bayesian normative approach for handling missing values because we derive the approach here using standard laws of probability. The method is another example of the use of mixture models [JJNH91] used for adaptive mixtures of local experts, and the equations derived have a related form. We do the analysis for the regression problem with Gaussian error and unknown standard deviation. The other cases are similar. We assume we have a model of some kind denoted F that gives rough prediction of the missing values from the known values. The model F might correspond to simple linear models or some other form readily calculated from the data.

The training sample consists of input vectors $x_1, \ldots, x_N$ and corresponding output values $y_1, \ldots, y_N$. But in this case some of the input vectors x_i have missing values. Let $unknown(x_i)$ denote the set of variables in the i-th input vector x_i that have missing values. For instance, given the i-th boolean pattern $x_i = (1, 0, ?, 0, ?, ?)$ where ? denotes a missing value, then $unknown(x_i)$ is the 3rd, 5th and 6th variables. Given an assignment to these, $x' \in unknown(x_i)$, let $x_i x'$ denote one possible completion for the input vector x_i. Then the likelihood of the pattern is now given by

$$l(y_i \mid x_i, w, \sigma, F) \;=\; \mathbf{E}_{x' \in unknown(x_i) | x_i, F}\left(l(y \mid x_i x', w, \sigma)\right)$$

where the expectation is done using the model F for predicting the unknown values. We can approximate this stochastically by selecting a few possible completions $compl(x_i)$ and weighting them to give

$$l(y_i \mid x_i, w, \sigma, F) \;\approx\; \sum_{x' \in compl(x_i)} p(x' \mid x_i, F) l(y \mid x_i x', w, \sigma) \;, \tag{11}$$

where the weights $p(x' \mid x_i, F)$ indicate likelihood of the different completions based on the model F. This formula can now be evaluated because each likelihood $l(y \mid x_i x', w, \sigma)$ can be determined as in Section 5.1. For instance, in the example above, we may select 3 completions according to the model for predicting missing values as

$$compl(x_i) = \{\, (1,0,0,0,0,1),\ (1,0,0,0,1,1),\ (1,0,1,0,1,0) \,\}$$

and give them equal weighting of 1/3 each. Completions and weightings can be determined once before the network training begins. Notice the crudest estimate is to use only the single modal or mean value for the missing values with a weight of 1, as is often done.

Network training now consists of using these modified likelihoods. In the regression case we are considering, the standard deviation can no longer be marginalized out so the cost function is now $-\log Pr(w, \sigma, \boldsymbol{y}|F, \boldsymbol{x})$. Suppose the patterns in *missing* have missing values and those in *complete* have all values given, then the cost is:

$$\sum_{i \in complete} \frac{1}{2\sigma^2}(y_i - o(x_i, w))^2 + (N+1)\log\sigma - \log Pr(w) + \text{constant}$$
$$- \sum_{i \in missing} \log\left(\sum_{x' \in compl(x_i)} p(x'|x_i, F) \exp\left(\frac{-1}{2\sigma^2}(y_i - o(x_i x', w))^2 \right) \right) .$$

While this looks more involved than the original cost function of Equation (4), its derivatives are not that different.

$$\begin{aligned} &\frac{\partial \text{Cost(w)}}{\partial w_{n,m}} \\ &= -\frac{\partial \log Pr(w)}{\partial w_{n,m}} - \frac{1}{2\sigma^2} \sum_{i \in complete} (y_i - o(x_i, w)) \frac{\partial o(x_i, w)}{\partial w_{n,m}} \\ &\quad -\frac{1}{2\sigma^2} \sum_{i \in missing} \sum_{x' \in compl(x_i)} w(x' \mid x_i, F, \sigma) \cdot (y_i - o(x_i x', w)) \frac{\partial o(x_i x', w)}{\partial w_{n,m}} , \end{aligned} \tag{12}$$

where the proportions $w(x'|x_i, F, \sigma)$ are calculated as

$$w(x'|x_i, F, \sigma) \equiv \frac{p(x' \mid x_i, F) \exp\left(\frac{-1}{2\sigma^2}(y_i - o(x_i x', w))^2\right)}{\sum_{x' \in compl(x_i)} p(x' \mid x_i, F) \exp\left(\frac{-1}{2\sigma^2}(y_i - o(x_i x', w))^2\right)} .$$

Compare the two sums in Equation (12). In the second, the term for the completion $x_i x'$ appears weighted by the proportion $w(x'|x_i, F, \sigma)$ but in the term for the first sum each (already complete) pattern x_i is effectively weighted by 1. For this reason we say each completion participates as a partial pattern. Notice, also that the proportions $w(x'|x_i, F, \sigma)$ essentially pick out that completion x' giving lowest squared error. Since the derivative of $o(x_i x', w)$ w.r.t. the completion x' is available after the back-propagation step, we could dynamically alter the completions x' during each learning step so their mean-square error become lower.

The derivative of Cost w.r.t. σ is in a similar form but this time a fixed-point equation (σ also occurs on the right-hand side) for σ can be derived:

$$\sigma = \frac{1}{N+1} \left(\sum_{i \in complete} (y_i - o(x_i, w))^2 + \sum_{i \in missing} \sum_{x' \in compl(x_i)} w(x' \mid x_i, F, \sigma) \cdot (y_i - o(x_i x', w))^2 \right) .$$

The standard deviation σ can be updated iteratively along with the weights w in each cycle. Again we see the "partial patterns" alter the usual form for the variance. This is a common result in mixture models.

A disadvantage of this method is that each pattern with missing values now produces a number of completed patterns, each of which must be run through the network. For instance, if every pattern has missing values and c different completions are used for each, then computation is increased by a factor of c. One way around this would be to have the network first learn for patterns with no missing values, and then to fine tune by including the remaining patterns. Of course, the extra computation should give improved performance as c is increased due to the normative justification of the approach. Also, the completed patterns can be dynamically altered during training with little extra overhead, to produce completions with lower mean-square error.

8 Conclusion

This paper has covered Bayesian theory relevant to the problem of training feed-forward connectionist networks. We now sketch out how this might be put together in practice, assuming a standard gradient descent algorithm as used during search.

For network training, the principle steps are as follows:

1. Choose an appropriate network structure and size based on prior knowledge about the application (see for instance the discussion in [LeC89] regarding choice of network), and select a prior on the weights. Notice a small number of different structures could be selected and the method will the best select the best.

2. Construct completions and their proportions $p(x'|x_i, F)$ for each training pattern with missing values. If stochastic training is used rather than epoch training, each set of completions should always be run through the network together so that the appropriate term in Equation (12) can be calculated.

3. Train to a local minima $\widehat{w}$ as per usual but incorporating the adjustments described for missing values in Section 7.3. Appropriate cost functions are given in Section 6.1. In principle, the weight variances, weights evaluation and predictions only apply if $\widehat{w}$ found is a true local minima to the cost function so epoch training might have to be used in the last few cycles to ensure this.

4. Possibly, use the weight elimination strategy of Section 7.1 to force some weights to zero, and continue training the network with forced weights remaining at zero.

5. Once a local minima is found, estimate the quality of the local minima by finding second derivatives for every training pattern and combining them in the Evaluation measure of Section 6.2. Calculation of second derivatives is described in [BW91].

6. Do random restarts of the network to repeat the last three steps and find other local minima. The estimated weight variances $\mathbf{v}_{w|\mathrm{x},\mathrm{y},near\ \widehat{w}}(w_{n,m})$ or $I(\widehat{w})$ and the evaluation measure $\mathrm{Eval}(\widehat{w})$ should be retained for each low cost local minima $\widehat{w}$ saved. If several different network structures are being tried, then repeat the last three steps for these as well.

7. Choose a few networks with the best evaluation ($\mathrm{Eval}(\widehat{w})$). Notice that the networks or local minima should not be chosen on the basis of their generalization error (as given in Section 7.2) because the generalization error for a particular local minima $\widehat{w}$ is estimated based on the assumption that the network structure is "correct" and the "true" weights are in fact quite near $\widehat{w}$.

8. Estimate the generalization error (for out-of-sample prediction) using the Formulas (8) or (10), possibly combined using the pooled versions at the end of Section 7.2. Now that a high posterior structure and set of weights has been chosen, the assumptions behind the formula are reasonable.

Once some local minima have been found, inference can be done on new patterns. Relevant approximate predictions are described in Section 7.2 that apply if the weight variances are small. Standard inference does forward propagation of the inputs for the new pattern to obtain the output. The adjustments described involve approximation of output posterior variance of the output (error bars), better approximation of expected output and variance, and averaging over multiple local minima.

These adjustments to the standard back-propagation procedure increase computation during back-propagation in most cases by at most a factor. Subsequent inference such as calculating variances require matrix calculations that can be done using fast standard matrix packages. While these methods come with the normative backing of Bayesian statistics, implementation often reveals lessons on how the various approximations and optimizations could be better made. One important issue here is the quality of the Gaussian approximation to the posterior of the weights. Since the whole approach rests on this, more evaluation and experience is required. A smooth transition also needs to be developed between Bayesian and uniform convergence methods to handle those cases where training samples become larger.

Acknowledgements

One of us (ASW) would like to thank David Rumelhart for many fruitful discussions. The other (WLB) would like to thank Robin Hanson likewise.

References

[Ame85] T. Amemiya. *Advanced Econometrics.* Harvard University Press, Cambridge, MA, 1985.

[BC91] A.R. Barron and T.M. Cover. Minimum complexity density estimation. *IEEE Trans. on IT*, ??, 1991.

[Ber85] J. O. Berger. *Statistical Decision Theory and Bayesian Analysis.* Springer-Verlag, New York, 1985.

[BH89] Eric B. Baum and David Haussler. What size net gives valid generalization? *Neural Computation*, 1:151–160, 1989.

[BL88] S. Becker and Y. Le Cun. Improving the convergence of back-propagation learning with second order methods. In David S. Touretzky, Geoffrey E. Hinton, and Terrence J. Sejnowski, editors, *Proceedings of the 1988 Connectionist Models Summer School*, pages 29–37. Morgan Kaufmann, 1988.

[Bri89] J.S. Bridle. Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In F. Fougelman-Soulie and J. Hérault, editors, *Neuro-computing: Algorithms, Architectures and Applications.* Springer-Verlag, 1989.

[Bun90] W.L. Buntine. Learning classification trees. Technical Report FIA-90-12-19-01, RIACS and NASA Ames Research Center, Moffett Field, CA, 1990. Paper presented at Third International Workshop on Artificial Intelligence and Statistics.

[Bun91] W.L. Buntine. *A Theory of Learning Classification Rules.* PhD thesis, University of Technology, Sydney, 1991.

[BW87] E.B. Baum and F. Wilczek. Supervised learning of probability distributions by neural networks. In D.Z. Anderson, editor, *Neural Information Processing Systems (NIPS)*, pages 52–61, 1987.

[BW91] W.L. Buntine and A.S. Weigend. Calculating second derivatives on feed-forward networks. Sunbmitted, 1991.

[Cot90] N.E. Cotter. The Stone-Weierstrass theorem and its application to neural networks. *IEEE Trans. on Neural Networks*, 1(4):290–295, 1990.

[DL91] John S. Denker and Yann Le Cun. Transforming neural-net output levels to probability distributions. In R.P. Lippmann, J.E. Moody, and D.S. Touretzky, editors, *Advances in Neural Information Processing Systems 3*, page 853. Morgan Kaufmann, 1991.

[EJM90] A. El-Jaroudi and J. Makhoul. A new error criterion for posterior probability estimation with neural nets. In *Int. Joint Conf. on Neural Networks*, pages III–185–192, San Diego, CA, 1990.

[Gol88] R.M. Golden. A unified framework for connectionist systems. *Biological Cybernetics*, 59:109–120, 1988.

[Goo83] I.J. Good. *The Goodbook.* University of Minnesota Press, Minnesota, 1983.

[Hau91] D. Haussler. A decision theoretic generalization of the PAC learning model and its application to some feed-forward neural networks. *Information and Control*, 1991. To appear.

[HHL86] E.J. Horvitz, D.E. Heckerman, and C.P. Langlotz. A framework for comparing alternative formalisms for plausible reasoning. In *Fifth National Conference on Artificial Intelligence*, pages 210–214, Philadelphia, 1986.

[HP90] J.B. Hampshire II and B.A. Pearlmutter. Equivalence proofs for multi-layer perceptron classifiers and the Bayesian discrimination function. In David S. Touretzky, Jeffrey L. Elman, Terrence J. Sejnowski, and Geoffrey E. Hinton, editors, *Proceedings of the 1990 Connectionist Models Summer School.* Morgan Kaufmann, 1990.

[HT90] T.J. Hastie and R.J. Tibshirani. *Generalised Additive Models.* Chapman and Hall, London, 1990.

[Ish90] M. Ishikawa. A structural learning algorithm with forgetting of link weights. Technical Report TR-90-7, Electrotechnical Laboratory, Life Electronics Research Center, Tokyo, 1990. Modified version of a paper presented at *IJCNN*, Washington D.C. 1989.

[JJNH91] R.A. Jacobs, M.I. Jordan, S.J. Nowlan, and G.E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1), 1991.

[LDS90] Yann Le Cun, John S. Denker, and Sara A. Solla. Optimal brain damage. In David S. Touretzky, editor, *Advances in Neural Information Processing Systems 2 (NIPS*89)*, page 589. Morgan Kaufmann, 1990.

[LeC88] Y. Le Cun. A theoretical framework for back-propagation. In David S. Touretzky, Geoffrey E. Hinton, and Terrence J. Sejnowski, editors, *Proceedings of the 1988 Connectionist Models Summer School*, pages 21–28. Morgan Kaufmann, 1988.

[LeC89] Y. Le Cun. Generalization and network design strategies. Technical Report CRG-TR-89-4, Dept. of Computer Science, University of Toronto, Toronto, M5S 1A4, Canada, 1989.

[LTS89] E. Levi, N. Tishby, and S.A. Solla. A statistical approach to learning and generalization in layered neural networks. In R. Rivest, D. Haussler, and M.K. Warmuth, editors, *COLT'89: Second Workshop on Computational Learning Theory*, pages 245–260, University of California, Santa Cruz, 1989. Morgan Kaufmann.

[Mac91] D.J.C. Mackay. A practical Bayesian framework for backprop networks. Submitted to *Neural Computation*, 1991.

[MN89] P. McCullagh and J.A. Nelder. *Generalised Linear Models*. Chapman and Hall, London, second edition, 1989.

[OH91] M. Opper and D. Haussler. Generalised performance of Bayes optimal classification algorithm for learning a perceptron. In *COLT'91: 1991 Workshop on Computational Learning Theory*. Morgan Kaufmann, 1991. Manuscript.

[Pan89] Panel. Discriminant analysis and clustering. *Statistical Science*, 4(1):34–69, 1989.

[Pea88] J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan and Kauffman, 1988.

[PFTV88] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, Cambridge, 1988.

[Pre89] S.J. Press. *Bayesian Statistics*. Wiley, New York, 1989.

[Qui86] J.R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[Qui89] J.R. Quinlan. Unknown attribute values in induction. In *Proceedings of the Sixth International Machine Learning Workshop*, Cornell, New York, 1989. Morgan Kaufmann.

[RHW86] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning internal representations by error propagation. In David E. Rumelhart, James L. McClelland, and the PDP Research Group, editors, *Parallel Distributed Processing*, page 318. MIT Press, 1986.

[Ris87] J. Rissanen. Stochastic complexity. *J. Roy. Statist. Soc. B*, 49(3):223–239, 1987.

[SST91] H.S. Seung, H. Sompolinsky, and N. Tishby. Statistical mechanics of learning from examples; i. general formulation and annealing approximation. Manuscript, 1991.

[TSN90] G.G. Towell, J.W. Shavlik, and M.O. Noordewier. Refinement of approximate domain theories by knowledge-based neural networks. In *Eighth National Conference on Artificial Intelligence*, pages 861–866, Boston, Massachusetts, 1990.

[Vap82] V. Vapnik. *Estimation of Dependencies Based on Empirical Data.* Springer-Verlag, New York, 1982.

[WF87] C.S. Wallace and P.R. Freeman. Estimation and inference by compact encoding. *J. Roy. Statist. Soc. B*, 49(3):240–265, 1987.

[WHR90] Andreas S. Weigend, Bernardo A. Huberman, and David E. Rumelhart. Predicting the future: a connectionist approach. *International Journal of Neural Systems*, 1:193–209, 1990.

[WRH91] Andreas S. Weigend, David E. Rumelhart, and Bernardo A. Huberman. Generalization by weight-elimination with application to forecasting. In Richard P. Lippmann, John Moody, and David S. Touretzky, editors, *Advances in Neural Information Processing Systems 3 (NIPS*90).* Morgan Kaufmann, 1991.

[Zel90] A. Zellner. Bayesian methods and entropy in economics and econometrics. In W.T. Grandy, Jr. and L. Schlick, editors, *Maximum Entropy and Bayesian Methods.* Kluwer, 1990.