Covariance-based weighting for optimal combination of model predictions

William D. Penny, Dirk Husmeier and Stephen J. Roberts

Department of Electrical and Electronic Engineering,
Imperial College, London SW7 2BT.
{w.penny,d.husmeier,s.j.roberts}@ic.ac.uk

Abstract

This paper introduces a method for calculating the covariance between different neural network solutions. It is based on a generalisation of the delta method for calculating the network Hessian and generates what we call the 'cross-covariance' matrix (its inverse is the 'cross-Hessian'). Using this matrix we are able to estimate the covariance between network predictions at each point in input space, using training data alone. Whilst this is a significant result in itself we have also applied the method to the problem of finding optimal linear combinations of models. This results in a 'covariance-based' weighted committee, where the weights are input-dependent. If the individual networks are unbiased then the covariance-based weighted committee is optimal in the sense of minimum expected prediction error.

1 Introduction

It is well known that combining predictions from many neural networks to form a single 'committee' prediction often results in reduced prediction error [1]. For regression problems, the simplest combining method is to take the mean prediction. More accurate predictions can however be made by forming linear combinations. The problem then remains of how to assign 'weights' to each model.

In this paper, we consider a weighting scheme in which the weights are input-dependent and are set according to the covariance of the model predictions at each point in input space (the so-called 'pointwise' covariance matrix). Tresp and Taniguchi [9] consider a diagonal approximation to this matrix which results in a weighting scheme where each predictor is weighted in proportion to the inverse variance of its prediction. The diagonal approximation, however, assumes that the network predictions are uncorrelated; clearly an unrealistic assumption considering that the networks have been trained on the same data to perform the same task.

In this paper, we show how a first-order Taylor series expansion can be used to estimate the pointwise covariance matrix. This estimate is based on training data alone and therefore, unlike other methods, no validation data set [5] or surrogate data set [4] is required. We then go on to show how the covariance matrices are used to define a covariance-based committee weighting scheme and compare it to variance-based weighting and uniform weighting.

2 A Single Network

We consider the nonlinear regression framework where $D = \{\boldsymbol{x}_i, y_i\}_{i=1}^N$ is a data set consisting of input-output pairs and e_i is independent and identically distributed (IID) additive target noise with variance σ^2. The target noise captures the notion that we may not be able to model the data perfectly. Data are assumed to be generated from the model $y_i = f(\boldsymbol{x}_i, \hat{\boldsymbol{w}}) + e_i$. We consider f to be a neural network where $\hat{\boldsymbol{w}}$ is a weight vector of dimension p and the network prediction is $\hat{y}_i = f(\boldsymbol{x}_i, \hat{\boldsymbol{w}})$. The goodness of fit of the network to the data can be assessed by a least squares cost function. The choice of $\hat{\boldsymbol{w}}$ that minimises this cost function can be found by an iterative optimization procedure such as conjugate gradients. This algorithm will find only local minima solutions. This is not considered a problem, however, as the committee weighting scheme will be used to combine predictions from many different local-minima solutions in an optimal way.

Different instantiations of target noise will result in a different estimated weight vector (so will different aspects of the training procedure, eg. different initialisation and convergence criteria, but, for the purposes of this paper, we assume these factors to be constant). The consequent variation around a single local minimum solution can be estimated from a first order Taylor series as [7]

$$\Delta \hat{\boldsymbol{w}} = (\boldsymbol{G}^T \boldsymbol{G})^{-1} \boldsymbol{G}^T \boldsymbol{e}$$

where $\boldsymbol{G}$ is an N-by-p matrix whose ith row consists of the vector $\boldsymbol{g}_i$, the vector of first derivatives of network output with respect to its weights on the ith data point, and $\boldsymbol{e}$ is the target noise vector. The weight variance is

$$Var(\Delta \hat{\boldsymbol{w}}) = \sigma^2 (\boldsymbol{G}^T \boldsymbol{G})^{-1} \qquad (1)$$

When the network is presented with a new data point this variance will give rise to a variance in the network prediction. The model prediction on a new data point, $\tilde{\mathbf{x}}$, is $\tilde{y} = f(\tilde{\boldsymbol{x}}, \hat{\boldsymbol{w}}) + \tilde{e}$. Note that we refer to $\hat{y} = f(\tilde{\boldsymbol{x}}, \hat{\boldsymbol{w}})$ as the 'network prediction' and to $\tilde{y}$ as the 'model prediction'. This distinction acknowledges that the network is part of a statistical model. Because $\hat{\boldsymbol{w}}$ and $\tilde{e}$ are assumed independent, the variance of the model prediction is

$$Var(\tilde{y}) = Var(f(\tilde{\boldsymbol{x}}, \hat{\boldsymbol{w}})) + Var(\tilde{e})$$

The first term captures the variance due to network uncertainty which has its original source in the target noise on the training data set. We refer to this as the variance in the network prediction and it is identical to the 'delta method' error bars described in [8]. The second term is due to the inherent target noise at the predicted point. Together, the two terms constitute the variance in the model prediction which, from [7], can be expressed as

$$Var(\tilde{y}) = \sigma^2 + \tilde{\boldsymbol{g}}^T \boldsymbol{H}^{-1} \tilde{\boldsymbol{g}} \qquad (2)$$

where $\boldsymbol{H}$ is the outer product approximation to the Hessian matrix and is given by the inverse of equation 1. We note that the variance of the model prediction is identical to that derived under a Bayesian framework [1]. From here on, when we refer to the prediction variance or 'error bar' we mean the variance of the *model* prediction.

3 Committees

We now look at the case where two networks have been trained to solutions, $\hat{\boldsymbol{w}}_j$ and $\hat{\boldsymbol{w}}_k$. These may correspond to different local minima in the same network or to solutions in networks with different structures.

3.1 Prediction covariance

From the analysis in the previous section we can arrive at the equivalent of equation 2 for each of the two solutions

$$\begin{aligned} \Delta \hat{\boldsymbol{w}}_j &= (\boldsymbol{G}_j^T \boldsymbol{G}_j)^{-1} \boldsymbol{G}_j^T \boldsymbol{e} \\ \Delta \hat{\boldsymbol{w}}_k &= (\boldsymbol{G}_k^T \boldsymbol{G}_k)^{-1} \boldsymbol{G}_k^T \boldsymbol{e} \end{aligned}$$

These approximations arise from a first order Taylor series expansion of each solution about the *same* true solution, $\boldsymbol{w}$. If the different solutions are two different local minima in the same network then it is unlikely that such an approximation can be valid in both cases. For two networks with different structures, however, the approximation is more plausible. The covariance of the two solutions is given by

$$\begin{aligned} \boldsymbol{C}(j,k) &= Var(\Delta \hat{\boldsymbol{w}}_j, \Delta \hat{\boldsymbol{w}}_k) \\ &= (\boldsymbol{G}_j^T \boldsymbol{G}_j)^{-1} \boldsymbol{G}_j^T \sigma^2 \boldsymbol{I} \\ &\quad \left((\boldsymbol{G}_k^T \boldsymbol{G}_k)^{-1} \boldsymbol{G}_k^T \right)^T \end{aligned}$$

Re-arranging gives,

$$\begin{aligned} \boldsymbol{C}(j,k) &= \sigma^2 (\boldsymbol{G}_j^T \boldsymbol{G}_j)^{-1} \boldsymbol{G}_j^T \boldsymbol{G}_k \\ &\quad (\boldsymbol{G}_k^T \boldsymbol{G}_k)^{-1} \end{aligned} \qquad (3)$$

We refer to $\boldsymbol{C}$ as the cross-covariance matrix (we can also define a further matrix $\boldsymbol{H}(j,k) = \boldsymbol{C}^{-1}(j,k)$ as the 'cross-Hessian'). This matrix gives rise to correlations in predictions when the networks are presented with new data points. If $\hat{\boldsymbol{w}}_j = \hat{\boldsymbol{w}}_k$ then $\boldsymbol{G}_j = \boldsymbol{G}_k = \boldsymbol{G}$ and we recover the estimate of weight variance for a single network given in equation 1. In practice σ^2 may be unknown. In these cases we can substitute the estimate $\sigma^2 = \hat{\sigma}_j \hat{\sigma}_k$ where $\hat{\sigma}_i$ is the root mean squared training error of the ith network. The model predictions on a new data point $\tilde{\mathbf{x}}$ are

$$\begin{aligned} \tilde{y}_j &= f(\tilde{\boldsymbol{x}}, \hat{\boldsymbol{w}}_j) + \tilde{e} \\ \tilde{y}_k &= f(\tilde{\boldsymbol{x}}, \hat{\boldsymbol{w}}_k) + \tilde{e} \end{aligned}$$

Because $\tilde{e}$ is independent of $\hat{\boldsymbol{w}}_j$ and $\hat{\boldsymbol{w}}_k$, the covariance of the predictions is

$$\begin{aligned} \hat{\boldsymbol{\Omega}}_{jk} &= Var(f(\tilde{\boldsymbol{x}}, \hat{\boldsymbol{w}}_j), f(\tilde{\boldsymbol{x}}, \hat{\boldsymbol{w}}_k)) \\ & \quad + Var(\tilde{e}) \end{aligned}$$

Now, from a local linear approximation

$$\begin{aligned} f(\tilde{\boldsymbol{x}}, \hat{\boldsymbol{w}}_j) &= f(\tilde{\boldsymbol{x}}, \boldsymbol{w}) + \tilde{\boldsymbol{g}}_j^T(\tilde{\boldsymbol{x}}) \Delta \hat{\boldsymbol{w}}_j \\ f(\tilde{\boldsymbol{x}}, \hat{\boldsymbol{w}}_k) &= f(\tilde{\boldsymbol{x}}, \boldsymbol{w}) + \tilde{\boldsymbol{g}}_k^T(\tilde{\boldsymbol{x}}) \Delta \hat{\boldsymbol{w}}_k \end{aligned}$$

where $\boldsymbol{g}_j(\tilde{\boldsymbol{x}})$ is the vector of first derivatives of network j's output with respect to its weights evaluated for input vector $\tilde{\boldsymbol{x}}$. The covariance of these quantities is induced solely by the target noise in the training set. Hence,

$$\begin{aligned} & Var(f(\tilde{\boldsymbol{x}}, \hat{\boldsymbol{w}}_j), f(\tilde{\boldsymbol{x}}, \hat{\boldsymbol{w}}_k)) \\ &= Var(\tilde{\boldsymbol{g}}_j^T \Delta \hat{\boldsymbol{w}}_j, \tilde{\boldsymbol{g}}_k^T \Delta \hat{\boldsymbol{w}}_k) \\ &= \tilde{\boldsymbol{g}}_j^T(\tilde{\boldsymbol{x}}) \boldsymbol{C}(j,k) \tilde{\boldsymbol{g}}(\tilde{\boldsymbol{x}})_k \end{aligned}$$

The overall covariance of the predictions is therefore

$$\hat{\boldsymbol{\Omega}}_{jk}(\tilde{\boldsymbol{x}}) = \sigma^2 + \boldsymbol{g}_j^T(\tilde{\boldsymbol{x}}) \boldsymbol{C}(j,k) \boldsymbol{g}_k(\tilde{\boldsymbol{x}}) \quad (4)$$

where $\boldsymbol{C}(j,k)$ is the cross-covariance matrix defined in equation 3. When $\tilde{y}_j = \tilde{y}_k$ the formula reduces to the prediction variance of a single model given in equation 2. The above formula captures the covariance in model predictions due to the correlation in network solutions induced by the target noise. By normalising with respect to the variance of each model prediction (equation 2) the *correlation* between predictions can be estimated.

3.2 Optimal weighting of network predictions

The output of a weighted committee of networks can be written as $\hat{t}(\boldsymbol{x}) = \boldsymbol{\alpha}(\boldsymbol{x})^T \boldsymbol{f}(\boldsymbol{x})$ where $\boldsymbol{f}(\boldsymbol{x})$ is a vector of network outputs and $\boldsymbol{\alpha}(\boldsymbol{x})$ is an input-dependent vector of weights. The pointwise expected squared error between the committee prediction and the true value is

$$E = \varepsilon_D[(\hat{t}(\boldsymbol{x}) - t(\boldsymbol{x}))^2]$$

where ε_D denotes an expectation over the training data set, D. In this paper the $\boldsymbol{x}$-values in the training set are considered fixed, but the y-values are not. The expectation is therefore in effect taken just over the y-values, that is, over different realisations of the target noise. In what follows we drop the dependence on $\boldsymbol{x}$, the test point, to simplify the notation although this dependence is implicitly assumed. The pointwise expected prediction error can be decomposed into a variance term and a bias term [9] $E = V + B$, where V is the committee variance and B is the squared committee bias

$$\begin{aligned} V &= \boldsymbol{\alpha}^T \boldsymbol{\Omega} \boldsymbol{\alpha} \\ B &= (\boldsymbol{\alpha}^T \boldsymbol{m} - t)^2 \end{aligned}$$

where $\boldsymbol{m}$ is a vector of expected network outputs, $m_j = \varepsilon_D[f_j]$, and $\boldsymbol{\Omega}$ is a covariance matrix having elements $\boldsymbol{\Omega}_{jk} = \varepsilon_D[(f_j - m_j)(f_k - m_k)]$. If the weights are constrained to sum to one ($\sum_i \alpha_i = 1$), then the weight values that give rise to the minimum prediction error are [9]

$$\boldsymbol{\alpha} = \frac{(\boldsymbol{\Omega} + \boldsymbol{\Phi})^{-1} \boldsymbol{u}}{\boldsymbol{u}^T (\boldsymbol{\Omega} + \boldsymbol{\Phi})^{-1} \boldsymbol{u}} \quad (5)$$

where $\boldsymbol{u}$ is a vector of ones and $\boldsymbol{\Phi}$ is a bias matrix having elements

$$\boldsymbol{\Phi}_{jk} = \varepsilon_D[(m_j - t)(m_k - t)]$$

By plugging our estimate of $\boldsymbol{\Omega}$ from equation 4 into equation 5 we have an optimal method for weighting networks in a committee. We call this 'covariance-based' weighting. The networks may be assumed to be unbiased, in which case $\boldsymbol{\Phi}$ is the null matrix, or the bias may be estimated by the difference between each network output and the committee output. From here on, in this paper, we assume that the networks are unbiased. In previous work [9] $\boldsymbol{\Omega}$ has been approximated by a diagonal matrix resulting in a 'variance-based' weighting scheme. This is intuitively satisfying as each network prediction is weighted inversely according to the variance of that prediction.

A problem with the above methods is that the weights can take on large negative and positive values giving rise to committee predictions which are beyond the range of predictions from individual networks [2]. To overcome this we can impose the additional non-negativity constraint, $\alpha_i > 0$. This can be implemented by expressing the weighting factors as softmax functions of hidden variables which are chosen so as to minimise the expected pointwise prediction error. This is the approach taken in this paper. It is, however, a nonlinear optimization problem requiring iterative solution. An alternative approach is to use a non-iterative regularization technique [3].

4 Results

The potential benefit of the covariance-based weighting scheme can be seen by looking at the effect of correlations between predictors in the ensemble. This effect is shown in Figure 1 for a committee composed of just two networks. The stronger the positive correlation between networks, the stronger the weight on just one of the networks; the one with the smallest variance. This is shown in Figure 1(a) where the low-variance network (network 1) is assigned a weight of 1 if the correlation is high. This makes sense because knowing the prediction from the second (high variance) network gives you an increasingly smaller amount of extra information. Figure 1(b) shows the committee variance for each of the three weighting schemes; covariance-based, variance-based and uniform. The covariance-based weighting scheme results in a committee with the lowest variance. The effect is greater as the correlation between predictors increases. We can summarise the effect of correlation by saying that the covariance-based weighting scheme will be beneficial if the variance of predictors is uneven and if the correlation is high.

Figures 2a and b show the output of Multi-Layer Perceptrons (MLPs) with 2 and 4 hidden units that have been trained on samples of a noisy sine wave function. The error bars, which were calculated according to equation 2, are also shown. The 4-unit network has smaller error bars in regions of high data density and the 2-unit network has smaller error in areas of low data density. These input-dependent error bars are used in the variance-based weighting scheme to find an optimal weighting of networks. The covariance-based scheme also uses these error bars but, additionally, also takes into account the correlation between predictors. Figure 2c shows this correlation as estimated by a normalised version of Equation 4. In regions of high data density the model predictions are positively correlated but are almost entirely uncorrelated in regions of low data density. Thus, in regions of low data density the pointwise covariance matrix becomes diagonal and the covariance-based scheme reduces to the variance-based scheme. Figure 2d shows the weight coefficients derived from the variance-based and the covariance-based weighting schemes. In regions of low data density both schemes do indeed give rise to similar weighting factors. In regions of high data density, the covariance-based weighting scheme relies more heavily on the predictions of the 4-hidden unit MLP; the more accurate network.

The methods were also tested on the robot arm data set [6] which describes the mapping for a two-joint robot arm. There are two input variables and one target. The data was grouped into two regions such that at half of the test points the networks are interpolating and at the other half they are extrapolating. Two four hidden unit MLPs were then trained to Normalised Mean Square Errors (NMSE) of 0.05 and 0.2, representing one good fit to the data and one bad fit. In the interpolation region the average correlation of predictions was 0.963. In the extrapolation region this reduced to 0.669. Thus, the correlation measure is qualitatively correct. Table 1 shows that the covariance-based committee weighting scheme results in a committee with a lower prediction variance, especially in the interpolation region. The actual prediction errors were 0.318 for uniform, 0.236 for variance and 0.222 for covariance. The difference between prediction variances and actual errors may be attributed to network bias.

5 Summary

This paper has introduced a method for calculating the covariance between different network solutions. It is based on a generalisation of the delta method for calculating the network Hessian and generates what we call the 'cross-covariance' matrix. Its inverse is the 'cross-Hessian'. Using this matrix we are able to calculate the point-wise covariance between model predictions. By normalising with respect to the prediction variance of each model we can estimate the correlation between predictors at each point in input space. Importantly, these estimates can be carried out with the use of training data alone. Application of the method to the problem of finding optimal linear combinations of models results in a 'covariance-based' weighted committee, where the weights are input-dependent. If the individual networks are unbiased then the covariance-based weighted committee is optimal in the sense of minimum expected prediction error. If the networks are biased, the committee is optimal in the sense of minimum prediction variance. In practice, the covariance-based scheme shows an improvement over a simpler 'variance-based' approach in areas of input space where the predictions are correlated (ie. in regions of high or intermediate training data density) and the networks have different prediction variances. This suggests using a range of networks with different complexities or networks trained to fit the data with a range of accuracies. The covariance-based scheme does, however, require more computation and storage than the variance-based scheme as the cross-covariance matrices must be calculated and stored.

Acknowledgements

William Penny is a research fellow funded by the UK EPSRC (GR/K79062), whose support we gratefully acknowledge. Dirk Husmeier is a research fellow funded by the Jefferiss Research Trust.

References

[1] C. M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, Oxford, 1995.

[2] L. Breiman. Stacked regression. Technical report, Department of Statistics, University of California at Berkeley, 1993.

Table 1: *Committee prediction variance on Robot Arm Data.*

Weighting	Interp	Extrap	Overall
Uniform	0.118	0.341	0.229
Variance	0.062	0.099	0.080
Covariance	0.043	0.081	0.062

[3] G. H. Golub and C. F. Van Loan. *Matrix Computations.* John Hopkins University Press, 3rd edition, 1996.

[4] A.S. Krogh and J. Vedelsby. Neural network ensembles, cross validation and active learning. In *Advances in Neural Information Processing Systems 7, edited by G.Tesauro, D.S.Touretzky and T.K. Leen.* MIT Press, 1995.

[5] M. LeBlanc and R. Tibshirani. Combining estimates in regression and classification. Technical Report, Department of Preventive Medicine and Biostatistics, University of Toronto, 1993.

[6] D.J.C. Mackay. A practical Bayesian framework for backprop networks. *Neural computation*, 4(3):448–472, 1992.

[7] G.A.F. Seber and C.J. Wild. *Nonlinear regression.* New York: John Wiley, 1989.

[8] R. Tibshirani. A comparison of some error estimates for neural network models. *Neural Computation*, 8:152–163, 1996.

[9] V. Tresp and M. Taniguchi. Combining estimators using non-constant weighting functions. In *Advances in Neural Information Processing Systems 7, edited by G.Tesauro, D.S.Touretzky and T.K. Leen.* MIT Press, 1995.

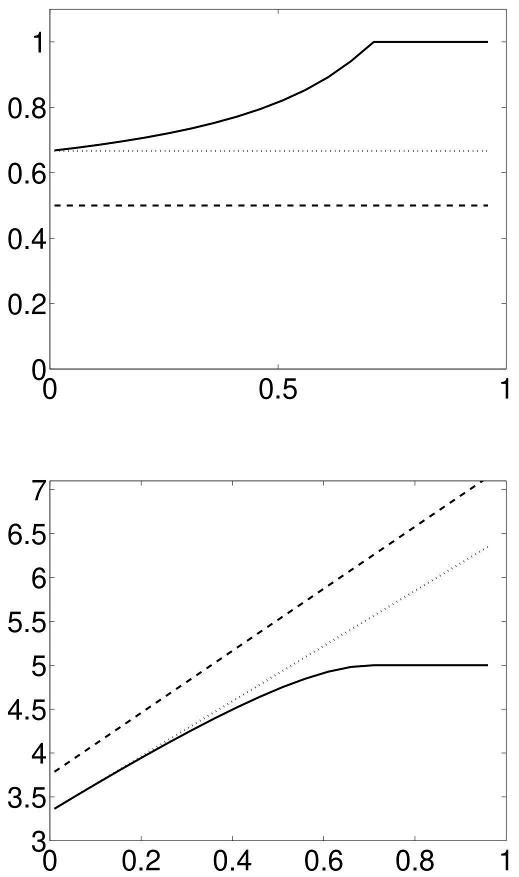

Figure 1: **Effect of correlation between predictors** *(a) weighting of first predictor versus correlation, (b) committee variance versus correlation. The first predictor in the committee has a variance of 5 and the second predictor has a variance of 10. The solid lines are for covariance-based weight values (from equation 5), dotted lines for variance-based weight values (from equation 5 using a diagonal* $\mathbf{\Omega}$*), and dashed lines for uniform weights. All three lines converge if the variances of the predictors are equal. The stronger the correlation between predictors the greater the benefit of using the covariance-based weighting scheme.*

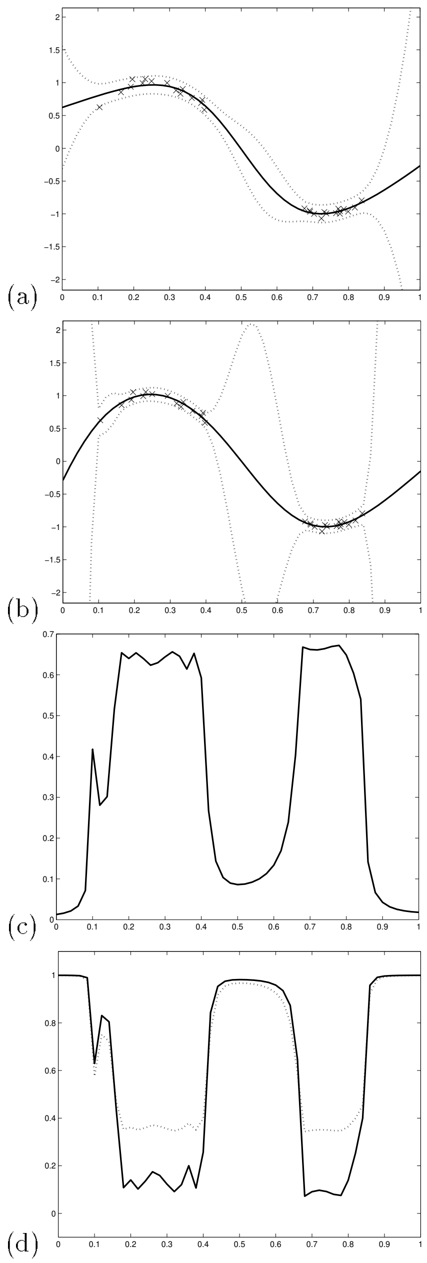

Figure 2: **Two networks trained on the sinewave data** *(a) MLP with two hidden units, (b) MLP with four hidden units. The crosses are training data points, the solid line is the prediction and the dotted lines are error bars at plus and minus 2 standard deviations. (c) correlation between model predictions as approximated by a normalised version of equation 4, (d) committee weighting factors for the 2-unit MLP from equation 5 and a non-negativity constraint; covariance-based (solid line) and variance-based (dotted line).*