Weight Averaging for Neural Networks and Local Resampling Schemes

Joachim Utans
London Business School
Sussex Place, Regent's Park
London NW1 4SA, UK
J.Utans@lbs.lon.ac.uk

Abstract

Recently methods for combining estimators have been popular and enjoyed considerable success. Typically, one obtains several models by considering different model architectures (e.g. linear or nonlinear) or uses different partitions of the data (overlapping or non-overlapping) for estimating the model. In a final step the outputs of each model are combined, either by assigning fixed weights *a priori* (e.g. equal weighting) or be estimating these weights from data as well. Here we propose a strategy similar to Breiman's (1994) "bagging": we use a resampling scheme (nonlinear Cross-Validation in this case but the idea is applicable to other local resampling schemes (bootstrap, jackknife)) to estimate the expected out-of-sample error as a model selection criterion in the context of neural network models. We want to make further use of the models estimated via resampling by combining them and propose that under certain conditions one can average model parameters directly instead of retaining all networks and combine their outputs. We demonstrate that for the examples considered the network performance is comparable to performance obtained by averaging outputs while offering improved recall speed and savings in storage requirements.

Introduction

The methods of combining estimators to improve their performance has a long history in statistics and econometrics (Granger, 1989). Recent years have seen increased interest in the technique in the Neural Network community. For certain classes of estimators a combined estimator is on average as good or better than the individual components (whereas for example nearest neighbor methods or discriminant analysis are not improved (Breiman, 1994)). Perrone (1993) has studied the linearly weighted combination of neural network models. Recent developments in the statistics community include Bootstrap Aggregation (Bagging, (Breiman, 1994)), Stacked Regression (Breiman, 1992) and Stacked Generalization (Wolpert, 1990). The weighting of estimators can simply be constant or inversely proportional to the uncertainty of each components estimate (Perrone, 1993). Tresp and Taniguchi (1995) have proposed choosing the weighting based on the input density, blurring the distinction between averaging models and the Mixture of Experts approach of Jacobs, Jordan, Nowlan and Hinton (1991). In the context of classification several "voting" schemes have been proposed where each model individually produces a classification and these are then combined, for example by choosing the decision of the majority (Battiti and Colla, 1994).

Combining estimators works trough reducing the variance of the combined output relative to the output of the individual networks while not increasing the bias (assuming that the component estimators are unbiased or have the same bias). With flexible models such as neural networks, high variance due to overfitting is preventing good generalization of the models. Model selection thus is faced with finding a balance between reducing variance and possibly increasing bias to find a model that generalizes optimally given the available (noisy) data (the Bias/Variance Dilemma (Geman, Bienenstock and Doursat, 1992)). Therefore the properties of the techniques for combining models are very attractive for neural networks models.

The cost of using a combination of models is that each of the constituent models must be estimated and then stored and its output must be calculated during the recall phase. The design procedure for neural networks typically includes the estimation of several models. In particular, when resampling methods are used to asses the generalization ability of the model in terms if its expected performance on out-of-sample data, a number of models are estimated an different partitionings or samples of the training data. Since these models have been estimated already the only additional overhead results from finding appropriate combination weights. For some applications the storage and computational requirement for recall can reduce the applicability of the methods (for example, in the financial industry the dominant computational tool is a spreadsheet running on a desktop computer).

Here we propose a scheme were, under certain conditions, the parameters of the constituent neural networks can be combined instead of their outputs. When only one neural network must be implement a significant speed improvement can be achieved. The methods makes use of the properties of Nonlinear Cross-Validation (Moody and Utans, 1994; Moody and Utans, 1992) which will be described in the next section.

Neural Network Model Selection via Nonlinear Cross-validation

For the presentation of nonlinear Cross-validation we follow (Moody and Utans, 1994) were a detailed description can be found. Consider a set of observations $D = \{(\vec{x}_j, t_j); j = 1 \ldots N\}$ and assume that the data-generating model is given as $y_j = \mu(x_j) + \epsilon_j$ where $\mu(x)$ is an unknown function, the inputs x_j are drawn independently with an unknown stationary probability density function $p(x)$, the ϵ_j are independent random variables with zero mean ($\bar{\epsilon} = 0$) and variance σ_ϵ^2, and the y_j are the observed target values. The task is to find an estimator $\hat{\mu}(\mathbf{x}, \mathbf{w}; D)$, realized in form of a feed-forward neural network.

For small size, noisy data-sets (as are common for example in finance and economics) the bias-variance dilemma refers to the trade-off between using small models with a large exemplars/parameter ratio possibly resulting in a biased model or choosing a larger model where the parameter estimates have high variance.

We have proposed a extension to standard resampling methods for nonlinear models (Moody and Utans, 1994; Utans, Moody, Rehfuss and Siegelmann, 1995) and presented applications of the nonlinear cross-validation algorithm for model selection for neural networks. To this end we construct a family of neural networks models of increasing complexity by using more input variables or adding more hidden units. The task is to select the model that has the smallest expected out-of-sample error.

With nonlinear models local minima can occur. We regard each such local minimum as a different model (although these models can all be *realized* using the same neural network architecture i.e. same number of inputs and number of hidden units, their weights differ). We thus are interested in analyzing the generalization properties of one such model corresponding a particular local minimum. The straightforward extension to standard resampling schemes is to choose as the initial weights a weight-vector close to the local solution. In contrast to methods that choose a new random starting point this procedure results in *perturbing* the local solution by resampling of the training data.

We briefly outline the method for NCV below for the mean squared error (MSE) loss function (a more detailed description of NCV can be found in (Moody and Utans, 1994), see also (Stone, 1978) and (Eubank, 1988)); the extension to other resampling schemes is straightforward. Let the data D be divided into v randomly selected disjoint subsets D_j of roughly equal size: $\cup_{j=1}^{v} D_j = D$ and $\forall i \neq j,\ D_i \cap D_j = \emptyset$. Let N_j denote the number of observations in subset D_j. Let $\hat{\mu}_{\lambda(D_j)}(x)$ be an estimator trained on all data except for $(x,t) \in D_j$. Then, the cross-validation average squared error for subset j is defined as

$$CV_{D_j}(\lambda) = \frac{1}{N_j} \sum_{(x_k, y_k) \in D_j} \left(y_k - \hat{\mu}_{\lambda(D_j)}(x_k) \right)^2 \quad . \qquad (1)$$

These are averaged over j to obtain the v-fold cross-validation estimate of prediction risk:

$$CV(\lambda) = \frac{1}{v} \sum_j CV_{D_j}(\lambda) \quad . \qquad (2)$$

Typical choices for v are 5 and 10. Leave–one–out CV is obtained in the limit $v = N$.

To obtain the starting point, a network is trained on the entire data set D to obtain a model $\hat{\mu}_\lambda(x)$ with weights $\mathbf{w}_0$. Each subset D_j is removed from the training data in turn. The network is re-trained using the remaining data starting at $\mathbf{w}_0$ (rather than using random initial weights). The Cross-Validation error computed as the average error for the "perturbed models" $\hat{\mu}_{\lambda(D_j)}(x)$ thus estimates the prediction risk for the model with locally-optimal weights $\mathbf{w}_0$ as desired, and not the performance of other predictors at other local minima.

However, the methods relies on good convergence properties of the optimization algorithm used for training. Cross-Validation relies on "forced over-fitting" of the model to different partitions of the data. Stopping training early, either deliberately or because of poor convergence properties of the minimization algorithm, has been shown to be equivalent to a form of regularization that biases towards the initial weights (Sjöberg and Ljung, 1992). This will lead NCV to underestimate the model variance.

Also, for the purpose of combining estimators one wants the constituent estimators as uncorrelated as possible (Bunn, 1989; Sollich and Krogh, 1995); in the case of NCV this requires local convergence to each $\mathbf{w}_i$.

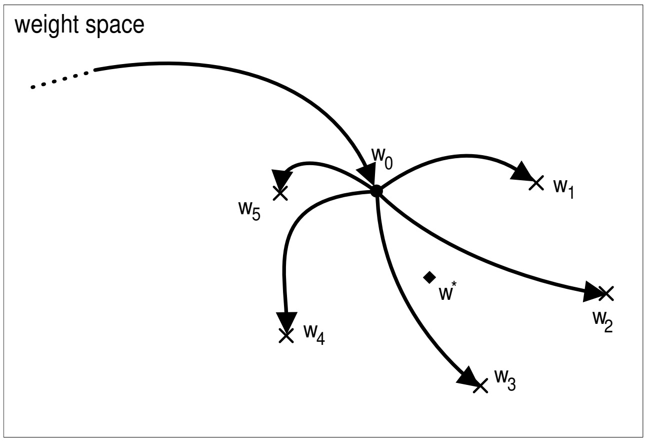

Figure 1: Illustration of the computation of 5–fold *nonlinear cross-validation (NCV)*. First, the network is trained on all data to obtain weights $\mathbf{w}_0$ which are used as starting point for the cross-validation. Each data subset $D_i, i = 1 \ldots 5$ is removed from the training data D in turn. The network is trained, starting at $\mathbf{w}_0$, using the remaining data. This *perturbs* the weights to obtain $\mathbf{w}_i$. The test error of the "perturbed model" w_i is computed on the hold-out sample D_i. The average of these errors is the 5-fold CV estimate of the prediction risk for the model with weights close to w_0.

Combining Networks by Averaging Weights

Having selected the best model via NCV one is left with a set of neural networks, each being trained on a different but overlapping partition of the training data. The primary objective for computing NCV is to estimate the generalization performance of a neural network model. Once the networks have been trained on different partitions of the data one would like to "reuse" these models by combining them rather than to choose a particular network as the final one (this reasoning moves in the opposite direction from Wolpert and Macready (1996) where the generalization error of the bagging estimator is estimated by examining the bootstrap errors of it's components).

Here we consider averaging the weights of the network directly instead of combining their outputs. For a set linear model trained on different partitions of a dataset the combined estimator $\hat{\mu}_A(\mathbf{x}, \mathbf{w})$ for the simplest case of weighted averaging is of the form

$$\begin{aligned} \hat{\mu}_A(\mathbf{x}, \mathbf{w}) &= \sum_i a_i \hat{\mu}_i(\mathbf{x}, \mathbf{w}) \qquad (3) \\ &= \sum_i a_i \sum_j w_j^i x_j \end{aligned}$$

where i indexes the individual models and a_i are weights. By rearranging the sum this can viewed as averaging parameters w_j^i

For neural network models averaging parameters normally is problematic because 1) starting from random initial weights a different local minimum could be found and 2) the same local solution can be represented by a permutation of the hidden units and their associated weights. Because a local resampling method is designed to explore weight space around a particular local minimum it is possible to directly compare weights from different networks.

White (1989) has shown that under certain conditions the limiting distribution of the weights $\hat{\mathbf{w}}_i$ for a well-specified model around the weights $\mathbf{w}^*$ of the best model (with minimum Kullback-Leibler divergence from $\mu(\mathrm{x})$) is a multivariate Gaussian distribution

$$(\sqrt{N}(\hat{\mathbf{w}} - \mathbf{w}^*) \rightarrow N(0, \mathbf{C}^*) \qquad (4)$$

with mean 0 and asymptotic covariance matrix $\mathbf{C}^* = \mathbf{A}^{*-1}\mathbf{B}^*\mathbf{A}^{*-1}$ with

$$\mathbf{A}^* = E\{\nabla_{\mathbf{W}} \nabla_{\mathbf{W}} MSE(\mathbf{w})\}$$

and

$$\mathbf{B}^* = E\{\nabla_{\mathbf{W}} MSE(\mathbf{w})\ \nabla_{\mathbf{W}} MSE(\mathbf{w})^T\}$$

It is assumed that only one local minimum exists (for a detailed account see (White, 1989; White, 1982), an empirical analysis of the weight distribution of neural network models can for example be found in (Zapranis, Utans and Refenes, 1996)).

This result, combined with the properties of NCV, allow us to approximate the sample mean $\mathbf{w}^*$ by averaging the weights of the best model found be NCV as

$$\hat{\mathbf{w}}^* \approx \frac{1}{v}\sum_{i=1}^{v}\hat{\mathbf{w}}^i \tag{5}$$

reducing the variance of the estimated weights. This results in a single model and thus offers a speed advantage for the recall phase because one does not have to compute the output for several models and average, as well a savings in storage requirements because only the weights of one neural network must be stored. The covariance can be used to calculate error-bars for the weights and weight their contribution to the sum similar to the form used for combining outputs (Perrone, 1993).

However, note that NCV does not guarantee that at least for some of the data-partitions the weights converge to a different local minimum. In fact, we initially examined the weight distribution to detect just these cases (i.e. by using a robust estimator for the mean and deviation, for example MAD (mean absolute deviation)). If networks converge to different local minima averaging their weights does not approximate the sample mean as intended. It is generally necessary to check for convergence of the individual networks.

In the context of NCV the simplest, most straightforward diagnostic is to examine the variations in the errors for the individual subsets, using a robust estimator for the mean and standard deviation, and discard networks from the sum that deviate from the mean by a large amount. For the simulation reported here this strategy was employ. Using the median and median deviation as a robust estimator networks with $CV_{D_j} > 3MAD$ were excluded from the sum. Similarly one can examine the deviation from the mean for each weight.

Another way to detect whether the networks converged to different minima is to perform a Principle Component Analysis (PCA) on their outputs. It is reasonable to expect that if all networks correspond to the same local minimum the first (dominant) principle component will account for most of the variance. On the other hand, with two or more local minima the first two (or first few) components will have a large contribution.

Simulations

We have applied the methods to a standard test case, the prediction of the sunspot count, and a macroeconomic forecasting problem, the prediction of the US Index of Industrial production. We compared weight averaging with averaging the outputs of the models obtained from NCV.

For the sunspot-count series the best network selected by NCV had 4 inputs and 3 hidden units. The data-set contains observations for years 1700–1988 and was was normalized to zero mean and unit standard deviation, data form years 1700–1920 was used for training, years 1921–1955 constitute the test set. We compare the averaging of the outputs of all networks from NCV to averaging the weight vector directly. Equal combination weight were given to each network in each case.

The data for the Index of Industrial Production consists of monthly observations of IP and other macroeconomic and financial series for the period from January 1950 to December 1989. Both the "unfiltered" series and various "filtered" versions are considered for inclusion in the model, for a total of 48 possible input variables. The target series and all 48 candidate input series are normalized to zero mean and unit standard deviation. Following prior work by Moody, Levin and Rehfuss (1993) and Utans *et al.* (1995) we chose a neural network with 3 hidden units, NCV selected a subset of 13 input variables.

The results show that weight averaging in these two cases performs slightly better for the optimal architecture selected by NCV as shown in Table 1, but the difference is likely not significant.

Figure 2 shows a comparison between output and weight averaging for all networks considered for the input selection for the sunspot data. Both averaging schemes perform similarly with both showing much lower variation than the test error for the network with weights $\mathbf{w}_0$. The variable selection using NCV (i.e. by resampling from the training set) indicated that four inputs give best performance whereas the test-error of the combined networks decreases further when 5 inputs are included, indicating that model variance is decreased by averaging (which is not taken into account for the NCV analysis). The five input variables are lags 1, 2, 3, 8, 11 of the series, the network with four input variables omits lag 11.

Weight averaging appears to offers results comparable to the widely practiced method of averaging network outputs given that the underlying assumptions are satisfied. However, only one network has to be retained instead of all networks computed for NCV (10 for the examples shown) reducing the storage needs and the computational requirements for recall.

Summary and Conclusion

We have proposed a methods for combining neural networks by averaging their parameters instead of combining outputs that is based on knowledge of the theoretical asymptotic weight distribution. Initial simulations

Normalized Test Set Mean Square Error		
	Averaging Outputs	Averaging Weights
Sunspot count	0.0888	0.0879
Index of Industrial production	0.3902	0.3900

Table 1: Results for the best model selected using NCV for the sunspot and IP data comparing averaging network outputs to averaging their weights.

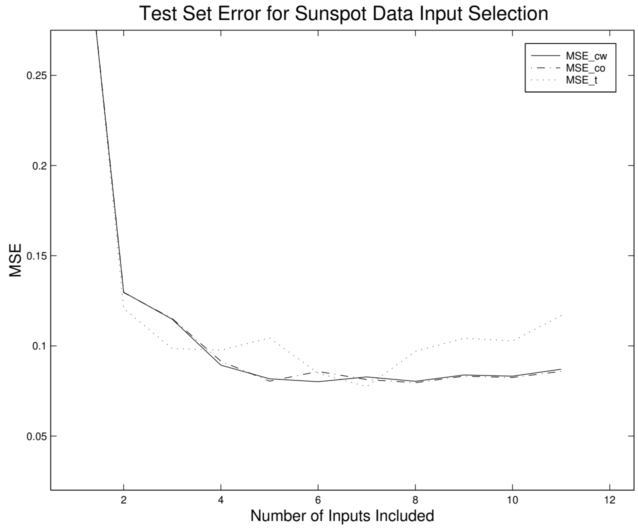

Figure 2: Comparison of output averaging (MSE_{co}) vs. weight averaging (MSE_{cw}) for all candidate networks architectures considered for the selection of inputs for the sunspot data. Both averaging schemes perform similarly with both showing much lower variation than the test error for the network with weights $\mathbf{w}_0$ (MSE_t). The variable selection using NCV (i.e. by resampling from the training set) indicated that four inputs give best performance whereas the test-error of the combined networks decreases further when 5 inputs are included, indicating that model variance is decreased by averaging (which is not taken into account for the NCV analysis). The five input variables are lags 1, 2, 3, 8, 11 of the series.

show that performance is comparable to the widely practiced method of averaging network outputs when the underlying assumptions are satisfied.

The scheme is limited to situations where all networks share the same architecture with model variance arising from training the networks using different subsets of the training data and where corresponding weights in the networks can be identified. As such the methods is considerably less flexible then methods that combine the outputs of neural networks (or a mix of estimators). On the other hand, when the underlying assumptions are satisfied one can reduce storage needs and computational requirements during the recall phase because only one network must be stored and evaluated.

Non-linear cross-validation is a resampling methods witch produces networks that satisfy these requirements and thus is well matched to the proposed method.

Acknowledgements

Initial work leading to this project was performed while the author was with the Oregon Graduate Institute where support as provided in part under grant ONR N00014-92-J-4062 and N00014-94-1-0071. Helpful discussions with J. Moody and A. Zapranis are gratefully acknowledged.

References

Battiti, R. and Colla, A. M. (1994), 'Democracy in neural nets: Voting schemes for classification', *Neural Networks* **7**(4), 691–707.

Breiman, L. (1992), Stacked regression, Technical Report 367, Dept. of Statistics, University of California, Berkeley, CA.

Breiman, L. (1994), Bagging predictors, Technical Report 421, Dept. of Statistics, University of California, Berkeley, CA.

Bunn, D. (1989), 'Forecasting with more than one model', *Journal of Forecasting* **8**, 161–166.

Eubank, R. L. (1988), *Spline Smoothing and Nonparametric Regression*, Marcel Dekker, Inc.

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

Granger, C. W. J. (1989), 'Combining forecasts – twenty years later', *Journal of Forecasting* **8**, 167–173.

Jacobs, R. A., Jordan, M. I., Nowlan, S. J. and Hinton, G. E. (1991), 'Adaptive mixtures of local experts', *Neural Computation* **3**, 79–87.

Moody, J. E. and Utans, J. (1992), Principled architecture selection for neural networks: Application to corporate bond rating prediction, *in* J. E. Moody, S. J. Hanson and R. P. Lippmann, eds, 'Advances in Neural Information Processing Systems 4', Morgan Kaufmann Publishers, San Mateo, CA, pp. 683–690.

Moody, J. and Utans, J. (1994), Architecture selection strategies for neural networks: Application to corporate bond rating prediction, *in* A. N. Refenes, ed., 'Neural Networks in the Captial Markets', John Wiley & Sons.

Moody, J., Levin, A. and Rehfuss, S. (1993), 'Predicting the U.S. index of industrial production', *Neural Network World* **3**(6), 791–794. special issue: *Proceedings of Parallel Applications in Statistics and Economics '93*, M. Novak (ed).

Perrone, M. P. (1993), Improving Regresssion Estimation: Averaging Methods for Variance Reduction with Extensions to General Convex Measure Optimization, PhD thesis, Dept. of Physics, Brown University.

Sjöberg, J. and Ljung, L. (1992), Overtraining, regularization, and searching for minimum in neural networks, Technical report, Dept. of Electrical Engineering, Linköping University, Linköping, Sweden.

Sollich, P. and Krogh, A. (1995), Learning with ensembles: How over-fitting can be usefull, *in* G. Tesauro, D. S. Touretzky and T. Leen, eds, 'Advances in Neural Information Processing Systems 7'.

Stone, M. (1978), 'Cross–validation: A review', *Math. Operationsforsch. Statist., Ser. Statistics* **9**(1).

Tresp, V. and Taniguchi, M. (1995), Combining estimators using non-constant weighting functions, *in* G. Tesauro, D. S. Touretzky and T. Leen, eds, 'Advances in Neural Information Processing Systems 7'.

Utans, J., Moody, J., Rehfuss, S. and Siegelmann, H. (1995), Input variable selection for neural networks: Application to predicting the U.S. business cycle, *in* 'Proceedings of the IEEE/IAFE conference on *Computational Intelligence for Financial Engineering (CIFEr)*', New York, NY.

White, H. (1982), 'Maximum likelihood estimation of misspecified models', *Econometrica* **50**(1), 1–25.

White, H. (1989), 'Learning in artificial neural networks: A statistical perspective', *Neural Computation* **1**, 425–464.

Wolpert, D. H. (1990), Stacked generalization, Technical Report LA-UR-90-3460, Complex Systems Group, Los Alamos, NM.

Wolpert, D. H. and Macready, W. G. (1996), An efficient method to estimate bagging's generalization error, Technical Report SFI-TR-96-03, Santa Fe Institute, 1399 Hyde Park Road, Santa Fe, NM 87501.

Zapranis, A., Utans, J. and Refenes, A. (1996), 'Specification tests for neural networks: A case study in tactical asset allocation'. In preparation.