EARLY STOP CRITERION FROM THE BOOTSTRAP ENSEMBLE

Lars Kai Hansen, Jan Larsen & Torben Fog

CONNECT, Department of Mathematical Modelling, Build. 321
Technical University of Denmark,
DK-2800 Lyngby, Denmark
emails: lkhansen,jl,tf@imm.dtu.dk

ABSTRACT

This paper addresses the problem of generalization error estimation in neural networks. A new early stop criterion based on a Bootstrap estimate of the generlization error is suggested. The estimate does not require the network to be trained to the minimum of the cost function, as required by other methods based on asymptotic theory. Moreover, in constrast to methods based on cross-validation which require data left out for testing, and thus biasing the estimate, the Bootstrap technique does not have this disadvantage. The potential of the suggested technique is demonstrated on various time-series problems.

1. INTRODUCTION

The goal of neural network learning in signal processing is to identify robust functional dependencies between input and output data (for an introduction see e.g., [3]). Such learning usually proceeds from a finite random sample of training data; hence, the functions implemented by neural networks are stochastic depending on the particular available training set. This opens the question of how robust the learned functions are to fluctuation and noise in the training set, and how well will they perform on new test data. Generalization is a key topic in the theory of supervised learning, and significant progress has been reported. The most universally valid results are due to Murata *et al.* [5], describing the asymptotic generalization ability of algorithms that are continuously parameterized. However, these generalization error estimators assume that the networks are trained to the minimum of the training error and that the training set is large compared to the number of degrees of freedom in the model.

The overtraining phenomenon is well documented in the neural network literature. When training a network with too many resources the network will initially learn the typical, generic aspects of the problem and then as training continues it will adapt to increasingly finer details of the training data. If the average error on an independent example (the generalization error) is monitored one typically experiences an initial decrease of error followed by an increase when the network weights are getting too specialized. The before mentioned asymptotic theories cannot cope with this phenomenon, since they are based on assumption that the network weights are trained to the minimal training error and that this minimum is close to the "true" weights. Current practice prescribes the use of a validation set. However, this means that data has to be set aside for testing, similarly the standard cross-validation technique is based on resampling without replacement, hence, the statistics obtained are biased.

In this contribution we use a Bootstrap resampling plan to estimate test errors. Bootstrap involves training an ensemble of networks on training sets resampled from the original training set. In contrast to cross-validation, Bootstrap uses sampling *with replacement*. Consequently, Bootstrap can approach the statistics of the learning problem at the full sample size available. Another complicated test error estimator based on mixing the concepts of cross-validation and bootstrap has been suggested in [4]. This estimator aims at estimating the test error of a bootstrap ensemle of networks rather than the test error of the individual network.

2. BOOTSTRAP BASED TEST ERROR ESTIMATE

Let the data set consist of N realted input-output examples: $\mathcal{D} = \{(\boldsymbol{x}_k, \boldsymbol{y}_k)\}_{k=1}^{N}$ where $\boldsymbol{x}, \boldsymbol{y}$ are the input and ouput vectors, respectively. Define the cost for neural network training by $\epsilon(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{w})$, with $\boldsymbol{w}$ denoting the vector of network weights. In the specific case of squared error for a network implementing the scalar function $f(\boldsymbol{x}, \boldsymbol{w})$ we have, $\epsilon(\boldsymbol{x}, y, \boldsymbol{w}) = (y - f(\boldsymbol{x}, \boldsymbol{w}))^2$. The network is trained by an iterative scheme such as gradient descent or a Newton based scheme as to minimize the cost function (or training error)

$$S_{\mathcal{D}}(\boldsymbol{w}) = \frac{1}{N} \sum_{k=1}^{N} \epsilon(\boldsymbol{x}_k, \boldsymbol{y}_k, \boldsymbol{w}). \quad (1)$$

Next we apply the Bootstrap technique, see e.g., [2], [6]. Consider Q resamples of the data set *with replacement*, $\mathcal{D}_q$, $q = 1, \cdots, Q$. Each set consists of N input-output examples drawn independently from $\mathcal{D}$ with probability $1/N$. Define $\widehat{\boldsymbol{w}}_q$ as the weight estimates obtained when training on the sets $\mathcal{D}_q$ and further define the indicator variable

$$\delta_{kq} = \begin{cases} 1 & , k \notin \mathcal{D}_q \\ 0 & , k \in \mathcal{D}_q \end{cases} \quad (2)$$

This research is supported by the Danish Natural Science and Technical Research Councils through the Danish Computational Neural Network Center. JL furthermore acknowledge the Radio Parts Foundation for financial support.

where $k = 1, 2, \cdots, N$ refers to the examples in the data set $\mathcal{D}$. The training error on $\mathcal{D}$ evaluated at $\widehat{\boldsymbol{w}}_q$ can be expressed as:

$$\frac{1}{N}\sum_{k=1}^{N} \epsilon(\boldsymbol{x}_k, \boldsymbol{y}_k, \widehat{\boldsymbol{w}}_q) = \frac{1}{N}\sum_{k=1}^{N}(1-\delta_{kq}) \cdot \epsilon(\boldsymbol{x}_k, \boldsymbol{y}_k, \widehat{\boldsymbol{w}}_q) + \frac{1}{N}\sum_{k=1}^{N} \delta_{kq} \cdot \epsilon(\boldsymbol{x}_k, \boldsymbol{y}_k, \widehat{\boldsymbol{w}}_q) \quad (3)$$

Consider a fixed q corresponding to a particular sample of training data $\mathcal{D}_q$. It is now possible to interpret the two terms on the right hand side of Eq. (3) as training and test error, respectively. That is, by performing an ensemble average over all possible data sets $\mathcal{D}$ of size N in Eq. (3) we get:

$$\frac{1}{N}\sum_{k=1}^{N} E_{\mathcal{D}}\{\epsilon(\boldsymbol{x}_k, \boldsymbol{y}_k, \widehat{\boldsymbol{w}}_q)\} = \frac{1}{N}\sum_{k=1}^{N}(1-\delta_{kq}) \cdot E_{\mathcal{D}}\{\epsilon(\boldsymbol{x}_k, \boldsymbol{y}_k, \widehat{\boldsymbol{w}}_q)\} + \frac{1}{N}\sum_{k=1}^{N} \delta_{kq} \cdot E_{\mathcal{D}}\{\epsilon(\boldsymbol{x}_k, \boldsymbol{y}_k, \widehat{\boldsymbol{w}}_q)\} \quad (4)$$

where the data set ensemble average is denoted by $E_{\mathcal{D}}\{\cdot\}$. Assume that the examples are drawn independently. Eq. (4) is further reduced by introducing three quantities:

1. The *training error*

$$S_{\mathcal{D}}(\widehat{\boldsymbol{w}}_q) = \frac{1}{N}\sum_{k=1}^{N} \epsilon(\boldsymbol{x}_k, \boldsymbol{y}_k, \widehat{\boldsymbol{w}}_q). \quad (5)$$

2. The *individual training error*

$$S_{\mathcal{D}_q}(\widehat{\boldsymbol{w}}_q) = \frac{1}{N}\sum_{k \in \mathcal{D}_q} (1-\delta_{kq}) \cdot \epsilon(\boldsymbol{x}_k, \boldsymbol{y}_k, \widehat{\boldsymbol{w}}_q). \quad (6)$$

3. The *average generalization error*

$$\Gamma = E_{\mathcal{D}}\{G(\widehat{\boldsymbol{w}}_q))\} \quad (7)$$

   where the generalization error is defined as

$$G(\widehat{\boldsymbol{w}}_q) = E_{\boldsymbol{x},\boldsymbol{y}}\{\epsilon(\boldsymbol{x}, \boldsymbol{y}, \widehat{\boldsymbol{w}}_q)\} = \int \epsilon(\boldsymbol{x}, \boldsymbol{y}, \widehat{\boldsymbol{w}}_q) \cdot p(\boldsymbol{x}, \boldsymbol{y})\, d\boldsymbol{x} d\boldsymbol{y} \quad (8)$$

   with $p(\boldsymbol{x}, \boldsymbol{y})$ denoting the joint (unknown) probability density of $(\boldsymbol{x}, \boldsymbol{y})$.

That is, using Eq. (4):

$$E_{\mathcal{D}}\{S_{\mathcal{D}}(\widehat{\boldsymbol{w}}_q)\} = \frac{1}{N}\sum_{k=1}^{N}(1-\delta_{kq}) \cdot E_{\mathcal{D}}\{S_{\mathcal{D}_q}(\widehat{\boldsymbol{w}}_q)\} + \frac{1}{N}\sum_{k=1}^{N} \delta_{kq} \cdot \Gamma \quad (9)$$

Finally averaging over all possible configurations of resamples gives:

$$E_q\{E_{\mathcal{D}}\{S_{\mathcal{D}}(\widehat{\boldsymbol{w}}_q)\}\} = (1-\beta) E_q\{E_{\mathcal{D}}\{S_{\mathcal{D}_q}(\widehat{\boldsymbol{w}}_q)\}\} + \beta\Gamma \quad (10)$$

where $\beta = (1 - 1/N)^N$ is the average number of examples in the test set i.e., equal to the probability that a specific example is not used in a resample of size N [1]. Aproximating the average w.r.t. q by the emprical average obtained from Q replicas, i.e.,

$$\langle X \rangle = \frac{1}{Q}\sum_{q=1}^{Q} X(\widehat{\boldsymbol{w}}_q). \quad (11)$$

where X is a arbitrary variable. Furthermore, we drop the average w.r.t. different data sets[2].

Finally, the relation becomes:

$$\langle S_{\mathcal{D}}(\widehat{\boldsymbol{w}}_q)\}\rangle \approx (1-\beta)\langle S_{\mathcal{D}_q}(\widehat{\boldsymbol{w}}_q)\rangle + \beta\Gamma \quad (12)$$

That is,

$$\Gamma \approx \frac{\langle S_{\mathcal{D}}(\widehat{\boldsymbol{w}}_q)\}\rangle - (1-\beta)\langle S_{\mathcal{D}_q}(\widehat{\boldsymbol{w}}_q)\rangle}{\beta} \quad (13)$$

3. EXPERIMENTAL RESULTS AND CONCLUDING REMARKS

While the generalization error estimate can be used for optimization of all aspects of neural net adaptation, including tuning of regularization parameters or selection of network architecture, we will here focus on it's use in the early stop context. This is a problem that can not be solved by the conventional statistical estimates like the that of Murata et al. [5], since these estimates assume the network to be close to optimal (i.e. within a second order Taylor expansion). Early stop is the oldest form of regularization, used in many practical implementations, and also subject to some analysis, see e.g., [7]. The idea is simply to inspect the test error on an independent set of validation data, when the validation error start to increase training is stopped to avoid overfitting.

For illustration, we use two time series problems. The first case involves prediction of a noisy time series appearing in a functional neuroimaging context. We show the early stop scenario for two different levels of regularization. The series consist of a stimulus series (an on-off signal) and a response series from a specific region in the visual cortex. The network is trained to model the response by one-step prediction of the response series using a simple lag-space input from both stimulus a response series. The training set consists of 250 input-output pairs, while another set of 200 datapoints are used for evaluation of the unbiased test error. All errors are normalized by the total variance of test and training series. Significant overtraining is seen and indeed expected with a low level of the weight decay, whereas by using the generalization error estimate we may hinder overtraining efficiently as shown in Fig. 1. In Fig. 2 we similarly show the result of training with a somewhat higher weight decay. In this case the minima in the test error and the test error estimate are rather shallow, but they do indeed coincide.

[1] Note that $\beta \to e^{-1}$ for $N \to \infty$ where e is the base of the natural logarithm.

[2] This is identical to the technique in order to derive the FPE [1] or NIC criteria [5].

Our second case is the wellknown sunspot prediction problem. The task is to predict the yearly average sunspot activity. We follow the conventional approach and use 12 input units encoding a simple lag-space, while the output unit predicts next years activity (see e.g. [8] for details on the sunspot prediction problem). The total series runs from 1700 to 1979. The 209 member training set covers 1700-1920, while we use the socalled test set I (1921-1955) to compute the test error. Like in the first case the aim is to eliminate overfitting. First we train an oversized network with eight hidden units and low regularization (just anough to stabilize our Newton optimizer). The training scenario is depicted in figure Fig. 3 and indeed significant overtraining is seen at training beyond 15 iterations. Also, we here train a well-regularized network (see figure Fig. 3). In this case there is virtually no overtraining, neither measured by the unbiased generalization error test set or by our bootstrap test error estimate. Hence, we conclude that the Bootstrap ensemble based estimate of the test error is a viable means for implementing an early stop rule.

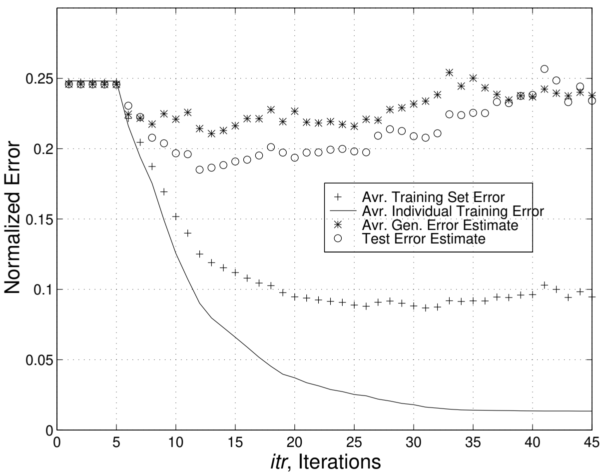

Figure 1: Overtraining scenario with small regularization (weight decay). The data set is a very noisy forecasting problem involving a stimulus series and a response series (the series was recorded in a functional neuroimaging experiment). The data set comprises $N = 250$ examples and an additional test set contains 200 examples. The Bootstrap ensemble had $Q = 15$ members. Each member is a feed-forward net with 20 input units, 8 hidden sigmoid units, and one output unit. The networks were trained by conventional backprop ($1 \leq itr \leq 5$), and by a second order pseudo Newton scheme ($6 \leq itr \leq 45$). The goal is to predict the noisy time series one-step-ahead based on a lag space of 10 previous values of the time series and 10 lagged values of an auxiliary time series coding the stimulus. In this run the network was only slightly regularized by weight decay. Errors are reported as normalized by the total variance of the series. Note that both the estimated and the measured test error suggest to stop training at around $itr = 12$.

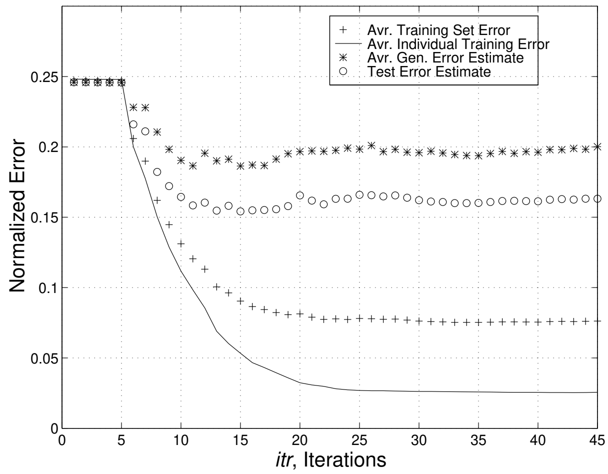

Figure 2: Overtraining scenario with near optimal regularization (weight decay). The data set and the networks are the same as in Fig. 1. In this case the network was more heavily regularized by weight decay. Note that both the estimated and the measured test error suggest to stop training around $itr = 15$. Note also that the overall level of test error is lower in this case, suggesting that careful control of the weight decay may be more efficient than early stop in optimizing generalization.

4. CONCLUSIONS

A technique for early stop has been presented. Training is terminated when a new Bootstrap based estimate of the generalization error has reached its minimum. Numerical examples showed excellent coherence among the Bootstrap based estimate and the test error on an independent test set. Furthermore, the numerical examples showed that one can not rely on early stop only; additional regularization (in the present case, weight decay) is necessary for achieving minimum generalization error.

5. REFERENCES

[1] H. Akaike: "Fitting Autoregressive Models for Prediction," *Ann. Inst. Stat. Mat.*, vol. 21, pp. 243–247, 1969.

[2] B. Efron & R.J. Tibshirani: *An Introduction to the Bootstrap*, New York, New York: Chapman & Hall, 1993.

[3] J. Hertz, A. Krogh & R.G. Palmer: *Introduction to the Theory of Neural Computation*, Redwood City, California: Addison-Wesley Publishing Company, 1991.

[4] J. Kindermann, G. Pass & F. Weber: "Query Construction for Neural Networks using the Bootstrap," *Proc. of ICANN'95*, EC2& lie, pp. 135–140, 1995

[5] N. Murata, S. Yoshizawa & S. Amari: "Network Information Criterion — Determining the Number of Hidden Units for an Artificial Neural Network Model," *IEEE Transactions on Neural Networks*, vol. 5, no. 6, pp. 865–872, Nov. 1994.

[6] J. Shao & D. Tu: *The Jackknife and Bootstrap*, Springer Series in Statistics, Berlin, Germany: Springer-Verlag, 1995.

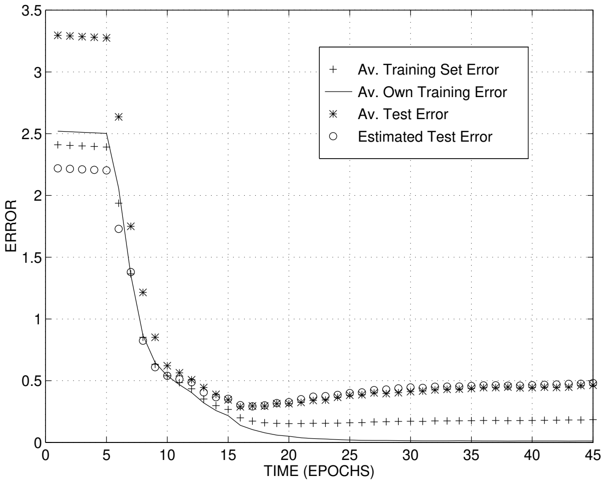

Figure 3: Overtraining scenario with large network and small regularization (weight decay). The data set is the wellknown sunspot prediction problem. The data set comprises $N = 209$ examples and an additional test set contains 35 examples. The Bootstrap ensemble had $Q = 15$ members. Each member is a feed-forward net with 12 input units, 8 hidden sigmoid units, and one output unit. The networks were trained by conventional backprop ($1 \leq itr \leq 5$), and by a second order pseudo Newton scheme ($6 \leq itr \leq 45$). The goal is to predict the noisy time series one-step-ahead based on a lag space of 12 previous values of the time series. In this run the network was only slightly regularized by weight decay. Errors are reported as normalized by the total variance of the series. Note that both the estimated and the measured test error suggest to stop training at around $itr = 12$.

[7] J. Sjöberg: *Non-Linear System Identification with Neural Networks*, Ph.D. Thesis no. 381, Department of Electrical Engineering, Linköping University, Sweden, 1995.

[8] C. Svarer, L.K. Hansen, and J. Larsen: "On Design and Evaluation of Tapped-Delay Neural Network Architectures," in H.R. Berenji *et al.* (eds.) *Proceedings of the 1993 IEEE Int. Conference on Neural Networks*, IEEE Service Center, NJ, vol. 1, pp. 46–51, 1993.

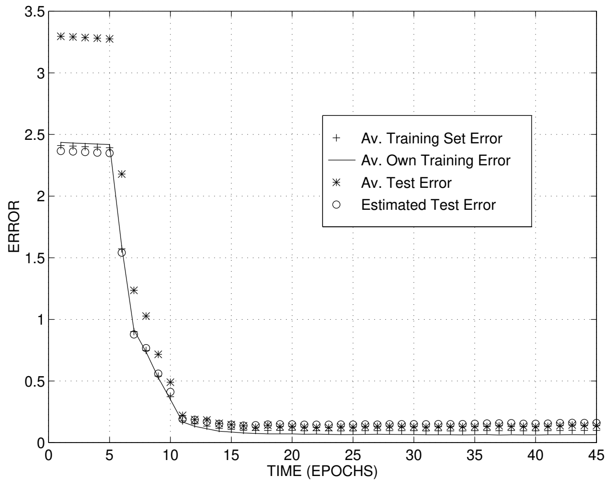

Figure 4: Overtraining scenario with a large network and near optimal regularization (weight decay). The data set and the networks are the same as in figure Fig. 3. Note that both the estimated and the measured test error suggest so stop training around $itr = 15$. Note also that like in the previous neuroimaging case, figures Fig. 1 - Fig. 2, the overall level of test error is lower in this case, suggesting that careful control of the weight decay may be more efficient than early stop in optimizing generalization.