On Design and Evaluation of Tapped-Delay Neural Network Architectures

Claus Svarer, Lars Kai Hansen, and Jan Larsen
CONNECT, Electronics Institute, B349
Technical University of Denmark,
DK-2800 Lyngby, Denmark
emails: claus, lars, jan@eiffel.ei.dth.dk

Abstract— **We address pruning and evaluation of Tapped-Delay Neural Networks for the sunspot benchmark series. It is shown that the generalization ability of the networks can be improved by pruning using the Optimal Brain Damage method of Le Cun, Denker and Solla. A stop criterion for the pruning algorithm is formulated using a modified version of Akaike's Final Prediction Error estimate. With the proposed stop criterion the pruning scheme is shown to produce succesful architectures with a high yield.**

I. Introduction

Needless to say, processing of time series is an important application area for neural networks, and the quest for application-specific architectures penetrates current network research. While the ultimate tool may be fully recurrent architectures, many problems arise during adaptation of these. Even worse, the generalization properties of recurrent networks are not well understood, hence, model optimization is difficult. However, the conventional Tapped-Delay Neural Net (TDNN) [11] may be analysed using statistical methods and the results of such analysis can be applied for model optimization. Here we demonstrate the power of this strategy within time series prediction. We aim at designing compact TDNN's using the so-called *Optimal Brain Damage* (OBD) method of Le Cun *et al.* [5]. The benefits from compact architectures are three-fold: They generalize better, they carry less computational burden, and they are faster to adapt if the environment changes. Further we show that the generalization ability of the network may be estimated, without extensive cross-validation, using a modification of Akaike's *Final Prediction Error* (FPE) estimate [1].

II. Time Series Prediction

The possibility of predicting the future fascinates. The techniques invoked through history cover oracles, crystal balls, feed forward neural networks and many more. While we can rule out long time predictions for chaotic systems; short time predictions may still be viable. Recent work by Priestly [9], and Weigend *et al.* [11] have established the *sunspot series* as a benchmark for time series prediction algorithms. The series is a scaled record of the yearly average sunspot activity for the period 1700-1979. The sunspot series is believed to be generated by a noisy, chaotic, dynamical system. The spectrum is dominated by a frequency corresponding to a 12 year period. Weigend *et al.* applied a weight decay scheme to design feed-forward networks that generalize better than conventional algorithms from the training set to an independent test set. In this work the OBD *pruning* method is shown to produce very compact networks for this problem, having only around 15 free parameters. The networks that we obtain use around one third of the parameters of the network published by Weigend *et al.* while having comparable performance.

We start the pruning procedure from the same initial network configuration as in [11]. The network is a *tapped delay line* architecture with 12 input units, 8 hidden sigmoid units and a single linear output unit, see Fig. 1. The initial network is fully connected between layers and implements a non-linear mapping from lag space $\mathbf{z}(k) = [x(k-1), ..., x(k-L)]$, $L = 12$, to the real axis:

$$\widehat{x}(k) = F_{\mathbf{u}}(\mathbf{z}(k)) \qquad \widehat{x} \in \mathcal{R}, \tag{1}$$

where $\mathbf{u}$=(w, W) is the N-dimensional weight vector and $\widehat{x}(k)$ is the prediction of $x(k)$.

The non-linear mapping can be written as:

$$F_{\mathbf{u}}(\mathbf{z}(k)) = \sum_{j=1}^{n_H} W_j \tanh\left(\sum_{i=1}^{L} w_{ij} x(k-i) + w_{i0}\right) + W_0, \tag{2}$$

where n_H is the number of hidden units. W_j are the hidden-to-output weights while w_{ij} connect the input and hidden units.

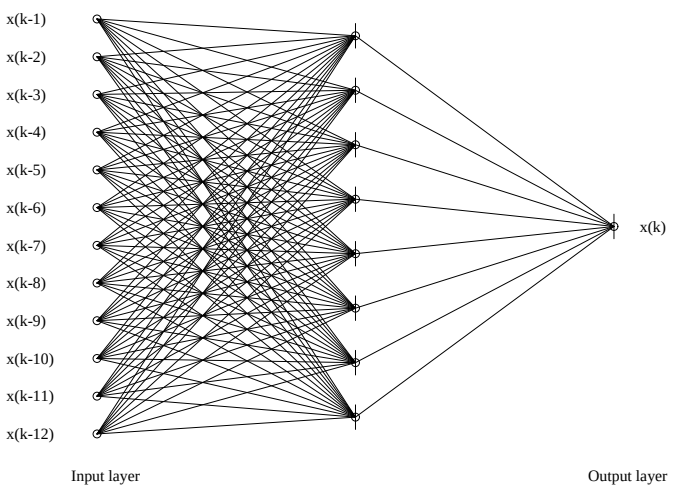

Fig. 1 Fully connected network used as a starting point for the pruning procedure. A vertical bar through a unit indicates an active threshold.

III. Training

The objective of the training procedure is single-step prediction. Hence, the network weights, $\mathbf{u}$, are trained to recognize the short time structure of the chaotic time series. We use the sum of squared errors to measure the prediction ability of the current network:

$$E_{train} = \frac{1}{p}\sum_{k=1}^{p}\left[x(k) - F_{\mathbf{u}}(\mathbf{z}(k))\right]^2, \tag{3}$$

where p is the number of training examples.

A state of the art simulator has been developed based on *batch mode*, second order local optimization. The second order scheme is implemented as a direct matrix-inversion identification of the hidden-to-output weights [2], while a pseudo Gauss-Newton method is used for identification of input-to-hidden weights, see e.g. [3].

To ensure numerical stability and for assisting the pruning procedure we augment the cost-function with a weight decay term. The cost-function can then be written as:

$$E = E_{train} + \frac{\alpha_w}{p}\sum_{ij}^{N_w} w_{ij}^2 + \frac{\alpha_W}{p}\sum_{j}^{N_W} W_j^2, \tag{4}$$

where N_w, N_W are the numbers of weights and thresholds in hidden and output units, respectively.

The second order pseudo Gauss-Newton method used for identification of input-to-hidden weights can be written as:

$$\Delta w_{ij} = -\eta\left(\frac{\partial E_{train}}{\partial w_{ij}} + \frac{2\alpha_w}{p}w_{ij}\right)\Bigg/\left(\frac{\partial^2 E_{train}}{\partial w_{ij}^2} + \frac{2\alpha_w}{p}\right) \tag{5}$$

where the parameter η is used to secure that all the weight updates lead to a decrease in the cost-function. η is initialized to 1 before each step, and iteratively dimished by powers of two, until the step leads to a decrease in the cost-function. As in [5] we approximate the second derivative by the positive semi-definite expression:

$$\frac{\partial^2 E_{train}}{\partial w_{ij}^2} \approx \frac{2}{p}\sum_{k=1}^{p}\left(\frac{\partial F_{\mathbf{u}}(\mathbf{z}(k))}{\partial w_{ij}}\right)^2. \tag{6}$$

IV. Pruning by Optimal Brain Damage

The OBD scheme proposed by Le Cun *et al.* [5] was succesfully applied to reduce large networks for recognition of handwritten digits [6]. The basic idea is to estimate the effect on the *training error* when deleting weights. The estimate is formulated in terms of weight saliencies s_l:

$$\delta E_{train} = \sum_{l\in D} s_l \equiv \sum_{l\in D}\left(\frac{2\alpha}{p} + \frac{1}{2}\frac{\partial^2 E_{train}}{\partial u_l^2}\right)u_l^2, \tag{7}$$

where u_l is a component of $\mathbf{u}$. The saliency definition used here takes into account that the weight decay terms force the weights to depart from the minimum of the training set error. The sum runs over the set of deleted weights D.

The following assumptions enter the derivation of OBD:

- The terms of third and higher orders in the deleted weights can be neglected.
- The off-diagonal terms in the Hessian, $\frac{\partial^2 E_{train}}{\partial u_l \partial u_{l'}}$, can be neglected.

Computationally the second order (diagonal) terms are reused from the training scheme (5), in particular we refrain from working on the full Hessian, which would scale poorly for large networks.

The recipe allows for *ranking* the weights according to saliency. The question of how many weights it may be possible to delete was not answered in [5]. To evaluate a network, hence, formulate a pruning *stop criterion*, we note that there are three objectives of pruning:

- Improve the generalization performance by limiting the network resources.
- Reduce the computational burden of prediction.
- Allow for fast on-line adaptation.

In this presentation we emphasize the first of these objectives. However, since the generalization error by definition, involves test on an independent data set, we cannot directly use the error on the training set, as estimated by OBD, to formulate a stop criterion. We may indeed accept an increased error on the training set if better generalization is obtained. Also, among networks with the same estimated test error we still prefer the minimal, because it has a lower computational burden, and typically needs less training examples for retraining if the environment changes. The latter is very important for on-line adaptation. If data are abundant we can formulate a stop criterion based on a validation set (an independent subset of the training set). This approach was criticized by Weigend *et al.*: The training data set is scarce for the sunspot series – and indeed for many other applications. We support their conclusion by the observation that even for the (90%)/(10%) splitting of the training set (in training and validation sets, respectively), as used by [11], the estimated validation error is an extremely noisy quantity.

In the usual case of limited data sets we follow the standard approach within *system identification* [7] and estimate the generalization error of the pruned networks using statistical arguments. In particular, we apply Akaike's FPE estimate [1, 7] of the test error in terms of the training error. In its standard form it reads:

$$\widehat{E}_{test} = \frac{p+N}{p-N} E_{train}, \tag{8}$$

where p is the number of training samples, and N is the number of parameters in the model. The left hand side of (8) is the average generalization error, averaged over all possible training sets of size p. The estimate is based on linearization of the networks as regards the fluctuations in the weights resulting from different training sets. The relation express the fact that the training error is a biased estimate of the noise level because each parameter during training has "absorbed" some of the noise in the training samples.

Since we have regularized the training procedure by weight-decay terms α_w, α_W, hence, suppressed the ability of the (otherwise) ill-determined parameters to model noise, we need to modify the classical FPE estimate by replacing the total number of parameters with the *effective* number of parameters see e.g. [4, 8]:

$$\widehat{E}_{test} = \frac{p+N_{eff}}{p-N_{eff}} E_{train}, \tag{9}$$

$$N_{eff} = \sum_{ij}^{N_w} \left(\frac{\lambda_{ij}}{\lambda_{ij} + 2\alpha_w/p} \right)^2 + \sum_{j}^{N_W} \left(\frac{\Lambda_j}{\Lambda_j + 2\alpha_W/p} \right)^2 . \tag{10}$$

Where the λ's are the second derivatives already computed in (7), $\lambda_{ij} \equiv \partial^2 E_{train} / \partial w_{ij}^2$, $\Lambda_j \equiv \partial^2 E_{train} / \partial W_j^2$.

In brief, the following assumptions enter the derivation of (8-10):

- Independence of input and error on output.
- Sufficient capacity, i.e., the network must be able to implement the rule.
- Many examples pr. weight: $N/p \to 0$.
- The off-diagonal elements of the second derivative matrix can be neglected.

With the above tool we can obtain a generalization error estimate for each pruned network. By selecting the network with the lowest estimated generalization error we have developed the stop criterion sought.

V. Experiments

Following Weigend *et al.* [11], the sunspot data are partioned into a training set (1700-1920) and a test set (1921-1979), and further we compute the test error for two separate sets, namely the periods 1921-1955 and 1956-1979. The sunspot series is rather non-stationary and the latter period is atypical for the series as a whole. The normalized errors on the training set and on the two test sets are calculated as:

$$E_{set} = \frac{1}{\sigma_{total}^2 \cdot p_{set}} \sum_{k=1}^{p_{set}} \left[x(k) - F_{\mathbf{u}}(\mathbf{z}(k)) \right]^2 , \tag{11}$$

where p_{set} is the number of examples in the data set in question. The squared errors of each data set are normalized as in [11] by the variance of the total data set σ_{total}^2.

An ensemble of 11 networks were trained and pruned. The weight decay parameters were set as: $\alpha_w = 0.02$ and $\alpha_W = 0.01$.

Fig. 2 shows the normalized training error and the two test set errors during training of the fully connected network. Note that the training set error decreases monotonously, while the error on the test sets start out decreasing, but after some training increase again. This is a generic *over-training scenario* in which the network overfits the training set.

In order to prevent the network from overfitting we limit its resources by pruning. We use OBD, rank the remaining weights according to saliency, and delete a number of these determined by $|D| = \lceil 0.02 \cdot N_{remaining} \rceil$, in a simple iterative procedure. The evolution of training and test errors during pruning are recorded for a specific network in Fig. 3. Further this figure shows the estimated test error as given by (9). The FPE estimate of the test error lies

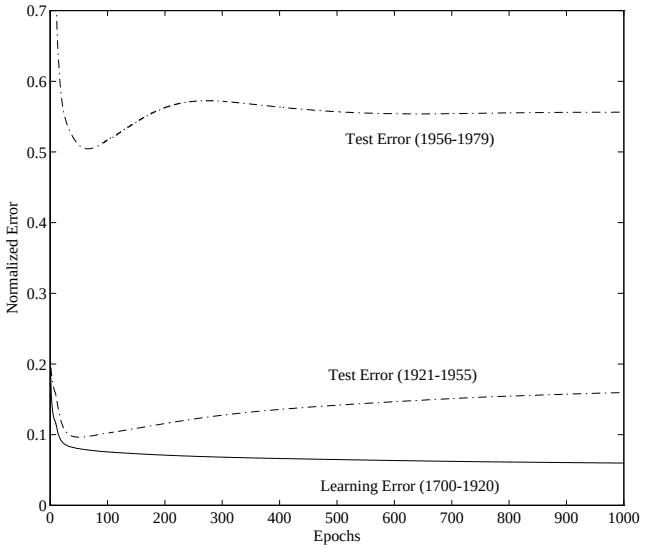

Fig. 2 Training and test error when training the fully connected network. An 'Epoch' is a full sweep through the training set.

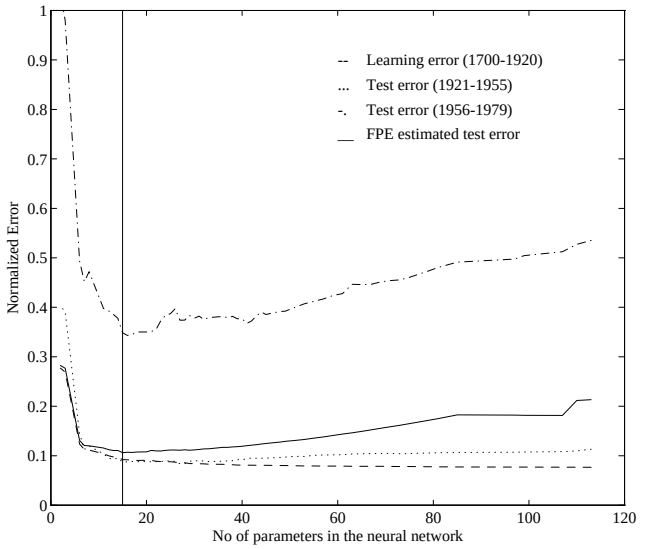

Fig. 3 The evolution of training and test errors during pruning. The FPE estimate of the test error is based on equation (9). The vertical line indicates the network for which the *estimated* test error is minimal.

between the two test sets. Most importantly, the estimate reproduce the common trend of the test sets, sharing a generalization error minimum just below 20 parameters, indicating that the statistical approach is viable. In the particular run, the stop criterion selects a network having 12 weights and 3 thresholds as indicated by the vertical line in Fig. 3.

Among the eleven networks pruned, the procedure selected 9 nets with a number of parameters in the range $12-16$, and two nets with more than 25 parameters. The 9 small architectures were considered succesfully pruned, and used as an ensemble for computing significance levels for the procedure. In order to fine tune the nine small networks, they were retrained without weight decay, resulting in the normalized errors on the three data sets: 0.090 ± 0.001 (1700-1920), 0.082 ± 0.007 (1921-55), and 0.35 ± 0.05 (1956-79). Four nets had 15 parameters, another four nets had 16 parameters, while one net had 12 parameters. The latter appears to be overpruned and carried higher errors than the rest. In Table I we compare our findings with other reported results for the sunspot series.

We illustrate the properties of one of the compact networks (presented in Fig. 4 and in Table II), in two ways: First we show, in Fig. 5, the retraining history of this network after pruning. As expected we see no overtraining. Both the training error and the two test errors are monotonously decreasing.

It is interesting to note that the network does not use

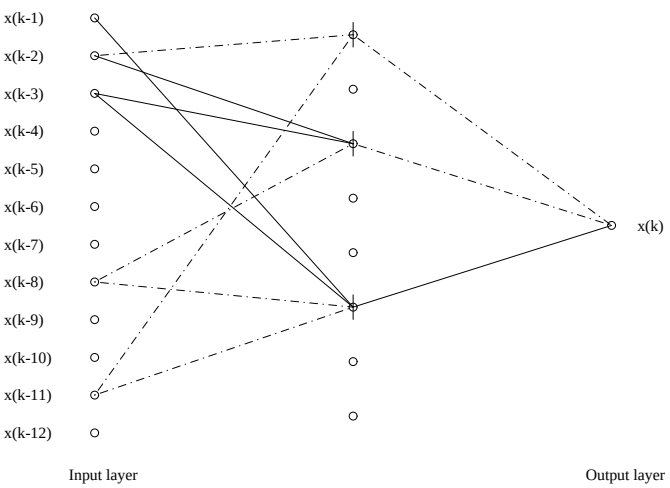

Fig. 4 Pruned network with 12 weights and 3 thresholds. Note that the network only uses a subset of the Tapped-Delay line. Dash-dotted lines indicate negative weights, and solid lines positive weights. The network parameters are given in Table II.

the full set of inputs. We interpret this to be a result of the finite training set. Within the noise level of the sunspot time series it is harmful to use more than a carefully selected subset of the available lag space. Among the succesfully pruned networks there is some consensus regarding which inputs to use. They all use the two most recent inputs, and one or more inputs among the "oldest" part of lag space.

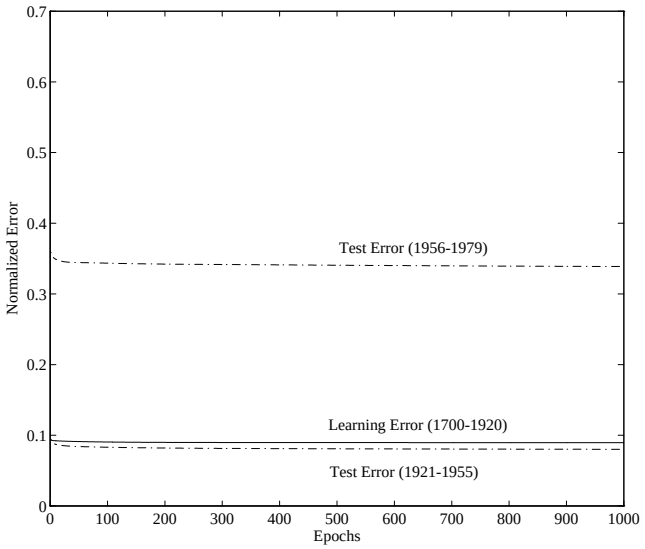

Fig. 5 Training and test error when re-training the pruned network without weight decay.

Secondly, in Fig. 6, the predicted sunspot activity using the pruned and fine tuned network is shown. We note that the variance of the sunspot activity is increased significantly in the period after about 1960.

In Fig. 7 we show the effective number of parameters as computed from (10). Judged from this figure the weight decays are important in the initial pruning phase where they limit the number of parameters to about 85. With higher weight decays we were unable to train the networks to error levels like those reported. On the other hand, with smaller weight decays the second order optimization scheme is plagued by numerical problems also leading to higher errors and a lower yield of useful architectures.

VI. Conclusion

We have discussed pruning and evaluation of Tapped-Delay Neural Networks. We have shown that the generalization ability on the sunspot data can be improved by pruning using the Optimal Brain Damage method. In particular, we have identified a set of compact networks with three hidden units employing around 15 weights and thresholds. These networks generalize well compared to

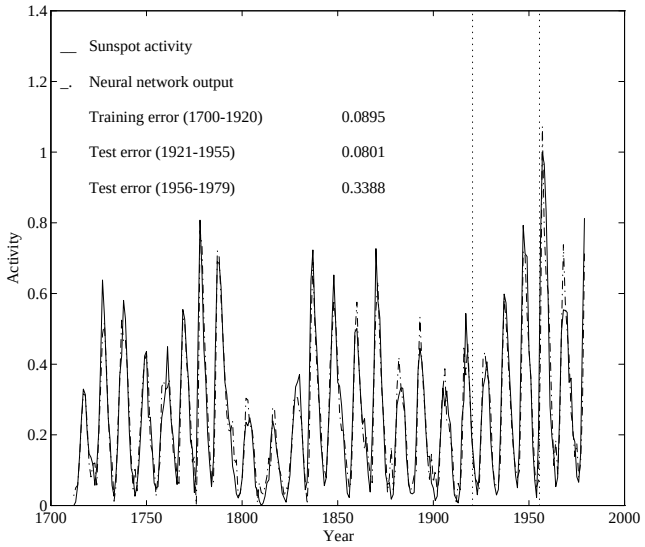

Fig. 6 Predicted sunspot activity using the pruned feed-forward network shown in Fig. 4.

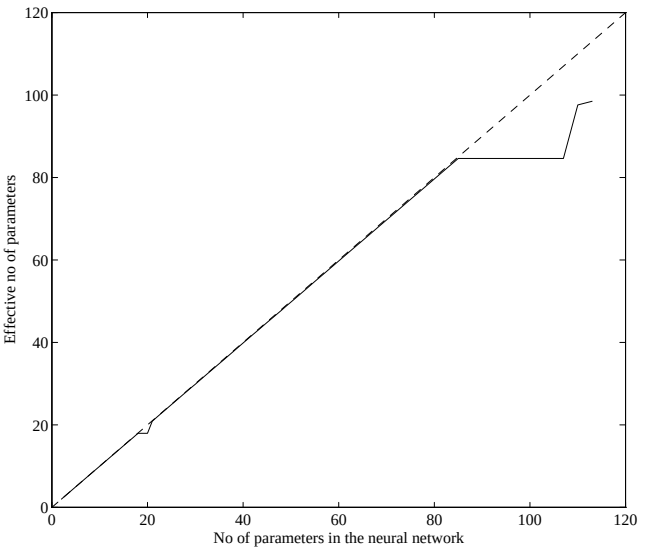

Fig. 7 The effective number of parameters in the neural network during the pruning session.

TABLE I
NORMALIZED ERROR

Model	Train (1700-1920)	Test (1921-55)	Test (1956-79)	Number of parameters
Tong and Lim [10]	0.097	0.097	0.28	16
Weigend *et al.* [11]	0.082	0.086	0.35	43
Linear model[1]	0.132	0.130	0.37	13
Fully connected network[2]	0.078 ± 0.002	0.104 ± 0.005	0.46 ± 0.07	113
Pruned network[3]	0.090 ± 0.001	0.082 ± 0.007	0.35 ± 0.05	$12 - 16$

1) Linear model is a single linear unit. 2) The initial pre-pruned networks, trained with the same weight decay terms as used during pruning. 3) Pruned networks retrained without weight decay. The mean and standard deviation are based on the networks selected for retraining (90 % of the initial set of networks).

TABLE II
PRUNED NETWORK WEIGHTS

Input	Hidden unit 1	Hidden unit 3	Hidden unit 6
Lag 1	0	0	1.399
Lag 2	-0.562	0.944	0
Lag 3	0	1.035	1.068
Lag 8	0	-0.435	-0.408
Lag 11	-0.279	0	-0.259
Threshold	0.192	0.236	0.411

Hidden	Output unit
Unit 1	-1.1544
Unit 3	-1.5537
Unit 6	1.5636
Threshold	0

previous studies. Further we have shown that the network performance may be evaluated using statistical methods and that the trend (in error versus number of parameters) of the estimate is in good agreement with those of the test sets. We have shown how the estimated generalization error may be used for selection of the optimal network architecture during a pruning session. The yield of the procedure was 90%: Out of eleven networks the procedure found nine useful architectures.

For the sunspot series we note that non-stationarity is a problem insofar that the normalized test error for the period 1956-1979 is four times higher than the test error on the more representative test set comprising the period 1921-1955. This means, that it is important to corroborate our results on other problems and time series, such work is in progress.

Acknowledgments

This research is supported by the Danish Natural Science and Technical Research Councils through the Computational Neural Network Center (CONNECT).

References

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

[2] S.A. Barton: "A Matrix Method for Optimization a Neural Network". *Neural Computation*, vol. 3, 450-459 (1990).

[3] J. Hertz, A. Krogh and R.G. Palmer: *Introduction to the Theory of Neural Computation*, Addison Wesley, New York (1991).

[4] J. Larsen: *Design of Neural Network Filters.* Ph. D. Thesis, Electronics Institute, Technical University of Denmark. In preparation, (1993).

[5] Y. Le Cun, J.S. Denker, and S.A. Solla: "Optimal Brain Damage". In *Advances in Neural Information Processing Systems 2*, 598-605, Morgan Kaufman. (1990).

[6] Y. Le Cun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, and L.D. Jakel: "Handwritten Digit Recognition with a Back-Propagation Network", In *Advances in Neural Information Processing Systems 2*, 396-404. Morgan Kaufman. (1990)

[7] L. Ljung: *System Identification: Theory for the user*, Prentice-Hall, Information and System Sciences series (1987).

[8] J.E. Moody: "Note on Generalization, Regularization and Architecture Selection in Nonlinear Systems". In *Neural Networks For Signal Processing*; Proceedings of the 1991 IEEE-SP Workshop, (Eds. S.Y. Kung, B.H. Juang, and C. Kamm), IEEE Service Center, 1-10, (1991).

[9] M.B. Priestly: *Non-linear and Non-stationary Times Series Analysis*, Academic Press (1988).

[10] H. Tong and K. S. Lim: "Threshold autoregression, limit cycles and cyclical data". *Journ. Roy. Stat. Soc. B*, vol. 42, 245 (1980).

[11] A.S. Weigend, B.A. Huberman, and D.E. Rumelhart: "Prediction the future: A Connectionist Approach", *Int. J. of Neural Systems*, vol. 3, 193-209 (1990).