

SIZING OF THE MULTILAYER PERCEPTRON VIA MODULAR NETWORKS

Hema Chandrasekaran¹, Kyung K. Kim², and Michael T. Manry³

^{1,3}Department of Electrical Engineering

The University of Texas at Arlington

Arlington, TX 76019

e-mail : manry@uta.edu

²Lockheed Martin Tactical Aircraft Systems, Mail Zone 1772

P.O. Box 748, Fort Worth, Texas 76101

Abstract

A fast method for sizing the multilayer perceptron is proposed. The principal assumption is that a modular network with the same theoretical pattern storage as the multilayer perceptron has the same training error. This assumption is analyzed for the case of random patterns. Using several benchmark datasets, the validity of the approach is demonstrated.

I. INTRODUCTION

Multilayer perceptrons (MLPs) with N_h nonlinear units in a single hidden layer have been established as universal function approximators [1,2]. Many investigators have proposed methods for choosing N_h , a process known as sizing. These methods include the leave-one-out technique, growing methods [3,4], pruning methods [5,6], and approaches based upon Akaike's information criterion [7,8]. Unfortunately, these approaches are very time consuming. Unlike in the leave-one-out and Akaike approaches, our goal is to predict MLP training errors for a wide range of N_h values, in less time than that required to train a single MLP.

It has been observed that different kinds of nonlinear networks, having the same theoretical pattern storage (or pattern memorization), produce very similar values of the training error E [9]. In order to verify the observation with networks having many free parameters, however, very efficient training methods are required. In our paper, we develop a fast sizing approach based upon the observation. In section II, we summarize a convergent technique for the fast design of piecewise linear (PWL) modular networks. In section III, we relate the pattern storage of the PWL and MLP networks, and describe the resulting sizing algorithm. In section IV, we analyze the bounds on the training error versus pattern storage for both PWL and MLP for the case of random training patterns. In section V, we demonstrate the algorithm on several well-known benchmark data sets.

II. PIECEWISE LINEAR MODULAR NETWORKS

The PWL modular network for N inputs and M outputs contains N_{mod} modules, each of which has an N -dimensional cluster center vector \mathbf{m}_n and a matrix $\mathbf{W}_{\text{pwl}(n)}$ of dimension M by $(N+1)$. The network employs a weighted sum of squares distance measure $d(\cdot)$. If the input vector \mathbf{x} belongs to the n th cluster such that $d(\mathbf{x}, \mathbf{m}_n) = \min_k d(\mathbf{x}, \mathbf{m}_k)$ then the output vector is $\mathbf{y} = \mathbf{W}_{\text{pwl}(n)} \cdot (\mathbf{x}^T:1)^T$. In this section, we summarize a PWL modular network training algorithm and give the network's pattern storage.

A. Training Algorithm

A convergent training algorithm for the PWL modular network [9] is described in the following.

- (1) Weights of the distance measure are found which de-emphasize less useful inputs, following the approach of [10].
- (2) We initialize the network with one cluster centered at the mean vector of input vectors. Two new cluster centers are found by perturbing this mean and reassigning patterns. A linear mapping is designed for each module or cluster through regression.
- (3) We add a new cluster by splitting the cluster contributing most to the mapping error. Again, a linear mapping is designed for each module or cluster through regression.
- (4) In each iteration, the number of clusters increases at most by one, till the specified number of clusters have been generated.

B. Convergence

Note that each regression step above will tend to reduce the network training error or leave it the same. However, it is possible for a new cluster to attract patterns from nearby clusters, which are not well approximated by a linear mapping, resulting in an increase in the network training error. If this happens, we delete the new module and pick a new cluster to split. Whenever the mapping error decreases, we store the network configuration. If we fail to achieve a lower mean square mapping error after making exhaustive attempts to split clusters, we backtrack the network to the previous optimum configuration and stop. The design algorithm clearly converges. The pattern storage of the PWL modular network is N_{mod} multiplied by the storage per module,

$$S_{\text{PWL}} = N_{\text{mod}} \cdot (N + 1) \quad (1)$$

Although the PWL modular network trains well, the resulting mapping is discontinuous at the boundaries between adjacent modules. When a discontinuous mapping is unacceptable, one may use a global type network such as the MLP.

III. SIZING ALGORITHM

In this section we describe a sizing algorithm for a fully connected MLP, which includes bypass weights, thresholds in the hidden layer, and thresholds in the output layer.

A. MLP Pattern Storage

It is well known that the Volterra filter can memorize a number of training patterns equal to the number of coefficients associated with one output. The upper bound on the MLP's pattern storage is the same [11]. Its pattern storage, S_{MLP} , is bounded above by P_{MLP}/M , where P_{MLP} is the total number of free parameters in the network. It has been shown [11] that this bound is fairly tight. Therefore we assume

$$S_{MLP}(N_h) = \left(\frac{(N+1+M)}{M} \right) \cdot N_h + (N+1) \quad (2)$$

$$N_h(S_{MLP}) = \left(\frac{M}{N+1+M} \right) \cdot S_{MLP} - \frac{(N+1) \cdot M}{N+1+M} \quad (3)$$

B. Fundamental Assumption

The fundamental assumption we make is that the PWL modular network and MLP have the same error performance if their respective theoretical pattern storages are the same. This assumption is certainly true in the following three situations:

- If the networks' outputs are constant for all training patterns, then both the PWL modular network and MLP have the same error performance and they both memorize 1 pattern only.
- If the networks' outputs are linear functions of the inputs (as when the PWL modular network has one module and the MLP has only bypass weights and thresholds), both networks are linear, have the same error performance, and can memorize $N+1$ patterns.
- If the both the PWL modular network and MLP memorize all the training patterns, then they both have the same training error (zero).

Our observation is that at intermediate storage capacities, the assumption also usually holds. Note, however that mappings can be constructed which are counterexamples to our assumption.

C. Algorithm

Consider the PWL modular network trained for N_{it} iterations. Let $E_{it}(i)$ be the training error and $N_{mod}(i)$ be the number of modules in the i th iteration. The pattern storage $S_{PWL}(i)$ in the i th iteration is given by

$$S_{PWL}(i) = N_{mod}(i) \cdot (N + 1) \quad (4)$$

Equating the pattern memorization of the MLP in equation (2) to that of the PWL modular network in equation (4), we have

$$N_h(i) \equiv N_h(S_{PWL}(i)) = \frac{M \cdot (N + 1) \cdot (N_{mod}(i) - 1)}{N + M + 1} \quad (5)$$

This formula helps us to estimate the number of hidden units in an equivalent MLP for various $E_t(i)$, $i = 1, 2, \dots, N_{it}$.

Case $M = 1$: When the number of outputs $M = 1$, equation (5) approximately reduces to $N_h = (N_{mod} - 1)$.

Case $M > 1$: In this case, overly large MLP networks result when the desired outputs are correlated and many free parameters are redundant. Then, a singular value decomposition (SVD) technique is employed to detect whether outputs can be compressed without significantly degrading the MSE performance. Compressing the outputs allows the sizing algorithm to predict a smaller, less complex MLP. The resulting mean square error E_t' after we compress the M outputs down to M' is given by

$$E_t' = E_t + \sum_{i=M'+1}^M \lambda_i$$

where λ_i 's are the i th singular values of the desired outputs' covariance matrix. For each value of i , the sizing algorithm generates all possible MLP configurations and their predicted training MSE using M' values between 1 and M . The algorithm sifts through these potential configurations and saves those that have the lowest training MSE for a given number of hidden units. The final output of our algorithm is the sequence of ordered pairs, $\{E_t'(i), N_h(i)\}$.

IV. ERROR VERSUS PATTERN STORAGE

Our goal in this section is to show that a failure of our principal assumption, as depicted in figure 1, does not happen when the training patterns are chosen randomly (all elements of patterns are statistically independent). For the following discussion, assume there are N_v random training patterns having N -dimensional input vectors, M -dimensional desired output vectors \mathbf{t}_p . We also assume that the outputs \mathbf{t}_p and $-\mathbf{t}_p$ are equally likely.

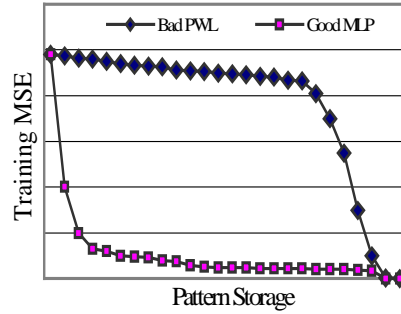


Figure 1 : Failure of Principal Assumption

A. Upper Bound on the MSE versus N_{mod} Curve

Each module in the PWL network memorizes $(N+1)$ patterns perfectly. We assume $(N_{\text{mod}}-1)$ modules memorize $(N_{\text{mod}}-1) \cdot (N+1)$ training patterns giving zero training error, leaving the rest of the training patterns to be handled by the last module. Let the output vector of the last module be $\mathbf{y} = E[\mathbf{t}_p]$ where \mathbf{t}_p is the desired output vector. Here $E[\cdot]$ denotes the expectation over all randomly chosen training sets, from a given joint probability density. Thus the total mean square training error E_t is given by

$$\begin{aligned} E_t &= \{N_v - (N_{\text{mod}} - 1) \cdot (N + 1)\} \cdot M \cdot E[(t_p - E(t_p))^2] \\ &= \{N_v - (N_{\text{mod}} - 1) \cdot (N + 1)\} \cdot M \cdot \sigma_t^2 \\ &= \{N_v + (N + 1)\} \cdot M \cdot \sigma_t^2 - N_{\text{mod}} \cdot (N + 1) \cdot M \cdot \sigma_t^2 \end{aligned} \quad (6)$$

where $\sigma_t^2 = E[(t_p - E(t_p))^2]$ is the variance of each output and $N_{\text{mod}} \geq 1$. We see that equation (6) represents a straight line with a slope of $-(N+1) \cdot M \cdot \sigma_t^2$. We conclude that the linear case is an upper bound on the E_t versus N_{mod} and E_t versus S_{PWL} curves.

B. Upper Bound on the MSE versus N_h Curve

In this subsection we show that on the average the E versus N_h curve for the MLP has a linear upper bound, for the case of random patterns.

We assume that the MLP with N_h hidden units has completely memorized N_v training patterns and we have applied the Gram-Schmidt(GS) procedure to orthonormalize and order the output basis functions. Also, we assume that we have eliminated any linearly dependent basis functions (hidden units). Consider any two consecutive hidden units whose basis functions are numbered j and $(j+1)$. Let the initial raw basis functions be $\sigma_1, \sigma_2, \sigma_3, \dots, \sigma_{N_u}$ where $N_u = N+1+N_h$, N is the input vector dimension, N_h is the number of hidden units, 1 for the threshold. Let the corresponding orthonormal basis functions after the GS procedure be $\phi_1, \phi_2, \phi_3, \dots, \phi_{N_u}$. Next, removing the effects of all the previous $(j-1)$ basis functions from the rest of the basis functions we obtain

$$x_k = \sigma_k - D_1\phi_1 - D_2\phi_2 - D_3\phi_3 - D_4\phi_4 \dots \dots \dots - D_{j-1}\phi_{j-1} \quad \text{where } j \leq k \leq N_u,$$

$$D_1 = \langle \phi_1, \sigma_k \rangle, D_2 = \langle \phi_2, \sigma_k \rangle, \dots, D_{j-1} = \langle \phi_{j-1}, \sigma_k \rangle. \text{ Here}$$

$$\langle \phi_1, \sigma_k \rangle = \frac{1}{N_v} \sum_{p=1}^{N_v} \phi_{1p} \cdot \sigma_{kp} \text{ following the definition of [15]. Now,}$$

$$t_i' = t_i - w_{i1}\phi_1 - w_{i2}\phi_2 - w_{i3}\phi_3 - w_{i4}\phi_4 \dots \dots \dots - w_{i,j-1}\phi_{j-1} \quad \text{where } 1 \leq i \leq M$$

where $w_{i1}, w_{i2}, \dots, w_{i,j-1}$ are the orthonormal weights connecting the basis functions to the i th output.

For the following derivation we consider only one output. Consider once again the basis functions $j, (j+1)$. Without loss of generality, we refer to them as basis

functions 1 and 2 from now on. Let the one output be t' . Let w_1, w_2 be the output weights for orthonormal basis functions 1 and 2. Define the following :

$$\begin{aligned} P_{11} = \langle x_1, x_1 \rangle &= \frac{1}{N_v} \sum_{p=1}^{N_v} x_{1p} \cdot x_{1p}, & P_{12} = \langle x_1, x_2 \rangle &= \frac{1}{N_v} \sum_{p=1}^{N_v} x_{1p} \cdot x_{2p} \\ Q_1 = \langle x_1, t \rangle &= \frac{1}{N_v} \sum_{p=1}^{N_v} x_{1p} \cdot t'_p, & Q_2 = \langle x_2, t \rangle &= \frac{1}{N_v} \sum_{p=1}^{N_v} x_{2p} \cdot t'_p \end{aligned} \quad (7)$$

From modified Gram-Schmidt orthonormalization procedure we have the following :

$$\begin{aligned} x'_1 &= x_1 / \sqrt{P_{11}} \\ x'_2 &= \frac{x_2 - \langle x'_1, x_2 \rangle x'_1}{\left\| x_2 - \frac{P_{12}}{P_{11}} \cdot x_1 \right\|} = \frac{x_2 - \frac{P_{12}}{P_{11}} \cdot x_1}{\sqrt{P_{22} - \frac{P_{12}^2}{P_{11}}}} = \frac{P_{11}x_2 - P_{12}x_1}{\sqrt{P_{11}} \cdot \sqrt{P_{11}P_{22} - P_{12}^2}} \end{aligned} \quad (8)$$

$$\begin{aligned} w_1 = \langle x'_1, t \rangle &= Q_1 / \sqrt{P_{11}} \\ w_2 = \langle x'_2, t \rangle &= \frac{P_{11}Q_2 - P_{12}Q_1}{\sqrt{P_{11}} \cdot \sqrt{P_{11}P_{22} - P_{12}^2}} \end{aligned} \quad (9)$$

If we choose x_2 as the first basis function, then the corresponding orthonormal weight would be,

$$w'_1 = \langle x'_2, t \rangle = Q_2 / \sqrt{P_{22}} \quad (10)$$

We state the following properties resulting from Schmidt procedure ordering and complete memorization:

1. All the basis functions we have are linearly independent; if not we can always eliminate those dependent hidden unit basis functions. This means that none of the orthonormal weights connecting the basis functions to the output are zero.
2. The training error $E = 0$ due to perfect memorization of the patterns.
3. $w_1^2 \geq w_1'^2$, a property of Schmidt procedure ordering. Therefore,

$$\frac{Q_1^2}{P_{11}} > \frac{Q_2^2}{P_{22}} \Rightarrow P_{11}P_{22}Q_1^2 > P_{11}^2Q_2^2$$

4. $P_{11}P_{22} > P_{12}^2$, since $(P_{11}P_{22} - P_{12}^2) = \|x_2 - \langle x_1, x_2 \rangle x_1\|^2$, thus positive.

$$\text{Consider } w_1^2 - w_2^2 = (P_{11}P_{22}Q_1^2 - P_{11}^2Q_2^2) + 2(P_{11}P_{12}Q_1Q_2 - P_{12}^2Q_1^2) \quad (11)$$

The term $(P_{11}P_{22}Q_1^2 - P_{11}^2Q_2^2)$ is always positive because of property 3. However, we can not say whether the second term $(P_{11}P_{12}Q_1Q_2 - P_{12}^2Q_1^2)$ is positive or negative for a particular realization of the network. So, we consider the ensemble average of $(P_{11}P_{12}Q_1Q_2 - P_{12}^2Q_1^2)$, which may be written as

$$E[P_{11}P_{12}Q_1Q_2 - P_{12}^2Q_1^2] = \sum_{j=1}^{N_v} \sum_{k=1}^{N_v} \sum_{m=1}^{N_v} \sum_{n=1}^{N_v} E[x_{1k}^2 x_{1m} x_{2m} x_{1n} t_n x_{2j} t_j] - \sum_{j=1}^{N_v} \sum_{k=1}^{N_v} \sum_{m=1}^{N_v} \sum_{n=1}^{N_v} E[x_{1k} x_{2k} x_{1j} x_{2j} x_{1m} t_m x_{1n} t_n] \quad (12)$$

Here the subscripts j,k,m, and n denote the pattern numbers in a given data set. The training patterns in a data set are assumed to be statistically independent. This is a valid assumption since we are considering completely independent random inputs and outputs.

We notice that the terms like $E[x_{1m}^i x_{2m}]$ are zero for every positive integer i and terms like $E[x_{1m} (x_{2m})^{2i} t_m^j]$ are zero for every nonnegative i and j for the following reason: The network performance is not affected if one or all of the basis functions and their corresponding output weights change sign simultaneously. Thus equation (12) now reduces to

$$E[P_{11}P_{12}Q_1Q_2 - P_{12}^2Q_1^2] = \sum_{j=1}^{N_v} \sum_{k=1}^{N_v} E[x_{1k}^2] \cdot E[x_{1j}^2 x_{2j}^2 t_j^2] - \sum_{k=1}^{N_v} \sum_{m=1}^{N_v} E[x_{1k}^2 x_{2k}^2] \cdot E[x_{1m}^2 t_m^2] \quad (13)$$

which is net positive. We have proved that in an ensemble sense $E[w_1^2 - w_2^2] > 0$. *This result is true even if we have multiple outputs.* Thus we have proved that the average E versus N_h curve for a multilayer perceptron and the E_t versus N_{mod} curve for PWL net lie below a linear curve.

Since the average E_t versus S_{PWL} curve and E versus S_{MLP} curves are below a linear curve, the worst case depicted in Figure 1 does not occur, on the average, for random patterns.

V. NUMERICAL RESULTS

A. Data Sets With One Output

X.tra : This training data set was created from a chaotic time series generated by the Mackey-Glass delay-difference equation[12] with $a = 0.2$, $b = 0.1$, and $\tau = 17$. The desired output is the sample at time $T+6$. The training data set consists of 4 inputs and 1 output and contains 2654 training patterns. The predicted and actual E versus N_h curves, illustrated in figure 2, show good performance by the sizing algorithm.

Hwang-t3 : The "hwang-t3" data set contains artificial data derived from a real valued function of two variables, used by J.N.Hwang and others to test non-parametric regression methods[13]. The function y is defined to be

$$y = 42.659 \cdot (0.1 + x_1 \cdot (0.05 + x_1^4 - 10 \cdot x_1^2 \cdot x_2^2 + 5 \cdot x_2^4))$$

The functions are all real-valued, and are defined on the two-dimensional domain $[0,1] \times [0,1]$. There are 8170 training patterns with 2 inputs and 1 output. The

predicted and actual E versus N_h curves, illustrated in figure 3, again show that the sizing algorithm is very effective.

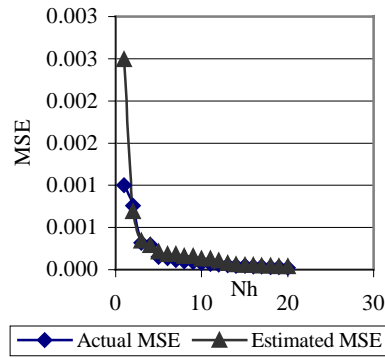


Figure 2 : MLP Sizing - x.tra

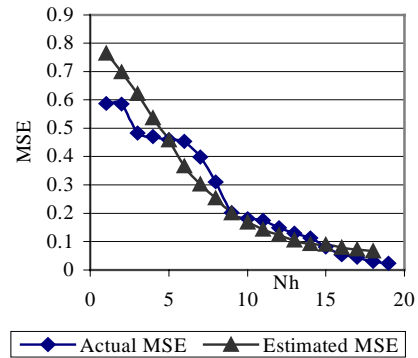


Figure 3 : MLP Sizing - hwang-t3.tra

Kin-8nm.tra : This data is part of the DELVE data set and contains 4918 training patterns. The data is artificial and describes forward kinematics of an 8 link robot arm . The data set is characterized by 8 inputs, 1 output, high non- linearity, and medium noise. For a given value of N_h in figure 4, the estimated training error is off by much less than an order of magnitude.

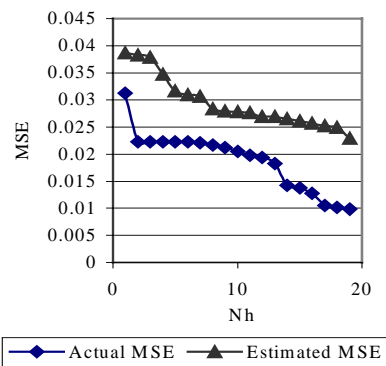


Figure 4 : MLP Sizing -kin-8nm.tra

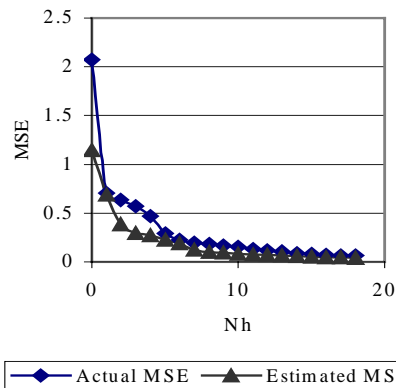


Figure 5 : MLP Sizing - single2.tra

B. Data Sets With Multiple Outputs

Single2.tra : The data set *Single2* has 16 inputs and 3 outputs, and represents the training set for inversion of surface permittivity ϵ , the normalized surface rms roughness $k\sigma$, and the surface correlation length kL found in backscattering models from randomly rough dielectric surfaces [14]. The training set contains 5992

patterns. The 3 outputs can be compressed down to 1 with less than 1% increase in training MSE of an equivalent MLP. The results, illustrated in figure 5, show good performance by the sizing algorithm.

Build3.tra : This file is part of the Proben1 benchmarking data sets. The data set represents the problem of predicting the hourly consumption of electrical energy, hot water, and cold water based on the date, time of day, outside temperature, outside air humidity, solar radiation, and wind speed. There are 8 inputs, and 3 outputs and 2104 training patterns in this data set. All the three outputs are relevant and no compression of the outputs is possible. The predicted and actual E versus N_h curves, illustrated in figure 6, coincide very well.

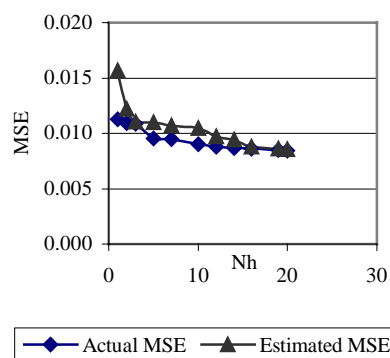


Figure 6 : MLP Sizing - build3.tra

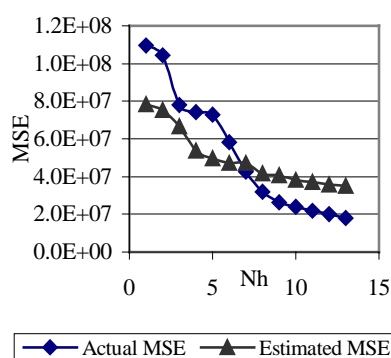


Figure 7 : MLP Sizing - f17.tra

F17.tra : The third multiple-output example is the data set *F17*, which contains 2823 training patterns for onboard *Flight Load Synthesis* (FLS) in helicopters. There are 17 inputs and 9 outputs for each pattern. This data was obtained from the M430 flight load level survey conducted in Mirabel Canada in early 1995 by Bell Helicopter Textron of Fort Worth. The 9 outputs can be compressed down to 2 with less than 1% increase in training MSE of an equivalent MLP. The predicted and actual E versus N_h curves, illustrated in figure 7, are not off by very much.

VI. CONCLUSIONS

In this paper we have described an efficient, accurate and fast method to size an MLP using PWL modular networks. We have shown theoretically that the principal assumption behind our algorithm is a good one, for the case of random patterns. Using several standard benchmark data sets, we have experimentally verified that the sizing algorithm works and is an order of magnitude faster than the brute force design of multiple MLPs. Much work remains to be done. Bounds on the error of our principal assumption need to be found for nonrandom datasets. We need to size networks, from training data, so that testing error is minimized.

Finally, we need to determine when PWL modular networks are acceptable substitutes for the MLP.

Acknowledgement

This work was funded by the Advanced Technology Program of the state of Texas under grant 003656-063.

VI. REFERENCES

- [1] K.Hornik, M.Stinchcombe, and H.White, "Multilayer feedforward networks are universal approximators", *Neural Networks*, Vol. 2, No. 5, pp. 359-366, 1989.
- [2] M.Leshno, V.Lin, A.Pinkus, and S.Schocken, "Multilayer feedforward networks with a nonpolynomial activation function can approximate any function", *Neural Networks*, Vol. 6, No. 6, pp. 861-867, 1993.
- [3] B.Fritzke, "Growing cell structures – A self-organizing network for unsupervised and supervised learning", *Neural Networks*, Vol. 7, No. 9, pp. 1441-1460, 1994.
- [4] J.N.Hwang, S.S.You, S.R.Lay, and I.C.Jou, "Casacade Correlation Learning: A Projection Pursuit Learning Perspective", *IEEE Transactions on Neural Networks*, Vol. 7, No. 2, pp. 278-289, Mar 1996.
- [5] R.Reed, "Pruning Algorithms – A Survey", *IEEE Transactions on Neural Networks*, Vol. 4, No. 5, pp. 740-747, Sep 1993.
- [6] Z.J.Yang, "Hidden-layer size reducing for multilayer neural networks using the orthogonal least-squares method", *Proceedings of the 1997 36th IEEE Society of Instrument and Control Engineers (SICE) Annual Conference*, pp. 1089-1092, July 1997.
- [7] H.Akaike, "A New Look at the Statistical Model Identification", *IEEE Transactions on Automatic Control*, Vol. AC-19, No. 6, pp. 716-723, Dec 1974.
- [8] D.B.Fogel, "An Information Criterion for Optimal Neural Network Selection", *IEEE Transactions on Neural Networks*, Vol. 2, No. 5, pp. 490-497, Sep 1991.
- [9] Hema Chandrasekaran and Michael T. Manry, "Convergent Design of a Piecewise Linear Neural Network", to appear in *International Joint Conference on Neural Networks (IJCNN'99)*, July 1999.
- [10] S.Subbarayan, K.Kim, M.T.Manry, V.Devarajan and H.Chen, "Modular Neural Network Architecture using Piecewise Linear Mapping", *Thirtieth Asilomar Conference on Signals, Systems & Computers*, Vol. 2, pp 1171-1175, Nov 1996.
- [11] A.Gopalakrishnan, X.Jiang, M.S.Chen, and M.T.Manry, "Constructive Proof of Efficient pattern Storage in the Multilayer Perceptron", *Twentyseventh Asilomar Conference on Signals, Systems & Computers*, Vol. 1, pp 386-390, Nov 1993.
- [12] A.Lapedes and R.Farber, "Nonlinear signal processing using neural networks: prediction and system modelling", *Technical Report LA-UR-87-2662*, Los Alamos National Laboratory, Los Alamos, NM, 1987.
- [13] Hwang, J.-N., Lay, S.-R., Maechler, M., Martin, R. D., and Schimert, J. "Regression modeling in back-propagation and projection pursuit learning", *IEEE Transactions on Neural Networks*, vol. 5, no. 3, pp. 342-353, 1994.
- [14] A.K.Fung, Z.Li, and K.S.Chen, "Backscattering from a randomly rough dielectric surface", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 30, No. 2, pp. 356-369, 1992.
- [15] A.Papoulis, *Probability, Random Variables, and Stochastic Processes*, McGraw-Hill, 1991.