TRAINABLE FUNCTIONAL LINK NEURAL NETWORK ARCHITECTURE

BASKIN I. TAPKAN & BOGDAN M. WILAMOWSKI
University of Wyoming, Laramie, WY 82071-3295
e-mail: baskin@uwyo.edu, wilam@uwyo.edu

ABSTRACT:

The difficulty in some pattern classifications and slowness of error back-propagation algorithm cause serious problems during training such as very long training period or not converging into the desired output space. In this paper with the trainable functional link neural network architecture we introduce new nonlinear terms as inputs to the output layer. These nonlinear terms are neurons with sigmoidal or Gaussian type of outputs which are fed simultaneously with the input pattern features to the output layer. We calculate a global error between the desired and the actual outputs using some learning rule, for example delta learning rule, then an extra neuron is connected by those input pattern features and trained to suppress this error. Additional neurons are added and the same procedure is followed so that a tolerable error is achieved at the output neurons. The proposed network was compared with the error back-propagation algorithm and the Cascade Correlation architecture. Many typical test benchmarks were used for comparison. The method is slightly better than the Cascade Correlation, but significantly outperforms the error backpropagation algorithm.

INTRODUCTION:

The slowness of the error backpropagation neural network training algorithms for the real world problems is a major drawback for the practical applications of neural networks. With the increasing dimension of the input space, the training becomes a tougher problem. In the literature several approaches are explored where various authors have tried to solve this problem with new training algorithms employing new ideas [1 - 11]. These techniques are called as constructive or growth algorithms. Some of these techniques rely on the idea of classification for as many patterns as possible at each iterative step, thus keeping each pattern correctly classified. Different solutions such as exploiting the graph theory [2] or learning in the neural sense [1, 4], and has been used and significant results have been accomplished. In this paper the trainable functional link neural network (TFLNN) learning is performed in the conventional neural sense. The nonlinear terms for the proposed neural network are the hidden neuron units, thus giving the name functional link neural networks [3]. A modified regression algorithm is adopted for the training purposes. The nonlinearly separable patterns such as the N-parity problems have been attempted for the TFLNN to solve. In this network structure new nonlinear hidden

neuron units with sigmoidal type of activation function is put to use and connected to the output layer after the training. The cascade correlation algorithm is investigated separately because of the similarities between the TFLNN and the results have been used for comparison.

NEURAL NETWORK ARCHITECTURE:

Input and Output Pattern Space: Input space could be any analog real value which could be defined by a training set of M patterns $P_r(r = 1,2,..., M)$. The rth pattern P_r is an input-output pair classified by N values $X_i^{(r)} = \{X_1, X_2,..., X_N\}$ where $X_i^{(r)} \in R^N$ of the input variables and the corresponding desired output set as $o_r = \{-1, 1\}$. The input space as defined above is an N dimensional real space ($X_i^{(r)} = \left\{X_1^{(r)}, X_2^{(r)},..., X_N^{(r)}\right\} \in R^N$).

A sigmoidal type of activation function is used in the network neurons to limit the output between plus or minus one,as well as the error modification which is defined as

$$E_{mj} = \tanh\left[k \sum_{j=1}^{o} \left(E_j - \overline{E}_j\right)\right] \tag{1}$$

where E_j is the local error, $\overline{E}_j$ is the mean error for the jth input pattern at the output layer, o is the actual output of main layer (output layer)and k is the gain. The reason for doing the above error modification is limiting so that the desired output is created to train the hidden unit over the existing input pattern space. This is done only once for one hidden neuron unit. A typical Functional Link Neural Network is depicted in Figure 1.

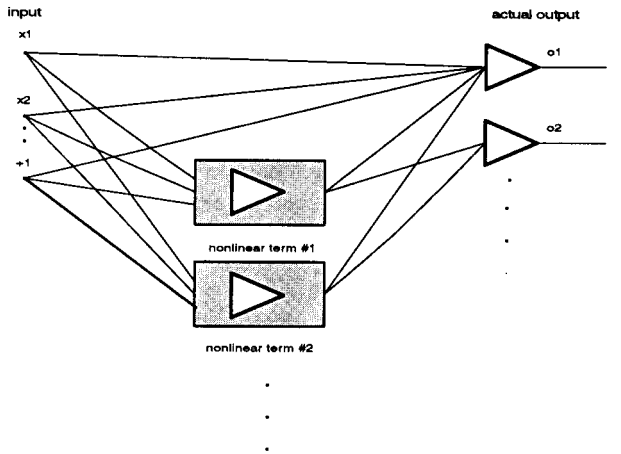

Figure 1: Functional Link Neural Network.

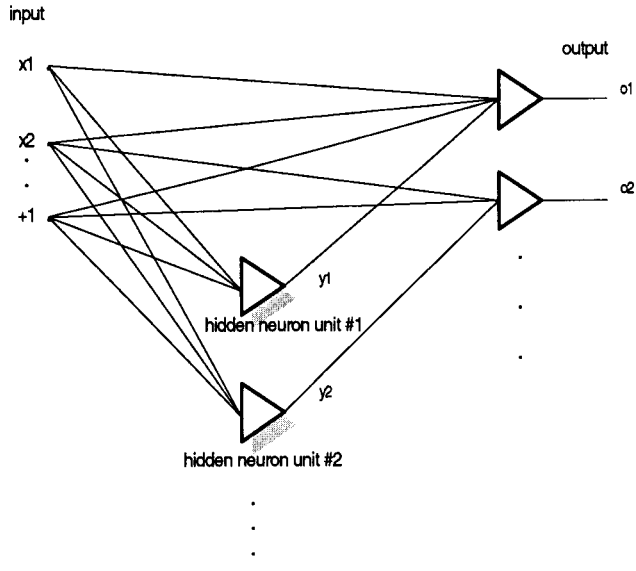

Figure 2: TFLNN Architecture.

Algorithm is very similar to the cascade correlation [1] with some exceptions. Unlike the Cascade Correlation the neurons are added to the same layer where one neuron form different layer at the each addition of hidden unit. Also the modified error in equation (1) is used to train the hidden neuron units where in the cascade correlation the correlation between the actual output and the desired outputs are used for training.

With the presented approach only single layer or single neuron is needed to be trained at any time. Many fast algorithms can be applied for that purpose including the quickprop (Fahlman), in our case we have used the implemented modified regression algorithm (Anderson). The basic scheme of TFLNN architecture is in Figure 2.

First, the main layer is trained using the modified regression training algorithm and error at the output layer is computed. This error is modified using the Equation (1). A hidden neuron unit is trained separately taking the modified error as its target output using the same training algorithm as mentioned above. Then this unit is connected to the main layer as an input and training is repeated. In practice a pool of candidate hidden neuron units are generated, trained separately and the best one is chosen as the winner which will be connected to the main layer. The weights of these hidden neuron units after connection are not altered and they become a permanent part of the structure. The flow chart for the algorithm is depicted in Figure 3.

After the training is completed the network is permanently constructed. The TFLNN architecture is one hidden layer network comprising additional nonlinear operators which generates extra inputs for the training. Separate training procedure is used for these neurons and neurons are added in the hidden layer as needed.

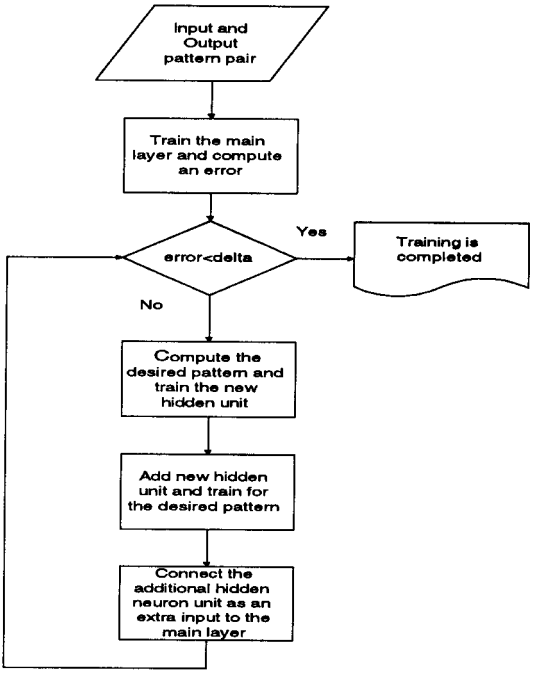

Figure 3. The training algorithm for the TFLNN Architecture.

EXAMPLES:

The nonlinearly separable problems are chosen to check the performance of TFLNN. The XOR problem is solved with the addition of only one hidden neuron unit. The neural network solution for the XOR problem with weights are shown in Figure 4. The total computation time for training is only three neurons once or three times the time of one single neuron. The results are shown in Table 1.

For the N-parity problem, three to eight bit parity problems have been solved using this algorithm.. Several simulations have been run using both the Cascade Correlation and the TFLNN. For the TFLNN the quicker conversion time then the Cascade Correlation is observed. The reason for this faster conversion time is the training time of the hidden neuron units where in the TFLNN each hidden neuron unit is trained over the same existing input pattern, however the input pattern space in the Cascade Correlation for the hidden neuron units increases proportionally at the each addition of a hidden neuron unit. In the case of backpropagation with the N-parity problems, the TFLNN is almost thousand times faster. However, as seen on the Table 2, TFLNN requires greater number of hidden neuron units with the increasing dimensionality. But this can be a trade off between the converging time and number of hidden neuron units in certain neural network applications.

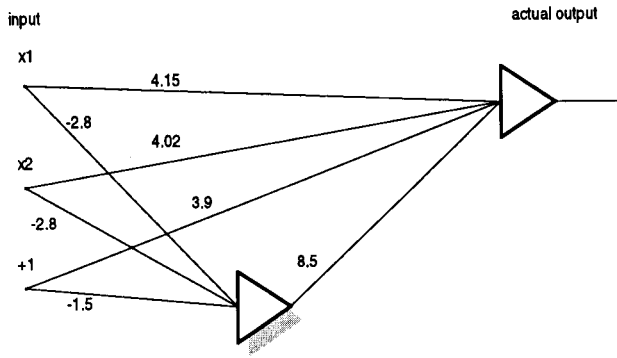

Figure 4. The XOR Problem

TABLE 1: THE RESULTS FOR THE XOR PROBLEM				
Patterns	x1	x2	Actual Value	Desired Value
1	-1	-1	0.9962	1
2	-1	1	-0.9979	-1
3	1	-1	-0.9959	-1
4	1	1	0.9975	1

TABLE 2. COMPARISON OF N-PARITY PROBLEM BETWEEN THE CASCADE CORRELATION AND TFLNN			
The Parity N	Number of Cases	The Cascade Correlation	TFLNN
2	4	1	1
3	8	1	1
4	16	2	3
5	32	2 - 3	4
6	64	3	5 - 10
7	128	4 - 5	8 - 14
8	256	4 - 5	10 - 18

CONCLUSION:

The most significant advantage we have noted with the TFLNN architecture is the converging time and its easiness for the application. With the given problem the open questions on number of hidden layers or number of hidden neuron units are resolved because the network decides on those issues by checking the error after each forward training where a lot of focus is given in the literature. When the error is in the tolerance limits which are set by the supervisor, the network stops the training automatically. Further studies employing the presented architecture are suggested with different training algorithms and input-output pattern pairs.

REFERENCES

Fahlman,S.E., and Lebiere,C., (1990). The Cascade Correlation Learning Architecture *Adv. Ner. Inf. Proc. Syst., 2, D. S. Touretzky Ed., Los Altos, CA: Morgan, Kaufmann* pp. 524-532.

Mascioli,F.M.F., and Martinelli, G., (1995). A Constructive Algorithm for Binary Neural Networks: The Oil-Spot Algorithm, *IEEE Trans. Neural Network, Vol6,* pp.794-797.

Poa,Y.H., (1989). Adaptive Pattern Recognition and Neural Networks, *Addison-Wesley.*

Zurada, J.M., (1992) Introduction to Artificial Neural Systems, *St. Paul West Publishing.*

Philip Chen,C.L., and LeClair,S.R., (1994). An Efficient Supervised Learning Neural Network: An Architecture and Algorithm, *Intelligent Engineering Systems Through Artificial Neural Networks.,* Vol 4, pp. 179-184.

Sugaya,F., and. Yatsuzuka,Y., (1994). A Parallel Binary Neural Network With Very Fast Learning *Intelligent Engineering Systems Through Artificial Neural Networks,* Vol 4, pp. 185-190.

Gallant, S.I., (1988). Perceptron Based Algorithms, *IEEE Trans. Neural Networks,* pp 179-191.

Anderson,T.J., and Wilamowski,B.M., (1995). A Modified Regression for Fast One-layer Neural Network Training., *World Congress on Neural Networks in Washington D.C.*

Mozer, M.C., (1989). Using Relevance to Reduce Network Size Automatically, *Connection Science* pp. 3-16.

Barman,F., and Biegler-Kanig,F., (1992). Class of Efficient Neural Network Algorithms for Neural Networks, *Neural Networks pp. 139-144.*

Hubel,N.F., and Hwarng,H.B., (1994). A Neural Network Model and Multiple Linear Regression: Another Point of View, *Intelligent Engineering Systems Through Artificial Neural Networks, Vol 4, pp. 199-204.*

INTELLIGENT ENGINEERING SYSTEMS THROUGH ARTIFICIAL NEURAL NETWORKS

VOLUME 5

FUZZY LOGIC AND EVOLUTIONARY PROGRAMMING

Editors:

Cihan H. Dagli
Metin Akay
C. L. Philip Chen
Benito R. Fernández
Joydeep Ghosh