Proceedings of the 17th World Congress
The International Federation of Automatic Control
Seoul, Korea, July 6-11, 2008

Channel Equalization Using Dynamic Fuzzy Neural Networks

Ming-Bin Li and Meng Joo Er

Intelligent Systems Centre, Nanyang Technological University, Nanyang Avenue, Singapore 639798 (e-mail: {mbli,EMJER}@ntu.edu.sg).

Abstract: In this paper, a dynamic fuzzy neural network (DFNN) is applying for communication channel equalization problem. By combining fuzzy rules with the learning ability of neural networks, DFNN can achieve the advantages of both fuzzy logic and neural networks. The simulation results show that DFNN equalizer is superior to other equalizers such as recurrent neural network (RNN) and minimal resource allocation networks (MRAN) in terms of bit error rate (BER).

1. INTRODUCTION

It is well known that in high-speed digital communication systems, the channel distorts the transmitted symbols in both amplitude and phase thereby causing interferences between adjacent symbols. Fig. 1 shows a standard baseband-equivalent model of a communication system and the channel output x_n is given below:

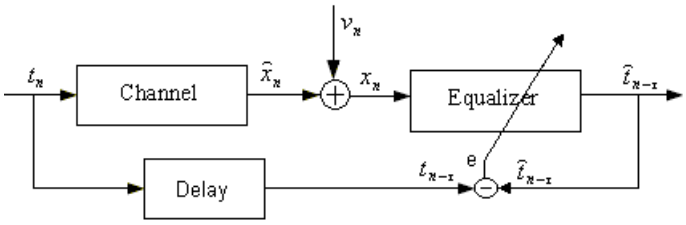

Fig. 1. Discrete model of a communication system

$$x_n = \sum_{i=1}^{n_h-1} h_i t_{n-i} + v_n \tag{1}$$

where n_h is the channel order. The signal sequence $\{t_n\}$ is passed through the linear/nonlinear channel; the channel noise-free output $\hat{x}_n$ is corrupted by additive zero-mean Gaussian noise v_n to generate the equalizer input signal x_n; after equalization, the equalizer output signal $\hat{t}_{n-\tau}$ is compared with the delayed channel input signal $t_{n-\tau}$ and the error is used to adjust the free parameters of the proposed equalizer. In Brief, the purpose of the equalizer is to reconstruct the transmitted signal $t_{n-\tau}$ based on the received symbol sequence $X_n = [x_n, x_{n-1}, \cdots, x_{n-m+1}]^T$, where τ is the equalizer decision delay and m is the equalizer input dimension.

Because neural networks have very good mapping and classification ability (Huang et al. (2000)), many types of neural network have been applied for equalization problems, such as Multi-Layer Perceptions (MLP), Radial Basis Function (RBF) networks and Recurrent Neural Networks (RNN) (Chen et al. (1990, 1991, 1993); Kechriotis et al. (1994); Kumar et al. (2000); Li et al. (2000); Parisi et al. (1997)). Chen et al. (1990) develop an adaptive equalizer using multilayer perceptron to overcome channel non-linearities and additive noise correlation. They also investigated a radial basis function neural network (RBFNN) equalizer to reconstruct binary signals in a dispersive channel and showed that the RBF nonlinear equalizer can realize optimal equalization and is beneficial in practical implementation (Chen et al. (1991, 1993)). Recurrent neural networks equalizer has gained great attention because of its feedback property and is developed by Kechriotis et al. (1994) and Parisi et al. (1997), respectively. Kechriotis et al. (1994) propose an adaptive RNN equalizer for linear and nonlinear channels. Another fast adaptive RNN digital equalizer is presented by Parisi et al. (1997), which is superior to the Kechriotis' RNN equalizer in terms of bit error rate (BER) and training samples. Kumar et al. (2000) present a RBF equalizer using minimal resource allocation network (MRAN) and evaluate performances of the proposed equalizer in different channels for 2-PAM and 4-QAM signals. The equalizer using the function link artificial neural networks (FLANN) is developed by Weng et al. (2004) and Yen et al. (2004). The performance of FLANN equalizer is compared with that of MLP equalizer and linear least-mean-square-based equalizer in several channels for QAM signals. Recently, self-organizing neural networks have generated great interest in many researchers. One sequential learning algorithm termed growing and pruning RBF (GAP-RBF) network is applied to solve the channel equalization problem (Li et al. (2000)). Using the growing and pruning criterion, the number of the hidden layer neurons is dynamically adjusted to achieve the compact network structure. Simulation results show that the GAP-RBF equalizer is superior to some existing neural-networks-based equalizers in term of BER and equalizer complexity.

In the past few years, fuzzy logic has gained great interest in various applications because fuzzy systems can approximate any continuous function on a compact set to any accuracy. It is well known that fuzzy logic incorporates a simple "IF-Then" rule-based approach to solve a control problem rather than modeling a system mathematically, which needs expert experiences in the design. However,

the designer will encounter great difficulty in conventional fuzzy system design when the system is too complicated to extract an appropriate number of fuzzy rules. Recently, a Takagi-Sugeno-Kang (TSK) fuzzy system implementing radial basis function (RBF) neural networks, termed dynamic fuzzy neural networks (DFNN), is proposed by Wu et al. (2000). By combining fuzzy rules with the learning ability of neural networks, the DFNN can achieve the advantages of both fuzzy logic and neural networks. The key idea of the DFNN is that the system starts with no hidden units and dynamically adds and deletes neurons according to their significance to system performance. A parsimonious structure can be achieved by self-adaptive learning algorithm. Many applications of the DFNN have been accomplished since it was proposed Er et al. (2005,?); Wu et al. (2001). In this paper, the DFNN is applied to the channel equalization problem and its performance is evaluated in several different channel models. Simulation results show that the DFNN equalizer can achieve superior performance compared with the Bayesian, MRAN and adaptive RNN equalizers (Parisi et al. (1997)) in terms of BER.

This paper is organized as follows. Section 2 briefly introduces the architecture of DFNN. Section 3 introduces the learning algorithm of DFNN. Section 4 shows a performance comparison of the DFNN equalizer with the Bayesian, MRAN and RNN equalizers. Section 5 concludes the paper.

2. ARCHITECTURE OF DYNAMIC FUZZY NEURAL NETWORKS

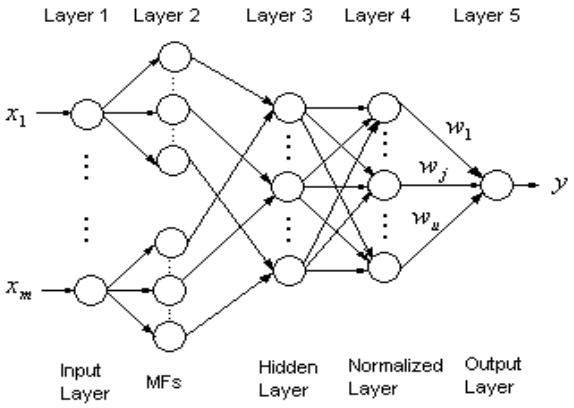

Fig. 2. Architecture of dynamic fuzzy neural networks

Fig. 2 depicts the architecture of DFNN. Consider a series of training samples $(\mathbf{X}_i, t_i)$, $i = 1, 2, \cdots, n$, where $\mathbf{X}_i = [x_{i1}, x_{i2}, \cdots, x_{im}]^T \in R^m$. The architecture of the DFNN is made up of five layers:

Layer 1: Input layer. Each node represents an input linguistic variable $x_i, i = 1, 2, \cdots, m$.

Layer 2: Each input variable x_i, has u membership functions μ_{ij}, which is in the form of a Gaussian function, i. e.

$$\mu_{ij} = \exp\left[-\frac{(x_i - c_{ij})^2}{\sigma_j^2}\right] \quad i = 1, \ldots, m \quad j = 1, \ldots, u \tag{2}$$

where c_{ij}, σ_j are the center and width of the Gaussian function and u is the number of membership functions, respectively.

Layer 3: Each node represents a possible IF-part for fuzzy rules. If the T-norm operator is chosen as multiplication to calculate each rule's firing strength, the output of the jth rule R_j is given by

$$R_j = \exp\left[-\frac{\sum_{i=1}^{m}(x_i - c_{ij})^2}{\sigma_j^2}\right] = \exp\left[-\frac{\|(X - C_j)\|^2}{\sigma_j^2}\right] \quad j = 1, \ldots, u \tag{3}$$

Layer 4: In this layer, the outputs from the previous layer are normalized to the interval $[0, 1]$.

$$a_j = \frac{R_j}{\sum_{k=1}^{u} R_k} = \frac{\exp\left[-\frac{\|(X-C_j)\|^2}{\sigma_j^2}\right]}{\sum_{k=1}^{u} \exp\left[-\frac{\|(X-C_k)\|^2}{\sigma_k^2}\right]} \tag{4}$$

Layer 5: This is the output layer. We have

$$y(X) = \sum_{j=1}^{u} w_j a_j \tag{5}$$

For the TSK model, w_j can be expressed as follows:

$$w_j = k_{j0} + k_{j1}x_1 + \cdots + k_{jm}x_m \tag{6}$$

For n training samples, the output of layer 4 with u fuzzy rules can be denoted by:

$$\psi = \begin{bmatrix} a_{11} & \ldots & a_{1n} \\ \vdots & \vdots & \vdots \\ a_{u1} & \ldots & a_{un} \end{bmatrix} \tag{7}$$

then, Eq.(5) can be rewritten in a more compact form

$$Y = W\Psi \tag{8}$$

where

$$W = [k_{10}, \ldots, k_{u0}, k_{11}, \ldots, k_{u1}, \ldots, k_{1m}, \ldots, k_{um}] \tag{9}$$

$$\Psi = \begin{bmatrix} a_{11} & \ldots & a_{1n} \\ \vdots & \vdots & \vdots \\ a_{u1} & \ldots & a_{un} \\ a_{11}x_{11} & \ldots & a_{1n}x_{1n} \\ \vdots & \vdots & \vdots \\ a_{u1}x_{11} & \ldots & a_{un}x_{1n} \\ \vdots & \vdots & \vdots \\ a_{11}x_{m1} & \ldots & a_{1n}x_{mn} \\ \vdots & \vdots & \vdots \\ a_{u1}x_{m1} & \ldots & a_{un}x_{mn} \end{bmatrix} \tag{10}$$

where a_{jk} is the normalized output from layer 4, $j = 1, \ldots, u, k = 1, \ldots, n$, n is the number of training data. The optimal coefficient vector W^* can be easily solved by the well-known linear least squares (LLS) method

$$W^* = Y \cdot (\Psi^T\Psi)^{-1}\Psi^T \tag{11}$$

3. LEARNING ALGORITHM OF DFNN

3.1 Criteria of neuron growing

For the ith observation (X_i, t_i), calculate the DFNN output error e_i and the distance $d_i(j)$ as follows:

$$\begin{aligned} \|e_i\| &= \|t_i - y_i\| \\ d_i(j) &= \|X_i - C_j\| \quad j = 1, \ldots, u \end{aligned} \tag{12}$$

If

$$\begin{aligned} \|e_i\| &> k_e \\ d_{\min} &> k_d \end{aligned} \tag{13}$$

where $d_{\min} = \arg\min(d_i(j))$ and k_e, k_d are two predetermined parameters which are chosen as follows.

$$\begin{aligned} k_e &= \max[e_{\max} \times \beta^i, e_{\min}] \quad 0 < \beta < 1 \\ k_d &= \max[\hat{d}_{\max} \times \gamma^i, \hat{d}_{\min}] \quad 0 < \gamma < 1 \end{aligned} \tag{14}$$

A new neuron is added to the DFNN and the center and width for the new generated neuron are set as

$$\begin{aligned} C_i &= X_i \\ \sigma_i &= k \times d_{\min} \end{aligned} \tag{15}$$

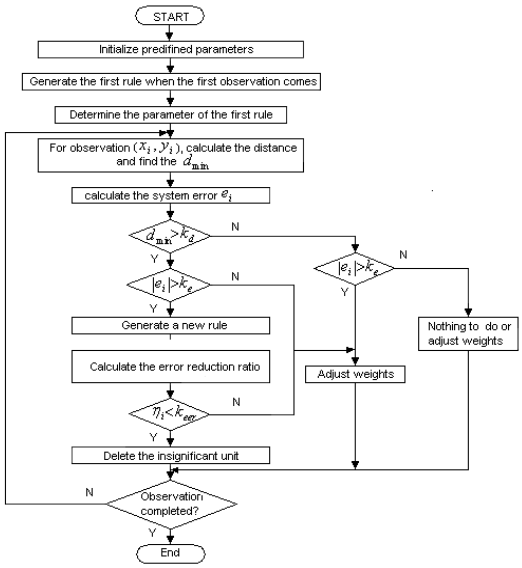

Fig. 3. Learning algorithm of dynamic fuzzy neural networks

3.2 Criteria of neuron pruning

In order to realize a compact network structure, inactive hidden neurons should be detected and removed during the learning progress. First, the error reduction ration (ERR) matrix $ERR = (\delta_1, \delta_2, \ldots, \delta_u) \in R^{(m+1)\times u}$ is calculated (please see Wu et al. (2000) for details of ERR matrix calculation).

If

$$\eta_i = \sqrt{\frac{\delta_i^T \delta_i}{m+1}} < k_{err} \tag{16}$$

The ith hidden neuron is inactive and should be deleted, where k_{err} is a prespecified threshold. Detailed descriptions of the learning algorithm are shown in Fig. 3.

4. SIMULATION RESULTS

The performance of the DFNN equalizer for 2-PAM signals is evaluated for three different channel models used in Kechriotis et al. (1994) and Parisi et al. (1997). In example 1 and example 2, the simulation environment is set as the same as that of Parisi et al. (1997) in order to compare with its results directly. 10^6 testing data were used to calculate the bit error rate (BER) after the a total of equalizer is trained at different signal-to-noise ratio (SNR).

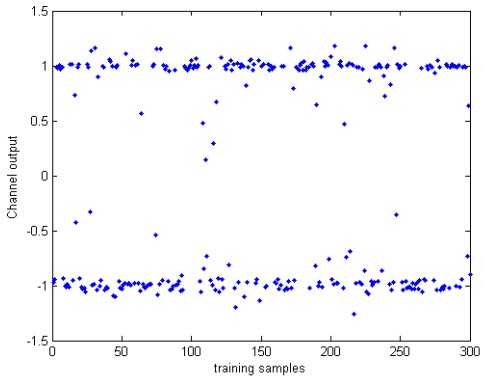

Fig. 4. equalizer output (Example 1)

Example 1: In this example, the third-order nonminimum-phase channel model is used to evaluate the DFNN equalizer performance. The channel transfer function (Kechriotis et al. (1994), Parisi et al. (1997)) is given by

$$H_2(z) = 0.3482 + 0.8704z^{-1} + 0.3482z^{-2} \tag{17}$$

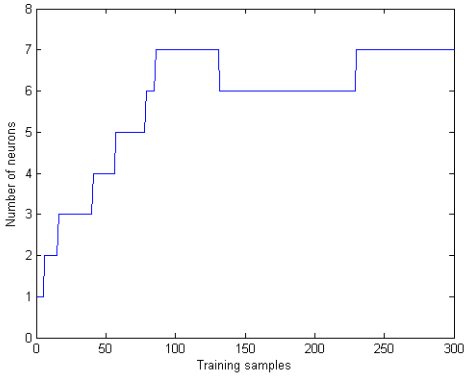

Fig. 5. Fuzzy rules generation (Example 1)

The input dimension of the DFNN equalizer was set to $m = 1$ and the equalizer decision delay was $\tau = 1$. The DFNN network parameters are set as follows: $\hat{d}_{\max} = 4, \hat{d}_{\min} = 0.3, e_{\max} = 1, e_{\min} = 0.02, k_{err} = 0.002, k = 0.9, \beta = 0.9, \gamma = 0.97$. The activation functions are chosen Gaussian functions for the MRAN, GAP-RBF and RBFNN equalizers and hyperbolic tangent function for the RNN equalizer. The RNN equalizer uses 3 units in the simulation in Kechriotis et al. (1994).

The DFNN equalizer is trained with 300 samples at different SNR and Fig. 4 is the DFNN equalizer output at 10dB SNR. Seven fuzzy rules have been generated by the

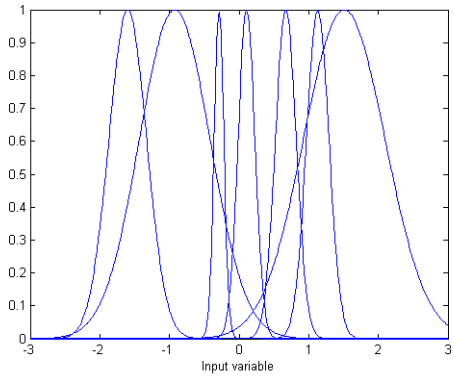

Fig. 6. Membership functions of input variable (Example 1)

DFNN equalizer at the end of the training process shown in Fig. 5. The membership functions of the input variable are shown in Fig. 6. Fig. 7 shows the curves of the BER versus SNR for Bayesian, MRAN, RNN and DFNN equalizers. As shown in Fig. 7, the Bayesian equalizer (star line) attains the best BER performance. Using the same training samples, the DFNN equalizer(circle line) always produces superior performance than the MRAN equalizer (dot line), the RNN equalizer (plus line), the GAP-RBF equalizer (square line) and the RBFNN equalizer (diamond line).

The fuzzy rules are listed as follows:

Rule 1: If x is $A(0.9, 0.7)$, then $t = -1.7 - 0.8x$.
Rule 2: If x is $A(1.5, 0.8)$, then $t = 0.1 + 0.6x$.
Rule 3: If x is $A(0.1, 0.2)$, then $t = 0.5 + 5x$.
Rule 4: If x is $A(1.1, 0.2)$, then $t = 1.3$.
Rule 5: If x is $A(0.7, 0.2)$, then $t = 0.9 + 0.5x$.
Rule 6: If x is $A(-0.3, 0.1)$, then $t = -0.5 + 0.7x$.
Rule 7: If x is $A(-1.6, 0.4)$, then $t = -2.8 - 0.9x$.

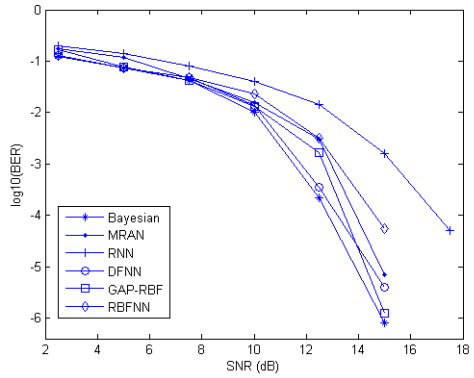

Fig. 7. Error probability. *: Bayesian. •: MRAN. +: RNN. o: DFNN. □: GAP-RBF. ◇: RBFNN equalizers (Example 1)

Example 2: In this case, a more complicated channel model will be used to evaluate the performance of the DFNN equalizer. The nonlinear channel transfer function (Kechriotis et al. (1994), Parisi et al. (1997)) can be given by

$$H_2(z) = 0.3482 + 0.8704z^{-1} + 0.3482z^{-2}$$
$$x_n = \hat{x}_n + 0.2\hat{x}^2 + v_n \quad (18)$$

where $\hat{x}_n$ is the linear noise-free channel output and v_n is the zero mean Gaussian white noise. The input dimension of the DFNN equalizer was set to $m = 1$ and the equalizer decision delay was $\tau = 1$. The DFNN network parameters are set as follows: $\hat{d}_{\max} = 4, \hat{d}_{\min} = 0.3, e_{\max} = 1, e_{\min} = 0.02, k_{err} = 0.002, k = 0.9, \beta = 0.9, \gamma = 0.97$.

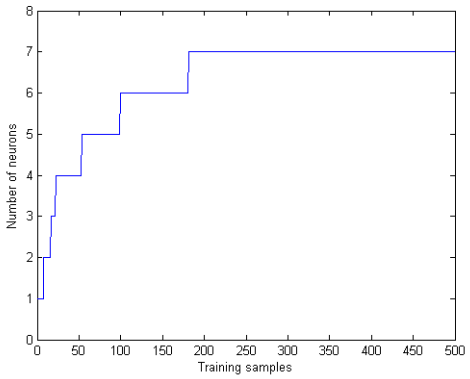

Fig. 8. Fuzzy rules generation (Example 2)

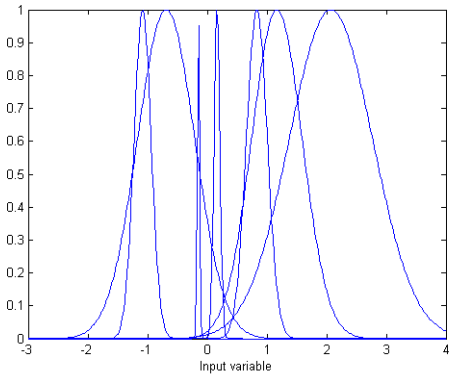

Fig. 9. Membership functions of input variable (Example 2)

The DFNN equalizer is trained at different SNR with 500 training samples. Fig. 8 and Fig. 9 show the generation of fuzzy rules and membership functions of the input variable, respectively. The fuzzy rules are listed as follows:

Rule 1: If x is $A(-0.7, 0.7)$, then $t = -0.9 + 0.2x$.
Rule 2: If x is $A(2.1, 1.0)$, then $t = 4.9 - 1.7x$.
Rule 3: If x is $A(0.2, 0.1)$, then $t = 2.6 - 7.5x$.
Rule 4: If x is $A(1.2, 0.6)$, then $t = 2.7 - 2.3x$.
Rule 5: If x is $A(-0.2, 0.02)$, then $t = -1.0 + 1.3x$.
Rule 6: If x is $A(0.8, 0.3)$, then $t = -1.3 + 2.1x$.
Rule 7: If x is $A(-1.1, 0.2)$, then $t = -0.5 + 0.4x$.

After the training process, 1000 000 test data at various SNRs were used for the BER evaluation. The BER results are compared with other five equalizers shown in Fig. 10. It can be seen from the figure that the DFNN equalizer

attains better BER performance than other equalizers, except Bayesian equalizer.

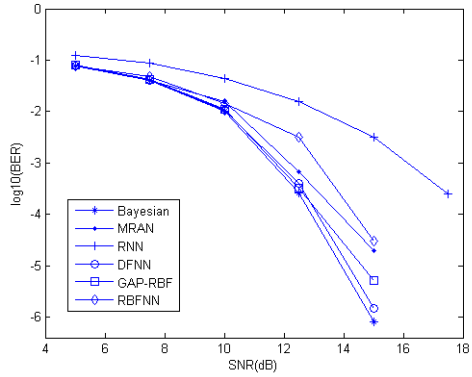

Fig. 10. Error probability. *: Bayesian. •: MRAN. +: RNN. o: DFNN. □: GAP-RBF. ◇: RBFNN equalizers (Example 2)

Example 3: The nonlinear channel model used in Kechriotis et al. (1994) and Kumar et al. (2000) is chosen for simulation studies. The channel transfer function is given by

$$H_1(z) = 1 + 0.7z^{-1}$$
$$x_n = \hat{x}_n + \hat{x}^2 + 0.7\hat{x}^3 + 0.5\hat{x}^4 + v_n \tag{19}$$

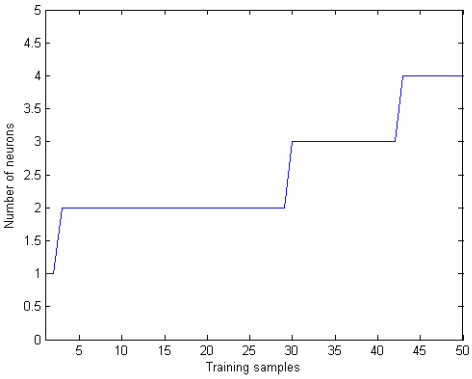

Fig. 11. Fuzzy rules generation (Example 3)

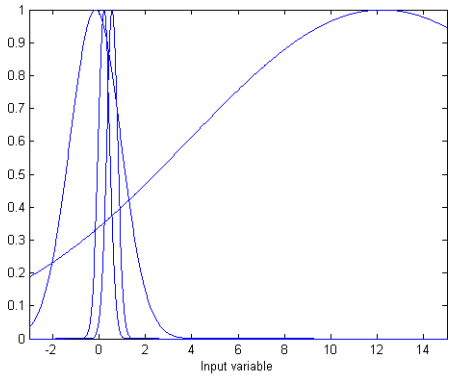

Fig. 12. Membership functions of input variable (Example 3)

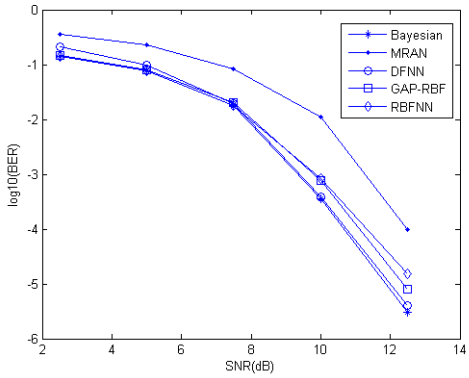

Fig. 13. Error probability. *: Bayesian. •: MRAN. o: DFNN. □: GAP-RBF. ◇: RBFNN equalizers (example 3)

The DFNN equalizer is trained with 50 training samples at 10dB SNR. The growth progress of the fuzzy rules during the training are given in Fig. 11. Fig. 12 shows the membership functions of the input variable. A total of 1000 000 test data at various SNRs were used for the BER evaluation. A comparison between the Bayesian, the DFNN and the MRAN equalizer in terms of BER is shown in Fig. 13. It can be seen from the figure that the DFNN equalizer is superior to the MRAN, the RBFNN and the GAP-RBF equalizers.

The fuzzy rules are listed as follows:

Rule 1: If x is $A(-0.2, 1.5)$, then $t = -0.8 - 2.7x$.

Rule 2: If x is $A(12.2, 11.7)$, then $t = 0.9 + 0.01x$.

Rule 3: If x is $A(0.2, 0.3)$, then $t = 1.4 - 27.7x$.

Rule 4: If x is $A(0.6, 0.3)$, then $t = 15.8 - 20.0x$.

4.1 Discussions

	Algorithms	Neuron	equalization time	
			training(s)	testing (ms/sample)
Example1	DFNN	7	0.206	0.119
	GAP-RBF	6	0.389	0.083
	MRAN	8	0.732	0.107
	RBFNN	8	0.089	0.132
	Bayesian	8	–	0.094
Example2	DFNN	7	0.234	0.104
	GAP-RBF	5	0.577	0.072
	MRAN	6	1.307	0.083
	RBFNN	8	0.112	0.136
	Bayesian	8	–	0.094
Example3	DFNN	4	0.031	0.108
	GAP-RBF	4	0.108	0.061
	MRAN	4	0.203	0.061
	RBFNN	4	0.018	0.068
	Bayesian	4	–	0.049

Table 1. Complexities of equalizers

From the simulation results given in the above section, we can see that the DFNN equalizer can obtain better equalization performance than other equalizers, except Bayesian equalizer. This is because the Bayesian equalizer

uses the noise-free channel states as its centers and the desired output as its priori probability, which can achieve an optimal solution. The limitation of Bayesian method is that the channel model needs to be exactly known, which is difficult to achieve in real-world applications. The RBFNN proposed by Chen et al. (1993) uses a supervised κ-means clustering procedure to eliminate the noise effect so that the RBF centers can converge to the desired states. The properties of the clustered centers will directly determine the equalization performance. For example 1 and example 2, the channel models are complicated which is the reason why RBFNN equalizer cannot perform better than the DFNN, GAP-RBF and MRAN equalizers in terms of BER. For example 3, the channel model is simple and the RBFNN equalizer can obtain almost the same equalization accuracy as the DFNN and the GAP-RBF equalizers. During the training, the RBFNN weights linking the hidden layer to the output layer are adjusted with the least mean square (LMS) algorithm. From the Table 1, it can be seen that the RBFNN can achieve the fastest learning speed in all methods by virtue of the simple training algorithm.

The DFNN, GAP-RBF and MRAN equalizers are all self-constructing neural networks and they use growing and pruning criterion to search the compact network structure. At the end of the training process, they generated almost the same number of hidden neurons as the Bayesian method which is the optimal solution (Table 1). For BER performance, the DFNN equalizer is the best in these three equalizers because it combines the advantage of fuzzy rules with the learning ability of neural networks. The linear least square (LLS) method used to determine the output weights enables the DFNN achieve global generalization property quickly and directly. Though the GAP-RBF and the MRAN equalizers can achieve the compact network structure, the output weights are modified using the extended Kalman filter (EKF) method which leads to more time and cost than the DFNN equalizer during the training (Table 1).

It can be seen from Table 1 that there is not much difference between the equalizer time of all equalizers. This is because the equalizer time (testing time) is mainly affected by the number of hidden neurons.

5. CONCLUSIONS

In this paper, channel equalization with 2-PAM signals is attempted by using the DFNN. A performance evaluation of the DFNN has been carried out using several channel models with increasing complexity. Simulation results show that the DFNN equalizer is superior to the MRAN, RBFNN, GAP-RBF and RNN equalizers in terms of BER.

REFERENCES

Chen S., G. J. Gibson, C. F. N. Cowan and P. M. Grant. Adaptive equalization of finite non-linear channels using multilayer perceptrons. *Signal Processing*, 20:107–119, 1990.

Chen S., G. J. Gibson, C. F. N. Cowan and P. M. Grant. Reconstruction of binary signals using an adaptive radial-basis function equalizer. *Signal Processing*, 22(1):77–93, 1991.

Chen S., B. Mulgrew, and P. M. Grant. A clustering technique for digital communications channel equalization using radial basis function networks. In *IEEE Transactions on Neural Networks*, 4(4):570–579, 1993.

Er M. J., Z. R. Li, H. N. Cai and Q. Chen. Adaptive Noise Cancellation Using Enhanced Dynamic Fuzzy Neural Networks. In *IEEE Transactions on Fuzzy Systems*, 13(3):331–342, 2005.

Er M. J. and D. Chen. Obstacle Avoidance of a Mobile Robot Using Hybrid Learning Approach. In *IEEE Transactions on Industrical Electronics*, 52(3):898–905, 2005.

Huang G. -B., Y. Q. Chen and H. A. Babri. Classification ability of single hidden layer feedforward neural networks. *IEEE Transactions on Neural Networks*, 11(3),799–801, 2000.

Kechriotis G., E. Zervas, and E. S. Manolakos. Using recurrent neural networks for adaptive communication channel equalization. *IEEE Transactions on Neural Networks*, 5(2):267–278, 1994.

Kumar P. C., P. Saratchandran, and N. Sundararajan. Minimal radial basis function neural networks for nonlinear channel equalisation. *IEE Proceedings, Vision, Image and Signal Processing*, 147(5):428–435, 2000.

Li M. B., G. -B. Huang, P. Saratchandran, and N. Sundararajan. Performance evaluation of GAP-RBF network in channel equalization. *Neural Processing Letters*, 22(2):223–233, 2005.

Parisi R.,E. D. D. Claudio, G. Orlandi, and B. D. Rao. Fast adaptive digital equalization by recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11),2731–2739, 1997.

Weng W. D. and C. T. Yen. Reduced-decision feedback FLANN nonlinear channel equaliser for digital communication systems. *IEE Proceedings on communications*, 155(4),305–311, 2004.

Wu S. and M. J. Er. Dynamic Fuzzy Neural Networks - A Novel Approach to Function Approximation. *IEEE Transactions on Systems, Man and Cybernetics-Part B: Cybernetics*, 30(2):358–364, 2000.

Wu S., M. J. Er and Y. Gao. A Fast Approach for Automatic Generation of Fuzzy Rules by Generalized Dynamic Fuzzy Neural Networks. *IEEE Transactions on Fuzzy Systems*, 9(4):578–594, 2001.

Yen C. T., W. D. Weng and Y. T. Lin. FPGA realization of a neural-network-based nonlinear channel equalizer. *IEEE Transactions on Industrical Electronics*, 51(2),771–779, 2004.