A Neural-Network-Based System for Testing Speakers

M.J. Er, T.H. Ooi, C.T. Toh and F.S. Toh

School of Electrical and Electronic Engineering, Nanyang Technological University
Nanyang Avenue, Singapore 2263.

Abstract

This paper presents a high performance neural-network-based system for testing speakers. A multi-layer neural network system with back-propagation learning algorithm is employed. It consists of 53 *input nodes, one hidden layer with* 10 *nodes and 1 output node. The normalized Total Harmonics Distortion (THD) values of the speakers at different frequencies are fed to the input of the system. The average training time is* 40 *minutes (on a* 486DX 50MHz *PC) for a training size of 100 patterns. The neural-network-based system is able to achieve a remarkable accuracy of* 95%.

1 Introduction

Frequently, strict inspection is carried out on incoming speakers in order to secure excellent quality in the final products in Motorola Singapore Pte Ltd. Currently, this inspection is performed manually by trained operators who are skilled at identifying audio defects. The manual procedure involves exciting the speaker under test with eight tones at different frequencies and recording the frequency response. Judgement on the speaker's performance is based on listening to the tone produced and by comparing the recorded frequency response with some "reference" responses laid down by experienced engineers. Listening test allows the operator to assess the distortion level whereas comparison with the reference curve allows evaluation of the speaker's gain response.

This manual test procedure is found to have the following disadvantages:

1. It is a very subjective test as the PASS/FAIL criterion varies from person to person.
2. As the operators are required to listen to a large number of speakers per day, their fatigue level, emotional state and work stress level do affect the evaluation.
3. Usually, a significant amount of "border-line" case speakers will get accumulated at the end of the week by "unsure" operators. This requires engineers to spend a lot of time to look into the issue.

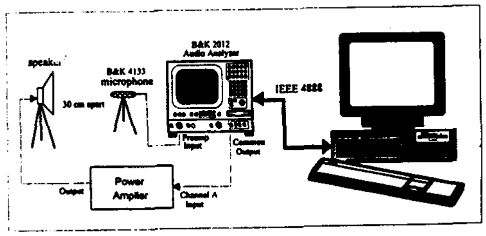

Figure 1: Neural-Network-Based System for Testing Speakers

In this paper, a neural-network-based system is proposed to replace the manual procedure. Why neural network ? The reasons are three fold: first, consistency. Due to its good fault tolerance that facilitates disturbance handling, neural network is able to provide consistent performance. Second, flexibility. Neural network involves no algorithmic programming; it learns and generates its own algorithm to solve the problem through training with examples. Changes in different models of the speakers and PASS/FAIL criterion will not lead to changes in the complete system; only re-training of the network is needed. Third, good noise tolerance. Frequently, due to facility constraint, speaker testing is carried out in the production floor. Such environment invariably introduces a noticeable amount of background noise to the testing process. As a consequence, a good speaker may be rejected when its frequency response is compared with the reference curve.

2 System Overview

The block diagram of the system is shown in Figure 1. The operation of the system is now briefly described. The neural-network-based system will prompt the audio analyser via IEEE-488 interface to sweep a tone from 300 Hz to 8 KHz in step of 25 Hz to the speaker under test via a Class A Power Amplifier. The speakers are tested

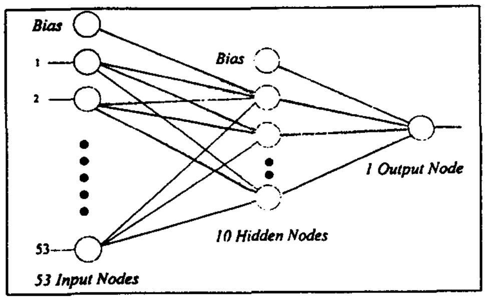

Figure 2: Neural Network Configuration

on frequency responses and Total Harmonic Distortion (THD) up to the *5th* harmonic. The frequency response provides information on the functionality of the speaker over human hearing range while the THD provides information on the Rub and Buff phenomenon. The speaker's response is in turn captured by the microphone located 1 *meter* away and fed back to the audio analyser for post processing. The frequency response and THD data generated from the audio analyzer are then sent back to the neural-network-based system.

3 Network Design

Multi-layer neural network configuration with back-propagation learning algorithm is utilized to implement the system for testing speakers. This network contains 53 input nodes, 1 hidden layer with 10 nodes and 1 output node. The input pattern is the normalized THD response. The speaker's response is sampled 53 times within the audio frequency range of $400Hz$ to $8kHz$. Each normalized THD value at its sampled frequency is fed as input to the neural network. The output node is responsible for indicating whether the speaker is good or bad. The final neural network configuration is shown in Figure 2.

3.1 Initial Network Design

The roles of the hidden layer and the effects of hidden nodes are particularly important because of their substantial impacts on the network's ability to learn. The hidden layer undertakes the non-linear mapping between the input pattern and the output. Many researchers have found that a single layer perceptron cannot represent arbitrary functions. To increase the network ability to learn complicated problem, one or more hidden layers are often necessary. However, simulation studies show that for our application, no significant improvement is obtained by increasing the number of hidden layers from 1 to 2. On the contrary, the training time increases exponentially. Hence, the network has only one hidden layer. An immediate question is how many nodes are required for the hidden layer ? Using the guideline of [1] which states that a good number of hidden nodes to start with can be obtained by taking the square root of the number of input nodes plus output nodes and adding a few more, the number of nodes in the hidden layer was set to 15.

3.2 Evaluation of Network Performance

The network performance is evaluated by the percentage correct. Percentage correct indicates the accuracy of the network in correctly identifying good and bad speakers. In general, this accuracy is closely linked to the degree of generalization the network has achieved after training. Good generalization means that the network has adequately explored and learnt the vital characteristics of both good and bad speakers during training. This generalized speaker's characteristic is used to compare the response of the speaker during actual testing. Therefore, good generalization is a pre-requisite for good accuracy. Basically, the degree of generalization of the neural network can be found by testing the network using some known patterns and recording its percentage correct. In this paper, there are a total of 160 known speaker's responses which have already been manually classified into good and bad. Out of these 160 known patterns, 100 were used to train the network and the remaining 60 were reserved for testing the network for percentage correct.

The procedure for checking the network's percentage correct is outlined in Figure 3.

3.3 Fine Tuning of Network

Fine tuning of the network is necessary in order to achieve the best compromise between accuracy and training speed through the adjustment of hidden layer size. The network may memorize a solution for each individual pattern in the training set rather than extracting a more general solution. On the other hand, insufficient hidden nodes will result in insufficient learning. Therefore, fine tuning the hidden layer is necessary to arrive at a trade-off for the number of nodes in the hidden layer. The trial and error procedure shown in Figure 4 is adopted.

The network performance is assessed based on its percentage correct on 60 test patterns which are unknown to the network. The percentage correct with 1,5,10,15 and 20 hidden nodes are experimented and the results are summarized in Figure 5. It is clear from Figure 5 that the hidden layer size markedly affects the network's ability of processing the information correctly.

Another factor which must be taken into consideration on deciding the 'optimum' hidden layer size is the training time needed. The variation of the training time with respect to the number of nodes in the hidden layer is

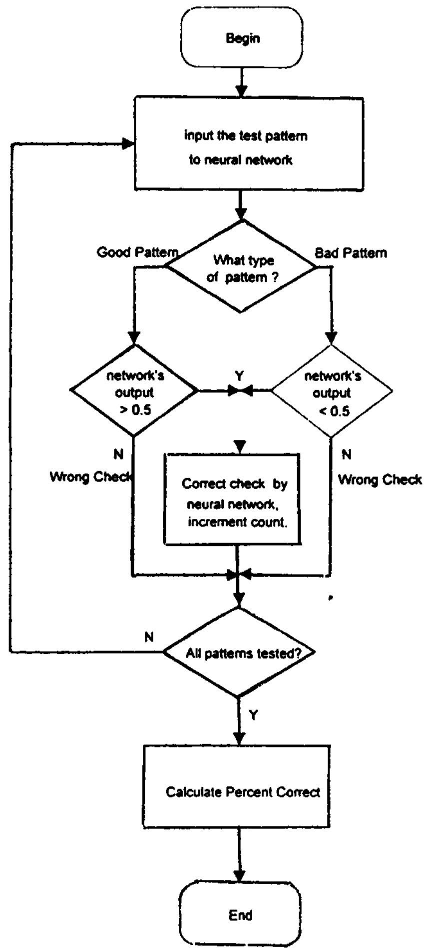

Figure 3: Procedure for Checking Percentage Correct

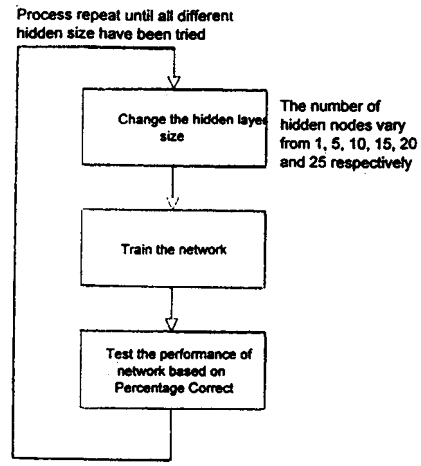

Figure 4: Trial and Error Procedure for Determining Hidden Layer Size

summarized in Figure 6. From Figure 6, we see that the training time increases exponentially with increases in the hidden layer size. This is because as the number of hidden nodes increases, the number of connection weights that needs to be trained also grows exponentially. From Figures 5 and 6, it is decided that the number of nodes in the hidden layer should be 10.

4 Network Training

The network was trained repeatedly until the desired error level was achieved. The sum squared error criterion is adopted. The sum squared error indicates how well

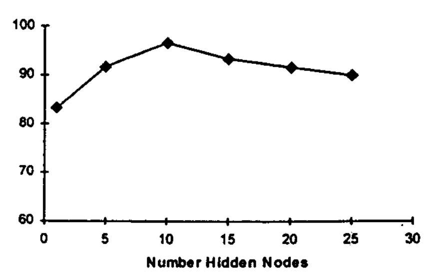

Figure 5: Relationship Between Percentage Correct and Hidden Layer Size

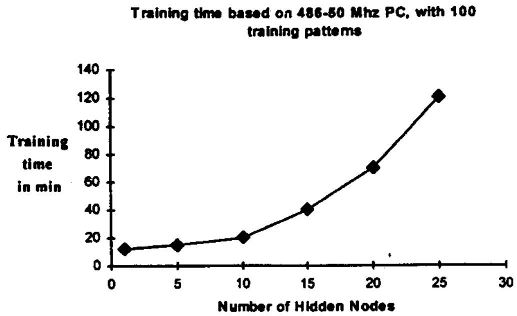

Figure 6: Relationship Between Training Time and Hidden Layer Size

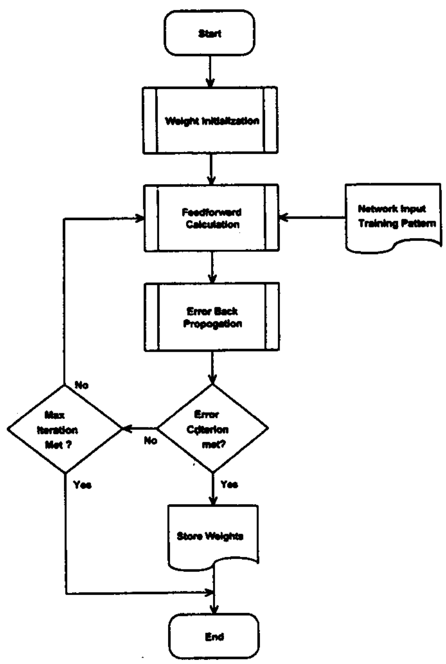

Figure 7: Major Steps in Network Training

training process is to drive the sum squared error to the desired value over all training patterns.

As implied in the major steps for network training shown in Figure 7, there are a few variables that affect the network's error convergence rate during training. These factors are: network structure, size of training set, learning rate and momentum. As mentioned in Section 3, network structure affects the training time exponentially. A neural network with larger hidden layer needs longer training time. However, as shown in the previous section, the network's size has already been fine tuned to achieve the best accuracy possible. Thus, it is not considered here.

The size of training set is another factor that affects the training speed in an exponential manner. Smaller training set requires shorter training time. However, it is not feasible to reduce training time by reducing the size of the training set. This is because small training set will limit the degree of generalization the network can achieve during training, resulting in lower accuracy as discussed in the previous section.

In general, learning rate does not affect the network performance in terms of correct classification. It merely affects the network convergence rate during training and hence the training time required. (The training time is inversely proportional to the learning rate.) From the weight adjustment equation used in the back-propagation algorithm, we can see that the learning rate is directly related to the amount of error correction, hence the amount of weight adjustment. Figure 8 shows the error curves with 0.3 and 0.8 learning rate. Note the reduction in training time as the learning rate increases from 0.3 to 0.8. However, when the learning rate gets too large (greater than 0.9), the training process becomes unstable. Oscillations of sum squared error commence and generally convergence to the required error tolerance is not possible. This is due to the larger weights correction required which in turn causes the resultant weights to oscillate about the desired value. In the event that convergence is possible, the larger learning rate will cause the error to drop steeply to the desired sum squared error. Figure 8 illustrates such oscillatory error curve when the learning rate is 2.

In addition to the learning rate, momentum is another factor used by the training algorithm to modify the network's weights. From the weight adjustment mechanism, we see that momentum allows a fraction of the previous weight change to be added to the current weight change. This mechanism prevents the network from being stuck at the shallow minimal of the error surface. When the momentum factor is zero, the weight change is based solely on the gradient descent direction. From Figure 9, we can see that without such momentum push, the network will sometimes fall into a local minimal. As the momentum increases, the network is able to pull itself out of any local minimal. From the error trajectories shown in

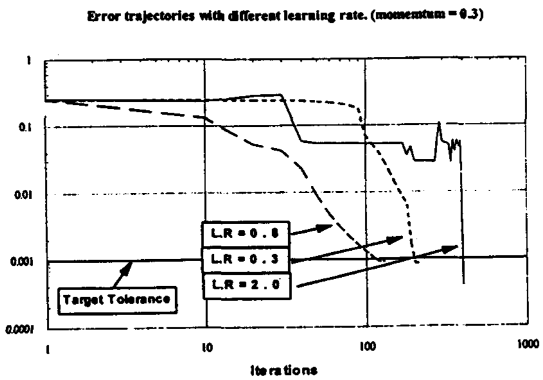

Figure 8: Sum Squared Error versus Number of Iterations for Different Learning Rates

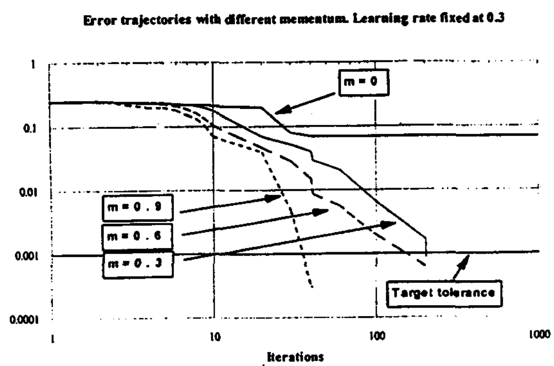

Figure 9: Sum Squared Error versus Number of Iterations for Different Momentum

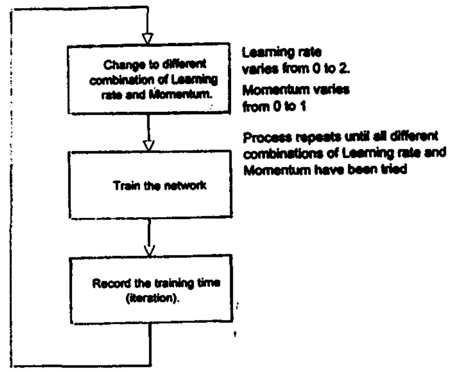

Figure 10: Iterative Process for Determining the Best Pair of Learning Rate and Momentum

Figure 9, we can also see that the number of iterations reduces as momentum increases from 0.3 to 0.9. However, for larger value of momentum, numerical overflow occurs during training.

From the previous study, it can be concluded that learning rate and momentum are the two factors that can be adjusted to achieve faster training without affecting the network's accuracy. In summary, by increasing the learning rate and the momentum to below a certain limit, we will be able to get shorter training time.

To our knowledge, there is no analytical method to determine the 'optimal' learning rate and momentum. To fine tune the learning rate and momentum for fast training yet securing convergence, trial and error procedure is adopted. The iterative process for training the network for the target error tolerance of 0.001 with 100 training data is shown in Figure 10.

It is observed that any combination with learning rate greater than 0.9 or with momentum greater than 0.8, the network may not converge to the required error tolerance. Furthermore, any combination with learning rate lower than 0.4 requires a longer training time and in the extreme case, the network cannot converge with zero learning rate. Similarly, with momentum smaller than 0.4, the learning is slow. Therefore, the best combination of learning rate and momentum to train the network falls in the range of 0.4 to 0.8 and 0.5 to 0.8 for learning rate and momentum respectively. The unshaded portions of Table 1 indicate the desired range.

Having obtained all the optimal training parameters, the network is trained for a target sum squared error tolerance of 0.001. Table 2 summarizes all the parameters of the network used. The error trajectory of the training process is shown in Figure 11.

Momentum	Learning Rate								
	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
0	∞	1170	600	390	320	270	210	180	160
0.1	∞	1070	530	360	270	230	200	170	150
0.2	∞	920	470	310	230	200	170	140	130
0.3	∞	860	410	270	210	170	150	120	110
0.4	∞	690	350	250	200	150	120	110	120
0.5	∞	580	290	210	150	120	110	100	120
0.6	∞	470	210	160	130	100	90	80	90
0.7	∞	360	180	120	90	70	60	70	100
0.8	∞	270	110	70	60	30	30	70	220
0.9	∞	90	40	30	40	30	50	380*	170*
1.0	∞	OFL	OFL	OFL	OFL	OFL	OFL	OFL	OFL

Momentum	Learning Rate				
	0.9	1.0	1.1	1.2	2.0
0	140	140	140	150	480*
0.1	140	120	110	120	290*
0.2	120	130	110	100	230*
0.3	90	110	80	110	130*
0.4	90	80	90	80	430*
0.5	80	110	100	90	1240*
0.6	90	90	100	90*	∞
0.7	100	120	150	220*	∞
0.8	170*	200*	190*	230*	∞
0.9	440*	460*	930*	∞	∞
1.0	OFL	OFL	OFL	OFL	OFL

Legend :
* - sometimes unable to converge
∞ - 10,000 iterations reached without converging
OFL - overflow occurs in calculation

Table 1: Number of Iterations for Different Values of Learning Rate and Momentum

Number of Input Nodes	53
Number of Hidden Nodes	10
Number of Output Nodes	1
Learning Algorithm	Back-propagation
Learning Rate	0.5
Momentum	0.7
Size of Training Set	100
Target Sum Squared Error Tolerance	0.001
Training Time Needed	40 min.
Number of Iterations	2500

Table 2: Final Network Settings

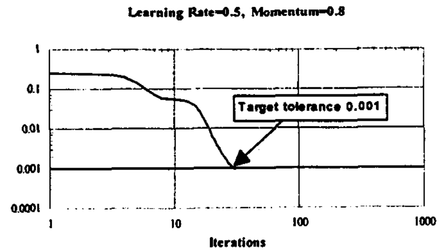

Figure 11: Error Trajectory for the Final Network Settings

5 Network Performance

Having trained the network with the optimal training parameters obtained in the previous section, the overall performance of the neural network-based system is evaluated based on percentage correct and noise tolerance. It turns out that the network produces a remarkable accuracy of 95%. Sixty known speakers which have been classified into good and bad manually were employed in this evaluation. Their THD responses were captured and fed to the network one at a time. Out of these 60 speakers, 57 of them were correctly identified by the proposed system.

6 Conclusions

In this paper, a high performance neural network-based speakers testing system has been successfully designed and implemented. A multi-layer neural network system with back-propagation learning algorithm is employed. The network consists of 53 input nodes, a hidden layer with 10 nodes and an output node. The system requires an average training time of 40 minutes (on a 486DX 50MHz PC) for a training size of 100 patterns. The system is able to achieve a remarkable 95% accuracy.

Acknowledgements

The authors would like to thank Mdm Chan Chew Lan and Mr Tai Chee Thiam, Engineering Section Managers of Motorola Singapore Pte Ltd for initiating the project, their kind assistance to the students and the use of the facilities at the plant.

References

[1] Russell C. Eberhart and Roy M. Dobbins, "Neural Network PC Tools- A Practical Guide".