Removing Noise from Speech Signals Using Different Approaches of Artificial Neural Networks

Omaima N. A. AL-Allaf
Faculty of Sciences & IT, Dept. of Basic Sciences, Al-Zaytoonah University of Jordan, P.O. Box130, Amman (11733), Jordan
E-mail: omaimaallaf@zuj.edu.jo

Abstract— In this research, four ANN models: Function Fitting (FitNet), Nonlinear AutoRegressive (NARX), Recurrent (RNNs), and Cascaded-ForwardNet were constructed and trained separately to become a filter to remove noise from any speech signal. Each model consists of input, hidden and output layers. Two neurons in the input layer that represent speech signal and its associated noise. The output layer includes one neuron that represent the enhanced signal after removing noise. The four models were trained separately on stereo (noisy and clean) audio signals to produce the clean signal. Experiments were conducted for each model separately with different: architecture; optimization training algorithms; and learning parameters to identify model with best results of removing noise from speech signal. From experiments, best results were obtained from FitNet and NARAX models respectively. TrainLM is the best training algorithm in this case. Finally, the results showed that the suggested architecture of the four models have filtering ability to remove noise form both trained and not trained speech signals samples.

Index Terms— Signal Enhancement, Artificial Neural Networks, Function Fitting (FitNet), Nonlinear Autoregressive (NARX), Recurrent (RNNs), and Cascaded-ForwardNet

I. INTRODUCTION

Artificial Neural Networks (ANNs) includes many neurons that operate in parallel and connected via weights [1]. ANNs have been used largely in the fields of image and signal processing [2]. Many ANN models that differ in architecture and training algorithms were suggested in the past. Multi-layer perceptron and backpropagation neural network (BPNN) are largely used ANN models. These models require large number of iterations for training network on any specified problem. Many literature approaches have been carried out to improve the speed up the network learning [3][4][28...32]. The nonlinear ANN nature and ability to learn from their environments in supervised and unsupervised ways make them highly suited for solving difficult signal processing problems. But all these available techniques have certain limitations.

Signal enhancement is process of performing nonlinear filtering of a signal for noise reduction. Is critical to understand the nature of problem formulation when dealing with signal processing applications, so that the most ANN approaches can be applied. Recently, ANNs are found to be a very efficient tool for signal enhancement. It is important to assess the impact of ANN on the performance, robustness, and cost-effectiveness of signal processing systems.

Another important issue is how to evaluate ANN approaches, learning algorithms, and structures for solving signal processing problems [5].

ANNs may provide a new approach for signal filtering. Recent experimental suggests that ANNs may be able to reduce noise for many applications [6]. There are few literature researches adopted different ANNs models for removing noise from signal. At the same time, the filtering ability of ANN architectures of most of these literature studies were limited to training set signals. These literature approaches were not able to remove noise from untrained signals or filtered the noise with bad quality and generate the signal with some noise. Other literature studies required long time for training process.

Therefore, we need ANN approaches and models for signal enhancement that remove noise from any speech signal. Also, we need ANN models that generate high values of Peak Signal to Noise Ratio (PSNR) and less values of Mean Square Error (MSE). We need to reduce the time required for the training process.

In this research, four ANN models were constructed and implemented (in MathLab 2013a) for signal enhancement system: Function Fitting (FitNet), Nonlinear Autoregressive (NARX), Recurrent (RNNs), and Cascaded-ForwardNet to improve the speed of convergence and enhance the signals by obtaining high PSNR, MSE and best values of R² (coefficient of determination of a linear regression). Three optimization algorithms were used in training process (LM, GD, and GDM). Many experiments were conducted to determine the ANN model with its architecture that lead to less learning time and best performance. Comparisons between the four models were conducted also. The research is organized as follows: section II includes related literature and section III describes signal enhancement. Section IV includes details about ANN used models. Section V includes research methodology. Section VI includes experiments and finally section VII concludes this work.

Copyright © 2015 MECS
II. RELATED LITERATURE

Many literature studies for noise reduction from speech signal were based on different approaches. Other literature studies were based on ANN for signal enhancement each with different model and architecture. The goal is to define a model that give best results for signal enhancement. At the beginning, Kevin, 1988 [6] explored BPNN approach to filtering noise from signals. Experiments were conducted using single sine wave inputs, multiple sine wave inputs, and human speech inputs. The networks’ outputs were then compared to the original signals to determine the network’s performance. But network’s filtering ability was strictly limited to signals from the training set. The networks were not able to generalize enough to filter signals whose frequencies had never been encountered.

Speech enhancement is concerned with neural processing of noisy speech to improve the quality of speech signal. Tlucak, 1999 [7] described ANNs approaches for speech enhancement. He described an experiment with implementation of two channel adaptive noise remover via direct time domain mapping approach.

Whereas, Badri, 2010 [8] used BPNN and Recurrent ANN (RNN) for noise reduction to enhance the captured noisy signals. They presented the effect of training algorithms and network architecture on ANN performance for a given application. The PSNR was not calculated to determine the enhanced signal. Whereas the minimum obtained MSE is 0.0112. Adaptive filtering technique is one of digital signal processing areas and used in noise cancellation. Noise cancellation is a common and necessary in today telecommunication systems. The LMS algorithm is one of the most efficient criteria for determining the values of adaptive noise cancellation coefficients but it suffers slow convergence rate under low Signal-to-Noise ratio (SNR). Therefore, Miry, et. al., 2011 [9] presented an adaptive noise canceller algorithm based fuzzy and neural network for noise canceling problem of long distance communication channel. Whereas, Pankaj and Anil, 2012 [10] reviewed some speech enhancement techniques to remove noise from speech. They concluded that the fuzzy and neural algorithms used for minimizing noise from a set of sound files. The Root Mean Square Error (RMSE) and number of epochs are less and at the same time the membership functions are also less.

While, Dehananda, et. al., 2012 [11] discussed the use of BPNN for noise filtering to remove noise from a noise signal with implications for the signal processing. Thus research lack of discussion related to experimental results such as PSNR and RMSE .

And, Chatterjee, et. al., 2013 [12] presented BPNN as an adaptive filtering technique for removing noise from ECG signal using the Recursive Least Square (RLS) method. They extracted 12 significant features from an echocardiogram (ECG) dataset. ECG signal is obtained after carrying out noise cancellation followed by Recursive Least Square method filtered.

Finally, Andrew, et. al., 2012 [13] introduced recurrent auto encoder ANN to remove noise from input signal for automatic speech recognition (ASR). They trained the model on noisy and clean audio signals to output clean signal. The model makes no assumptions about how noise affects signal, nor the existence of noise environments.

The results of literature related to noise enhancement showed that the performance of recently suggested ANN models and approaches to enhance the untrained signal is low against the signals used in training process. At the same time most of the recently suggested models in literature studies require long time for training process. Thus, we need ANN model for signal enhancement that leads to: less ANN training time; less MSE and highest values for PSNR.

III. SIGNAL ENHANCEMENT

Signal enhancement is plays an important role in many applications. There are many literature researches for removing the noise from the signal. All these available techniques have certain limitations. Speech enhancement techniques tend to reduce or remove the noise from speech signal. Different types of speech enhancement systems using a single microphone have been proposed and tested. All of these systems are based on techniques which recover the clean speech signal by enhancing the Signal to Noise Ratio (SNR). The performances depend on type of noise and the information they require about noise. It should be noted that the increase of SNR will improve the quality of the speech signal [14]. There are four types of speech enhancement methods:

A. Filtering

Adaptive filtering techniques like Kalman filtering have been used for speech enhancement [15]. The Kalman filter yields an optimal solution to the adaptive filtering problem based on LMSE. Unless the noise is stationary and perfectly known that must be done iteratively [16]. According to changing of noise over a time period and overlapping of frequencies of noise and signal, the Adaptive filtering become important. The ability of adaptive filters to operate satisfactorily in unknown and time-varying environments without user intervention and improving their performance during operation by learning statistical characteristics from current signal observations has made them more efficient [12].

B. Noise subtraction

The noise and speech are assumed to be uncorrelated and additive. The power spectrum of cleaned speech is obtained by subtracting the noise power spectrum from the spectrum of noisy speech [17]. The method assumes that the noise varies slowly so that the noise estimation obtained during a pause can be used for suppression.

C. Space Mapping

Speech enhancement is a process of transforming noisy speech into clean speech by mapping. For instance, spectral mapping has been implemented by a set of rules obtained by vector quantization techniques. It is possible
to implement arbitrarily complex space transformations thanks to connectionist ANNs. Even multi-layer perceptrons have been trained on learning samples to realize a mapping of noisy signals to noise-free speech which has been tested with success in an auditory preference test with human listeners [18].

D. Hidden Markov Models (HMM)

Decomposition is a method which makes it possible to separate speech from additive noise [19]. It is assumed that speech and noise are both modeled by separate HMMs and that noisy speech corresponds to a composition model combining these two models. This method is rather computationally demanding, but it has been demonstrated to perform satisfactorily even in bad SNR conditions.

IV. NEURAL NETWORKS MODELS

Fig.1 shows Multilayer feed-forward ANNs that consist of neurons arranged in layers with only forward connections to units in subsequent layers [20]. The connections have weights associated with them. The input layer units distribute the inputs to units in subsequent layers. In the following layers, each unit sums its inputs and adds a bias to the sum and nonlinearly transforms the sum to produce an output. This nonlinear transformation is called the activation function of unit [5].

![Multilayer feed-forward ANN](image)

The output layer units have linear activations. The training data set consists of Np training patterns \( \{(x_p, t_p)\} \), where \( p \) is the pattern number. The input vector \( x_p \) and desired output vector \( t_p \) have N and M dimensions. \( y_p \) is network output vector for \( p \)th pattern. The thresholds are handled by augmenting input vector to an element \( x_p(N+1) \) and it is equal 1. For the \( j \)th hidden unit, the net input \( net_p(j) \) and output activation \( Op_p(j) \) for \( p \)th training pattern are shown in (1) and (2) [1][3][5].

\[
net_p(j) = \sum_{i=1}^{N+1} w(j,i) \cdot x_p(i), \quad 1 \leq j \leq N_h
\]

\[
Op_p(j) = f\left(net_p(j)\right)
\]

where \( w(j,i) \) denotes the weight connecting the \( i \)th input unit to the \( j \)th hidden unit. For MLP networks, a typical sigmoid activation function \( f \) is shown in (3) [1][3][5].

\[
f\left(net_p(j)\right) = \frac{1}{1 + e^{-net_p(j)}}
\]

For trigonometric networks [23], the activations are Sines and Cosines. The \( k \)th output for the \( p \)th training pattern is \( y_{pk} \) and is given by (4) [1][3][5]:

\[
y_{pk} = \sum_{i=1}^{N+1} w_{hp}(k,i) \cdot x_p(i) + \sum_{j=1}^{N_h} w_{hj}(k,j) \cdot Op_p(j), \quad 1 \leq k \leq M
\]

(4)

Where \( w_{hp}(k,i) \) denotes the output weight connecting the \( i \)th input unit to the \( k \)th output unit and \( w_{hj}(k,j) \) denotes the output weight connecting the \( j \)th hidden unit to the \( k \)th output unit. The mapping error for the \( p \)th pattern is calculated using (5) [1][3][5].

\[
E_p = \sum_{k=1}^{M} \left| t_{pk} - y_{pk} \right|^2
\]

(5)

Where \( t_{pk} \) denotes the \( k \)th element of \( p \)th desired output vector. In order to train ANN in batch mode, the mapping error for \( k \)th output unit is defined as (6) [1][3][5].

\[
E(k) = \frac{1}{N_p} \sum_{p=1}^{N_p} \left| t_{pk} - y_{pk} \right|^2
\]

(6)

The overall performance of ANN an be measured as MSE and written as (7) [1][3][5].

\[
E = \sum_{k=1}^{M} E(k) = \frac{1}{N_p} \sum_{p=1}^{N_p} E_p
\]

(7)

Then, the adjustment of ANN weights to minimize the error between network’s prediction and actual output. This is to adjust weights values when ANN gaining extra knowledge after each iteration. Output \( y_{pk} \) is compared with target output \( T_{pk} \) using (8) [1][3][5]:

\[
\delta_k = (T_{pk} - y_{pk}) \cdot (1 - y_{pk})
\]

(8)

For every neuron in hidden layer, the error term is calculated using (9) [1][3][5]:

\[
\delta_k = y_{pk} \cdot (1 - y_{pk}) \sum \delta_{pk} w_{pk}
\]

(9)

Where \( \delta_{pk} \) is error of output layer and \( w_{pk} \) is weight between hidden and output layer. The error is propagated backward from output layer to input layer to update weight of connections using (10) [1][3][5]:

\[
w_{jk}(t+1) = w_{jk}(t) + \eta \delta_k y_{kj} + \alpha (w_{jk}(t) - w_{jk}(t-1))
\]

(10)

Both of \( \eta \): learning rate and \( \alpha \): momentum factor are specified at the start of training and determine speed and stability of ANN [1][3][5].

A. Function Fitting ANN

The FitNet is a type of feed forward networks, which are used to fit an input output mapping relationship. A FitNet with one hidden layer and enough neurons [21][22]. The output is calculated directly from the input
Removing Noise from Speech Signals Using Different Approaches of Artificial Neural Networks

B. Nonlinear Auto-Regressive eXogenous (NARX)

NARX is a nonlinear autoregressive model which has exogenous inputs in time series modeling. The NARX is a recurrent dynamic network with feedback connections enclosing several layers of network. The model relates the current value of a time series which one would like to explain or predict to both: past values of the same series; and also current and past values of the driving (exogenous) series. That is, of the externally determined series that influences the series of interest. The model contains: error term which relates to the fact that knowledge of the other terms will not enable the current value of the time series to be predicted exactly. Such a model can be stated algebraically as shown in (11) [23].

\[
y_t = F \left( y_{t-1}, y_{t-2}, y_{t-3}, \ldots, u_t, u_{t-1}, u_{t-2}, u_{t-3}, \ldots \right) + \epsilon_t
\]  

Here \( y \) is the variable of interest, and \( u \) is the externally determined variable. Information about \( u \) helps predict \( y \), as do previous values of \( y \) itself. Here \( \epsilon \) is the error term (noise). For example, \( y \) may be air temperature at noon, and \( u \) may be the day of the year (day-number within year). The function \( F \) is some nonlinear function (polynomial). \( F \) can be ANN, a wavelet network, or a sigmoid network [23]. Although dynamic networks (time serious) can be trained using the same gradient-based algorithms (BP) that are used for static networks, the performance of the algorithms on dynamic networks can be quite different, and the gradient must be computed in a more complex way [21][22][23][24]. Fig.4 shows the architecture of RNN model that conducted in this research.

D. Cascaded Forward Neural Networks

These are similar to feed forward networks such as BPNN with exception that they have a weight connection from the input and every previous layer to the following layers. Fig.5 shows the architecture of Cascaded-ForwardNet model that was conducted in this research and has connections from layer 1 to layer 2, layer 2 to layer 3, and layer 1 to layer 3. This ANN also has connections from input to all three layers.
E. Training Algorithms for ANN models

The optimization training algorithms adjusted the ANN weights with the goal to minimize the performance function. In static feed forward networks or RNN, the performance function to be minimized is taken to be the MSE, between model-desired output and actual response. In each synapse between connected neurons the training function calculates the output error and determines the adjustments to the network’s weights and bias. There are many training algorithms that can be used for ANN training such as: Gradient descent (GD); Gradient descent with momentum (GDM); Gradient descent momentum and an adaptive learning rate (GDX); Levenberg-Marquardt (LM) and Bayesian regularization (BR). These training algorithms can be used to train any ANN to adjust networks' weights to reduce errors as possible as.

V. RESEARCH METHODOLOGY

In this research, four ANN models: FitNet (Fig.2), NARX (Fig.3), RNN (Fig.4), and Cascaded-ForwardNet (Fig.5) were constructed and implemented to remove noise from any signal and obtaining good results of RMSE, PSNR and reducing the training time. Each model was suggested to include 3 layers (input, hidden and output). In this research, three different optimization training algorithms were used to train the four ANN models separately to get best results for noise removing system. These optimization training algorithms are as follows [22]: GD; GDM and LM. The network architecture and parameters are shown in details in Table 1. Whereas, Fig. 2 shows the general view of noise removal system.

A. Algorithm of Training Process

The main steps of training process of each model are as follows:
1) Initialization of network weights with random numbers in the range [0.001 and 0.009];
2) Initialization of network parameters (η and α) each with value in the range [0.1 and 0.9]
3) Set iterations to zero.
4) Set Threshold error with 0.0000001.
5) Total_error = zero;
6) iterations \(\rightarrow\) iterations+1
7) Set i with 0 (where i: signal sequence)
8) Open the file which contains the sound signal (s_i) to be used in training process and read the signal (s_i)
9) Apply this sound file to input layer neuron.
10) Initialize target output to be same as input signal.
11) Add noise to this sound signal by applying randomly noise to second input neuron.
12) Calculate the outputs of hidden layer units using (1), (2) and (3).
13) Calculate the outputs of output layer units using (4).
14) Calculate the error(0) of the signal (s_i)= desired output – actual output using (5)
15) If there is another sound signal to be used in training process, then i=i+1, then go to step8, otherwise, go to step 16.
16) Calculate overall ANN error using (6) and (7).
17) Adjust weights between output and hidden layer units using (8), (9) and (10).
18) Adjust weights between hidden and input layer units using (8), (9) and (10).
19) If Total_error \(\leq\) Threshold_error then stop (the network in trained well). Otherwise (need more training), then go to step 5.

B. Testing Process

The main steps of testing process of each model are:
1) Open file which contains sound signal to be tested.
2) Apply this sound file to input layer neuron.
3) Add noise to this sound signal by applying randomly noise to second input neuron.
4) Calculate outputs of hidden layer units using (1), (2) and (3).
5) Calculate outputs of output layer units using (4).
6) Calculate the MSE, PSNR of input noisy signal, PSNR of output clean signal and R^2.

Table 1. Network architecture and parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of layers</td>
<td>3 layers (input, hidden and output)</td>
</tr>
<tr>
<td>Size of signal sample</td>
<td>50000</td>
</tr>
<tr>
<td>Size of noise sample</td>
<td>50000</td>
</tr>
<tr>
<td>Number of neurons in input layer</td>
<td>2 input units (One unit for signal and other unit for the noise)</td>
</tr>
<tr>
<td>Number of neurons in hidden layer</td>
<td>10 hidden units</td>
</tr>
<tr>
<td>Number of neurons in output layer</td>
<td>One output unit that represent the signal after removing noise.</td>
</tr>
<tr>
<td>Training Algorithms</td>
<td>LM, GD, and GDM</td>
</tr>
<tr>
<td>Training patterns</td>
<td>Audio files including music with song from MathLab Library (2013a)</td>
</tr>
</tbody>
</table>

VI. EXPERIMENTAL RESULTS

In this research, four simulation programs to implement the learning and testing processes of the four ANN models: FitNet (Fig.2), NARX (Fig.3), RNN (Fig.4), and Cascaded-ForwardNet (Fig.5). The MathLab 2013a software was used to write programs of speech signal enhancement system. The training data were taken from MathLab library. Also many other sound files were downloaded from website "Odyssey FX | Wav Sound Effects” and used as testing data. Odyssey FX delivers a great source of wonderful sound effects.
The Sample Pack includes 70 24-bit 100\% royalty free wav sound effects samples, Swishes, sweeps and falls Weird hits and stuttered glitch, Retro noises, bleeps and atmospherics inspired by the early Radiophonic Workshop [27].

Number of iterations, MSE, R\(^2\) and PSNR were used to check the performance of the signal enhancement system. PSNR will be calculated between the output clean signal and input noisy signal to verify the quality of signal enhancement system. The efficiency of this system was tested by several speech signals. Each model consists of 3 layers (input layer with 2 neurons, hidden layer with 10 neurons and output layer with one neuron).

A. **Gradient Descent with Momentum (GDM)**

The first four experiments were based on training the four models separately using GDM training algorithm. One simulation program for each experiment. Table 2 shows: MSE, PSNR, R\(^2\) and the number of iteration required to train each model with GDM algorithm.

<table>
<thead>
<tr>
<th>Model</th>
<th>Iterations</th>
<th>MSE</th>
<th>PSNR</th>
<th>R(^2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>NARX</td>
<td>678</td>
<td>0.0027</td>
<td>33.73</td>
<td>0.95</td>
</tr>
<tr>
<td>Layrecnet</td>
<td>810</td>
<td>0.0019</td>
<td>33.24</td>
<td>0.95</td>
</tr>
<tr>
<td>Cascaded</td>
<td>988</td>
<td>0.0008</td>
<td>32.59</td>
<td>0.93</td>
</tr>
<tr>
<td>FitNet</td>
<td>465</td>
<td>0.0006</td>
<td>34.76</td>
<td>0.98</td>
</tr>
</tbody>
</table>

We can note from Table 2 that, best results (smallest number of iterations, high value of PSNR, less MSE) were obtained using the FitNet model. Also the next good results were obtained from using the NARX model.

B. **Gradient Descent (GD) Algorithm**

Another four experiments were based on training the four models separately using the GD training algorithm. One simulation program for each experiment. Table 3 shows: MSE, PSNR, R\(^2\) and number of iterations required to train each model with GD algorithm.

<table>
<thead>
<tr>
<th>Gradient Descent with Momentum (GDM)</th>
<th>No. of hidden units =10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>Iterations</td>
</tr>
<tr>
<td>NARX</td>
<td>678</td>
</tr>
<tr>
<td>Layrecnet</td>
<td>835</td>
</tr>
<tr>
<td>Cascaded</td>
<td>1013</td>
</tr>
<tr>
<td>FitNet</td>
<td>473</td>
</tr>
</tbody>
</table>

We can note from Table 3 that, best results were obtained using FitNet model. The results of experiments related to GDM algorithm are near to results related to experiments related to GD algorithm.

C. **Levenberg-Marquardt (LM) Algorithm**

Other four experiments were based on training the four models using LM algorithm. Simulation program for each experiment. Table 4 shows MSE, PSNR of input noisy signal, PSNR of output clean signal, R\(^2\) and number of iterations to train each model with LM algorithm.

<table>
<thead>
<tr>
<th>Levenberg-Marquardt (LM)</th>
<th>No. of hidden units =10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>Iterations</td>
</tr>
<tr>
<td>NARX</td>
<td>623</td>
</tr>
<tr>
<td>Layrecnet</td>
<td>745</td>
</tr>
<tr>
<td>Cascaded</td>
<td>932</td>
</tr>
<tr>
<td>FitNet</td>
<td>413</td>
</tr>
</tbody>
</table>

From Table 4, the best values of PSNR were obtained from FitNet model. The values of MSE of all models are small whereas the values of R\(^2\) are high.

We can note from Table 2, Table 3 and Table 4 that, the Cascaded model requires number of iterations more than the other models to convergence. Whereas FitNet model requires less number of iterations. Also, best values of: number of iterations, PSNR, MSE, R\(^2\) were obtained from all models when they were trained using Levenberg-Marquardt (LM). Therefore, PSNR of input noisy signal was computed for each model based on LM algorithm. This is done to determine the models ability to remove noise from speech signals.

We can note from Table 4 that, the values of PSNR of output signal is high according to the values of PSNR of the corresponding input noisy signal. Therefore, all the models have the ability to remove noise from any speech signal with small differences between PSNR values. Also the best value of PSNR was obtained from FitNet model.

D. **Impact of learning rate value**

In this section, other four experiments were conducted for each one of the four models separately. Each model is trained using LM with 10 hidden units. The four experiments used different values of learning rate such as (0.05, 0.005, 0.0003 and 0.00001). Table 5 shows: number of iterations, MSE, PSNR and R2 for each model for every value of learning rate.
We can note from Table 5 that there are no notable differences between the values of MSE, PSNR and $R^2$ for all models when decreasing the value of learning rate. But the number of iterations were decreased when decreasing learning rate.

### E. The Effect of Hidden Layer Neurons

Many other experiments were conducted based on training the four models separately on different number of hidden layer neurons (10, 20 and 30 respectively). Table 6 shows the impact of number of hidden neurons on: number of iterations, MSE, PSNR and $R^2$ factors.

<table>
<thead>
<tr>
<th>No. node in hidden=10</th>
<th>Levenberg-Marquardt (LM)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>Iters</td>
</tr>
<tr>
<td>NARX</td>
<td>623</td>
</tr>
<tr>
<td></td>
<td>589</td>
</tr>
<tr>
<td></td>
<td>511</td>
</tr>
<tr>
<td></td>
<td>491</td>
</tr>
<tr>
<td>RNN</td>
<td>745</td>
</tr>
<tr>
<td></td>
<td>702</td>
</tr>
<tr>
<td></td>
<td>678</td>
</tr>
<tr>
<td></td>
<td>622</td>
</tr>
<tr>
<td>Cascaded</td>
<td>932</td>
</tr>
<tr>
<td></td>
<td>893</td>
</tr>
<tr>
<td></td>
<td>845</td>
</tr>
<tr>
<td></td>
<td>809</td>
</tr>
<tr>
<td>FitNet</td>
<td>413</td>
</tr>
<tr>
<td></td>
<td>397</td>
</tr>
<tr>
<td></td>
<td>365</td>
</tr>
<tr>
<td></td>
<td>311</td>
</tr>
</tbody>
</table>

We can note from Table 6 that increasing the number of hidden layer neurons for each model will lead to increasing number of iterations. At the same time, there is no big differences in values of MSE, PSNR and $R^2$.

### F. Impact of MSE

At the same time, we can note from all experiments related to all models, that the value of MSE start with large value at the beginning of the training process, then decreased gradually during the training process. According to the value of MSE, the training process will stopped at value of MSE near to 0.0000001. Fig. 6 shows the decreasing in MSE values during the training process.

![Fig. 6. Learning rate during training significance of figure in caption.](image)

### G. Results of Testing Process

Other 12 experiments were conducted based on testing the four models separately using LM, GDM and GD training algorithms respectively. One simulation program for each experiment. The PSNR is calculated for the output signal after removing the noise from it. Table 7 shows MSE and PSNR of input noisy signal, and PSNR of output clean signal for each one of the four model when trained using LM algorithm.

<table>
<thead>
<tr>
<th>Model</th>
<th>MSE</th>
<th>PSNR of input noisy signal</th>
<th>PSNR of output signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>NARX</td>
<td>0.0007</td>
<td>20.12</td>
<td>33.88</td>
</tr>
<tr>
<td>RNN</td>
<td>0.0005</td>
<td>19.25</td>
<td>33.65</td>
</tr>
<tr>
<td>Cascaded</td>
<td>0.0004</td>
<td>17.64</td>
<td>32.54</td>
</tr>
<tr>
<td>FitNet</td>
<td>0.00007</td>
<td>16.22</td>
<td>34.22</td>
</tr>
</tbody>
</table>

Whereas, Table 8 shows MSE and PSNR of input noisy signal, and PSNR of output clean signal for each one of the four model when trained using GDM algorithm.

<table>
<thead>
<tr>
<th>Model</th>
<th>MSE</th>
<th>PSNR of input noisy signal</th>
<th>PSNR of output signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>NARX</td>
<td>0.0034</td>
<td>20.12</td>
<td>33.53</td>
</tr>
<tr>
<td>RNN</td>
<td>0.0023</td>
<td>19.25</td>
<td>33.11</td>
</tr>
<tr>
<td>Cascaded</td>
<td>0.0012</td>
<td>17.64</td>
<td>32.35</td>
</tr>
<tr>
<td>FitNet</td>
<td>0.0009</td>
<td>16.22</td>
<td>34.55</td>
</tr>
</tbody>
</table>
And finally, Table 9 shows MSE and PSNR of input noisy signal, and PSNR of output clean signal for each one of the four model when trained using GD algorithm.

<table>
<thead>
<tr>
<th>Model</th>
<th>MSE</th>
<th>PSNR of input noisy signal</th>
<th>PSNR of output signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>NARX</td>
<td>0.0067</td>
<td>20.12</td>
<td>PSNR</td>
</tr>
<tr>
<td>RNN</td>
<td>0.0051</td>
<td>19.25</td>
<td>33.55</td>
</tr>
<tr>
<td>Cascaded</td>
<td>0.0032</td>
<td>17.64</td>
<td>33.42</td>
</tr>
<tr>
<td>FitNet</td>
<td>0.0023</td>
<td>16.22</td>
<td>32.47</td>
</tr>
</tbody>
</table>

We can note from Table 7, Table 8 and Table 9 that, the best results were obtained from the FitNet model. Also, we can note that the smallest values of MSE and PSNR of all models (NARX, Layrecnet, Cascaded, and FitNet) were obtained when using the LM training algorithm.

Therefore, PSNR of the input noisy signal was calculated for all models based on LM algorithm to determine the models ability to enhance noisy speech signals. From Table 7, Table 8 and Table 9, the PSNR values of output signal is high according to PSNR values of the corresponding input noisy signal. Thus, the four models have the ability to remove noise from any speech signal with small differences between PSNR values.

H. Ability to Remove Noise from Untrained Signal

The testing process is based on the trained and untrained speech signals. Fig.7 shows the input noisy speech signal which was already used in training process.

Fig.7 shows also the trained speech signal after the filtering (removing noise) using FitNet model. Also, the four models were succeeded to filter untrained speech signal (S1) from noise. We can note from Fig.7 that the FitNet model can successfully remove noise any trained speech signal. Untrained speech signal (S1) is used to test the FitNet model that is trained using LM. Fig.8 shows untrained clean signal (S1), noisy signal (S1), and finally clean signal (S1) after removing noise using FitNet model. Other experiments for testing FitNet model based on LM on other 5 untrained speech signals (S2..S6). Fig.9 shows the original signal, noisy signal and its enhanced version of four untrained speech signals (S2, S3, S5 and S6). Fig.9 shows the ability of FitNet model to remove noise from any noisy speech signal. Table 10 shows the MSE, PSNR of input noisy signal and PSNR of output clean signal of the 5 untrained signals. Table 10 shows high values of PSNR and low values of MSE.

<table>
<thead>
<tr>
<th>Network</th>
<th>Signal</th>
<th>MSE</th>
<th>PSNR Of input signal</th>
<th>PSNR Of output signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>FitNet, LM Algorithms, 10 hidden units</td>
<td>S1</td>
<td>0.000081</td>
<td>20.45</td>
<td>34.55</td>
</tr>
<tr>
<td></td>
<td>S2</td>
<td>0.000073</td>
<td>21.12</td>
<td>34.24</td>
</tr>
<tr>
<td></td>
<td>S3</td>
<td>0.000073</td>
<td>16.22</td>
<td>34.36</td>
</tr>
<tr>
<td></td>
<td>S5</td>
<td>0.000077</td>
<td>19.07</td>
<td>34.22</td>
</tr>
<tr>
<td></td>
<td>S6</td>
<td>0.000083</td>
<td>20.35</td>
<td>34.37</td>
</tr>
</tbody>
</table>

Fig. 7. Original signal, signal with noise, and signal after filtering for already trained speech signal using FitNet model.
Removing Noise from Speech Signals Using Different Approaches of Artificial Neural Networks

Fig. 8. Original signal, signal with noise, and signal after filtering for already untrained speech signal (S1) using FitNet model

Fig. 9. Shows the original signal, noisy signal and its enhanced version of four untrained speech signals (S2, S3, S5 and S6) (FitNet model)
VII. CONCLUSION

In this research, Four ANN models: FitNet (Fig.2), NARX (Fig.3), RNN (Fig.4), and Cascaded-ForwardNet (Fig.5) were constructed with 3 layers (input, hidden and output) for speech signal enhancement to remove noise from any signal. Two neurons in input layer to represent speech signal and its associated noise. The output layer includes one neuron that represent the enhanced signal after removing noise. Different number of hidden neurons were used for each model (10, 20 and 30). Three training algorithms (GD; GDM and LM) were used to train each one of the four models separately to become a filter. The MathLab 2013a software was used to write programs of training and testing the four models for speech signal enhancement system. The four models were trained on many stereo speech signals (noisy and clean) to produce the clean signal of the noisy input signal.

The training data were taken from MathLab 2013a. Also other sound files were taken from Odyssey FX | Wav Sound Effects [27] and used as testing data.

Many experiments were conducted for training/testing each model separately with different: number of hidden layer neurons, learning rate, and training algorithm to identify model with best results of removing noise from any speech signal.

Best results (smallest number of iteration, highest PSNR, lowest MSE) were obtained from FitNet model, then from NARX models respectively. The experiments shows that, decreasing the learning rate parameter will decrease the number of iterations required to train each model. Experiments shows also that, increasing number of hidden neurons will lead to increase number of iterations for learning process. From experiments, TrainLM is the best training algorithm in all experiments. The worst results obtained from Cascaded model.

Finally, from experiments, the suggested filtering approach based on FitNet, NARX , RNN and Cascaded models respectively has the ability to remove noise from speech signal as possible as. This is done according to the calculation of the PSNR of the input noisy signal (17.88dB) and PSNR of output clean signal (35.35dB) for the FitNet model. Also there is a big difference between the two values (PSNR of input and PSNR of output) for the other models (NARX, RNN, Cascaded). The four models approved in removing noise from speech signal. The filtering based ANN four approaches was tested on both trained and untrained speech signal. The best values of PSNR of output clean signal were obtained from FitNet model (34.22dB) and NARX (33.88dB) respectively.

REFERENCES


Removing Noise from Speech Signals Using Different Approaches of Artificial Neural Networks


Author’s Profiles

Dr. Omaima N. A. AL-Allaf is an Assistant Professor at Basic Sciences department, Faculty of Sciences and IT, AL-Zaytoonah University of Jordan, Amman-Jordan (from 15Feb2009). She received her B.Sc. in CS from Dept. of CS, Faculty of Sciences, University of Mosul (1990-1994), and received her M.Sc. in Dept. of CS, Faculty of Computer & Mathematical Sciences, University of Mosul (1997-1999), Mosul, Iraq. She worked as Teacher B in Applied Science University (Private)-Amman-Jordan from 2000 to 2004. She received her PhD in CIS from the Department of CIS, AABFS- Amman-Jordan (2004-2008). During 2004-2008, she worked as a part time lecturer in the AABFS. Her research interests are in the fields of: Artificial Intelligence, Artificial Neural Networks, Image Processing, Web Engineering, and Software Process Models.

How to cite this paper: Omaima N. A. AL-Allaf,"Removing Noise from Speech Signals Using Different Approaches of Artificial Neural Networks", International Journal of Information Technology and Computer Science(IJITCS), vol.7, no.7, pp.8-18, 2015, DOI: 10.5815/ijitcs.2015.07.02