

Intelligent Arabic letters speech recognition system based on mel frequency cepstral coefficients

Anas Quteishat^{1,2}, Mahmoud Younis², Ahmed Qtaishat³, Anmar Abuhamdah⁴

¹Electrical Engineering Department, Faculty of Engineering Technology, Al-Balqa Applied University, Al-Salt, Jordan

²Electrical and Computer Engineering Department, Sohar University, Al Batana, Sultanate of Oman

³General Foundation Program, Sohar University, Al Batina, Sultanate of Oman

⁴Management Information Systems Department, Faculty of Business Administration, Taibah University, Al Madina, Saudi Arabia

Article Info

Article history:

Received Jun 8, 2022

Revised Jul 3, 2022

Accepted Jul 5, 2022

Keywords:

Back-propagation

Mel frequency cepstral coefficients

Neural networks

Voice recognition

ABSTRACT

Speech recognition is one of the important applications of artificial intelligence (AI). Speech recognition aims to recognize spoken words regardless of who is speaking to them. The process of voice recognition involves extracting meaningful features from spoken words and then classifying these features into their classes. This paper presents a neural network classification system for Arabic letters. The paper will study the effect of changing the multi-layer perceptron (MLP) artificial neural network (ANN) properties to obtain an optimized performance. The proposed system consists of two main stages; first, the recorded spoken letters are transformed from the time domain into the frequency domain using fast Fourier transform (FFT), and features are extracted using mel frequency cepstral coefficients (MFCC). Second, the extracted features are then classified using the MLP ANN with back-propagation (BP) learning algorithm. The obtained results show that the proposed system along with the extracted features can classify Arabic spoken letters using two neural network hidden layers with an accuracy of around 86%.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Anas Quteishat

Electrical Engineering Department, Faculty of Engineering Technology, Al-Balqa Applied University
Al-Salt, Jordan

Email: anas.quteishat@bau.edu.jo

1. INTRODUCTION

Communication is a channel that is used to transfer information, and emotions between humans. Speech is one of the most ancient, effective, and preferred methods of communication between people [1]. Over the past few decades' speech recognition has been the center of attraction for researchers [2]. The applications of speech recognition can be seen in telephony-based applications [3], mobile applications [4], car assistance applications [5], healthcare applications [6], and educational applications [7].

The process of recognizing spoken words involves many stages and these stages depend on the language's complexity. However, there are two main steps for this process, which are feature extraction and classification. Many feature extraction techniques exist in the literature, one type of feature extraction technique is linear predictive coding (LPC) [8]. This method compares two feature vectors in real-time at the expense of computational power, which makes this method unpopular in real-time applications.

Another extraction technique uses the cepstrum which can be defined as the inverse Fourier transform of the logarithm of the magnitude spectrum. Simple Euclidean distance can be computed by using the cepstrum which allows the similarity between two cepstral feature vectors. Furthermore, false-acceptance rate (FAR) and false-rejection rates (FRR) [9] for a speaker verification system derived from the LPC features result in

the best performance as demonstrated by Atal in his paper. Consequently, in this paper, the LPC-derived cepstrum is used as the feature extraction method.

Mel frequency cepstral coefficients (MFCCs) is the most popular feature extraction technique in speech recognition. MFCC technique evolved from the human peripheral auditory system. Several feature extraction techniques are present, but features extracted with the help of MFCCs are extensively used in automatic speech recognition (ASR) as well as in speaker recognition [10]–[13].

The MFCC is widely used in the literature for feature extraction for voice identification and recognition. In [14] a voice command recognition system using MFCC and vector quantization is presented. The system utilized the Euclidian distance to feature matching. A robust computer voice recognition is proposed in [15], the system is based on an improved MFCC algorithm. Pattern matching algorithms are used to identify the computer commands and an accuracy of 80% was achieved by this system. A deep learning algorithm was presented in [15], the system used enhanced MFCC to improve voice recognition rates. The presented work aimed to reduce the size of data in order to use voice recognition on devices with small memory.

The second stage of speech recognition is classification. When it comes to classifications artificial neural networks (ANNs) are known for their classification capabilities. Different neural networks are designed in the literature for classifications [16]–[24]. Such networks are the Multilayer perceptron. These capabilities can be found in different applications in the literature [17], [25], [26]. ANNs have been used in different applications such as medical applications [27]–[29], industrial applications [30], [31], and biometric security [32].

This paper aims to classify Arabic spoken letters and optimize the performance of multi-layer perceptron (MLP) ANN to obtain the highest testing accuracy. The optimization was done by changing the number of hidden layers and the number of neurons per layer. The methodology used in this paper consisted of generating a dataset for Arabic spoken letters, then extracting meaningful features using MFCC. The extracted features are then used to train different MLP neural networks and their performances are compared. The remaining of the paper is organized as follows; section 2 will introduce the method used which explains the MFCC feature extraction techniques. And the MLP neural network. The results and discussion are presented in section 3. Finally, section 4 will present the conclusion of the work.

2. METHOD

This section presents the methodology used to present this work. First the MFCC is presented and explained in details, such that the meaningful features can be extracted from it. Once the features are extracted, the training of MLP neural network will take place using these features.

2.1. Mel frequency cepstral coefficients

MFCC is commonly utilized for speech recognition. Introduced in the 1980s, MFCC are been the focus of scientists ever since [12], [32]–[36]. Previously linear prediction coefficients (LPCs) and linear prediction cepstral coefficients (LPCCs) techniques are used for feature extraction but MFCC is the dominant method among the feature extraction techniques [13]. This technique is given in the 1980s by Davis and Mermelstein [37]. This technique is based on the frequency domain which uses the Mel scale whose main source is the human ear scale. Features are based on two types which are frequency and time domain, as compared to time-domain frequency-domain features give more accurate results. To extract features from the audio recorded files, each file will undergo the following steps these steps are shown in Figure 1 and are described as:

- 1) Pre-Emphasis, in this step a high-frequency filter is used to emphasize the high frequencies in the audio signal [14]. Figure 2 shows the speech waveform shows that the spoken word is between 1 and 1.5 seconds. Figure 3 shows the pre-emphasis output waveform.
- 2) Frame blocking, a segmentation process of the speech signal is done in this step. Segments of 20-30 ms referred to as frames are created. A voice signal is segmented into N samples, and the adjacent frames are being separated by M, where ($M < N$). Usually $M=100$ and $N=256$ [14].
- 3) Hamming Windowing, for each frame created in step two, is multiplied with a hamming window to maintain the continuity of the original signal [14].

$$Y(n) = X(n) * W(n) \quad (1)$$

- where $W(n)$: the window function. Figure 4 shows the effect of applying the Hamming window on a signal.
- 4) Fast Fourier transform (FFT), is used to transform the signal from the time domain to the frequency domain, the result will be a magnitude frequency response for each frame [14].
 - 5) Band pass filters (BPFs), 20 triangular BPFs are multiplied by the magnitude of the frequency response, this helps in reducing the size of the features extracted from the MFCC [14].

$$\text{Mel}(f) = 1125 * \ln(1 + f/700) \tag{2}$$

6) The final step is the discrete cosine transform (DCT), the DCT is applied on the triangular BPFs to obtain the 13 Mel-scale cepstral coefficients [14] as shown in Figure 5.

The MFCC starts with the pre-emphasis phase. In this step, to obtain the same amplitude for all the high-frequency energies are emphasized and the first-order response of the pre-emphasis is done. A total of 13 MFCCs are extracted in this paper, and these 13 coefficients represent the training vector for one audio sample. Once the Mel coefficients are extracted, they are used to train the MLP neural network. The next section will briefly explain the MLP neural network and its learning algorithm.

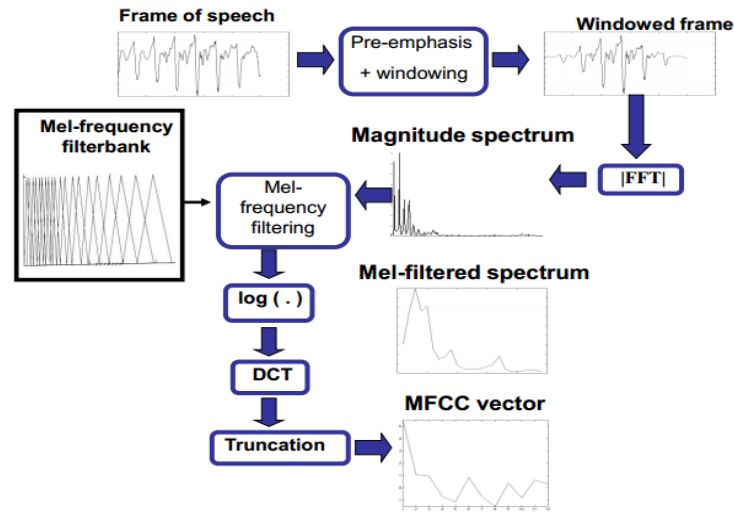


Figure 1. Mel frequency cepstral coefficients extraction

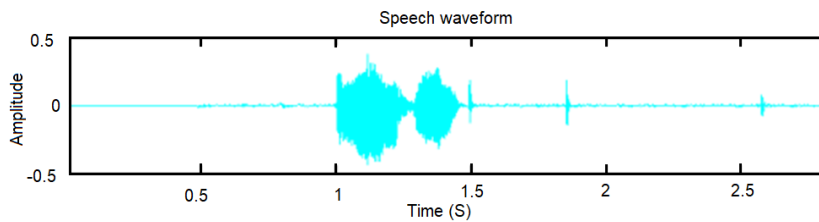


Figure 2. Speech waveform

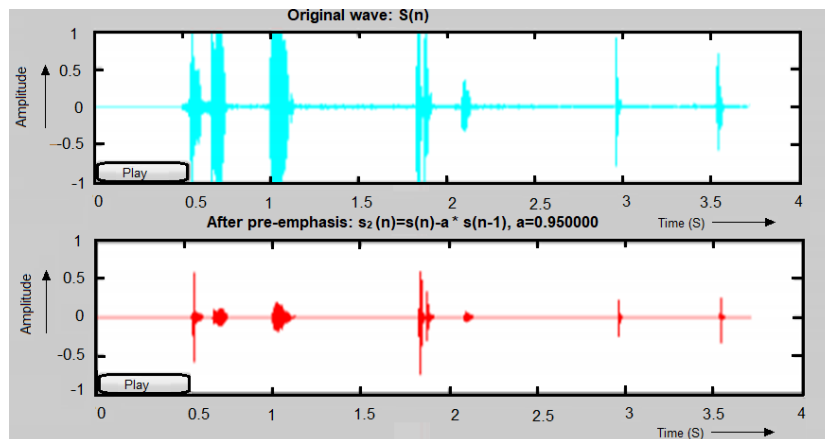


Figure 3. Pre-Emphasis waveform

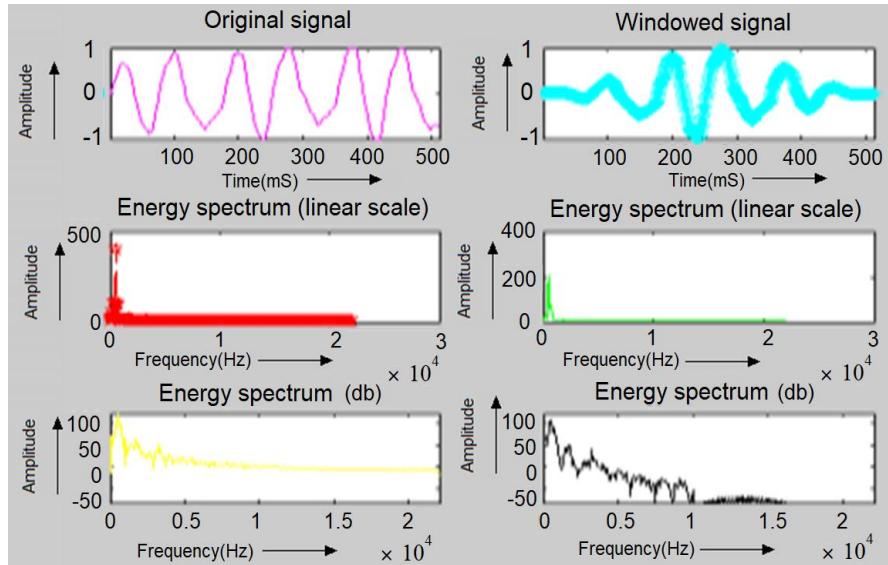


Figure 4. Edges become sharp by using hamming window

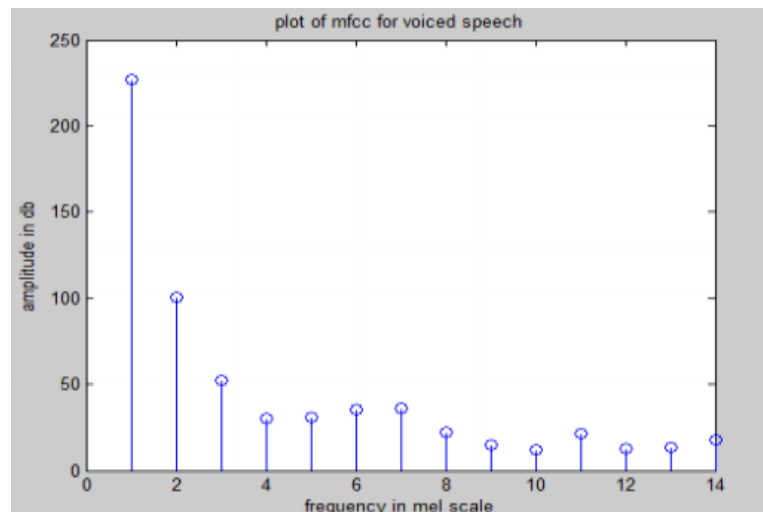


Figure 5. The plot of the MFCCs for one audio sample

2.2. Multi-layer perceptron

Artificial intelligence is a field that gains importance in the last two decades and growing eventually. MLP is one of the efficient types of neural network. The single neuron model is known as to be perceptron which is the originator of a large neural network. MLP constitutes a large set of perceptrons as shown in Figure 6. Each time addition is done to the network to enhance its efficiency so MLP is the addition to the feed-forward neural network (FFNN) [38]. This network mainly constitutes three layers which are input, output, and hidden layer. Signals are received at the input layer as an input to this layer which is processed to perform the required task. Hidden layers which are placed between the input and output layers are the computation engine for this network where all the tasks are performed.

Hidden layers are placed randomly in the system to find the optimal results [39]. MLP is mainly used for prediction and classification at the output layer. Data flow in MLP is in the forward direction (input to output layer) as in FFNN [40]. The neural network contains the neuron which has to be trained to get effective results so in MLP neurons are trained with a backpropagation learning algorithm. In Training the model parameters or weights are adjusted to minimize the error rate [41]. MLP is used to solve the non-linear, supervised, or continuous function type of problems. MLP gives efficient results in pattern classification, recognition, prediction, and approximation [42].

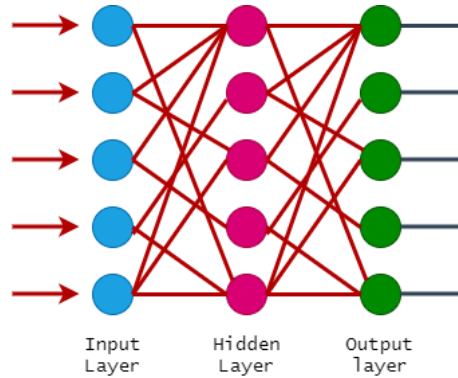


Figure 6. Representation of MLP network

There exist different types of learning algorithms for MLP neural networks. In this paper, the back-propagation (BP) learning algorithms are used. The BP learning algorithms consist of the following steps:

- Initialization

Randomly assign all the weights and thresholds within the following range:

$$\left(-\frac{2.4}{F_i}, +\frac{2.4}{F_i}\right) \quad (3)$$

where F_i represents the input neurons i , and the weight initialization is done for every neuron [42].

- Activation

Applying inputs $X_1(p), X_2(p), \dots, X_n(p)$ to the neural network, and compare it with the desired outputs $Y_{d1}(p), Y_{d2}(p), \dots, Y_{dn}(p)$. Calculate the actual outputs of the neurons in the hidden layer:

$$Y_j(p) = \text{sigmoid} [\sum_{i=1}^n x_i(p) * w_{ij}(p) - \phi_j] \quad (4)$$

where n is the number of inputs of neuron, j in the hidden layer, and sigmoid is the sigmoid activation function. Calculate the actual outputs of the neurons in the output layer:

$$Y_k(p) = \text{sigmoid} [\sum_{j=1}^m X_{jk}(p) * W_{jk}(p) - \phi_k] \quad (5)$$

where m is the number of inputs of neuron k in the output layer [42]

- Weight training

Update the weights in the BP network propagating Backward the errors associated with output neurons. Find the error gradient for the neurons in the output layer:

$$\delta_k(p) = y_k(p) * (1 - Y_k(p)) * e_k(p) \quad (6)$$

where:

$$e_k(p) = Y_{d,k}(p) - Y_k(p) \quad (7)$$

Determine the weight corrections:

$$\Delta W_{jk}(p) = \alpha * Y_j(p) * \delta_k(p) \quad (8)$$

Calculate the updated the weights at the output neurons:

$$W_{jK}(P+1) = W_{jk}(p) + \Delta W_{jk}(p) \quad (9)$$

Estimate the error gradient for the neurons in the hidden layer:

$$\delta_j(p) = Y_j(p) * [1 - Y_j(p)] * \sum \delta_k(p) * W_{jk}(p) \quad (10)$$

Evaluate the weight corrections:

$$\Delta W_{ij}(p) = \alpha * x_i(p) * \delta_j(p) \quad (11)$$

Update the weights at the hidden neurons:

$$W_{ij}(p+1) = W_{ij}(p) + \Delta W_{ij}(p) \quad (12)$$

- Iteration

The iteration step is needed to make sure that the stopping condition is satisfied. The iteration is done by the increment of the value of the iteration p by one, if the error criterion is not satisfied go to step 2, else stop.

3. RESULTS AND DISCUSSION

This section will present the data collected for the experiment, and the results obtained from training the neural network with features extracted using MFCC. MATLAB application provides a neural network pattern recognition tool that helps in selecting data, creating and training the network, and evaluating its performance. Throughout this section, all these procedures are explained.

The dataset used in this paper consisted of all 28 Arabic alphabetical letters. Thirty-three people of different ages and gender participated in generating this dataset. Each individual recorded 28 recordings one for each letter. A total of 924 (28x33) samples belonging to 28 different classes.

An MFCC code was developed to enable MATLAB application understands these voices. Using MFCC instructions three matrixes are obtained [FBEs, frames, MFCCs], such that each matrix contains 13 numerical row features associated with signals voice. The dataset used to train the MLP consisted of 924 vectors belonging to 28 classes. Each vector consists of 13 features.

The dataset is divided into three sets, training validation, and testing. The training dataset contained 60% of the total number of samples, whereas the validation and testing datasets each contained 20% of the dataset. The datasets are randomized before each run on the neural network to ensure different learning.

To distinguish between different types of voices MFCCs matrix is adopted and normalizing the numerical numbers between 0's and 1's by using Microsoft excel to build the neural network. The network is trained using different feature matrixes many times, and different results are obtained that distinguish between different sample features. In Table 1, MFCCs features are the best between filter-bank energies (FBEs) and MFCCs methods, because it has the highest correctness percentage, the network took a low number of epochs, and the performance was low. Therefore, MFCCs features are used in this paper.

Table 1. Neural network features

Features	Number of Epochs	Percentage of Correct	Performance
<i>FBEs</i>	1400	78.680%	0.0188
<i>MFCCs</i>	960	84.740%	0.0180
<i>FBEs and MFCCs</i>	1128	78.742%	0.0198

MATLAB NNTool is used for creating a neural network system. This tool requires input data and target data, which classify the inputs into the set of target categories. The input data is considered as features that had been extracted from 33 letters voice samples in which the voices are already known. Target data was a (28x925) matrix, this matrix contain 28 classes for letters. The NN training tool demonstrates the network being trained and the algorithms used for training. It also displays the training state during training and the stopping criterion for training is highlighted in green. During the training process, some criteria are tested until a trusted and reliable trained network is reached, these criteria are explained and discussed as:

- Maximum number of epochs: training neural network requires a large dataset that involves many data items. Iteration in a neural network that is being trained is data items pass into the neural network one by one, which is called iteration, and when the whole dataset goes through it, that is called an epoch. Determining the maximum number of epochs means that only the affixed number of passes through the training data cannot be exceeded. To train the network system, 1,700 epochs were obtained as a maximum number of epochs.
- Error rate: by observing the performance plot of the trained network, it is obvious that the network stops automatically when the mean square error (the average squared difference between output and target) gradient falls into its minimum value. To reach the level of speed/accuracy desired min-grad value was set to (1e-005) in the trained network, where zero means no error.

- Learning rate: learning rate is the amount of change the neural network weight can exhibit after each iteration. For a neural network to learn faster, a learning factor is set to a higher value. However, the network may not learn in a good manner if there is a large variability in the input set. Therefore, it is better to set the factor to a small value and increase it gradually if the learning rate seems slow. To train the system network, the learning rate of 0.01 is obtained as the best value.
- Size of hidden layers: deciding your overall neural network architecture requires deciding the number of neurons in the hidden layers. However, layers do not directly interact with the external environment, they have a fabulous influence on the final output. Hence, consideration should be made in both the number of hidden layers and the number of neurons in each of these hidden layers. In the following subsections, the number of neurons and the number of hidden layers are studied to get the optimum network size.

The conducted experiments are divided into three different experiments, the first set of experiments studied the change in the number of alphabets the network tries to classify. The second set of experiments studied the change in a number of neurons with a single hidden layer. While the third set of experiments studied the change in a number of hidden layers along with the number of neurons per layer. A sample of the experimental setup is shown in Figure 7.

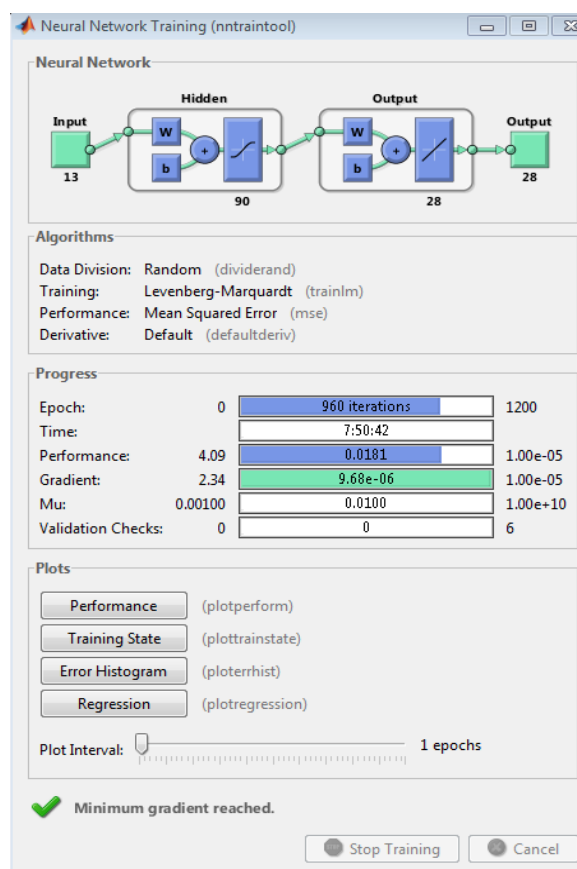


Figure 7. An experimental setup samples

3.1. Number of alphabets

In this subsection, the change in the number of alphabets is studied. The study was conducted using a neural network with one hidden layer and 10 neurons. Table 2 shows that when the network trained 5 times using 10 different letters, where the number of epochs change for each train, the average for the number of epochs was 285.6, and the percentage of correct did not change for each train. The average percentage of correctness is 100%.

Table 3 shows when the network is trained 5 times with 18 letters, the number of epochs changed for each train, and their average number is 597.8, results show the percentage of correctness changes slightly as the number of training letters increases to 18 giving an accuracy around 98.51%. According to Tables 2 and 3, when more letters are used for training, the number of epochs increases, and the percentage of correct decreases

gradually. Table 4 results parameters when the network is trained 5 times with 28 letters and 10 neurons. Results show the number of epochs changes for each new train, the average for a number of epochs is 125.6, and the percentage of correctness changes for each extra train, the average percentage of correct is 30.45%, which is really low.

Table 2. Training data for (10) letters with 10 neurons and one hidden layer

Number of trains	Number of epochs	Percentage of correct
1	177	100%
2	372	100%
3	431	100%
4	192	100%
5	256	100%
Average	285.6	100%

Table 3. Training data for (18) letters with 10 neurons and one hidden layer

Number of trains	Number of epochs	Percentage of correct
1	350	97.98%
2	444	99.16%
3	348	97.98%
4	847	98.32%
5	1000	97.31%
Average	597.8	98.51%

3.2. Number of neurons with one hidden layer

In this subsection, the number of neurons in a single hidden layer is increased and the changes in the testing accuracy are observed. As discussed in Table 4, the testing accuracy decreased dramatically when all 28 alphabets are trained on a 10-neuron neural network. Table 5 illustrates results when the number of neurons is increased to 30, the average percentage of correctness increases significantly and becomes 50.21%. Also interestingly, the average number of required epochs is increased to 170.8.

However, in Table 6 results when the number of neurons is set to 90, the average percentage of correctness becomes more accurate 84.83%. However, the required average number of epochs becomes 839. The time needed to train this network was around 7 hours and (50) minutes.

Table 4. Training data for (28) letters with 10 neurons and one hidden layer

Number of trains	Number of epochs	Percentage of correct
1	53	30.62%
2	99	32.57%
3	103	29.22%
4	88	29.65%
5	285	30.41%
Average	125.6	30.45%

Table 5. Training data for (28) letters using 30 neurons with one hidden layer

Number of Train	Number of epochs	Percentage of correct
1	168	50.64%
2	153	52.72%
3	184	49.13%
4	207	48.81%
5	142	49.78%
Average	170.8	50.21%

Table 6. Training data for (28) letters 90 neurons with one hidden layer

Number of Train	Number of epochs	Percentage of correct
1	960	84.74%
2	1200	86.58%
3	303	83.23%
4	533	84.96%
5	1200	84.63%
Average	839	84.83%

3.3. Number of hidden layers and neurons

In this subsection, both the number of hidden layers and the number of neurons per layer are studied. The number of hidden layers is increased from 1 to two hidden layers, and the number of neurons per layer has changed. Results in Table 7 when the network is trained 5 times using 28 letters with [30 30] hidden layers, the average for the number of epochs is 1,700, and the percentage of correct changes for each train such that the average of correctness percentages is 73.76%. Comparing these results with a single-layer neural network having 30 neurons Table 5 one can notice the improvement of adding another layer to the network. This improvement comes with the cost of increasing the network complexity.

In Table 6, a total of 90 neurons were used to improve results, the hidden layers are increased to [30 60], Table 8 shows training results for 5 times, and the average of correctness percentages is improved to 85.78%. Table 9 on the other hand, has hidden layers set to [45 45], to improve the accuracy, even more, therefore, the testing accuracy percentages are improved to 89.65%, However, the improvement is small and the training time increased with more layers.

Table 7. Data for (28) letters when hidden layers [30 30]

Number of Train	Number of epochs	Percentage of correct
1	1700	77.272%
2	1700	72.186%
3	1700	75.000%
4	1700	71.864%
5	1700	72.511%
Average	1700	73.766%

Table 8. Data for (28) letters when hidden layers [30 60]

Number of Train	Number of epochs	Percentage of correct
1	1700	88.420%
2	1700	81.170%
3	1700	85.988%
4	1700	86.688%
5	1700	86.670%
Average	1700	85.787%

Table 9. Data for (28) letters when hidden layers [45 45]

Number of Train	Number of epochs	Percentage of Correct
1	1700	88.745%
2	1700	89.419%
3	1700	91.017%
4	1700	90.369%
5	1700	88.744%
Average	1700	89.658%

4. CONCLUSION

This paper introduced an intelligent system used to recognize the Arabic alphabets which are a total of 28. The proposed system utilized the MFCC technique to extract meaningful features from the spoken Arabi letters. These features are then used to train a multilayer perceptron neural network with a back propagation learning algorithm. The dataset is collected from the different native speakers of the language which include people of different ages and gender. A total of 33 different people participated in creating this dataset. A series of experiments were conducted and testing accuracy of around 90% was achieved.




REFERENCES

- [1] M. Malik, M. K. Malik, K. Mehmood, and I. Makhdoom, "Automatic speech recognition: a survey," *Multimedia Tools and Applications*, vol. 80, no. 6, pp. 9411–9457, Mar. 2021, doi: 10.1007/s11042-020-10073-7.
- [2] M. Benzeghiba *et al.*, "Automatic speech recognition and speech variability: A review," *Speech Communication*, vol. 49, no. 10–11, pp. 763–786, Oct. 2007, doi: 10.1016/j.specom.2007.02.006.
- [3] S. Kantharajah, "Real time telephony based speech recognition solution for mobile phones to assist the hearing - impaired," PhD Thesis, University of Westminster (UOW), 2017.
- [4] Y. He *et al.*, "Streaming end-to-end speech recognition for mobile devices," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 6381–6385, doi: 10.1109/ICASSP.2019.8682336.
- [5] A. Czyzewski, B. Kostek, P. Bratoszewski, J. Kotus, and M. Szykalski, "An audio-visual corpus for multimodal automatic speech recognition," *Journal of Intelligent Information Systems*, vol. 49, no. 2, pp. 167–192, Oct. 2017, doi: 10.1007/s10844-016-0438-z.
- [6] K. C. Santosh, "Speech processing in healthcare: Can we integrate?," in *Intelligent Speech Signal Processing*, 2019, pp. 1–4.
- [7] L.-P. Tuca and A. Iftene, "Speech recognition in education: Voice geometry painter application," in *2017 International Conference on Speech Technology and Human-Computer Dialogue (SpED)*, Jul. 2017, pp. 1–8, doi: 10.1109/SPED.2017.7990446.
- [8] D. O'Shaughnessy, "Linear predictive coding," *IEEE Potentials*, vol. 7, no. 1, pp. 29–32, Feb. 1988, doi: 10.1109/45.1890.
- [9] Z. Zhang, H. Wang, A. V. Vasilakos, and H. Fang, "ECG-cryptography and authentication in body area networks," *IEEE Transactions on Information Technology in Biomedicine*, vol. 16, no. 6, pp. 1070–1078, 2012, doi: 10.1109/TITB.2012.2206115.
- [10] A. Ailane, M. Eljourmi, and S. Khamlich, "ANN MFCC and its applications in speaker recognition," in *Fourth International Conference on Advances in Computing, Electronics and Communication - ACEC 2016*, Dec. 2016, doi: 10.15224/978-1-63248-113-9-75.
- [11] M. Bezoui, A. Elmoutaouakkil, and A. Beni-hssane, "Feature extraction of some quranic recitation using mel-frequency cepstral coefficients (MFCC)," in *2016 5th International Conference on Multimedia Computing and Systems (ICMCS)*, Sep. 2016, pp. 127–131, doi: 10.1109/ICMCS.2016.7905619.
- [12] H. A. Elharati, M. Alshaari, and V. Z. Kępuska, "Arabic speech recognition system based on MFCC and HMMs," *Journal of Computer and Communications*, vol. 08, no. 03, pp. 28–34, 2020, doi: 10.4236/jcc.2020.83003.
- [13] N. Dave, "Feature extraction methods LPC, PLP and MFCC in speech recognition," *International Journal for Advance Research in Engineering and Technology*, vol. 1, no. 6, pp. 1–5, 2013.
- [14] M. Shaneh and A. Taheri, "Voice command recognition system based on MFCC and VQ algorithms," *World Academy of Science, Engineering and Technology International Journal of Computer and Information Engineering*, vol. 3, no. 9, pp. 2231–2235, 2009.
- [15] C. Goh and K. Leon, "Robust computer voice recognition using improved MFCC algorithm," in *2009 International Conference on New Trends in Information and Service Science*, Jun. 2009, pp. 835–840, doi: 10.1109/NISS.2009.12.
- [16] A. Quteishat, C. P. Lim, and K. S. Tan, "A modified fuzzy min-max neural network with a genetic-algorithm-based rule extractor for pattern classification," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 40, no. 3, pp. 641–650, May 2010, doi: 10.1109/TSMCA.2010.2043948.
- [17] N. Arunkumar, M. A. Mohammed, S. A. Mostafa, D. A. Ibrahim, J. J. P. C. Rodrigues, and V. H. C. Albuquerque, "Fully automatic model-based segmentation and classification approach for MRI brain tumor using artificial neural networks," *Concurrency and Computation: Practice and Experience*, vol. 32, no. 1, Jan. 2020, doi: 10.1002/cpe.4962.




- [18] A. Quteishat and C. P. Lim, "A modified fuzzy min-max neural network with rule extraction and its application to fault detection and classification," *Applied Soft Computing*, vol. 8, no. 2, pp. 985–995, Mar. 2008, doi: 10.1016/j.asoc.2007.07.013.
- [19] P. G. Campos, E. M. J. Oliveira, T. B. Ludermir, and A. F. R. Araujo, "MLP networks for classification and prediction with rule extraction mechanism," in *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No.04CH37541)*, 2004, vol. 2, pp. 1387–1392, doi: 10.1109/IJCNN.2004.1380152.
- [20] C. P. Lim and R. F. Harrison, "Online pattern classification with multiple neural network systems: an experimental study," *IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews)*, vol. 33, no. 2, pp. 235–247, May 2003, doi: 10.1109/TSMCC.2003.813150.
- [21] C.-F. Juang and C.-M. Chang, "Human body posture classification by a neural fuzzy network and home care system application," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 37, no. 6, pp. 984–994, Nov. 2007, doi: 10.1109/TSMCA.2007.897609.
- [22] R. O. Duda and P. E. Hart, *Pattern classification and scene analysis*. Wiley; 1st edition, 1973.
- [23] D. Nauck and R. Kruse, "A neuro-fuzzy method to learn fuzzy classification rules from data," *Fuzzy Sets and Systems*, vol. 89, no. 3, pp. 277–288, Aug. 1997, doi: 10.1016/S0165-0114(97)00009-2.
- [24] S. Mukhopadhyay, S. Peng, R. Raje, M. Palakal, and J. Mostafa, "Multi-agent information classification using dynamic acquaintance lists," *Journal of the American Society for Information Science and Technology*, vol. 54, no. 10, pp. 966–975, Aug. 2003, doi: 10.1002/asi.10292.
- [25] M. N. A. Alqumboz and S. S. Abu-Naser, "Avocado classification using deep learning," *International Journal of Academic Engineering Research (IJAER)*, vol. 3, no. 12, 2020.
- [26] E. Baralis, S. Chiusano, and P. Garza, "A lazy approach to associative classification," *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 2, pp. 156–171, Feb. 2008, doi: 10.1109/TKDE.2007.190677.
- [27] G.-P. K. Economou *et al.*, "Medical diagnosis and artificial neural networks: a medical expert system applied to pulmonary diseases," in *Proceedings of IEEE Workshop on Neural Networks for Signal Processing*, pp. 482–489, doi: 10.1109/NNSP.1994.366018.
- [28] D. L. Hudson, M. E. Cohen, and M. F. Anderson, "Use of neural network techniques in a medical expert system," *International Journal of Intelligent Systems*, vol. 6, no. 2, pp. 213–223, Mar. 1991, doi: 10.1002/int.4550060208.
- [29] A. Quteishat and C. P. Lim, "Application of the fuzzy min-max neural networks to medical diagnosis," in *Knowledge-Based Intelligent Information and Engineering Systems*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 548–555, doi: 10.1007/978-3-540-85567-5_68.
- [30] A. Abdelhadi and L. H. Mouss, "An overview of artificial immune system algorithms for industrial monitoring," *International Review on Computers and Software*, vol. 6, no. 2, pp. 269–274, 2011.
- [31] E. E. Mangina, S. D. J. McArthur, J. R. McDonald, and A. Moyes, "A multi agent system for monitoring industrial gas turbine start-up sequences," *IEEE Transactions on Power Systems*, vol. 16, no. 3, pp. 396–401, 2001, doi: 10.1109/59.932274.
- [32] A. Quteishat, C. P. Lim, C. C. Loy, and W. K. Lai, "Authenticating the identity of computer users with typing biometrics and the fuzzy min-max neural network," *International Journal of Biomedical Soft Computing and Human Sciences: the official journal of the Biomedical Fuzzy Systems Association*, vol. 14, no. 1, pp. 47–53, 2009, doi: 10.24466/ijbschs.14.1_47.
- [33] M. Sahidullah and G. Saha, "Design, analysis and experimental evaluation of block based transformation in MFCC computation for speaker recognition," *Speech Communication*, vol. 54, no. 4, pp. 543–565, May 2012, doi: 10.1016/j.specom.2011.11.004.
- [34] M. Xu, L.-Y. Duan, J. Cai, L.-T. Chia, C. Xu, and Q. Tian, "HMM-based audio keyword generation," in *Advances in Multimedia Information Processing - PCM 2004*, 2004, pp. 566–574, doi: 10.1007/978-3-540-30543-9_71.
- [35] V. Bhardwaj and V. Kukreja, "Effect of pitch enhancement in punjabi children's speech recognition system under disparate acoustic conditions," *Applied Acoustics*, vol. 177, Jun. 2021, doi: 10.1016/j.apacoust.2021.107918.
- [36] A. M. Quteishat, M. Al-Batah, A.-M. Anwar, and S. H. Alnabelsi, "Cervical cancer diagnostic system using adaptive fuzzy moving k-means algorithm and fuzzy min-max neural network," *Journal of Theoretical and Applied Information Technology*, vol. 57, no. 1, pp. 48–53, 2013.
- [37] S. B. Davis and P. Mermelstein, "Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 28, no. 4, pp. 357–366, 1980, doi: 10.1109/TASSP.1980.1163420.
- [38] N. K. Manaswi, "Multilayer perceptron," *Deep Learning with Applications Using Python*, pp. 45–56, 2018.
- [39] W. Castro, J. Oblitas, R. Santa-Cruz, and H. Avila-George, "Multilayer perceptron architecture optimization using parallel computing techniques," *PLOS ONE*, vol. 12, no. 12, Dec. 2017, doi: 10.1371/journal.pone.0189369.
- [40] J. Tang, C. Deng, and G.-B. Huang, "Extreme learning machine for multilayer perceptron," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 4, pp. 809–821, Apr. 2016, doi: 10.1109/TNNLS.2015.2424995.
- [41] L. Grippo, A. Manno, and M. Sciandrone, "Decomposition techniques for multilayer perceptron training," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 11, pp. 2146–2159, Nov. 2016, doi: 10.1109/TNNLS.2015.2475621.
- [42] H. Ramchoun, M. A. J. Idrissi, Y. Ghanou, and M. Ettaouil, "Multilayer perceptron: Architecture optimization and training with mixed activation functions," in *ACM International Conference Proceeding Series*, 2017, no. 1, pp. 26–30, doi: 10.1145/3090354.3090427.

BIOGRAPHIES OF AUTHORS






Anas Quteishat    received his B.Sc. in Electronic Engineering from Princess Sumaya University for Technology, Jordan in 2003. In 2005 he finished his M.Sc. in Electronic System Design from the University of Science Malaysia. He got his Ph.D. in 2009 in Computational Intelligence from the University of Science Malaysia. His research interest includes artificial neural networks, pattern recognition, intelligent systems, and embedded systems. He has published many research papers in international journals. Currently, he is the program coordinator for Electrical and Computer Engineering at Sohar University-Oman. He is also on leave from Al-Balqa Applied university-Jordan, where he was an associate professor. He can be contacted at email: anas.quteishat@bau.edu.jo.






Mahmoud Younis    is an associate professor at the Faculty of Engineering at Sohar University, Oman. He received his Ph.D. in Power Electronics Engineering from the University of Malaya, Malaysia. His research interest is in the field of electrical power engineering, with a particular focus on power electronics and renewable energy systems. Dr. Younis has participated in many collaborative research projects funded by various bodies including the Oman Research Council. Dr. Younis is the author (co-author) of several research papers indexed by the Web of Science. He can be contacted at email: myounis@su.edu.om.



Ahmed Qtaishat    has a bachelor's degree in computer science from Applied Science University-Jordan in 2005, he finished his master's degree from University Utara Malaysia, in intelligent systems in 2007. He joined Sohar University in 2011. From 2012 until 2017 he worked as a coordinator in the General Foundation Program. His research area focuses on artificial intelligence, machine learning, and genetic algorithm. He has published several international papers. He can be contacted at email: aqtaishat@su.edu.om.



Anmar Abuhamdah    is an Associate Professor in the Department of MIS, Faculty of Business Administration, at Taibah University, KSA. He received his bachelor's degree from the Computer Science Department at Princess Sumaya University for Technology, Jordan in 2003. He received his master's degree from Intelligent Systems Department at Utara University in Malaysia in 2007. He received his Ph.D. in Intelligent Research algorithms-Computer Science from the National University of Malaysia in August 2011. His research interests are mainly directed to Metaheuristics and Combinatorial Optimization Problems including Course and Exam Timetabling, Travelling Salesman, Nurse Rostering Problems, real time system, and software development. He has published a number of high-quality research papers in international journals and conferences. He can be contacted at email: aabuhamdah@taibahu.edu.sa.