ACCEPTED: A. Beg and S. K. Hasnain, "A Speech Recognition System for Urdu Language," in Lecture Notes in Computer Science (LNCS) Wireless Networks, Information Processing and Systems: Springer, 2008, pp. 118-126.

A Speech Recognition System for Urdu Language

Azam Beg[1] and S. K. Hasnain[2]

[1]College of Information Technology, UAE University, United Arab Emirates
[2]Pakistan Navy Engineering College, Karachi, Pakistan

[1]abeg@uaeu.ac.ae, [2]hasnain@pnec.edu.pk

Abstract. This paper investigates use of a machine learnt model for recognition of individually words spoken in Urdu language. Speech samples from many different speakers were utilized for modeling. Original time-domain samples are normalized and pre-processed by applying discrete Fourier transformation for speech feature extraction. In frequency domain, high degree of correlation was found for the same words spoken by different speakers. This helped produce models with high recognition accuracy. Details of model realization in MATLAB are included in this paper. Current work is being extended using linear predictive coding for efficient hardware implementation.

Key words: Urdu speech processing, feature extraction, speaker independent system, machine learning, data pre-processing, modeling.

1 Introduction

Speech processing is a diverse, well-researched topic that finds applications in telecommunication, multi-media and other fields. Speech processing in real-time is more challenging than off-line processing. The processing has many facets, for example, distinguishing different utterances, speaker identification, etc.

Many years ago, von Kempelen showed that the speech production system of the human beings could be modeled. He demonstrated this by building a mechanical contrivance that “talked.” The paper by Dudley and Tarnocyz [1] relates the history of von Kempelen’s speaking machine.

Sounds are mainly categorized into these groups: *voiced* sounds (e.g., vowels and nasals), *unvoiced* sounds (e.g., fricatives), and stop-consonants (e.g., plosives). The speech starts in lungs but is actually formed when the air passes through larynx and vocal tracts [2]. Depending on the status of vocal fold in larynx, the sound can be grouped into: voiced sound that is time-periodic in nature and harmonic in frequency; and the unvoiced sound which is more noise-like [3].

Speech modeling can be divided into two types of coding: *waveform* and *source* [4]. In the beginning, the researchers tried to mimic the sounds as is, and called the technique waveform coding. This method tries to retain the original waveform using quantization and redundancy. An alternative approach makes use of breaking the sound up into individual components which are later modeled, separately. This method of utilizing parameters is referred to as source coding.

Different characteristics of speech can used to identify the spoken words, the gender of the speaker, and/or the identity of the speaker. Two important features of speech are *pitch* and *formant frequencies* [5]:

(a) Pitch is a significant distinguishing factor among male and female speakers. The frequency of vibration of vocal folds determines the pitch, for example, 300 times per second oscillation of the folds results in 300 Hz pitch. Harmonics (integer multiples of fundamental frequency) are also created while the air passes through the folds. The age also affects the pitch. Just before puberty, the pitch is around 250 Hz. For adult males, the average pitch is 60 to 120 Hz, and for females, it is 120 to 200 Hz.

(b) The vocal tract, consisting of oral cavity, nasal cavity, velum, epiglottis, and tongue, modulates the pitch harmonics created by the pitch generator. The modulations depend on the diameter and length of the cavities. These reverberations are called formant frequencies (or resonances). The harmonics closer to the formant frequencies get amplified, while others are attenuated.

While the humans are speaking, the formants vary depending on the positions of tongue, jaw, velum and other parts of the vocal tract. Two related key factors are: bandwidth of each formant, and formant's membership in a known bandwidth. The vowels for all human beings tend to be similar [5].

Each vowel uttered by a person generates different formants. So we can say that the vocal tract is a variable filter whose inputs are (1) the pitch, and (2) the pitch harmonics. The output of the filter is the gain or the attenuation of the harmonics falling in different formant frequencies. The filter is called *variable filter* model. The transfer function for the filter is determined by the formant frequencies [5].

Rabiner and Schafer's [6] discrete-time model make makes use of linear prediction for producing speech. The vocal tract and lip radiation models use discrete excitation signal. An impulse generator emulates voiced speech excitation; the impulses are passed through a glottal shaping filter. The unvoiced speech is generated by a random noise generator.

Ideally, any features selected for a speech model should be (1) not purposely controlled by the speaker, (2) independent of his/her health condition, and (3) tolerant to any environmental/acquisition noise.

Although the pitch can be easily varied by a speaker, the pitch can be easily filtered for any electrical noise, by using a low-pass filter. Formants on the other hand, are unique for different speakers, and are useful in individual speaker identification. In general, a combination of pitch and formant can be used in a speech recognition system.

Many different schemes have been used in the past for speech feature extraction, for example, *discrete Fourier transform (DFT), linear predictive coding (LPC), cepstral analysis,* etc [7]. (In this paper, we only use the DFT technique).

Much research has been on done for English (and other major languages) speech processing/recognition, but application of these time-tested techniques has not been investigated for Urdu language. So we consider ours to be the first effort in developing an Urdu language speech recognition system. In this paper, we provide the system (in the form of MATLAB code) so that other researchers can build up on our current efforts.

In Section 2, we include overviews of Fourier transformation and neural networks (NNs). Section 3 covers the speech data acquisition and pre-processing, whereas Section 4 discusses how NNs are used for modeling using MATLAB. In the same section, the results are also included. At the end, we present conclusions and our plan for extending the current work.

2 Preliminaries

2.1 Overview of Discrete Fourier Transform

DFT is itself a sequence rather than a function of continuous variable and it corresponds to equally-spaced frequency samples of discrete time Fourier transform of a signal. Fourier series representation of the periodic sequence corresponds to discrete Fourier transform of finite length sequence. So we can say that DFT is used for transforming discrete time sequence *x(n)* of finite length into discrete frequency sequence *X[k]* of finite length [2].

DFT is a function of complex frequency. Usually the data sequence being transformed is real. A waveform is sampled at regular time intervals T to produce the sample sequence of N sample values; n is the sample number, from $n = 0$ to N-1.

$$\{x(nT)\} = x(0), x(T), \ldots, x[(N-1)T] \quad (1)$$

For length-N input vector X, the DFT is a length-N vector X that has elements:

$$X(k) = \sum_{n=1}^{N} x(n).e^{-2\pi\, j.(k-1).(\frac{n-1}{N})},\ \ k = 1 \ldots N. \quad (2)$$

MATLAB function *fft(X)* calculates the DFT of vector *X*. Similarly, *fft(X, N)* calculates *N*-point FFT, but padded with zeros if *X* has fewer than *N* points, and truncated if it has more.

2.2 Overview of Neural Networks

NNs have proven to be a powerful tool for solving problems of prediction, classification and pattern recognition [8]-[20]. The NNs are based on the principle of biological neurons. An NN may have one or more input and output neurons as well as one or more *hidden* layers of neurons interconnecting the input and output neurons. In one of the well-known NN types, the outputs of one layer of neurons send data (only) to the next layer (Fig. 1), thus being called *feed-forward NNs. Back-propagation* is a common scheme for creating (*training*) the NNs. During the process of NN-creation, internal *weights* ($w_{i,j}$) of the neurons are iteratively adjusted so that the outputs are produced within desired accuracy. The training process requires that the *training set* (known examples/input-output datasets) be chosen carefully. The selected dataset usually needs to be pre-processed prior to being fed to a NN [19][20].

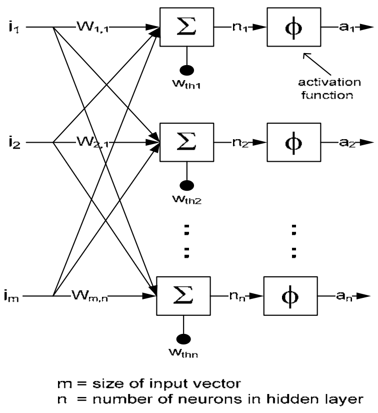

Fig. 1. Structure of a simple feed-forward neural network.

3 Speech Data Acquisition and Processing

The speech recognition system presented in this paper is limited to individual Urdu numerals (0 to 9). The data was acquired by speaking the numerals into a microphone connected to MS-Windows-XP based PC. Fifteen speakers uttered the same number set (0 to 9), specifically, *aik, do, teen, chaar, paanch, chay, saat, aat,* and *nau.* All sound samples were curtailed to the same time duration.

The following data processing steps can be generally used for preparing the data for NN training [19]-[22]:

(1) *Pre-processing*: This step may involve logarithmic or some transformation. Normalization may also be needed.

(2) *Data length adjustment*: DFT execution time depends on exact number of the samples (*N*) in the data sequence [*xK*]. It is desirable to choose the data length equal to a power of two.

(3) *Endpoint detection*: This step isolates the word to be detected from the following 'silence'.

(4) *Windowing*: A window is applied to select a smaller time than the complete sample. *Hamming window* is one of the widely used windowing algorithms.

(5) *Frame blocking*: Here we make use of the property that the sounds originate from a mechanically slow vocal tract. This assumed stationarity allows overlapping frames of 100 ms or so.

(6) *Fourier transform*: MATLAB's built-in DFT function provides symmetric data from which the lower half can be used for NN training/testing. Frequency spectrum for numeral *aik* (one) is shown in Fig. 2.

We observed that Fourier description of the same number uttered by different speakers had high correlation. Fig. 3 shows the surface plot for correlation of number *aik* (one) by 15 different speakers.

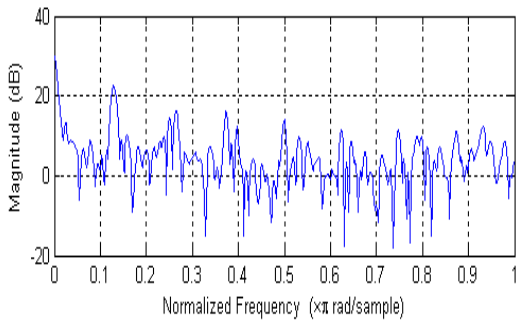

Fig. 2. Frequency representation of number *aik* (one).

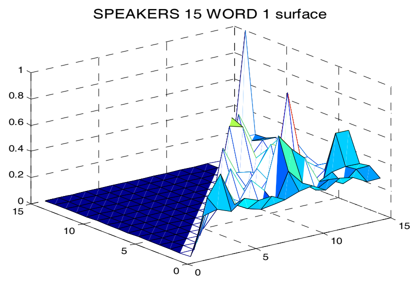

Fig. 3. The surface plot of the correlation of the spoken Urdu numbers spoken *aik* (one) by 15 different speakers.

4 Neural Network Topology and Implementation in MATLAB

We used multi-layer feed-forward networks in this research. An N-layer feed-forward NN is created by *newff* MATLAB command [23]:

```
net = newff(PR,[S1, S2, ..., SNl],{TF1
TF2...TFNl},BTF,BLF,PF)
```

where

PR =	Rx2 matrix of min and max values for R input elements
S1 =	Size of ith layer, for Nl layers
TFi =	Transfer function of ith layer. (Default = 'tansig')
BTF =	Backprop network training function. (Default = 'trainlm')
BLF =	Backprop weight/bias learning function. (Default = 'learngd')
PF =	Performance function (default = 'mse')

The NN is created and trained with the following set of MATLAB commands:

```
% p[] = array of training inputs
% t[] = array of targets/outputs
% minmax_of_p = array containing
%        mins and maxs of p[]

% network configuration
S1 = 10; % layer-1 neurons
S2 = 10; % layer-2 neurons
net = []; % network reset

% two-layer network declaration
net = newff(minmax_of_p,[S1, S2],
{'tansig','purelin'},'traingd');

% training and display parameters
net.trainParam.show = 50;
net.trainParam.lr = 0.01;% lrn rate
net.trainParam.goal = 0.01; % trg tol
net.trainParam.epochs = 5000;
% finally the training happens with:
net = train(net,p,t);
```

From the overall dataset of 150 speech samples (10 numbers spoken by 15 different speakers), we used 90% as the *training set*, and set aside the remaining 10% as *validation/testing set*. The learning rates (LRs) of 0.01 and 0.05, and training error tolerance of 0.01 were used. Maximum *epoch* count was limited to 5000.

The NNs used 64 FFT magnitudes as inputs, and produced ten separate predictions (one for each number). This means that we had 64 neurons in the input layer, and 10 neurons in the output layer. We experimented with many different sizes of the hidden layer(s), i.e., 10 to 35 neurons. In single hidden-layer NNs, the best learning accuracy was observed with 10 neurons; additional neurons resulted in accuracy deterioration (Table 1). However, among the 2-hidden layer NNs, the largest network (64-35-35-10) had the best learning accuracy. Table 2 shows the testing/validation accuracies for the trained NNs; the accuracies are reasonably high, i.e., between 94-100%. Unexpectedly, the training iterations (epochs), however, reduced with higher number of hidden layer neurons.

Table 1. Training/learning of neural networks with different configurations.

S. No.	Layer configuration	Learning accuracy %
1	64-10-10	81.5%
2	64-20-10	81.5%
3	64-30-10	72.6%
4	64-40-10	83.7%
5	64-60-10	62.9%
6	64-80-10	45.9%
7	64-10-10-10	64.4%
8	64-15-15-10	69.6%
9	64-20-20-10	58.5%
10	64-25-20-10	71.9%
11	64-25-25-10	71.9%
12	64-30-30-10	72.6%
13	64-35-35-10	94.0%

Table 2. Testing of neural network with different speakers.

S. No.	Training speaker	Testing speaker	Prediction accuracy %
1	15	5	94.3
2	12	4	95.0
3	10	3	96.7
4	7	2	95.1
5	5	2	100.2

The effect of LR on training accuracy is shown in Figs. 4 and 5. In our experiments, a larger value of LR (0.05) took longer to train as compared to the LR of 0.01, although in both cases, the NNs were able to meet the maximum prediction error criterion of 0.01.

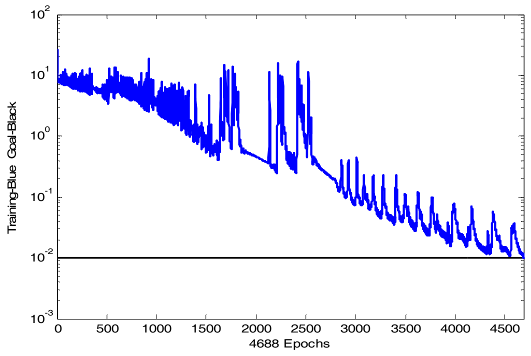

Fig. 4. Training epochs and error with LR=0.05.

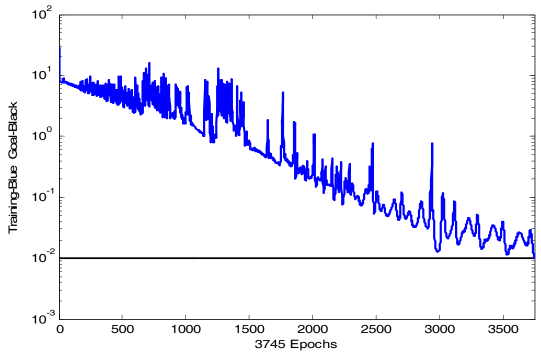

Fig. 5. Training epochs and error with LR=0.01.

5 Conclusions and Future Work

Ours is the first known attempt at using an NN to recognize spoken Urdu language words (numbers, in this case). The DFT of the acquired data was used for training and testing the speech recognition NN. The network made predictions with high accuracy. Currently, we are investigating use of LPC and cepstrum analysis for the same purpose. Our ultimate goal is to implement the Urdu speech recognition in hardware, for example, a DSP chip. The proposed system can be used in applications such as multi-media, telecommunications, and voice-driven customer service, in countries such as Pakistan and India.

References

1. H. Dudley and T. H. Tarnoczy, “The speaking machine of Wolfgang von Kempelen,” J. Acoust. Soc Am. 22: 1950, pp. 151 – 166.
2. T. Parsons. Voice and Speech Processing. McGraw-Hill College Div., Inc, 1986.
3. G. C. M. Fant. Acoustic Theory of Speech Production. Mouton, Gravenhage, 1960.
4. J. R. Deller and J. G. Proakis, J. H. L. Hansen. Discrete-Time Processing of Speech Signals. Prentice-Hall, Inc., Englewood Cliffs, NJ, 1993.
5. D. O’Shaughnessy. Speech Communication: Human and Machine. Addison Wesley Publishing Co., 1987.
6. L. R. Rabiner and R. W. Schafer. Digital Processing of Speech Signals. Prentice-Hall, Inc., Englewood Cliffs, 1978.
7. S. Varho. New Linear Predictive Systems for Digital Speech Processing. PhD dissertation, Helsinki University of Technology, Finland, 2001.
8. M. Caudill. AI Expert: Neural Network Primer. Miller Freeman Publications, 1990.
9. A. Beg, P. W. C. Prasad, and A. Beg. Applicability of Feed-Forward and Recurrent Neural Networks to Boolean Function Complexity Modeling. Expert Systems With Applications (Elsevier), (in press) November 2008, Vol. 36, No. 1.
10. A. Beg and Y. Chu. Modeling of Trace- and Block-Based Caches. Journal of Circuits, Systems and Computers. August 2007, Vol. 16, No. X.
11. A. Beg and Y. Chu. Utilizing Block Size Variability to Enhance Instruction Fetch Rate. Journal of Computer Science and Technology. April 2007, Vol. 7, No. 2, pp. 155-161, 2006.
12. A. K. Singh, A. Beg and P. W. C. Prasad. Modeling the Path Length Delay (LPL) Projection. In Proc. International Conference for Engineering and ICT, ICEI 2007, Melaka, Malaysia, November 27-28, 2007, pp. X.
13. A. Beg and W. Ibrahim. An Online Tool for Teaching Design Trade-offs in Computer Architecture. In Proc. International Conference on Engineering Education, Coimbra, Portugal, September 3-7, 2007.
14. A. Beg. Predicting Processor Performance with a Machine Learnt Model. IEEE International Midwest Symposium on Circuits and Systems, MWSCAS/NEWCAS 2007, Montreal, Canada, August 5-8, 2007, pp. 1098-1101.
15. P. W. C. Prasad and A. Beg. A Methodology for Evaluation (APL) Time Approximation. In Proc. IEEE International Midwest Symposium on Circuits and Systems, MWSCAS/NEWCAS 2007, Montreal, Canada, August 5-8, 2007, pp. 776-778.
16. A. Beg, P. W. C. Prasad, and S.M.N.A. Senanayake. Learning Monte Carlo Data for Circuit Path Length. In Proc. International Conference on Computers, Communications & Control Technologies, CCCT 2007, Orlando, Florida, July 12-14, 2007.
17. A. Beg, P. W. C. Prasad, M. Arshad, and K. Hasnain. Using Recurrent Neural Networks for Circuit Complexity Modeling", In Proc. IEEE INMIC Conference, Islamabad, Pakistan, December 23-24, 2006, pp. 194-197.
18. P. W. C. Prasad, A. K. Singh, A. Beg, and A. Assi. Modeling the XOR/XNOR Boolean Functions' Complexity using Neural Networks. In Proc. IEEE International Conference on Electronics, Circuits and Systems, ICECS 2006, Nice, France, December 10-13, 2006, pp. 1348-1351.

19. A. Beg and P. W. C. Prasad. Data Processing for Effective Modeling of Circuit Behavior. In Proc. WSEAS International Conference on Evolutionary Computing EC'07, Vancouver, Canada, June 18-20, 2007, pp. 312-318.
20. P. W. C. Prasad and A. Beg. Data Processing for Effective Modeling of Circuit Behavior. Expert Systems with Applications, (in press) Vol. 38, No. 4.
21. J. Koolwaaij. Speech Processing. www.google.com/ (current).
22. M. A. Al-Alaoui, R. Mouci, and M. M. Mansour, R. Ferzli. A Cloning Approach to Classifier Training. IEEE Trans. Systems, Man and Cybernetics – Part A: Systems and Humans, Vol. 32, No. 6, pp. 746-752, 2002.
23. MATLAB User's Guide. Mathworks Inc., 2006.