Spoken Digits Endpointing and Recognition

Final Project Report

ELEN E4810
Digital Signal Processing
Professor Dan Ellis

Weiguang Shi
ws2330
December 16th, 2010

1. Introduction

The objective of my final project is to find the beginning and end point of each spoken digit in a recording of digits strings and to build a system to recognize the these spoken digits. Based on the recognition of these speech utterances, we can implement the application of voice calcular or voice phone. In my project, I use the methods of digital signal processing to extract signal features and build a speaker-dependent recognition system. Figure 1 shows the overview of my system.

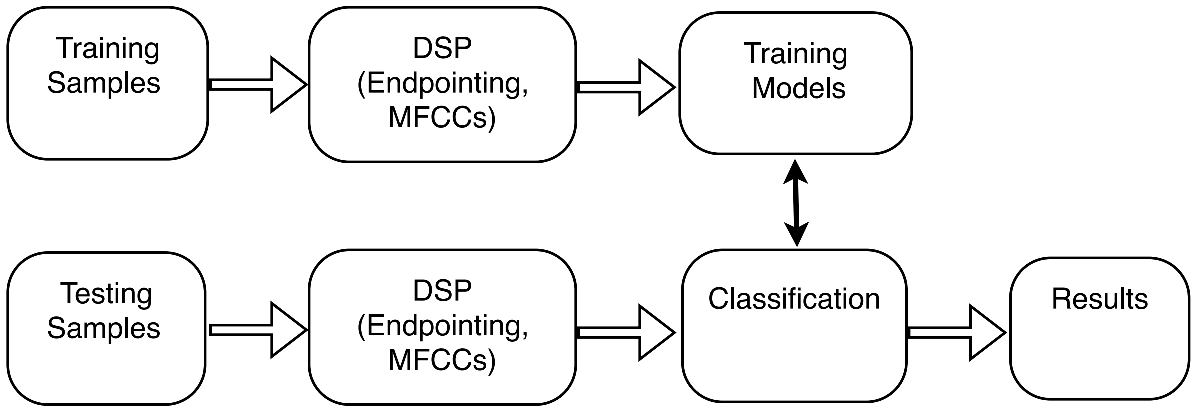

Figure 1 : System overview

From above figure, in this system, I find the beginning and end of recording utterances using the methods of digital signal processing, extract the discriminate signal features by computing the mel-frequency cepstral coefficients (MFCCs), and get the final classification results by nearest neighbor classifier.

2. Data Acquisition

I downloaded the sound samples of digit strings and cleans digits(TIDIGITS)[1] from the website of Professor D. Ellis.

Also, I used microphone and windows recorder to record digits. All the speech samples were sampled at 8kHz and saved as '.wav' file. I repeated each digit (from 0 to 9) or digit strings several times with different volume, pace and tones to test the accuracy of my system.

3.Speech Signal Processing

Each spoken digits has its special features in amplitude, energy, and spectrum. Through speech signal processing, we can extract these features to discriminate the different spoken digits.

3.1 Signal Preprocessing

In the speech signal, high frequency formants have smaller amplitude with respect to low frequency formants. A preemphasis of high frequency is therefore required to obtain similar amplitude for all formants. Such processing is usually obtained by filtering the speech signal with a first order FIR filter, whose transfer function in the z-domain is [5] :

$$H(z)=1-0.95*Z^{-1}$$

3.2 Windowing

In the speech signal, the signal keeps stationary within short time intervals. To reduce the complexity of processing, in the time domain, usually using the rectangular window to get the frames of signal. In the frequency domain, Hamming Window is a good choice [5].

3.3 Endpoint Detection

The endpoint detection is applied to find the beginning and end point of each digit in the recorded digit strings, also is used to extract the region of conversational speech signal, remove the silent and noise region. There are two famous endpoint detection algorithms. The first the algorithm uses signal features based on energy level and the second algorithm uses signal features based on the rate of zero crossings[2].

3.3.1 Zero-crossing Rate

Zero-crossing rate is the rate of zero-crossings in each frame of the signal. In the silent frames of signal, the zero-crossing rate is very low.

3.3.2 Short-time Energy

Short-time energy indicates the amplitude of signal in each frame. The energy of voiced speech is much greater than the energy of unvoiced speech. Also, the energy of the conversational speech is higher than the energy of noise.

In my work, I want to make the detection more accuracy and do something new, so I decide to combine above two algorithms. I set the two constant thresholds of energy and zero crossing rate. If the testing signals achieve one of these two thresholds, and continue for a long time (not a short-time noise), it will be decided as the beginning of the voiced speech. The same method for deciding the endpoint. (See Matlab Program *PointsDetect.m*)

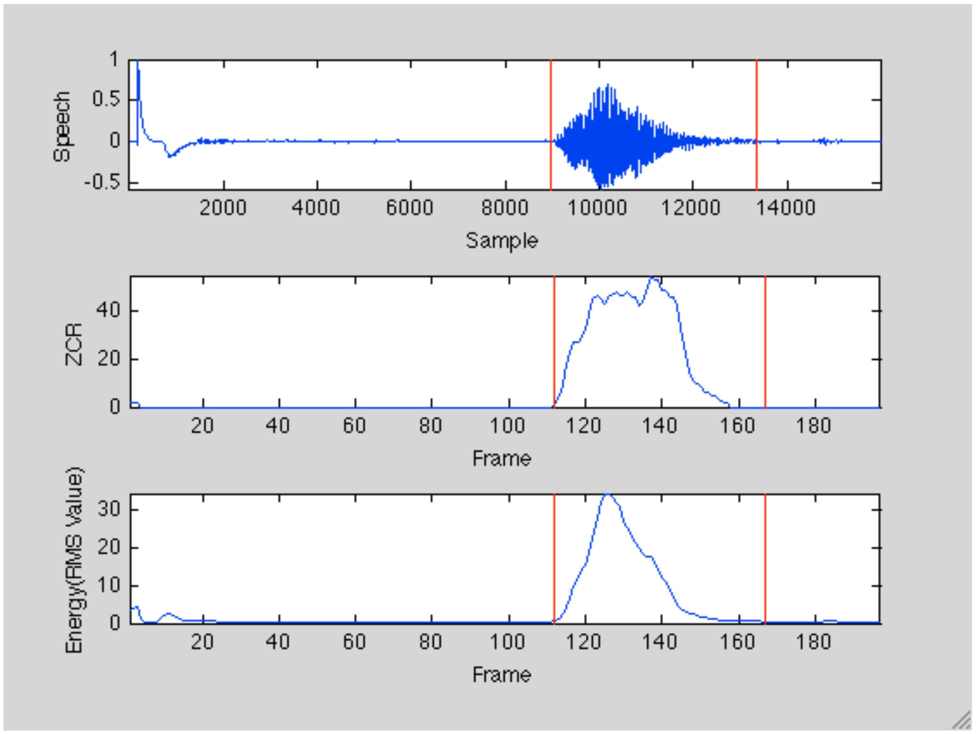

Figure 2 : Endpoint detection for single digit utterance. The testing digit is ‘One’. The segment between the two red color lines is the speech voice part.

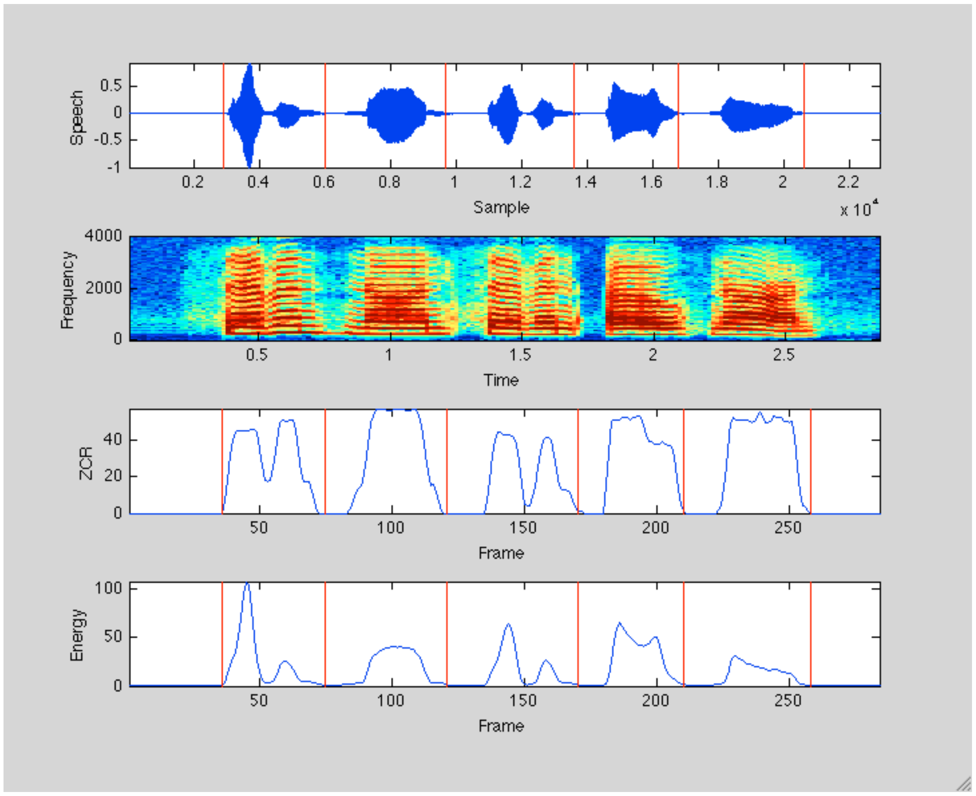

Figure 3 : Endpoint detection for digit strings utterance. The testing sample is ‘seven nine seven oh nine’ (TIDIGITS). The segment between the two red color lines is the part for each digit. From above two figures, we can see that my program works well when the threshold set properly.

3.4 Features Extraction

I mainly used the mel-frequency cepstral coefficients(MFCCs) to discriminate different spoken digits. The mel-frequency cepstrum (MFC) is the short-term power spectrum of a speech, based on a linear cosine transform of a log power spectrum on a nonlinear mel scale of frequency. Mel-frequency cepstral coefficients (MFCCs) are coefficients that make up an MFC.

MFCCs are commonly derived as follows:[3]

1. Take the Fourier transform of (a windowed excerpt of) a signal.
2. Map the powers of the spectrum obtained above onto the mel scale, using triangular overlapping windows.(See figure 4)
3. Take the logs of the powers at each of the mel frequencies.
4. Take the discrete cosine transform of the list of mel log powers, as if it were a signal.
5. The MFCCs are the amplitudes of the resulting spectrum.

In the computation of MFCCs, I used the *melbankm.m* function in the Voicebox tools[4] to determine matrix for a mel-spaced filterbank. The computed MFCCs are the matrix of Mx24 values. I reduced the dimension to 24 values by getting maximum values from each column.

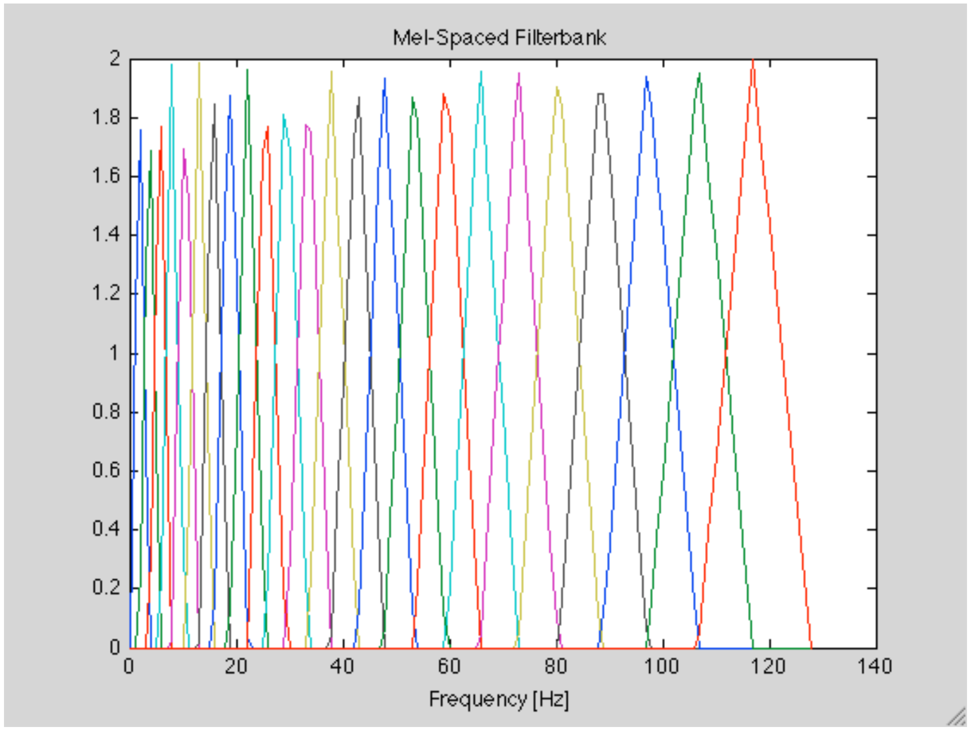

Figure 4: Mel-Spaced Filterbank (24 filters in the filterbank)

In my program, I processed the speech signals, extracted features and saved the necessary information of all the spoken digits into dataset (.mat file). For specific person recognition, the dataset file was saved as ‘soundinfo1.mat’. For non-specific person recognition, the dataset files were saved as ‘TItrain.mat’ and ‘TItest.mat’. (See Matlab Program *SigProcess.m*)

4.Classification

After speech endpointing, we are able to write the new sound file for each digit from digit strings. So in the classification, I only make recognition of the single spoken digit. There are two parts of classification.

In the first part, I used the 100 speech samples (each digit is repeated tenth) of myself to do the speech recognition for specific person. Two speech samples with rising and falling tones of each digit for training, and other 80 samples for testing. (See *classifier1.m*)

In the second part, I used the 120 speech samples from clean digits(TIDIGITS)[1] by 3 female and 3 male speakers to do the speech recognition for non-specific person. 60 samples for training, and 60 samples for testing. (See *classifier2.m*)

I applied the K-Nearest Neighbor(KNN) classifier for classification. The training sample is one or two of each digit in my system, so I chose 1 as the value of K. I calculated the euclidean distance between each testing sample and every training samples, then made classification by selecting the class type of training sample which has the shortest distance with testing sample.

To run these two classifiers, input:
classifier1('soundinfo1.mat','classifier1_output.txt')
classifier2('TItrain.mat','TItest.mat','classifier2_output.txt')
in the matlab command window.

The format of the output file is: “Filename (space) Classification Result”
For example, '0_01.wav 0' means number zero is classified as zero (Correct Result)
'9_04.wav 8' means number eight is classified as eight (Incorrect Result)

5.Results

(1)Endpoint Detection of Digit Strings

The 8 digit strings(Filtered) from TIDIGITS are used to do the endpoint detection. My system works well when the threshold set properly, and successfully finds the beginning and end point of each digit.

(2)Spoken Digits Recognition for Specific Person

Using the dataset of MyDigits, loading the file of “ soundinfo1.mat ”. 20 training samples(two training samples with rising and falling tones for each digit), 80 testing samples.

Digits	0	1	2	3	4	5	6	7	8	9
Correct No.	8	8	8	6	7	8	8	6	7	8
Incorrect No.	0	0	0	2	1	0	0	2	1	0
Accuracy	100%	100%	100%	75%	87.5%	100%	100%	75%	87.5%	100%

Overall Accuracy : 92.5%

(2)Spoken Digits Recognition for Non-specific Person

Using the dataset of TIDIGITS, loading the file " TItrain.mat " and " TItest.mat ".

Digits	0	1	2	3	4	5	6	7	8	9
Correct No.	4	6	6	6	5	5	6	3	6	5
Incorrect No.	2	0	0	0	1	1	0	3	0	1
Accuracy	67.7%	100%	100%	100%	83.3%	100%	100%	50%	100%	83.3%

Overall Accuracy : 86.7%

The system for non-specific person recognition is the speaker-dependent system. Each user has to do the training before use. It increases the complex of the system, but ensures the accuracy. There are some methods to build a speaker-independent recognition system in several papers, however the time is constraint. I hope I can do it in the future work.

6.Conclusion

In the final project of digital signal processing course, I have implement this spoken digits endpointing and recognition system. This system proved to be fairly reliable for both endpoint detection and digits recognition. In the noisy environment, my system did not perform very well. I hope I can implement some new methods to improve the robustness of system in the future work. I learned a lot from the DSP course, and I applied the knowledge to complete my course project. It is really a precious experience for me. Thanks for Professor, TA, classmates and all who helped me.

7.Reference

[1] D. Ellis. Clean Digits and Digit Strings (Sound Examples), http://www.ee.columbia.edu/~dpwe/sounds/tidigits/

[2] Hanchate, D.B.; Nalawade, M.; Pawar, M.; Pophale, V.; Kumar Maurya, P., "Vocal Digit Recognition using Artificial Neural Network," *ICCET 2010*, Vol.6, pp. V6-88 - V6-91, 2010.

[3] Min Xu et al. (2004). "HMM-based audio keyword generation" In Kiyoharu Aizawa, Yuichi Nakamura, Shin'ichi Satoh. *Advances in Multimedia Information Processing - PCM 2004*: *5th Pacific Rim Conference on Multimedia*. Springer. ISBN 3540239855.

[4] M. Brookes. VOICEBOX: Speech Processing Toolbox for MATLAB http://www.ee.ic.ac.uk/hp/staff/dmb/voicebox/voicebox.html

[5] Claudio Becchetti; Lucio Prina Ricotti, "Speech Recognition Theory and C++ Implementation" John Wiley and Sons, LTD. ISBN 0-471-97730-6

8.Matlab Code List

Reference:

Voicebox[4]:
melbankm.m, enframe.m, PointDetect.m (for single digit)

Self Code:

(1) Endpoint Detection : PointsDetect.m(for digits strings)
(2) Signal Processing and Build the dataset : SigProcess.m
(3) Computation of MFCCs : Mfcc.m
(3) Specific person recognition : classifier1.m
(4) Non-specific person recognition : classifier2.m