Voice Command Recognition Design and Implementation

WANG Jia-ping (200228013202862)

Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100871, China

Abstract: This paper presents a simple method of recognize a small voice command set of specific person. I use DTW algorithm to measure the distance two FFT spectrum vector sequences and FFT algorithm to generate FFT spectrum (0-8K) vector sequences.

1. Overview

The process of voice command recognition on still image can be divided into two stages: Training and Recognizing. Training stage produce 200 stock FFT spectrum vector sequences (4 sequences per command as to the supplied training speech database). When Recognizing I compare the incoming FFT spectrum vector sequences (FSVS) with stock FSVSes one by one and regard the closest one as the result.

2. Detailed Process

2.1 Training stage

For every wave file in training set do:

1. Volume Equalization: Adjust the volume of the wave file to a fixed level.
2. Silence Segmentation: Kick out the silence period at the head and tail of the wave file.
3. Build FSVS: Do FFT step by step on the wave to get a sequence of FFT Spectrum.
4. Store: Save the FDFS with the corresponded command name.

I use the 1-4th files in the speech database as the training set and the 5th files as the test set.

2.2 Recognizing stage

For any wave file in test set, I first do same steps 1-3 in Training stage. Then compare the FSVS generated from incoming wave file with every stock FSVSes and find the closest one. The command name associated with the closest one is the result of recognition.

3. Detailed algorithm

3.1 Volume Equalization

Adjust the volume of the wave file into a fixed level.

Definition. Let

1) W_i be the input waveform, $i \in [0, L]$

2) H_i $i \in [0,128]$ be the histogram of the input W_i, $H_i = \sum_{i \cdot 256 < |W_j| < (i+1) \cdot 256 - 1} 1$

3) U be the minimum value that satisfies $\left(\sum_{i=U}^{255} H_i \right) \Big/ L < 0.006$

Then the Equalized waveform:

$$E_i = \begin{cases} -32767 & M_i < -256 \cdot U \\ 127 \cdot M_i / U & |M_i| < 256 \cdot U \\ 32767 & M_i > 256 \cdot U \end{cases} \qquad i \in [0, L]$$

3.2 Silence Segmentation

Kick out the silence period at the head and tail of the wave file.

Definition. Let

1) W_i be the input waveform, $i \in [0, L]$

2) Constant D be the width of observing window. (Currently 512)

3) Power sequence $P_i = \sum_{i}^{i+D} |W_i| \Big/ D \qquad i \in [0, L-D]$

4) V be the maximum value that satisfies $\sum_{i=0}^{V} P_i \Big/ V < 600 \qquad V \in [0, L-D]$

5) U be the minimum value that satisfies $\sum_{i=U}^{L-D} P_i \Big/ (L-D-U) < 600 \qquad U \in [0, L-D]$

Then the segmented waveform: $S_i = W_{i+V} \quad i \in [0, U-V]$

3.3 Build FSVS

Do FFT step by step on the wave to generate a sequence of FFTSpectrum vectors of M dimension. The detail algorithm is stated as the following:

0. (Input)

   A waveform $W_i \; i \in [0, L]$

1. (Initialization)
   1) Let Constant D be the width of observing window. (Currently 512).
   2) Let Constant M be the dimension of every FFT Spectrum vector.
   3) Let F_i be the result sequence of FFT Spectrum vectors $i \in \left[0, \frac{2 \cdot (L-D)}{D}\right]$

2. (Forward Inclusion)

   Slider observing window O_i from W_0 to W_{L-D}, move D/2 elements for every step:

   (When O_i at $W_x \; x \in \left[0, \frac{2 \cdot (L-D)}{D}\right]$)

   1) Do FFT $O_i \rightarrow O_i' \quad i \in [0, D]$.
   2) Map O_i' into M dimension FFT Spectrum vector V: $V_i = \frac{M}{D} \left(\sum_{j=\frac{i \cdot D}{M}}^{\frac{(i+1) \cdot D}{M}} O_j' \right)$.
   3) Let $V \rightarrow F_x$.

3. (Output)

   $F_i, \quad i \in \left[0, \frac{2 \cdot (L-D)}{D}\right]$

3.4 Comparing FSVSes

I use standard Dynamic Time Warping (DTW) algorithm to calculate the distance between two sequences of FFT Spectrum vectors (FSVS). The sample Euler distance is used when measure

distance of two vectors in each sequences.

4. Try different M[1]

When different M is used, different recognition error rate appears. In my experiment, recognition error rate is calculated as the follow:

0) (Input)
   250 FFT spectrum vector sequences generated from 250 wave files in speech database
1) (Initialization)
   0→ErrorCounter
2) (Forward inclusion)
   For every sequences S in the 250 FSVSes:
   1) Find the closest FSVS S' in the rest 249 ones
   2) If S and S' is not associated with the same command name then:
      ErrorCounter+1→ErrorCounter
3) (Output)
   ErrorCounter/250

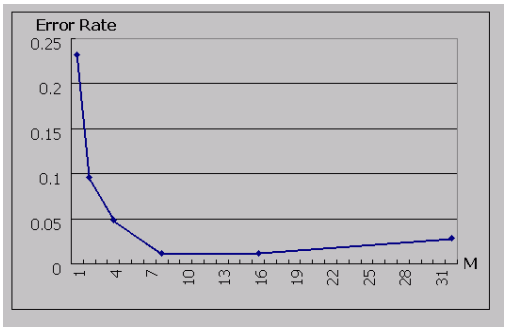

Figure 1. Error rate when different M

Figure 1 demonstrates variable error rates when different M is used. We can see from M =8 to 16 error rate reach minimum. So in my implementation I use 8 as the dimension M of FFT Spectrum vector.

Figure 2 and 3 show distribution of DTW distances of random different FSVS pairs. Dots above X-axis are DTW distances of the same command names and dots below X-axis are those of different command names.

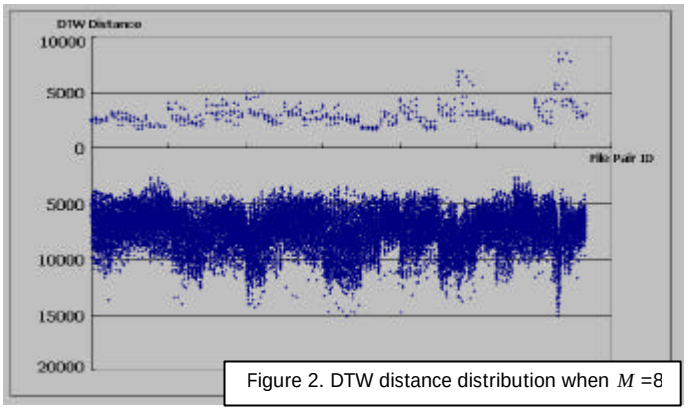

Figure 2. DTW distance distribution when M =8

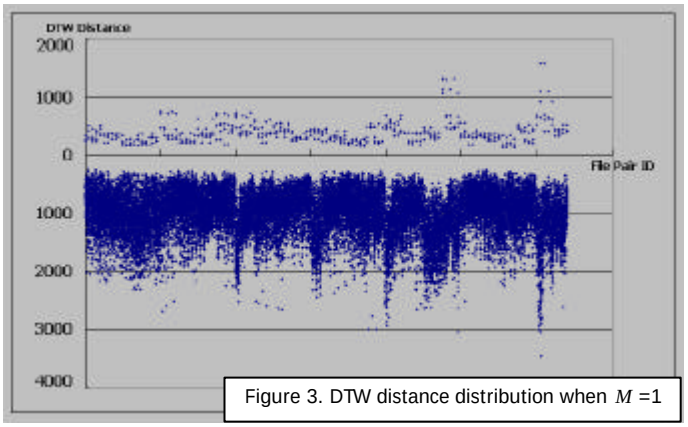

Figure 3. DTW distance distribution when M =1

When M=8 (in figure 2), the DTW distance of most FSVS pairs of same commands is less than 5000 and most different ones is great than 5000. So same/different command FSVS is classifiable.

When M=1 (in figure 3), the DTW distance of most FSVS pairs of same commands is less than 800 but about half different ones have a DTW distance less than 800. So when M=1 same/different command FSVS is not classifiable.

[1] The M is refer to the one mentioned in 3.3

5. Implementation

Based on previous project (Face Detection), I use Visual C++.Net to develop all components. All classes associated with waveform data process is under namespace **Audio**. In additional I add the following classes to the original project (Kernel) to maintain user interface for voice recognition.

CVoicePane:

It is derived from CWnd of MFC. It is used to display waveform and FFT Spectrums. It also performs all steps of recognizing stage.

In **Audio** namespace, there're classes for digital signal process. CVoicePane::StartNewSession() use them.

Class **CWave** encapsulates waveform data. It provides load and play a wave file.

Class **CWaveEx** is derived from **CWave**. It provides:

1) Generate FFT Spectrum.
2) Calculate average power (used in display waveform).
3) Generate sequence of M dimension FFT Spectrum vectors.
4) Volume equalization.
5) Silence segmentation.

Class **CFFTStream** encapsulates sequence of M dimension FFT Spectrum vectors (FSVS)

Class **CWaveClassifier** is a template class that performs DTW and Euler-distance calculating. I also build FDFS of specific M. Its template parameter **SepSeg** specified the dimension of FFT Spectrum vector M.

Class **CSample** encapsulates trained samples. It contains stock FFT spectrum vector sequence and a string of the corresponded command name.