Voice Recognition

By: Tim Lindquist & Alex Christenson
Overview

- Project Objective
- Background
- Feature Extraction Process
- Feature Matching Process
- Implementation
- Demonstration
- Python
Objective

Develop a real time speaker identification system using Python

Project Status:

MATLAB=working

Python=in progress
Background

Speaker Identification:
- understanding who is speaking

Speaker Verification:
- is the process of accepting or rejecting the identity claim of a speaker

Speech Recognition vs. Speaker Recognition:
- identifying what is said vs. who said it
Overall Process
Feature Extraction

Input audio signal sampled at $fs=10000Hz$

Human voice max frequency is 3000Hz ($fs$ satisfies Nyquist rate)
Frame Blocking

Blocking: Signal is blocked into frames of N samples. With overlap N-M

N=256  M=100

Figure 3. Block diagram of the MFCC processor
Windowing

each frame is windowed to minimize discontinuities at the end points of each frame

Size $0 < n < N - 1$ using Hamming window

Figure 3. Block diagram of the MFCC processor
FFT

DFT: using FFT function, converts each frame from time domain into the frequency domain

Figure 3. Block diagram of the MFCC processor
Mel-Frequency Wrapping

Filterbank with triangular bandpass frequency response

Linear frequency spacing <1000 Hz<Logarithmic frequency spacing

Human Speech € BL[300, 3000] Hz

k=number of mel spectrum coefficients=20

Figure 3. Block diagram of the MFB process
Cepstrum

DCT: converts the mel spectrum coefficients back to time domain

Provides a good representation of the local spectral properties for a given frame

Output is a set of coefficients called an acoustic vector
Feature Matching

Vector Quantization (VQ): Process of mapping vectors to a finite number of regions in space

Cluster: The region the VQ maps to

Codeword: Center of a cluster

Codebook: Collection of codewords
Feature Matching

Speaker 1- Acoustic vector (circles)

Speaker 2- Acoustic vector (triangles)

Acoustic vector = clusters of speaker samples

Codewords (black shapes) = center of clusters

Codebook (yellow box) = collection of codewords

*Figure 5. Conceptual diagram illustrating vector quantization codebook formation. One speaker can be discriminated from another based on the location of centroids. (Adapted from Song et al., 1987)*
Clustering the Training Vectors

1. Design a 1-vector codebook
2. Split codebook according to rule
3. Search for the Nearest neighbor
4. Update the centroid
5. Iterate 3, 4 until average distance< threshold ($\varepsilon$)
6. Iterate 2,3 and 4 until a codebook size (M) is designed

Figure 6. Flow diagram of the LBG algorithm (Adapted from Rabiner and Juang, 1993)
Implementation

Training Phase

- Input: signal used as reference for verification
- Output: vector quantized codebook

Process

1. Read audio signal
2. Block into frames of 256 samples
3. Hamming filter blocks
4. Compute DFT of blocks
5. Compute power spectrum & Mel filter
6. Take DCT to produce Mel frequency cepstral coefficients
7. Assemble code book through VQLBG algorithm

Testing Phase

Input: new signal & reference codebook
Output: The reference signal that matches

Process

1. Steps 1-6 again
2. Find minimum distance to codeword
3. Identify speaker from cluster
Demonstration

code=train('traindir2', 2);

test('testdir2', 2, code);

test('testdir1', 4, code);

Trained with 44 english sounds

- short -a- in and, as, after
- short -e- in pen, hen, lend
- short -i- in it, in
- short -o- in top, hop
- short -u- in under, cup
Python Code

Found libraries that use MATLAB commands

Manually rewriting scripts

So far

- Record audio from mic, automatically split when silence occurs
- Progress making melfb and mfcc functions
Sources

http://www.ifp.illinois.edu/~minhdo/teaching/speaker_recognition/


https://en.wikipedia.org/wiki/Vector_quantization