UNIVERSITY OF SCIENCE AND TECHNOLOGY OF HANOI

SPEAKER RECOGNITION

Digital Signal Processing Final Report

Group 8

June, 2023

Table of Contents

1. Introduction

1.1. Overview

The challenge of recognizing a speaker from a speech recording is known as speaker recognition. It is a crucial subject in speech signal processing and has many uses, particularly in security systems. Devices that use voice control mainly rely on speaker recognition.

This issue has previously been extensively studied; thus, my goal was to use Python to implement some well-known current methods for speaker detection rather than to develop a new algorithm. We used Python for scientific computing. The NumPy, SciPy, and Matplotlib packages, which provide a vast library for matrix manipulation, signal processing, and charting, were my main tools for this.

The extraction of speaker-specific characteristics from speech, followed by training on a data set and testing, is the core idea of speaker identification. We have relied extensively on the approach given in our code, where they train each speaker's Mel-Frequency Cepstral Coefficients with Vector Quantization (using the LBG algorithm). By extracting the Linear Prediction Coefficients (LPCs) for training, I have also experimented with Vector Quantization. The data set we used for training and testing was taken from our code and comprises of the words "zero" spoken by 8 distinct female speakers. With just 8 training and test sets, this data set is not large enough to provide definitive results, and the outcomes are not good enough. However, our intention was to learn about and implement algorithms in Python, not carry out accurate tests. we wish to gather more data to extend this work.

1.2. Authors

BA11-054 Nguyễn Trường Khải

BA11-041 Đinh Văn Hiệp

BA11-025 Nguyễn Hữu Đức

BA11-046 Lê Vũ Hoàng

BA11- 005 Lê Tuấn Anh

BI12-022 Nguyễn Nhật Anh

BA11-059 Hoàng Trung Kiên

2. Feature Extraction Techniques for Speaker Recognition

2.1. Speech Signal

As seen in **Figure 1a**, digital speech is a one-dimensional, time-varying discrete signal. The Autoregressive Model and the Sinusoidal + Residual Model are two examples of the many mathematical models of speech that are available. According to a common theory of speech creation, speech is made up of a train of impulses with a period equal to its pitch, random noise, a voiced/unvoiced switch, and the vocal tract, which acts as a time-varying filter.

Speech has a mostly fixed character. The signal is steady for brief durations, although it changes in frequency over longer time frames. In order to display its frequency content, a Short-Time Fourier Transform is required, as shown in **Figure 1b** (also performed in Python with FFT size = Hanning window size = 256 samples and overlap of 50%). A spectrogram is a 3D STFT graphic with time and frequency along the x and y axes with log amplitude represented by the intensity of the color.

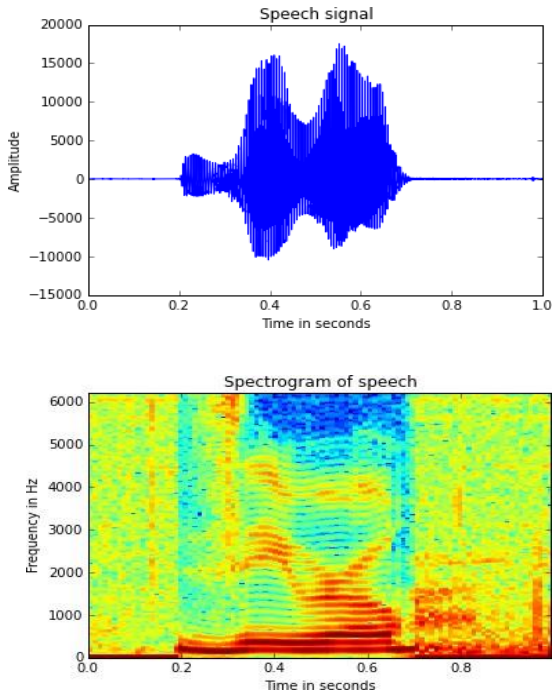

Figure 1

2.2. Feature extraction

Choosing which highlights to extricate from discourse is the foremost critical portion of speaker acknowledgment. A few well-known highlights are: MFCCs, LPCs, Zero-Crossing Rates etc. In this work, we have concentrated on MFCCs and LPCs. Here could be a brief diagram of these highlights.

2.2.1. Mel Frequency cepstral coefficient (MFCC)

Human hearing isn't straight but logarithmic in nature. This implies that our ear acts as a filter. MFCC's are based on the known variation of the human ear's critical bandwidths with frequency. Filters spaced linearly at low frequencies and logarithmically at high frequencies have been used to capture the phonetically important characteristics of speech. This is expressed in the mel-frequency scale. The relationship between frequency in Hz and frequency in Mel scale is given by:

$$m = 1125 \times \ln\left(1 + \frac{f}{700}\right)$$

$$f = 700 \times (e^{\frac{m}{1125}} - 1)$$

To calculate MFCCs, the steps are as follows. A schematic of this process is given in this figure:

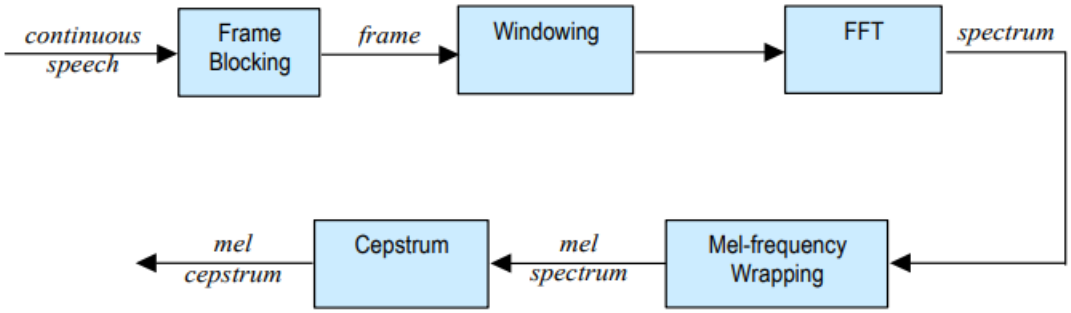

Figure 2. MFCC Calculation Schematic

1. The speech signal is divided into frames of 25ms with an overlap of 10ms. Each frame is multiplied with a Hamming window.
2. The periodogram of each frame of speech is calculated by first doing an FFT of 512 samples on individual frames, then taking the power spectrum as:

$$P(k) = \frac{1}{N}|S(k)|^2$$

Where P(k) refers to power spectral estimate and S(k) refers to Fourier coefficients for the kth frame of speech and N is the length of the analysis window. The last 257 samples of the periodogram are preserved since it is an even function.

3. The entire frequency range is divided into ‘n’ Mel filter banks, which is also the number of coefficients we want. ‘For ‘n’ = 12, the filter bank is shown in this Figure - a number of overlapping triangular filters with increasing bandwidth as the frequency increases.

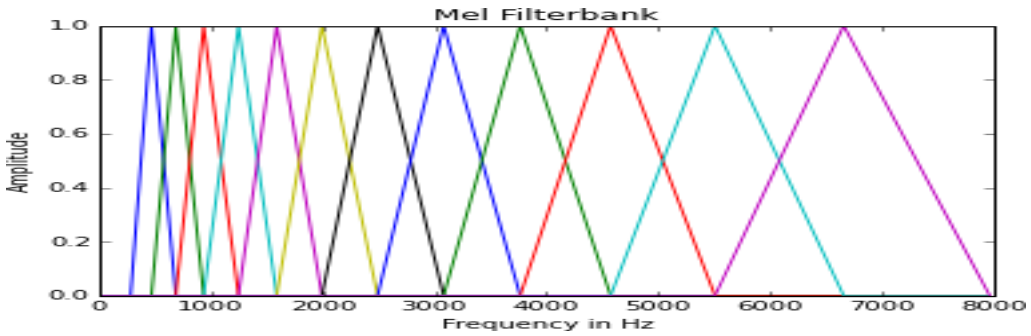

Figure 3. Mel Filter banks

4. To calculate filter bank energies, we multiply each filter bank with the power spectrum, and add up the coefficients. Once this is performed, we are left with ‘n’ numbers that give us an indication of how much energy was in each filter bank.
5. To obtain the final MFCCs, we compute the discrete cosine transform of the logarithm of these 'n' energies.

2.2.2. Linear prediction coeficients (LPC)

Another well-liked feature for speaker identification is LPC. Using data from a linear predictive model, LPC is a method primarily used in audio signal processing and speech processing to capture the spectral envelope of a digital signal of speech in compressed form. It is one of the most effective methods for encoding high-quality speech at a low bit rate and one of the most effective strategies for speech analysis. It also offers incredibly precise estimations of speech characteristics.

The simplicity of construction and the low level of code complexity are advantages of this method. In reality, it could just take a very short amount of time to train and test the samples and get a reliable result. On the other hand, the performance may not be as good as other methods.

This block diagram shows the overall flow of the LPC technique:

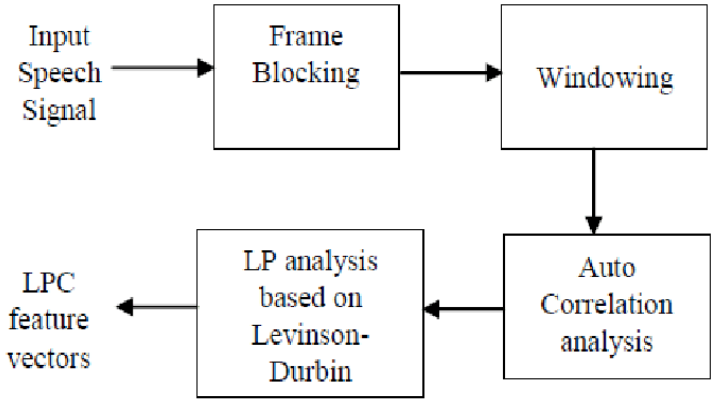

Figure 4. LPC technique

We first need to comprehend the Autoregressive model of speech in order to grasp LPCs. A p^{th} order AR process may be used to represent speech, with each sample provided by:

$$x(n) = -\sum_{k=1}^{p} a_k x(n-k) + u(n)$$

Following the addition of a Gaussian noise u(n), each sample at the n^{th} instant is dependent on 'p' prior samples.

This concept is based on the idea that a buzzer at the end of a tube (spoken sounds) generates a speech signal with the addition of hissing and popping noises on occasion.

The LPC coefficients are supplied by. We utilize the Yule-Walker equations to estimate the coefficients. The autocorrelation function R_x is employed. The formula for autocorrelation at lag l is:

$$R(l) = \sum_{n=1}^{N} x(n)x(n-l)$$

The Box-Jenkins technique, which adjusts the correlation at each lag by the sample variance so that the autocorrelation at lag 0 equals unity, is used to calculate ACF in Python.

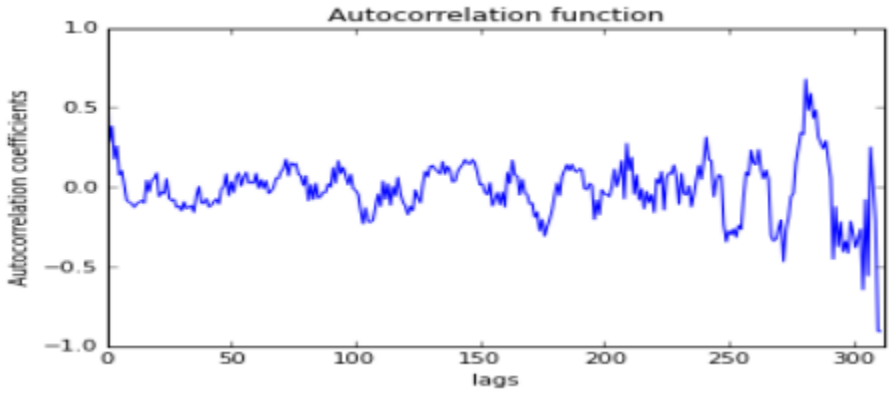

Figure 5. ACF for one speech frame

The final form of the Yule-Walker equations given by:

$$\sum_{k=1}^{p} \alpha_k R(l-k) = -R(l)$$

$$\begin{bmatrix} R(0) & R(1) & \dots & R(p-1) \\ R(1) & R(0) & \vdots & R(p-2) \\ \vdots & \vdots & \ddots & \vdots \\ R(p-1) & R(p-2) & \dots & R(0) \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_p \end{bmatrix} = - \begin{bmatrix} R(1) \\ R(2) \\ \vdots \\ R(p) \end{bmatrix}$$

The solution for α:

$$\boldsymbol{\alpha = -R^{-1}r}$$

Auto-correlations are what are used in the Yule-Walker equation as R. They are only the errors, or the separation between each "point" (sample) and the model that is assumed. We import the built-in auto-correlation function from package numpy for this project.

Once the LPC has been generated, it's crucial to store them compactly. This is made possible by vector quantization (VQ), which can map vectors from a huge vector space into a limited number of regions. Each region is referred to as a cluster and has a codeword in its core. A codebook is a compilation of all codewords.

2.3. Feature Matching

For speaker recognition, Dynamic Time Warping (DTW), Hidden Markov Model (HMM), and Vector Quantization (VQ) are the most often used feature matching techniques. I've used Vector here.

2.3.1. Vector Quantization (VQ)

Using VQ, vectors from a large vector space are mapped to a limited number of places in that space. A codeword serves as the hub of each region, which is referred to as a cluster. A codebook is a compendium of every codeword.

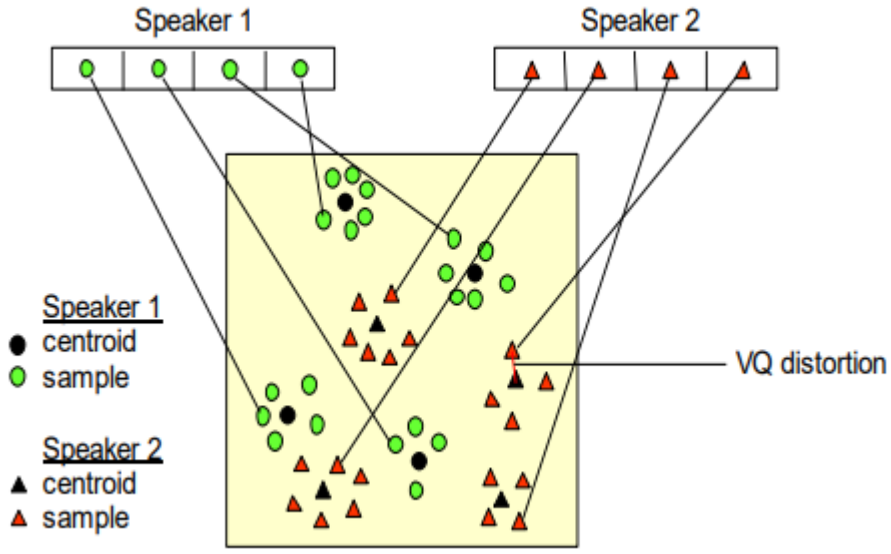

Figure 6a. Conceptual codebooks for 2 speakers

Figure 6a shows a conceptual diagram to illustrate this recognition process. On a 2D plane, just two feature dimensions are displayed for two separate speakers. In the plane, there are vector groupings that may be seen.

A codeword is selected for each cluster during training by reducing the distortion between each cluster's vector and the codeword. A codebook personal to each speaker is created from the group of codewords. The LBG algorithm, which is discussed later, chooses the codebook for each speaker.

The distortion of a speaker's characteristics relative to all taught codebooks is calculated in order to identify the speaker. It is determined which codebook has the least amount of distortion with the speaker's attributes.

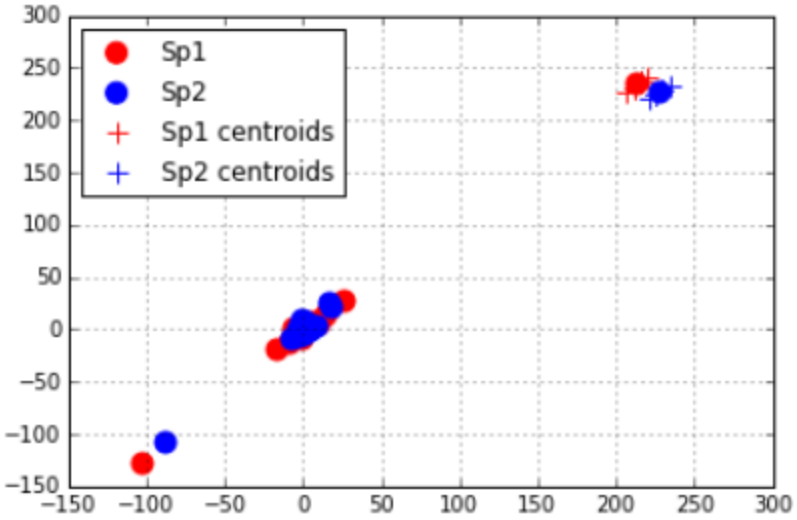

Figure 6b. Actual codebooks for 2 speakers

Figure 6b depicts a real-world 2D depiction representing the fifth and sixth characteristics of two speakers, together with their respective codebooks.

2.3.2. LBG Algorithm

The LBG algorithm [Linde, Buzo and Gray], is used for clustering a set of L training vectors into a set of M codebook vectors. The formal implementation of the algorithm is the following recursive process:

1. Create a 1-vector codebook, this vector represents the centroid of the complete collection of training vectors, therefore this step doesn't need iteration.

2. Double the codebook's size by splitting each existing codebook (y_n) in half in accordance with the rule, where n ranges from 1 to the codebook's present size and ε is a splitting parameter(we choose ε =0.01).

$$\mathbf{y}_n^+ = \mathbf{y}_n(1+\varepsilon)$$
$$\mathbf{y}_n^- = \mathbf{y}_n(1-\varepsilon)$$

3. Nearest-Neighbor Search: For each training vector, choose the closest (based on similarity measurement) codeword in the current codebook, and then assign that vector to the cell (associated with that codeword).

4. Centroid Update: Use the centroid of the training vectors provided to each cell to update the codeword in each cell.

5. Iteration 1: repeat steps 3 and 4 until the vector distortion for the current iteration is less than a percentage of the distortion from the previous iteration. To make sure the process has converged, do this.

6. Iteration 2: repeat steps 2, 3 and 4 until a codebook size of M is designed.

The LBG method logically creates an M-vector codebook in phases. It begins by creating a 1-vector codebook, splits the codewords to begin the search for a 2-vector codebook, and keeps separating the codewords until it finds the M-vector codebook that is wanted.

3. Result

It's time to put our speaker recognition system to the test. We can identify the speaker by comparing the speaker's feature vector to the codebooks of all trained speakers and calculating the shortest distance between them.

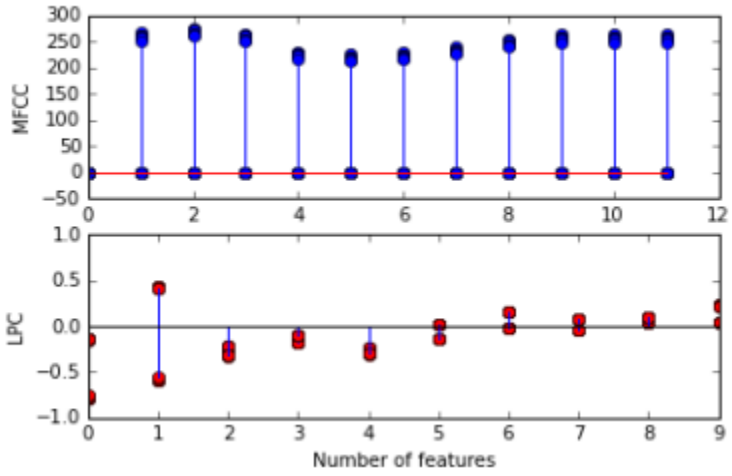

Figure 7. Codebooks for MFCC and LPC features

Be aware that the findings, which produced an accuracy of 37.5% with MFCC and 50% with LPC, are not as precise as we had anticipated. Lack of sufficient training data may be the cause of this low accuracy. Other sophisticated classification algorithms like ANNs and SVMs ought to produce better outcomes. We found that the greatest results come from training with 12 MFCC features and LPC coefficients of order 15. While computing MFCCs, various factors can be changed, including the amount of codewords in a codebook and the FFT size. It's feasible that another set of them will provide more precision.

The results of each speaker's identification using the MFCC and LPC coefficients and the Vector Quantization with LBG technique are shown in the following **Figure 8**:

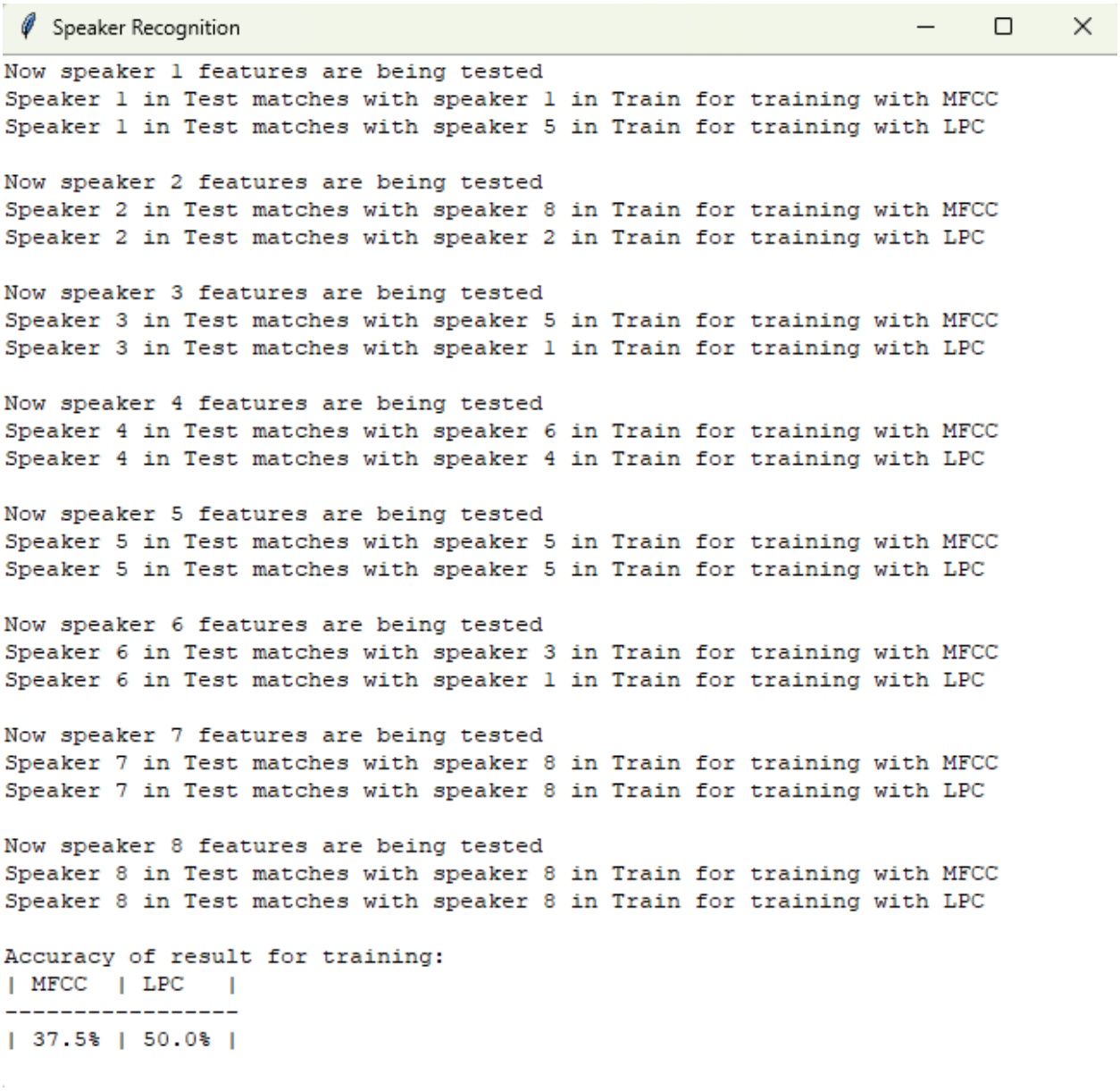

Figure 8

4. Comparison and analysis of results

LPC (Linear Predictive Coding) and MFCC (Mel Frequency Cepstral Coefficients) are two popular methods in speech processing and speech recognition. Here is a comparison of their similarities, differences, and an analysis of the results in speech recognition.

4.1. Similarities

- Both methods are used to represent speech as digital signals and extract features from speech signals.

- Both methods are based on the modeling of human speech and utilize fundamental concepts of audio signals and spectra.

4.2. Differences

- LPC focuses on modeling and predicting the acoustic parameters of speech, such as linear predictive coefficients and gain.
- LPC often allows for detailed representation of the acoustic information and spectral features of speech.
- MFCC generates a set of cepstral coefficients based on the Mel frequency scale, where information about the acoustic and spectral features of speech is relatively simply represented.
- MFCC minimizes unnecessary information and enhances important speech features through the use of Mel frequency filters and cepstral transformation.

4.3. Analysis of the speech recognition results.

The obtained accuracy of 37.5% with MFCC and 50% with LPC for the word "zero" spoken by 8 different individuals indicates that LPC performed better than MFCC in this specific case.

The higher accuracy of LPC could be attributed to its ability to capture more detailed acoustic and spectral information. LPC focuses on modeling and predicting the acoustic parameters of speech, such as linear predictive coefficients and gain. By capturing finer details of the speech signal, LPC may have generated better prediction models for each specific speaker, resulting in higher accuracy.

On the other hand, MFCC utilizes a different approach by mapping the speech signal to the Mel frequency scale and extracting cepstral coefficients. This representation may have resulted in slightly lower accuracy compared to LPC in this particular scenario.

However, it's important to note that the higher accuracy of LPC over MFCC in this case doesn't necessarily generalize to all situations and speech recognition applications. The performance of these methods can vary depending on the specific dataset, variability in speech patterns, and the complexity of the recognition task at hand.

To draw more conclusive insights, it would be beneficial to analyze additional factors such as the size and diversity of the dataset, the quality of recordings, the characteristics of the speakers, and any preprocessing or classification techniques employed in the speech recognition system.

5. Unresolved issues

Insufficient data collection: The project's data set was too little to guarantee the identification and categorization of a wide variety of speakers. To solve this problem, a larger data collection is required in order to improve speaker recognition accuracy.

1. Training data that was skewed: The training data was strongly impacted by outside elements like the recording setting and equipment. It is required to either eliminate these elements by recording in the same area with the same tools or to diversity the settings in order to prevent the emergence of patterns that feature extraction algorithms can take advantage of in order to solve this problem.
2. Sophisticated algorithms and their variations: For speaker recognition, there are a large number of sophisticated algorithms and their variations. For this job, neural networks may also be used. The research done for this project was restricted to simple voice recognition. To investigate more sophisticated algorithms and improve the speaker identification system's functionality and accuracy, more study and development is needed.
3. Handling noise and variability: Handling noise and variability in voice signals is a serious difficulty. Background noise, pronunciation changes, and other elements in real-world voice signals can affect how well they are recognized. These problems may be solved and system performance can be increased by putting noise reduction techniques, reliable feature extraction techniques, and adaptive modeling strategies into practice.
4. Generalization to different speakers: When faced with speakers outside of the training set, the speaker identification system could have trouble. The problem of generalizing the system to accommodate various speakers, including those with various accents, speaking styles, or languages, is still open. It would be advantageous to develop methods to increase the system's capacity to recognize a larger variety of speakers.

Research possibilities and prospects for breakthroughs in the area of voice processing and speaker recognition are presented by these unanswered problems.