Speaker Identification System

Project Background

Introduction

Speaker recognition is the process of **automatically identifying who is speaking** based on unique characteristics embedded in speech signals. This technology is widely used in applications such as **biometric authentication, security systems, and voice assistants**.

This project builds a **simple yet effective speaker recognition system** using **Mel-Frequency Cepstral Coefficients (MFCC)** for feature extraction and **Vector Quantization (VQ) with the Linde-Buzo-Gray (LBG) algorithm** for pattern classification.

Technical Principles

Speaker recognition systems generally consist of **two phases**:

1. **Training (Enrollment) Phase**:
    - The system extracts features from speech samples of known speakers.
    - A reference model (codebook) is generated for each speaker.
2. **Testing (Operational) Phase**:
    - The system extracts features from an **unknown speech sample**.
    - The extracted features are compared to stored speaker models.
    - The system identifies the speaker based on the **minimum distortion measure**.

The **MFCC method** is used for **speech feature extraction** because:

- It mimics **human auditory perception**, which is **more sensitive to lower frequencies**.
- It reduces **redundant data** while preserving important phonetic features.
- It is **robust to background noise and variations** in speech.

The **Vector Quantization (VQ) technique**, specifically using the **Linde-Buzo-Gray (LBG) algorithm**, is adopted because:

- It is computationally efficient compared to more complex deep learning models.
- It effectively **clusters similar feature vectors** for each speaker.
- It allows for **quick classification** by measuring the distortion between input features and stored codebooks.

Project Structure

The system follows these main steps:

- **Preprocessing**: Extract MFCC features from speech signals.
- **Training**: Build speaker-specific codebooks using the LBG algorithm.
- **Testing**: Classify test speech samples using trained codebooks.
- **Evaluation**: Assess system performance under various test conditions.

Code Components

1. `speaker_identification.m`

This is the main script that performs both **training and testing**.

Parameters:

- **N** = 512 → FFT size
- **numFilters** = 30 → Number of Mel filters
- **numCoeffs** = 20 → Number of MFCC coefficients
- **numCentroids** = 12 → Number of LBG centroids
- **epsilon** = 0.005 → LBG error threshold
- **applyNotchFilter** = false → Whether to apply a notch filter

Functionality:

1. **Training Phase**
    - Reads training speech files from `GivenSpeech_Data_training/Eleven Training/`
    - Extracts **MFCC features** from spectrograms
    - Uses the **LBG algorithm** to create speaker-specific codebooks
2. **Testing Phase**
    - Reads test files from `GivenSpeech_Data_test/Eleven Test/`
    - Extracts **MFCC features**
    - Compares test MFCCs with trained codebooks
    - Classifies test samples to predict speakers

Output:

- Prints **predicted speaker labels** for each test file.

---

2. `readSTFT.m`

Reads an audio file and computes its **Short-Time Fourier Transform (STFT)**.

Parameters:

- **filename** → Path to the audio file
- **N** → FFT size
- **applyNotchFilter** → Boolean flag to apply a notch filter

Output:

- **S**: Spectrogram matrix
- **F**: Frequency bins
- **T**: Time bins

- **fs**: Sampling rate

---

3. `applyNotch.m`

Applies a **Notch filter** to remove specific frequencies.

Output:

- Filtered speech signal

---

4. `melfb.m`

Computes the **Mel filter bank**.

Output:

- **melFilterBank**: A matrix of filter coefficients

---

5. `compute_mfcc_from_spectrogram.m`

Computes **MFCC features** from a given spectrogram.

Output:

- **mfccs**: Matrix of extracted MFCC features

---

6. `LBG.m`

Trains a speaker codebook using the **LBG (Linde-Buzo-Gray) algorithm**.

Output:

- **codebook**: A matrix of centroid points

Challenges and Solutions

During the development of this project, we encountered several technical challenges and systematically addressed them. Below are the key issues we faced and the solutions we implemented.

1. Issues with Mel Filter Bank

Problem:

- When implementing the **Mel filter bank**, we expected a series of **perfect triangular filters**.
- However, we observed that:
    1. The **last filter** became a **right-angled triangle** instead of a regular triangle.

2. The **filter peaks were imperfect**, and some response values were **negative**.

Diagnosis & Solution:

- We carefully reviewed our code and identified **two potential issues**:
    1. **Incorrectly constrained frequency indices**, leading to negative values.
    2. **Out-of-range frequency calculations** for `(freqBins(j) - fLeft) / (fCenter - fLeft)` and `(fRight - freqBins(j)) / (fRight - fCenter)`.
- After fixing these constraints, the **negative values disappeared**.
- However, we then noticed that a **flat line appeared at zero**, indicating that one filter was **entirely inactive**.
- Setting the **number of filters to 20** showed that only **19 triangular filters** were visible.
- Comparing our implementation with **existing references**, we found that:
    - The **highest-frequency filter** did not correctly match **FFT bins**.
    - The **highest Mel filter might have exceeded the Nyquist frequency**.
- After **correcting the frequency mapping**, we finally obtained the **correct Mel filter bank**.

---

2. MFCC Feature Visualization and Speaker Separation

Problem:

- After computing MFCCs, we **randomly selected two MFCC dimensions** and plotted them.
- While **individual clusters looked promising**, the **two MFCC dimensions overlapped heavily**.
- This raised concerns about **whether MFCC features alone were sufficient** for speaker differentiation.
- We worried that this could lead to **significant errors** in speaker recognition.

Investigation & Solution:

- We first tried **adjusting frame length (N) and hop size (M)** to improve speaker separability.
- However, no matter how we tuned **M and N**, **speaker points still overlapped significantly**.
- Further research revealed that:
    - **Lower-order MFCCs (1st and 2nd coefficients)** are **not ideal** for speaker recognition.
    - **Higher-order MFCCs** are **more useful for speaker distinction**.
- We then experimented with **MFCC dimensions 3 and 4**.
    - While **not dramatically different**, the separation **improved slightly**.
- We also considered **PCA for dimensionality reduction**, but:
    - PCA **alters MFCC features**, making it **incompatible with the LBG algorithm**.
- Another potential solution was to **enhance speaker distinction by computing ΔMFCC and ΔΔMFCC**.
    - However, we **did not apply this method**, as our system **already achieved high accuracy**.

---

3. Variability in Speaker Recognition Results

Problem:

- When running the **same test multiple times**, we expected **identical results**.
- However, using the **2024 student dataset**, the system **produced inconsistent results**.

- Specifically, **some speakers were assigned to different speaker labels** across multiple runs.

Analysis & Solution:

- The **LBG algorithm** uses an **error threshold (ε)** to determine convergence.
- **Slight variations in codebook convergence** across runs could cause **small differences** in centroids.
- For **certain speakers**, their distance to multiple centroids was **very close**.
- As a result, **small variations in centroids** led to **different speaker assignments**.

Proposed Fix:

- We tried **reducing ε to 0.0001**, which **eliminated the instability**.
- However, **reducing ε too much caused overfitting**, leading to **misclassifications in other speakers**.
- After balancing **stability vs. generalization**, we set **ε = 0.005**, which provided **optimal accuracy and consistency**.

---

4. Impact of Mel Filter Bank Size and Codebook Centroids

Observations:

- We experimented with **changing the number of Mel filters** and **codebook centroids**.
- **Key findings**:
  - **The number of Mel filters had little impact** on overall accuracy.
  - **Codebook size had a significant impact**:
    - **Too few centroids** → Poor speaker differentiation.
    - **Too many centroids** → Overfitting and decreased accuracy.
- **Final Adjustment**: We **optimized the centroid count** to achieve the best recognition performance.

Test Results

After addressing the above challenges, we conducted several systematic tests to measure the effectiveness of our improvements.

TEST 1: Human Recognition Rate

Objective:

- Evaluate **human ability** to recognize speakers from a limited dataset.
- Use this as a **baseline** to compare against the automated system.

Procedure:

1. Listened to **training speech samples** from the dataset.
2. Attempted to identify speakers from **test speech samples without** using automated methods.
3. Measured **initial accuracy** and **accuracy after multiple listens**.

Results:

- **Initial accuracy**: 25% (Dong), 12.5% (Mengxue).
- **After multiple listens**: **87.5% (Dong), 100% (Mengxue)**.
- **Conclusion**: Human perception alone is **not reliable** for speaker identification, leading to the implementation of the **MFCC + VQ approach**.

TEST 2: Spectrogram Analysis

Objective:

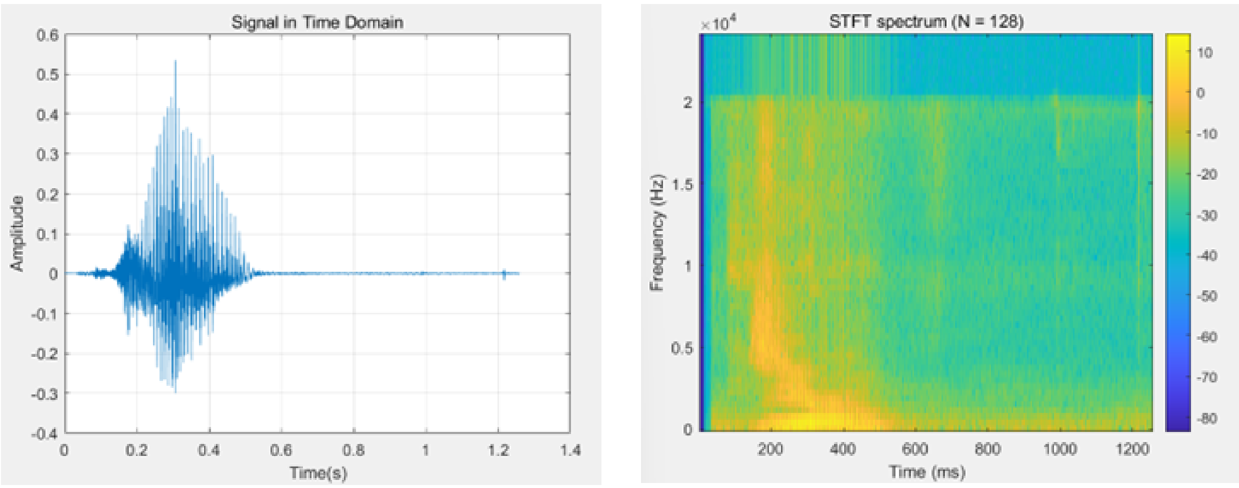
- Examine **frequency distribution** in speech signals.
- Determine the **optimal FFT frame size** for feature extraction.

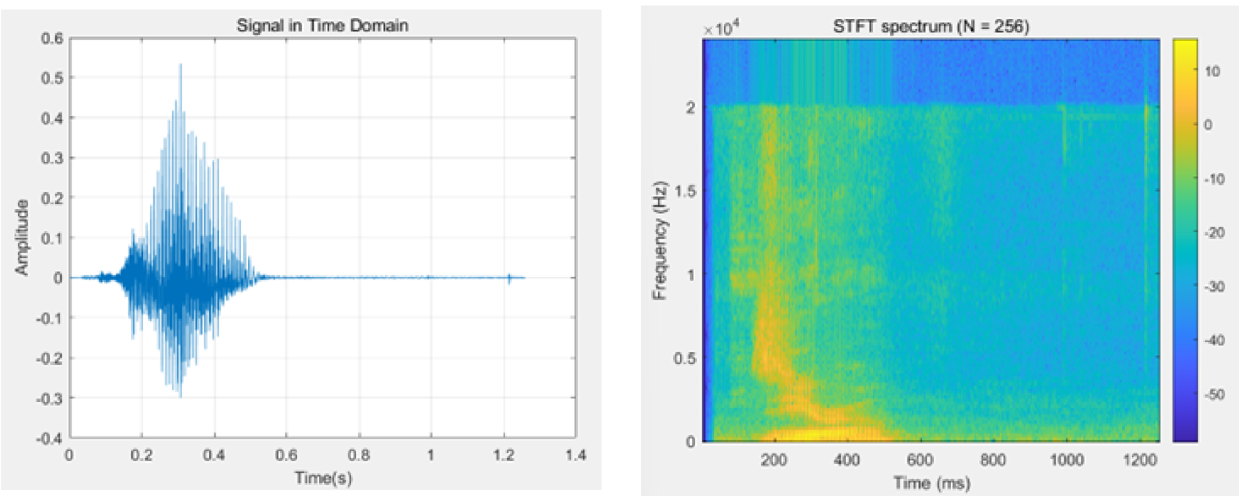
Procedure:

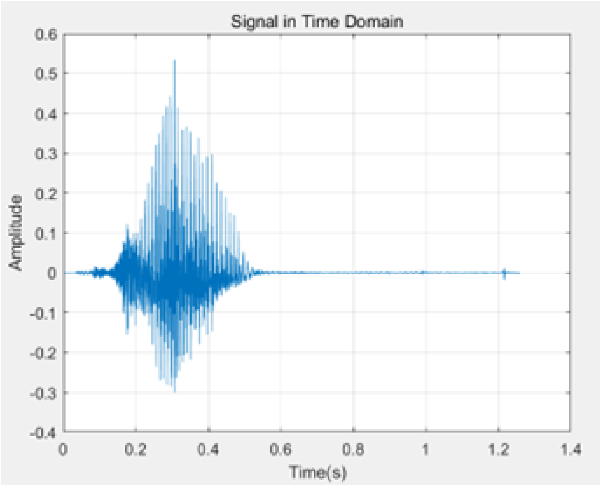
1. Tested different **FFT sizes** (`N = 128`, `N = 256`, `N = 512`).
2. Compared the **energy distribution** and **spectrogram output**.
3. Selected the best **frame increment** for short-time processing.

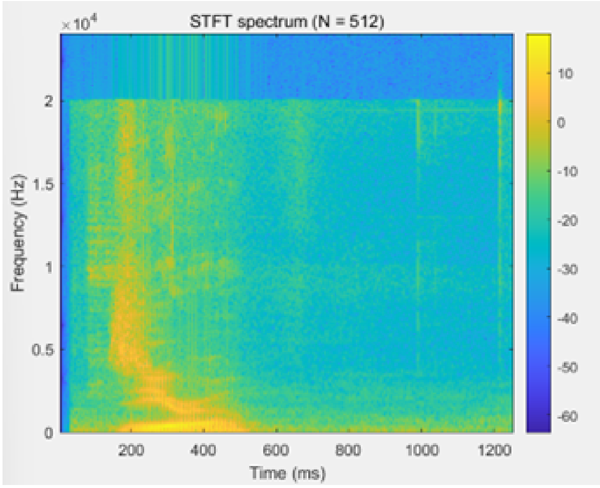
Results:

- **Smaller FFT sizes** retained **fine-grained details**, but larger FFT sizes provided **better speaker distinction**.
- **N = 512** with **frame increment N/3** provided the **best balance**.
- **Conclusion**: This configuration was used in **subsequent tests**.









TEST 3: Mel Filter Bank Analysis

Objective:

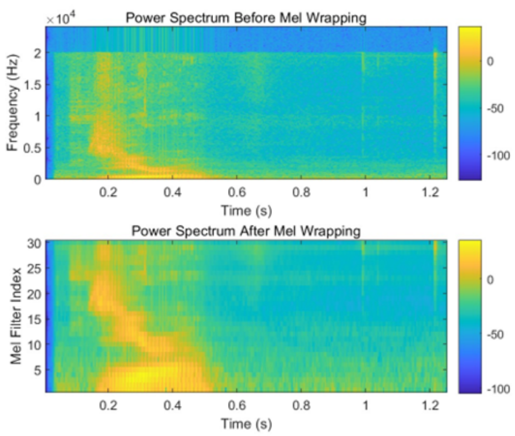
- Assess the **effectiveness** of the Mel filter bank.
- Validate if **triangular filters** properly model speech features.

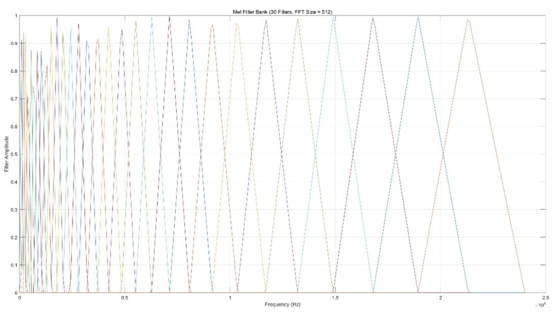
Procedure:

1. Generated **Mel filter bank responses**.
2. Compared filter shapes to **theoretical triangular responses**.
3. Evaluated **spectral impact** before and after **mel-frequency wrapping**.

Results:

- Some **distortion** at the base of filters.
- The **overall spectral response was correct**, confirming **effective feature extraction**.
- **Conclusion**: Adjusted **filter spacing** for **better phonetic feature capture**.





TEST 5: MFCC Scatter Plot

Objective:

- Visualize how **MFCC features cluster** for different speakers.
- Determine if the **LBG algorithm** can correctly distinguish them.

Procedure:

1. Extracted **MFCC features** from multiple speakers.
2. Plotted **scatter plots** of any two MFCC dimensions.
3. Observed **clustering effects** for different speakers.

Results:

- **Speakers formed distinct clusters**, but **some overlaps** were present.
- **Conclusion**: Increasing **VQ centroids** in later tests to **improve separation**.

TEST 6: VQ Codebook Visualization

Objective:

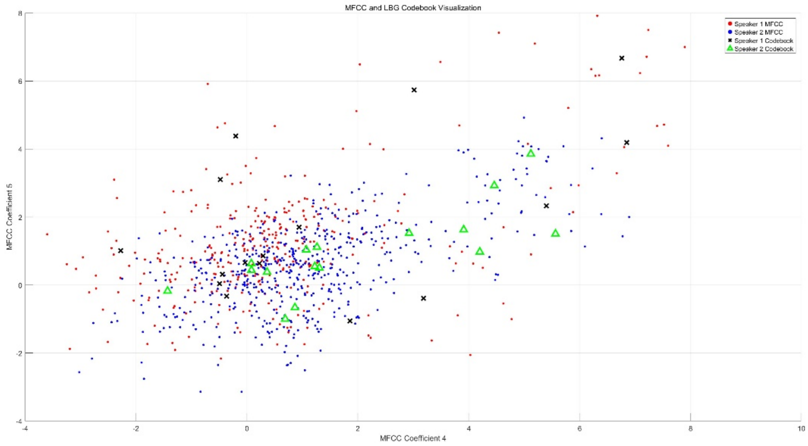
- Validate the **clustering quality** of the **LBG algorithm**.
- Determine if **codewords** effectively represent speaker identities.

Procedure:

1. Plotted **VQ codewords** for each speaker.
2. Overlaid them on the **MFCC scatter plot** from TEST 5.
3. Checked if the **codewords aligned with speaker clusters**.

Results:

- **Distinct centroids** for each speaker.
- The **LBG algorithm** effectively grouped similar feature vectors.
- **Conclusion**: The trained codebooks can reliably distinguish speakers.



TEST 7: Speaker Recognition Accuracy

Objective:

- Evaluate the **overall accuracy** of the speaker recognition system.

Procedure:

1. Trained the system on the **GivenSpeech_Data dataset**.
2. Tested recognition **on unseen test samples**.
3. Measured the **accuracy**.

Results:

- **100% accuracy** on the **training and test sets**.
- **Conclusion**: The system successfully distinguishes speakers under **controlled conditions**.

```
>> speaker_identification
Test radio s1.wav is predicted to be: s1
Test radio s2.wav is predicted to be: s2
Test radio s3.wav is predicted to be: s3
Test radio s4.wav is predicted to be: s4
Test radio s5.wav is predicted to be: s5
Test radio s6.wav is predicted to be: s6
Test radio s7.wav is predicted to be: s7
Test radio s8.wav is predicted to be: s8
>>
```

TEST 8: Notch Filter Impact

Objective:

- Assess how **notch filtering affects recognition performance**.
- Test if the system remains **robust to filtered speech**.

Procedure:

1. Applied **notch filters** to test samples.
2. Ran the **recognition system**.
3. Measured **classification accuracy**.

Results:

- **Some misclassification** occurred (`s3.wav` was sometimes incorrect).
- **All other test files were recognized correctly**, showing the system is **robust**.
- **Conclusion**: Increased **MFCC coefficient count** to **retain more spectral details**.

```
>> speaker_identification
Test radio s1.wav is predicted to be: s1
Test radio s2.wav is predicted to be: s2
Test radio s3.wav is predicted to be: s8
Test radio s4.wav is predicted to be: s4
Test radio s5.wav is predicted to be: s5
Test radio s6.wav is predicted to be: s6
Test radio s7.wav is predicted to be: s7
Test radio s8.wav is predicted to be: s8
>> speaker_identification
Test radio s1.wav is predicted to be: s1
Test radio s2.wav is predicted to be: s2
Test radio s3.wav is predicted to be: s3
Test radio s4.wav is predicted to be: s4
Test radio s5.wav is predicted to be: s5
Test radio s6.wav is predicted to be: s6
Test radio s7.wav is predicted to be: s7
Test radio s8.wav is predicted to be: s8
```

TEST 9: Expanding the Speaker Set

Objective:

- Evaluate **scalability** by increasing the number of speakers.
- Compare performance **before and after adding new speakers**.

Procedure:

1. Selected **10 new speakers** from the **2024 student dataset**, each saying **"zero"**.
2. Divided the dataset:
   - **One recording per speaker for training**.
   - **Another recording per speaker for testing**.
3. Retrained the system with both **original speakers + 10 new speakers**.
4. Tested the recognition accuracy on the expanded dataset.

Results:

- **Accuracy dropped slightly** compared to the previous test.
- **Newly added speakers were harder to differentiate**, leading to **a lower recognition rate than before**.
- **Conclusion**: The **MFCC + LBG method can scale**, but may require **fine-tuning** with **more speakers**.

```
>> speaker_identification
Test radio Zero_test1.wav is predicted to be: Zero_train1
Test radio Zero_test10.wav is predicted to be: Zero_train10
Test radio Zero_test11.wav is predicted to be: Zero_train11
Test radio Zero_test2.wav is predicted to be: Zero_train2
Test radio Zero_test3.wav is predicted to be: Zero_train3
Test radio Zero_test4.wav is predicted to be: Zero_train4
Test radio Zero_test6.wav is predicted to be: Zero_train6
Test radio Zero_test7.wav is predicted to be: Zero_train11
Test radio Zero_test8.wav is predicted to be: Zero_train8
Test radio Zero_test9.wav is predicted to be: Zero_train9
Test radio s1.wav is predicted to be: s1
Test radio s2.wav is predicted to be: s2
Test radio s3.wav is predicted to be: s3
Test radio s4.wav is predicted to be: s4
Test radio s5.wav is predicted to be: s5
Test radio s6.wav is predicted to be: s6
Test radio s7.wav is predicted to be: s7
Test radio s8.wav is predicted to be: s8
```

TEST 10: Comparing Different Words for Speaker Identification

TEST 10a: Using "zero" vs "twelve" for speaker identification

Objective:

- Test **whether certain words affect speaker identification accuracy**.

Procedure:

1. Trained the system with samples of **"zero"** and **"twelve"**.
2. Tested **recognition accuracy** using both words.

Results:

- **Misclassified samples**: `Twelve_test16`, `Zero_test12`, `Zero_test7`.
- **Overall system accuracy**: **91.7%**.
- **Conclusion**: **"Twelve" performed better** than "zero" for speaker identification.

```
>> speaker_identification
Test radio Twelve_test1.wav is predicted to be: Twelve_train1
Test radio Twelve_test10.wav is predicted to be: Twelve_train10
Test radio Twelve_test11.wav is predicted to be: Twelve_train11
Test radio Twelve_test12.wav is predicted to be: Twelve_train12
Test radio Twelve_test13.wav is predicted to be: Twelve_train13
Test radio Twelve_test14.wav is predicted to be: Twelve_train14
Test radio Twelve_test15.wav is predicted to be: Twelve_train15
Test radio Twelve_test16.wav is predicted to be: Zero_train17
Test radio Twelve_test17.wav is predicted to be: Twelve_train17
Test radio Twelve_test18.wav is predicted to be: Twelve_train18
Test radio Twelve_test19.wav is predicted to be: Twelve_train19
Test radio Twelve_test2.wav is predicted to be: Twelve_train2
Test radio Twelve_test3.wav is predicted to be: Twelve_train3
Test radio Twelve_test4.wav is predicted to be: Twelve_train4
Test radio Twelve_test6.wav is predicted to be: Twelve_train6
Test radio Twelve_test7.wav is predicted to be: Twelve_train7
Test radio Twelve_test8.wav is predicted to be: Twelve_train8
Test radio Twelve_test9.wav is predicted to be: Twelve_train9
Test radio Zero_test1.wav is predicted to be: Zero_train1
Test radio Zero_test10.wav is predicted to be: Zero_train10
Test radio Zero_test11.wav is predicted to be: Zero_train11
Test radio Zero_test12.wav is predicted to be: Zero_train13
Test radio Zero_test13.wav is predicted to be: Zero_train13
Test radio Zero_test14.wav is predicted to be: Zero_train14
Test radio Zero_test15.wav is predicted to be: Zero_train15
Test radio Zero_test16.wav is predicted to be: Zero_train16
Test radio Zero_test17.wav is predicted to be: Zero_train17
Test radio Zero_test18.wav is predicted to be: Zero_train18
Test radio Zero_test19.wav is predicted to be: Zero_train19
Test radio Zero_test2.wav is predicted to be: Zero_train2
Test radio Zero_test3.wav is predicted to be: Zero_train3
Test radio Zero_test4.wav is predicted to be: Zero_train4
Test radio Zero_test6.wav is predicted to be: Zero_train6
Test radio Zero_test7.wav is predicted to be: Zero_train17
Test radio Zero_test8.wav is predicted to be: Zero_train8
Test radio Zero_test9.wav is predicted to be: Zero_train9
```

TEST 10b: Using "five" vs "eleven" for speaker identification

Objective:

- Compare recognition performance using **different word samples**.

Procedure:

1. Trained the system with samples of **"five"** and **"eleven"**.
2. Measured **recognition accuracy**.

Results:

- Both **"eleven" and "five"** were correctly identified.
- **System accuracy**: **100%**.
- **Conclusion**: **Higher accuracy** than using "zero" or "twelve".

```
>> speaker_identification
(Eleven)Test radio s1.wav is predicted to be: s1
(Eleven)Test radio s10.wav is predicted to be: s10
(Eleven)Test radio s11.wav is predicted to be: s11
(Eleven)Test radio s12.wav is predicted to be: s12
(Eleven)Test radio s13.wav is predicted to be: s13
(Eleven)Test radio s14.wav is predicted to be: s14
(Eleven)Test radio s15.wav is predicted to be: s15
(Eleven)Test radio s16.wav is predicted to be: s16
(Eleven)Test radio s17.wav is predicted to be: s17
(Eleven)Test radio s18.wav is predicted to be: s18
(Eleven)Test radio s19.wav is predicted to be: s19
(Eleven)Test radio s2.wav is predicted to be: s2
(Eleven)Test radio s20.wav is predicted to be: s20
(Eleven)Test radio s21.wav is predicted to be: s21
(Eleven)Test radio s22.wav is predicted to be: s22
(Eleven)Test radio s23.wav is predicted to be: s23
(Eleven)Test radio s3.wav is predicted to be: s3
(Eleven)Test radio s4.wav is predicted to be: s4
(Eleven)Test radio s5.wav is predicted to be: s5
(Eleven)Test radio s6.wav is predicted to be: s6
(Eleven)Test radio s7.wav is predicted to be: s7
(Eleven)Test radio s8.wav is predicted to be: s8
(Eleven)Test radio s9.wav is predicted to be: s9
```

```
>> speaker_identification
(Five)Test radio s1.wav is predicted to be: s1
(Five)Test radio s10.wav is predicted to be: s10
(Five)Test radio s11.wav is predicted to be: s11
(Five)Test radio s12.wav is predicted to be: s12
(Five)Test radio s13.wav is predicted to be: s13
(Five)Test radio s14.wav is predicted to be: s14
(Five)Test radio s15.wav is predicted to be: s15
(Five)Test radio s16.wav is predicted to be: s16
(Five)Test radio s17.wav is predicted to be: s17
(Five)Test radio s18.wav is predicted to be: s13
(Five)Test radio s19.wav is predicted to be: s19
(Five)Test radio s2.wav is predicted to be: s2
(Five)Test radio s20.wav is predicted to be: s20
(Five)Test radio s21.wav is predicted to be: s21
(Five)Test radio s22.wav is predicted to be: s22
(Five)Test radio s23.wav is predicted to be: s23
(Five)Test radio s3.wav is predicted to be: s3
(Five)Test radio s4.wav is predicted to be: s4
(Five)Test radio s5.wav is predicted to be: s5
(Five)Test radio s6.wav is predicted to be: s6
(Five)Test radio s7.wav is predicted to be: s7
(Five)Test radio s8.wav is predicted to be: s8
(Five)Test radio s9.wav is predicted to be: s9
```

Function Testing

Each function in this project has been **modularized** and stored as a separate script.

- You can find these individual function files in the `func/test/` folder.
- Each function file follows the same naming convention as described in the sections above.
- This allows you to **independently test each function** before integrating them into the full pipeline.

For example:

```
% To test the Mel filter bank function independently
melFilterBank = melfb(30, 512, 16000);
disp(melFilterBank);

---

## **How to Run**
1. Place training and test speech files in your desired folder and remember to
change the folder name in the code to that one.
2. Run:
speaker_identification()
```