Certainly! Here’s a more detailed and structured MATLAB code for a voice biometric system using Gaussian Mixture Models (GMM) and Mel-Frequency Cepstral Coefficients (MFCC).

**Requirements**

1. **MATLAB**: Ensure you have MATLAB installed.
2. **Voicebox Toolbox**: Download the Voicebox toolbox for MFCC computation from here and add it to your MATLAB path.

**Directory Structure**

* data/training/: Directory containing training audio files.
* data/testing/: Directory containing testing audio files.

**MATLAB Code**

% Voice Biometric System in MATLAB

% Add necessary paths (you need the Voicebox toolbox for MFCC)

addpath('voicebox'); % Make sure you have Voicebox toolbox for MATLAB

% Parameters

numComponents = 16; % Number of Gaussian components

numMFCC = 12; % Number of MFCC features

fs = 16000; % Sampling frequency

% Load training data (enrollment phase)

trainingData = loadAudioData('data/training', fs);

% Train GMM model

gmmModel = trainGMM(trainingData, numComponents, numMFCC);

% Load testing data (authentication phase)

testingData = loadAudioData('data/testing', fs);

% Authenticate using GMM model

authenticate(gmmModel, testingData, numMFCC);

% Function to load audio data

function data = loadAudioData(directory, fs)

audioFiles = dir(fullfile(directory, '\*.wav'));

data = cell(1, length(audioFiles));

for i = 1:length(audioFiles)

[audio, originalFs] = audioread(fullfile(directory, audioFiles(i).name));

if originalFs ~= fs

audio = resample(audio, fs, originalFs); % Resample to target frequency

end

data{i} = audio;

end

end

% Function to train GMM model

function gmmModel = trainGMM(trainingData, numComponents, numMFCC)

allMFCC = [];

for i = 1:length(trainingData)

mfccFeatures = extractMFCC(trainingData{i}, numMFCC);

allMFCC = [allMFCC; mfccFeatures];

end

gmmModel = fitgmdist(allMFCC, numComponents, 'CovarianceType', 'diagonal', 'RegularizationValue', 0.01);

end

% Function to authenticate using GMM model

function authenticate(gmmModel, testingData, numMFCC)

for i = 1:length(testingData)

mfccFeatures = extractMFCC(testingData{i}, numMFCC);

logLikelihood = sum(log(pdf(gmmModel, mfccFeatures)));

fprintf('Log-Likelihood for test sample %d: %f\n', i, logLikelihood);

end

end

% Function to extract MFCC features

function mfccFeatures = extractMFCC(audio, numMFCC)

% Apply pre-emphasis filter

preEmphasized = filter([1 -0.97], 1, audio);

% Compute MFCC features

mfccFeatures = melcepst(preEmphasized, 16000, 'M', numMFCC, floor(3\*log(16000)), 0.025\*16000, 0.01\*16000);

end

**Explanation**

1. **Loading Audio Data**:
   * The loadAudioData function reads all .wav files in the specified directory, resampling them to the target frequency (fs).
2. **Training GMM Model**:
   * The trainGMM function extracts MFCC features from all training audio files using the extractMFCC function.
   * These features are then used to train a GMM model with the specified number of components and diagonal covariance type.
3. **Authenticating**:
   * The authenticate function extracts MFCC features from each testing audio file and calculates the log-likelihood of these features under the trained GMM model.
4. **MFCC Extraction**:
   * The extractMFCC function applies a pre-emphasis filter to the audio signal and then computes the MFCC features using the melcepst function from the Voicebox toolbox.

**Steps to Run the Code**

1. **Download and add Voicebox toolbox**:
   * Download Voicebox toolbox from here.
   * Unzip and add the Voicebox directory to your MATLAB path using addpath('path\_to\_voicebox\_directory');.
2. **Prepare Data**:
   * Place your training and testing audio files in the data/training and data/testing directories, respectively.
3. **Run the Script**:
   * Execute the script in MATLAB.

This code provides a structured approach to building a voice biometric system, focusing on clarity and modularity for ease of understanding and further development.

4o

where i can collect testing and training data

ChatGPT

For collecting testing and training data for a voice biometric system, you have several options:

**1. Publicly Available Datasets**

There are several publicly available datasets that can be used for training and testing voice biometric systems. Here are a few options:

**VoxCeleb**

* **Description**: Large-scale speaker identification dataset.
* **Link**: VoxCeleb Dataset

**LibriSpeech**

* **Description**: Corpus of approximately 1000 hours of 16kHz read English speech.
* **Link**: LibriSpeech ASR Corpus

**TIMIT**

* **Description**: Contains recordings of 630 speakers of eight major dialects of American English.
* **Link**: TIMIT Acoustic-Phonetic Continuous Speech Corpus

**2. Recording Your Own Data**

If you want data specific to your application, you can record your own audio data.

**Steps to Record Your Own Data:**

1. **Recording Environment**: Ensure a quiet environment to minimize background noise.
2. **Microphone**: Use a good quality microphone for clear recordings.
3. **Recording Software**: Use software like Audacity (free) or any built-in recording software on your computer or smartphone.
4. **Sample Rate**: Record at a sample rate of 16kHz to match the preprocessing steps in the code.

**Sample Recording Procedure:**

1. **Prepare Prompts**: Create a list of phrases or sentences that users will read aloud.
2. **Collect Data**: Record multiple samples from each user. Ensure consistency in the recording environment.
3. **Save Files**: Save each recording as a .wav file with a meaningful filename, e.g., user1\_sample1.wav, user1\_sample2.wav, etc.

**3. Using Voice Assistants for Data Collection**

You can also leverage voice assistants like Google Assistant, Siri, or Alexa to collect voice data by interacting with these systems. Ensure to obtain permission from users if collecting data this way.

**Example: Preparing Data for Training and Testing**

**Directory Structure:**

* data/training/user1/: Directory containing training audio files for user1.
* data/testing/user1/: Directory containing testing audio files for user1.

**Recording Script Example in MATLAB:**

You can create a script to automate the recording process.

matlab

Copy code

% Voice Recording Script in MATLAB

fs = 16000; % Sample rate

duration = 5; % Duration of each recording in seconds

numSamples = 5; % Number of samples per user

for user = 1:5

userDir = sprintf('data/training/user%d', user);

if ~exist(userDir, 'dir')

mkdir(userDir);

end

for sample = 1:numSamples

fprintf('Recording sample %d for user %d\n', sample, user);

recObj = audiorecorder(fs, 16, 1);

disp('Start speaking.');

recordblocking(recObj, duration);

disp('End of Recording.');

% Get audio data and save to file

audioData = getaudiodata(recObj);

filename = sprintf('%s/user%d\_sample%d.wav', userDir, user, sample);

audiowrite(filename, audioData, fs);

end

end

This script records audio samples for multiple users and saves them in the appropriate directory.

**Final Steps**

1. **Organize Data**: Ensure the recorded audio files are well-organized in directories.
2. **Preprocessing**: Perform any necessary preprocessing (e.g., resampling) before using the data for training/testing.

By following these steps, you can collect and prepare high-quality training and testing data for your voice biometric system.