CS578 Speech Processing
Laboratory 5
Speaker Identification with GMM

George Manos
csd4333@csd.uoc.gr

Alexandros Angelakis
csd4334@csd.uoc.gr

20 January 2023

1 Implementation Details

This project is split into 3 subsections, as described on the project description. The first one is responsible for extracting the speech features given a train set (i.e. select the train folder from TIMIT), the second creates and stores the Gaussian models, and the third one identifies the speaker based on the previously created models, given a test set (i.e. select the test folder from TIMIT).
The theory presented below is based on the course's book, Discrete Speech Signal Processsing, Principles and Practice of T. Quatieri. We also used functions from the Voice Box repository*

1.1 Feature Extraction

For the feature extraction part, we use the function v_melcepst from voicebox, to extract the mel-cepstrum coefficients. For the better part of our experiments, the number of those coefficients will be equal to 12. Also, the number of filters from the filterbank will be the default and equal to $\lfloor 3\log(f_s) \rfloor$, where f_s is the sampling rate. The window length will be equal to 20 and the step size equal to 5, as suggested by the description.

1.2 GMMs

After creating .mat files that contain the aforementioned coefficients, we start creating the GMM models. For this part, we also use a function from voicebox, v_gaussmix to produce the mixture model parameters ($\Theta_i = \{w_i, \mu_i, \Sigma_i\}$ for the mixture weight, mean and covariance matrix respectively). For the most part, we will be producing 12 mixture models and using a full covariance matrix.

Essentially, GMMs are a set of individual multi-dimensional Gaussian pdfs that represent the variability of the speech produced by a speaker, e.g. in the context of the vocal tract shape and glottal flow. The Gaussian pdf is state-dependent in that there is assigned a different Gaussian pdf for each acoustic sound class. We can think of these states at a very broad level, such as individual phonemes. The probability of a feature vector being in any one of I states (or acoustic classes) for a particular speaker model, denoted by Θ, is represented by the mixture of different Gaussian pdfs:

$$p(x|\Theta) = \sum_{i=1}^{I} p_i b_i(x) \tag{1}$$

Where $b_i(x)$ are the component mixture densities:

$$b_i(x) = \frac{1}{\sqrt{2\pi}^R \sqrt{|\Sigma_i|}} e^{-\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1}(x-\mu_i)} \tag{2}$$

Once created the GMMs for a speech file, the model parameters are stored in 3 separate .mat files, for the same speech sample, whose label will be included in the folder's name.
Now that we have a probabilistic model for the feature vectors, we must train, i.e., estimate, parameters for each speaker

*Available here

model, and then classify the test utterances. Essentially, we will be using expectation-maximization (EM) algorithm. The EM algorithm iteratively improves on the GMM parameter estimates by increasing (on each iteration) the probability that the model estimate λ matches the observed feature vectors, i.e., on each iteration $p(X|\Theta^{k+1}) > p(X|\Theta^k)$, k being the iteration number.
The fitting procedure will be specific to the implemented function we are using from Voicebox. That specific one uses one of several initialization methods to create an initial guess for the mixture centres and then uses the EM algorithm to refine the guess. Although the EM algorithm is deterministic, the initialization procedures use random numbers and so the routine will not give identical answers if called multiple times with the same input data. The maximum loop count value is configurable, and we will be using the default one, which will correspond to $\frac{1}{n^2}$, where n corresponds to the number of data values. The iteration will cease if the increase in log likelihood density per data point is less than this value.

1.3 Predicting

Finally, once the GMMs are created, we again extract features from the test dataset using the same feature extraction method as above. For all M frames, and given that we trained GMMs over S speakers, the speaker identification solution is:

$$\hat{S} = \max_{1 \le j \le S} \sum_{m=0}^{M-1} \log(p(x_m|\lambda_j)) \tag{3}$$

where x_m is a feature vector.

2 Our Voices

We both gathered our friends and asked them to kindly provide some recording samples of their voices, in order to construct our own database. We will be using spoken utterences in English and Greek, with a variety of accents, including 2 female speakers and 9 male ones. The ideal would be if they were all recorded using the same microphone, in order to avoid any bias introduced by the microphone. However, given that the sampling rate is small (16khz), we hope that this will not cause many problems. Also, it would be interesting to observe such behaviors. The dataset includes samples from George's sibling (AlexManos). These 2 speech samples, along with Shyrvana, were all recorded using the same microphone, while the rest were totally different ones and independent of each other.
Finally, to avoid speaker frequency bias, we tried to include a similar number of train speech files for most speakers (4-5), while using more personal ones (7-8 respectively). The test ones respectively usually include 1 or 2 speech files per speaker, while George's test folder includes 4 separate ones.

3 Testing

For the TIMIT dataset, having 12 mixture models, 12 features and a full covariance matrix, the resulting classification accuracy is 0.775. However, since it is a really slow process as there are many different speakers on the train set, for our experiments we will be using our own database, as described on the previous section.

4 Experiments

Once again, we will be using our personal voices database for the following experiments since the conditions are set by us. The accuracy is always pretty high, and therefore we can say that our GMM models can correctly discriminate almost all speakers correctly, while indeed the incorrectly classified speech samples are aurally quite similar to one another. On our code results, we can even see which speech file it matched the most, so that one may compare the test speech file with the highest corresponding log likelihood speech file.

4.1 Number of GMMs

We've constructed three different speaker identification modules with different order for the GMM and different types of covariance matrices (diagonal or full). For this section, the number of features extracted is always equal to 12. All of the modules had a slow prediction time for each speech signal in the test data set, but the prediction accuracy was very high.
The results are the following:

- 5 mixtures, full covariance matrix. Accuracy = 0.9375

- 12 mixtures, full covariance matrix. Accuracy = 0.875
- 30 mixtures, full covariance matrix. Accuracy = 0.5625
- 5 mixtures, diagonal covariance matrix. Accuracy = 0.9375
- 12 mixtures, diagonal covariance matrix. Accuracy = 1
- 30 mixtures, diagonal covariance matrix. Accuracy = 0.875

We can interpret the GMM as a "soft" representation of the various acoustic classes that make up the sounds of the speaker; each component density can be thought of as the distribution of possible feature vectors associated with each of the I acoustic classes, each class representing possibly one speech sound (e.g. a particular phoneme) or a set of speech sounds (e.g., voiced or unvoiced). Because only the measured feature vectors are available, we can think of the acoustic classes as being "hidden" processes, each feature vector being generated from a particular class i with probability p_i, on each analysis frame. However, for some specified number of mixtures in the pdf model, generally one cannot make a strict relation between component densities and specific acoustic classes.
Regardless of an acoustic class association, however, a second interpretation of the GMM is a functional representation of a pdf. The GMM, being a linear combination of Gaussian pdfs, has the capability to form an approximation to an arbitrary pdf for a large enough number of mixture components. For speech features, typically having smooth pdfs, a finite number of Gaussians (e.g., 8-64) is sufficient to form a smooth approximation to the pdf.
However, on our experiments we can see that the more mixtures we have for the full covariance matrix, the more the performance drops. Having too many could mean that we have too many models, and therefore many models will likely be similar to one from different speaker, resulting in missclassifications, given that our number of features is kept constant. For the diagonal covariance matrix, since only the variance of the coefficient is kept, we can see that the classification accuracy still drops after increasing too much the number of mixture models. The other accuracy between 5 mixtures and 12 requires more testing before claiming it is statistically significant.

4.2 Number of Features

In this section we will be experimenting with the number of extracted features, essentially the number of mel-cepstrum coefficients produced by the aforementioned proceedure. The GMM production configuration will be the same as the default one (12 mixtures, full covariance matrix) and we will be using our database.
The results are the following:

- 5 features. Accuracy = 1
- 12 features. Accuracy = 0.875
- 25 features. Accuracy = 1
- 40 features. Accuracy = 0.625

Once again, no particular conclusion can be made about the number of features, other that for high numbers, we can see the accuracy drop significantly.

4.3 Wrapping Up

In this section, we will experiment with different combinations of number of features and number of mixtures. We will be using a full covariance matrix for reference, and compare with how those parameters previously affected our results.
The results are the following:

- 5 mixtures, 5 features. Accuracy = 1
- 5 mixtures, 25 features. Accuracy = 1
- 30 mixtures, 5 features. Accuracy = 0.9375
- 30 mixtures, 25 features. Accuracy = 0.5625

Tampering with both variables we can start seeing a pattern. Increasing both of them seems to negatively impact our GMMs accuracy, while the accuracy is 100% when they are both low. Again, further testing is required to prove that those results are statistically significant, yet due to the time constraint they won't be covered by our work.

5 Appendix

Figure 1: Gauss, I summon thee!