Electrical and Computer Engineering
University of Thessaly (UTH)

ECE443 - Speech Processing

Fall Semester — Educational year 2024-2025

Image Stitching Project

Vasileios Stergioulis - AEM: 03166

Emmanouil Pantopoulos - AEM: 03222

Abstract

The goal of this project is to develop a pattern recognition system using Gaussian Mixture Models (GMMs) and MFCC (Mel-Frequency Cepstral Coefficients) features. More specifically the identification system is split into two modules; the features extraction module and the classification or machine learning module. The thematic area of this project focuses around Speech Recognition: Recognizing specific words (5 digits with 250 sound files each) from audio recordings, essentially a digit recognition task.

Contents

1 Feature Extraction

For our Feature Extraction implementation, we choose **HTK MFCC MATLAB** [1]. We chose this particular code package from Matlab Ceentral because we had also worked with the openSmile toolkit, which exports the features in a similar if not identical way. Another reason is the influence HTK toolkit had on the other speech Processing toolkits and because of its robustness (still used in plethora of applications). Except the given fixed parameters, we also introduce to our project another three:

I Pre-emphasis Coefficient α which is equal to 0.97.

II The number of triangular Mel filterbanks M, which is equal to 20 and

III The cepstral sine parameter (lifter) L, which is equal to 22

These extra parameters ensure that MFCCs are as robust as possibly could be. The resulting features are saved in `.mat` format. It should be noted that this toolbox returns the MFCCs for every frame of our speech signal. Due to difference in audio lengths and to make our code more light, we will use the first 50 frames of every signal.

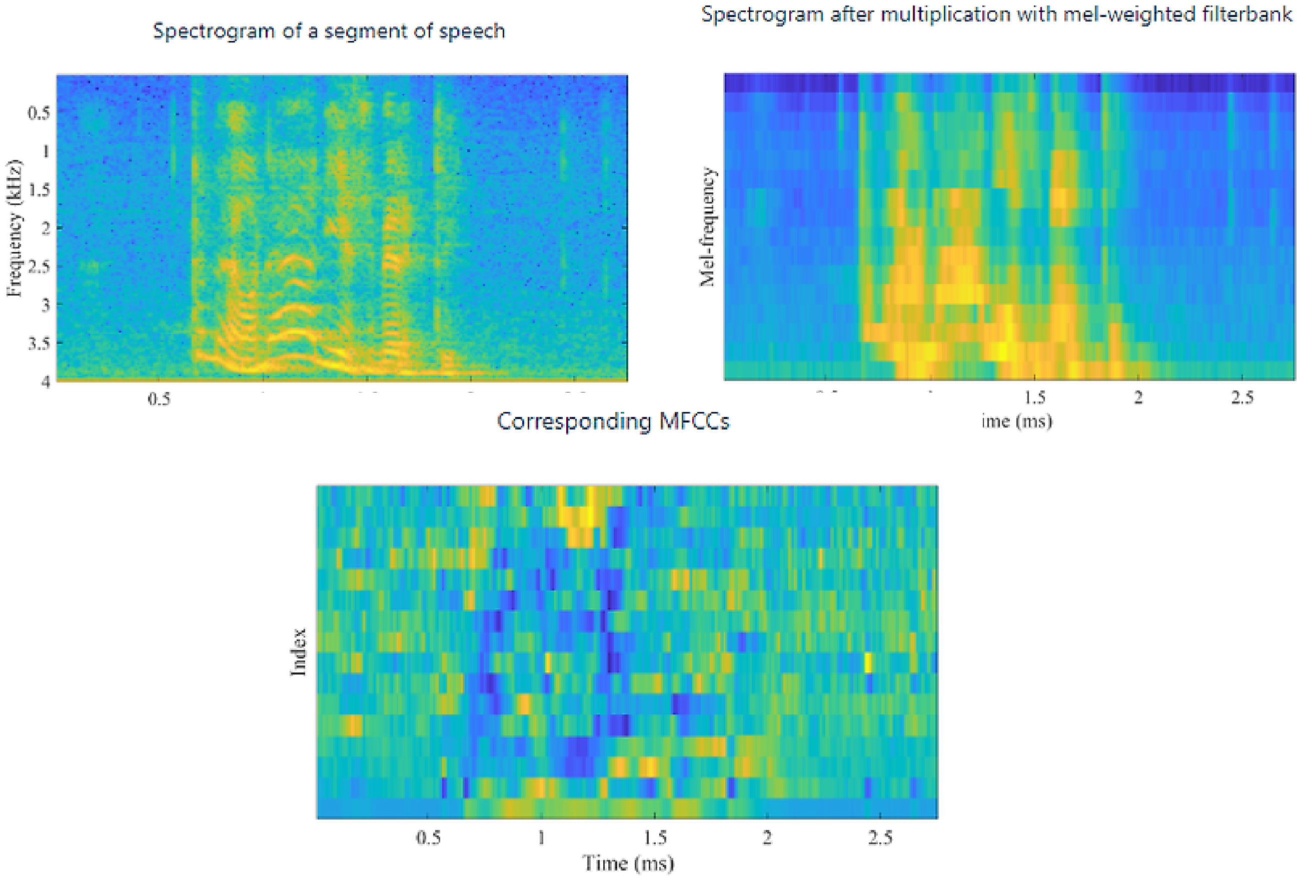

Figure 1: Difference between spectograms and the corresponding MFCCs

[1] https://www.mathworks.com/matlabcentral/fileexchange/32849-htk-mfcc-matlab

2 GMMs

For the creation of our GMMs, we used **EM Algorithm for Gaussian Mixture Model (EM GMM)**[2] toolbox, in order to calculate the GMM parameters $I = [w_i, \mu_i, C_i^2]$, with w being the mixture weight, μ the mean and C_i^2 the covariance matrix. Each parameter is also saved in `.mat` format. For our classification task, we create M total mixtures for every class. Every gaussian mixture is trained with the MFCCs of every class and the probability of a sample belonging to one of the five classes is denoted by the following formula:

$$p(x|I) = \Sigma_{j=1}^{5} p_j g_j(x),$$

where $g_i(x)$ is the gaussian probability density function.

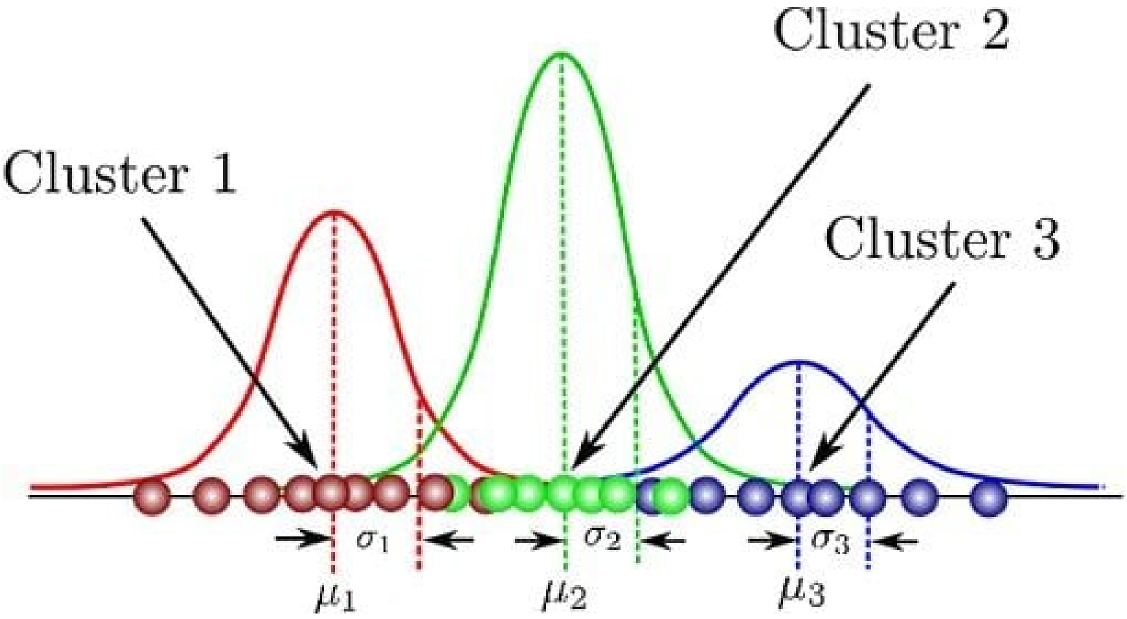

Figure 2: Gaussian Mixture Model

3 Classification

The Maximum a Posteriori (MAP) criterion is used to classify an input signal by selecting the class with the highest posterior probability. Given an observation X and a set of possible classes O_i, the MAP decision rule is defined as:

$$O^* = \arg\max_{O_i} P(O_i \mid X)$$

Under the assumption of equal a priori probabilities, this reduces to maximizing the likelihood:

$$O^* = \arg\max_{O_i} P(X \mid O_i)$$

[2] https://www.mathworks.com/matlabcentral/fileexchange/26184-em-algorithm-for-gaussian-mixture-model-em-gmm

where $P(O_i \mid X)$ is the posterior probability of class O_i given the input X, and $P(X \mid O_i)$ is the likelihood of observing X given that it belongs to class O_i. This approach is commonly used in classification tasks such as speaker recognition, digit recognition, and emotion classification. In order to simplify our task we use log-likelihood. In more simpler words, for every observation-sample X, we assign a probability score for every class and we classify it in teh class with the biggest probability score. We also have to note, that by using GMMs as our classifiers, we essentially perform clustering classification (like K-Nearest Neighbours).

4 Results

In the following Table we present the experiment results

№of Features	**№of Mixtures**	**Accuracy**
Features as a constant		
20	3	60 %
20	6	80 %
20	10	80 %
20	15	80 %
Mixtures as a constant		
5	6	80%
12	6	20 %
24	6	80 %
30	6	80 %

Table 1: Experimental Results. In the first part we keep the number of features constant, whilst on the second the number of GMMs.

4.1 Detailed Analysis of Results

The experimental results presented in Table 1 demonstrate the performance of the Gaussian Mixture Model (GMM) classifier under varying conditions. Specifically, we investigate the impact of two key parameters on the classification accuracy: the number of MFCC features and the number of Gaussian mixtures. The results are divided into two parts: (1) keeping the number of features constant while varying the number of mixtures, and (2) keeping the number of mixtures constant while varying the number of features.

4.1.1 Number of Features as a Constant

With three mixtures, the classifier achieves an accuracy of 60%, probably a result of underfitting, as we don't have enough models ot perform classification correctly. By increasing the number of mixtures, we see that our accuracy score stabilizes at 80 %, which means that four out five samples are classified correctly, without giving any hint of what the “weak class” (the class that gets misclassified the most) is . Thus, **six mixtures** seems to be the optimal number for this task having simultaneously the lowest complexity and the highest accuracy.

4.1.2 Number of Mixtures as a Constant

Now that we have concluded what the total number of mixtures should be, we have to find also the optimal number of MFCCs. Again here, not a lot can be determined, because in 80 % of our experiments, the accuracy is 80 %. The only interesting find, is when 12 features are used, the accuracy drops significantly, indicating that our model gets “confused” and cannot correctly classify our features. The optimal optimal number of features for this task, is **5 MFCCs**

4.2 Discussion

From the above results although we can certainly identify the best parameters for our system, we cannot say assuredly if the №MFCCs, or №GMMs is more crucial for classification. However, we can definitely see some under- and over-fitting, but no distinct pattern of classification can be identified. The misclassification rate, can be a product of the feature type we use (probably by using Logfbes or even the pure spectrogramms could produce better results), or of the type of classifier; GMMs, are a clustering algorithm mostly used for feature selection. If using another classifier (SVM, perceptorns, etc) maybe better results would occur.