# ELEC-E5510 – Speech Recognition

# Assignment 1

Nguyen Xuan Binh, 887799

1/11/2023

## Question 1

**a) What are the properties of MFCC features that make them well suited for automatic speech recognition?**

MFCC is well-suited for ASR because they have these following properties:

1. Human Hearing Perception Modeling: MFCCs are based on the Mel scale, which is linear at low frequencies and logarithmic at high frequencies. It's similar to how our ears perceive sound.

2. Decorrelation: MFCCs transform the signal in such a way that the resulting features are less correlated with each other compared to the raw audio data, using CDT. This makes the data more suitable for modeling and classification tasks in ASR.

3. Noise Robustness: MFCCs are quite robust to noise, making them suitable in noisy channels

4. Practicality: MFCCs have been proven to be effective in various ASR tasks in scientific papers. Additionally, MFCC is easy to be implemented in SR systems due to existing, well-established algorithms and libraries (Python/MATLAB) for computing MFCCs.

**b) Why wouldn't spectrogram or mel-spectrum features work so well?**

**Definitions:**

The spectrogram is a heatmap of the spectrum of frequencies in a sound signal as they vary with time. The intensity is usually measured in decibels.

The mel spectrum is based on the mel scale, after the spectrogram under short FFT transform being filtered by the mel triangle filter banks. It reflects human audio system

Spectrogram and mel-spectrum features wouldn't work so well because:

1. Lack of Phase Information:

Spectrogram: It shows how the signal power is distributed across different frequencies at each point in time. However, it does not capture the phase information of the signal.

The phase of a signal contains the initial time position of the waveforms, which is crucial if we want to reconstruct the original time-domain signal from its frequency representation. While phase information is not always required for SR, it can be important for other audio processing tasks, such as reconstructing the original audio. The same argument is true for the mel spectrum. Therefore, using the spectrogram can lose phase information.

2. Time and Frequency Resolution balance:

When we create a spectrogram, we need to choose the Hamming window size to analyze the signal. A larger window provides better frequency resolution but worse time resolution. Vice versa, a smaller window provides better time resolution but worse frequency resolution.

In a spectrogram, this trade-off is visible, and it might not capture rapid changes in the signal well, or it might blend together frequencies that are close together.

The mel-spectrum, while it provides a human-perception frequency scale, also suffers from this trade-off. Additionally, the mel-scale of the frequency axis can reduce the resolution at higher frequencies, and it has many correlated features not fitting for training the GMM models.

Overall, MFCC is the best candidate for SR task because MFCC features are decorrelated due to applying DCT, whereas the spectrogram and mel-spectrum are not.

**Question 2:**

Variables “test_data” and “test_class” contain the same kind of data as the training data but independent of the training data. In this case it was obtained from different speakers than the training data. Using the training data, train phoneme models with [5, 10, 15, 20, 25, 30, 35] components and evaluate their performance with both the training set and the test set.

Plot the error rates of both train and test sets with respect to the components number in GMMs.

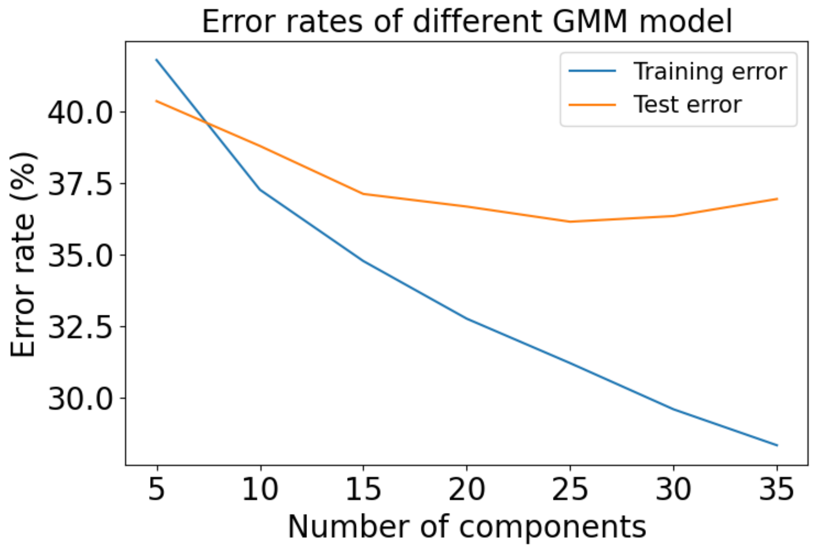

**Figure 1: Error rates of different GMM model**

The training and testing error of different GMM models in number is:

n_components =    [   5,        10,      15,       20,       25,     30,       35    ]

train_error_rates =  [ 41.80, 37.26, 34.76, 32.75, 31.19, 29.58, 28.32 ] %

test_error_rates =   [ 40.36, 38.79, 37.11, 36.67, 36.14, 36.34, 36.93  ] %

**a) Why are the recognition results with the train and the test set different?**

- The training and testing sets may come from different distributions. Specifically, the training data comes from the speech of 50 different Finnish males, while no information is provided regarding sound origin for the testing data.
- As the number of components in the GMM increases, the model becomes more complex and can start to fit the training data too closely, capturing noise as if it were a real pattern. This results in lower training error but can lead to higher test error because the model doesn't generalize well to unseen data, which is called overfitting.

**b) What is a good number of components for recognizing an unknown set of samples?**

The best number of GMM components should return the lowest training error on the testing dataset to prevent overfitting. Therefore, the best number of components is 25, with the lowest testing error at 36.14%. However, 25 components does not necessarily result in minimum loss, as the training loss can be reduced more and more if we continue to increase the components.

## Question 3

Using your best model, classify the test data and generate a confusion matrix using the provided function "confusion_matrix"

The best model is when n_components = 25 (minimum testing data error rate)

```
S = train_gmm(test_data_normalized, test_class, n_components=25)
predictions = predict(test_data_normalized, S, n_components=25)
C = confusion_matrix(test_class, predictions)
```

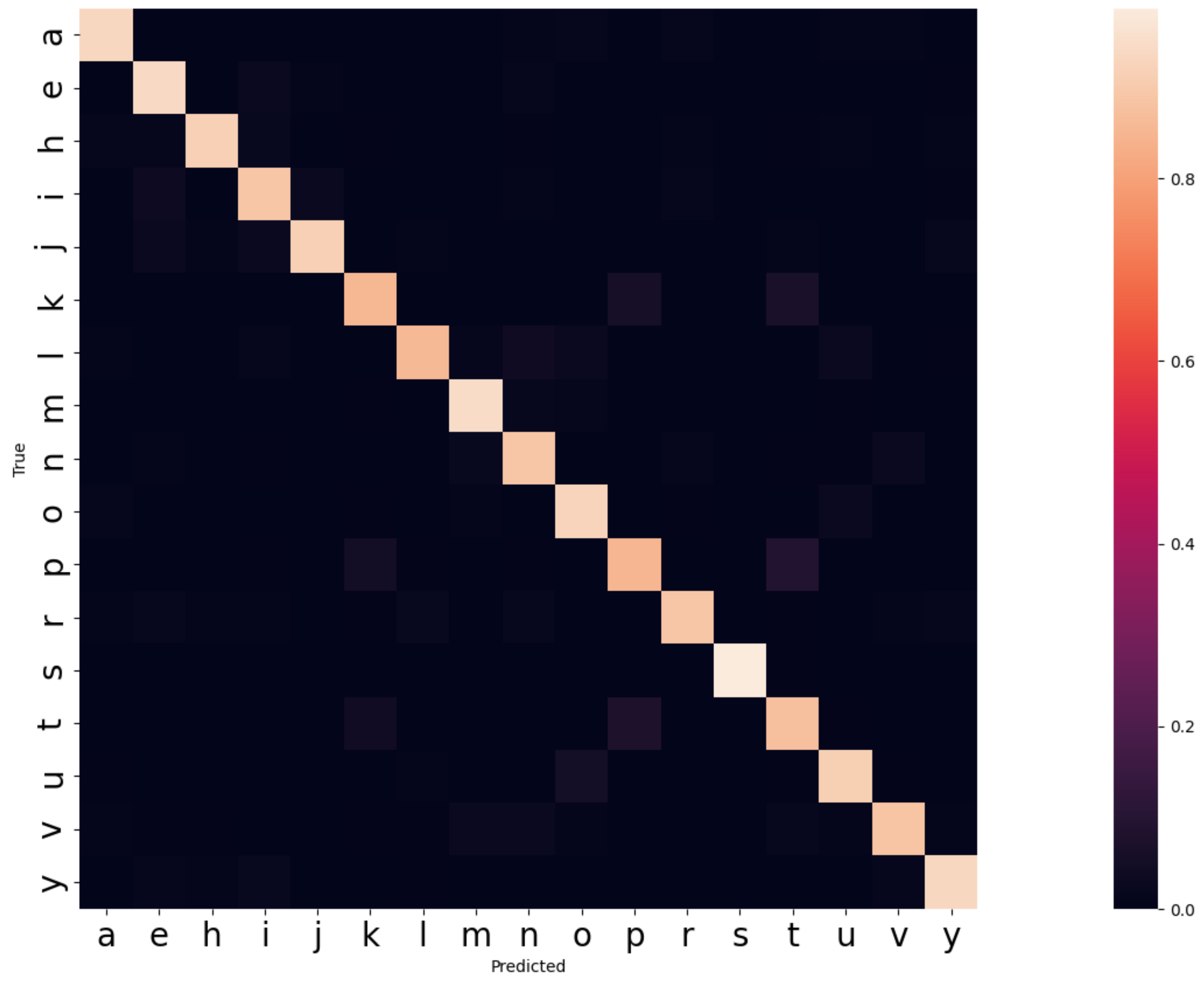

**Figure 2: Confusion matrix on the testing data using GMM, n_components = 25**

**a. Based on the confusion matrix, what can you conclude about phoneme recognition as a task and recognition performance of different phoneme classifiers?**

Phoneme Recognition is the recognition of individual sounds (such as vowels and consonants in this case) in spoken words. By listening to the sound, GMM models predict the phonemes of the testing dataset.

Based on the confusion matrix, the GMM model manages to consistently predict the correct phonemes over 80% of the time, which is much higher than random chances of choosing 1 out of 17 phonemes. Based on the confusion matrix colors, it appears that s and m have the highest accuracy, possibly due to their unique articulation in the Finnish language that no other phonemes share (For example, s could have much lower accuracy in Chinese as this language has many fricative consonants). In brief, the phoneme recognition as a task is very successful.

**b) Give examples of difficulties this classifier has.**

Some difficulties that this classifier has could be:

- The unaspirated stop consonants ”k”, ”p”, ”t”  are usually mistaken by one another.
- Closed back vowels “o” and ”u” are also likely to be mistaken by one another.
- Alveolar sounds ”l” and “n” are sometimes mixed up.

**c) Include the (visualized) confusion matrix with the answer.**

The confusion matrix image is attached above

## Question 4

Variables “tw1”, “tw2” and “tw3” contain the feature representations of three Finnish words. Classify their features using your best model.

Answer the following questions:

**a) Try to identify the words based on the classification result. (We believe this demonstration works especially well if you don't speak Finnish. Points are rewarded for attempts, not correct answers.)**

The classification results and my prediction of the words for each variable are as follows:

- Prediction for tw1 word:
  ptkpkvpkkptttkkkkkkoooooooooooooooooolllllllllllaoolmmmmmmmmmlmmrieeerreeey
  The word that I guess is possibly “kolme” (three)
- Prediction for tw2 word:
  sssssssssssssssssssreeerreeiiireeeeeiiiiiiiiyiisssssssssssssssssssssttt
  The word that I guess is possibly “seis” (stop)
- Prediction for tw3 word:
  ehhhhheejeeeeeeeierlllllrrrmmmmmmmmmviyiiiktktkkkkuouuuuuuuuuuuuuuuuoovorssss
  ssssssssssssssssssssavararuaaluvn
  The word that I guess is possibly “helmikuussa” (in February)

**b) What problems do you see in the frame based classification if one wants to recognize whole words?**

Some problems that I see in the frame based classification are

- The beginning and ending of each words is cluttered with random characters, which can be the effect of noise channels.
- At the beginning of the tw1 prediction (kolme) and middle of tw3 prediction (helmikuussa), there is a cluster of “p”, “t” and “k”, suggesting that the classifier struggles to tell apart these stop consonants. This is evident from the confusion map in task 3b
- The s phoneme detection in tw3 is not interrupted by any other phoneme, suggesting that the classifier is highly confident when “s” is pronounced. This can be seen from the confusion matrix again, as “s” has the highest accuracy score.
- Some other classifier’s mistakes could be mixing up “i” and “y”, “o” and “u” and so on.

**c) Describe ideas to improve the results.**

Some ideas that might improve the results:

- Most straightforward solution is to increase the training dataset size
- Develop models that are specifically trained to distinguish between commonly confused phonemes, like "p", "t", and "k", or "o" and "u". We can explore deep learning-based models for this problem. Also, we can consider different Hamming window sizes. Shorter window frames provide better time resolution for rapid changes in phonemes, such as fast speech or plosive sounds (p,t,k), while longer window frames lead to better frequency resolution to differentiate vowels and sustained sounds.
- Training data should include a more diverse set of speakers, accents, and speaking styles to improve the model’s ability to generalize across different pronunciations. The training dataset only contains adult standard male Finnish speeches, which limits the classifier’s accuracy.

## Question 5

DNNs can leverage large datasets, but here we have quite a small training set.

**a) Which model performs classification better, the DNN or your best GMM?**

DNN performs the classification task better. This is because DNN has average accuracy of 68.14% while GMM with 25 components has an accuracy of (100 - 31.19) = 63.86%

**b) The number of trained parameters in the DNN is reported above. Now look inside your best GMM model, the dictionary returned by “train_gmm”. Note: strictly speaking, there is a separate GMM for each class (phoneme), so count the total number of parameters for all the classes altogether. How many trained (estimated) parameters does your best GMM have (Remember: we used diagonal-covariance matrix GMMs)? Which model type used more parameters?**

```
mu = S['mu']

sigma = S['sigma']

prior = S['prior']

weight = S['weight']

model = S['model']
```

mu.shape:  (17, 25, 26)

sigma.shape:  (17, 25, 26)

prior.shape:  (17,)

weight.shape:  (17, 25)

model.shape:  17

From the matrix shape, we can see that there are 17 phonemes or classes. For each phoneme, we have 25 mixture components. Each of the component in turns has
- 26 parameters for the means
- 26 parameters from the diagonal covariance matrix
- 1 parameter for the prior

Each of the class has a weight parameter, adding 17 more tunable parameters
In total, the GMM with 25 components has (17×25×26)+(17×25×26)+(17×25)+17 = 22542 parameters. Therefore, our best GMM has fewer tunable parameters than DNN, which has 193817 parameters.

**Question 6**

**a) Which model has lower classification error on the sampled MFCCs? Why might that be?**

GMM has lower classification error on the sampled MFCCs. (GMM has an accuracy of 70.58 % and  DNN has an accuracy of 47.05 %). This might be because of

- DNNs require a large amount of data to make accurate predictions. Because the training sample dataset used to train the DNN is relatively small, the GMM outperforms the DNN due to its ability to model the data distribution with fewer samples.
- GMMs assume that the phonemes are generated from a mixture of several Gaussian distributions, each phoneme corresponding to a different component in the data. The sampled Mel-Frequency Cepstral Coefficients (MFCCs) used for training and testing are decorrelated, may match the assumptions of GMM better than the assumptions of DNN.