Tutorial - Exercise sheet 5

Pattern and Speech Recognition

Introduction

For this tutorial, you are going to learn how to initialize a gaussian mixture model (GMM). The presented algorithms are used in real world system like HTK [1]. However, the versions implemented in such systems are optimized to deal with huge amount of data. That's why you will notice big differences between your implementation and these ones.

We are also considering a open source speech corpus : the corpus CMU US SLT ARCTIC (you can find informations here http://festvox.org/cmu_arctic/dbs_slt.html)

1 Data preparation [2pts]

The data set, that you are going to use, is composed by the *mel frequency cepstral coefficients (MFCCs)* extracted from the signal of the corpus described earlier. For those of you who are interested with the extraction methodology, you will find informations in [2]. The main property of these *MFCCs* is that the dimension 0 (first coefficient) corresponds to the gain. We are going to ignore this information. Furthermore, the number of *MFCCs* $M = 50$ and the number of samples $m = 677970$.

The data set is stored in a binary format. Therefore, you will have to use dedicated primitives to load the data. For the users of Matlab you should use *fread* and the data type is **float**; for the users of python/numpy, you should use *numpy.fromfile* and the data type is **numpy.float32**.

In order to visualize more easily the behavior of the algorithms, which you will have to implement, we need to reduce the dimension. To achieve that part, you will use the *principal component analysis (PCA)*. You can use the function *pca* for Matlab and the class *sklearn.decomposition.PCA* for python.

Ex. 1 — Download and extract the data set from https://www.dropbox.com/s/bg2f4u1ujd9qpkj/corpus?dl=0.

Ex. 2 — Load the data set into a matrix $m \times M$ and remove the useless dimension.

Ex. 3 — Transform your data using a *PCA* and select the **first dimension of the projected data !**

Ex. 4 — Plot the **distribution** of the data and indicate the number of clusters you will choose. Justify your answer.

2 Clustering using k-means[6pts]

In this part, we are going to focus on the implementation of the *K-Means* that you have seen during the lecture.

2.1 Cluster association

The key part of the k-means algorithm is to determine to which cluster each sample is associated. To achieve this part, you need to have a distance function. In our case, the distance function we are going to use is the *root mean square error (RMSE)*. It is defined by:

$$RMSE(v1, v2) = \frac{1}{N} \sum_{n=1}^{N} (v1[n] - v2[n])^2 \tag{1}$$

where $v1$ and $v2$ are two vectors of length N.

The coefficient, that you are currently using, are *MFCCs* stripped from the gain coefficients. Generally, to compute a distance between to *MFCCs* vectors we are using a distance named *mel cepstral distortion (MCD)*. This distance is defined as the *RMSE* on *MFCCs* stripped from the dimension 0. Therefore, you are going to apply a distance used in real cases.

Based on this distance function, we can now determine how far an sample is far from the mean. The cluster associated to the sample is the one whose the mean is the closest to the sample.

Ex. 5 — Implement the `rmse` function.

Ex. 6 — Implement the `associate` function.

2.2 Compute means

The next important part of the *K-Means* algorithm is the cluster means computation. To achieve this part, we need the following informations:

- the data set;
- the vector which contains the cluster index associated to each sample;
- k

This function should return a matrix $k \times N$.

Ex. 7 — Why the matrix returned should be $k \times N$?

Ex. 8 — Implement the `compute_means` function

2.3 Initialization

Initialize the *K-Means* algorithm is a difficult part and multiple strategies exist: random, user defined by an expert and based on another algorithm result.

Ex. 9 — As an expert, propose the initialization values. Justify your choice!

2.4 K-means

You are now ready to implement the *K-Means*.The specification you **have to follow** are

- the function has two parameters:
  1. the matrix $m \times N$ where m is the number of samples and N the number of features;
  2. a matrix $k \times N$ containing the initialization values.
- the function returns
  1. a matrix $k \times N$
  2. a vector m which contains the cluster index for each sample

Ex. 10 — Implement the *K-Means* algorithm in the *kmeans* function

Ex. 11 — Plot on the x-axis the means of each cluster

3 GMM Initialization [2pts + 1pts bonus]

Ex. 12 — For each cluster, compute the mean vector and covariance matrix. Compute the weight of each cluster.

Ex. 13 — Explain why, based on these informations, we have defined a GMM.

Ex. 14 — **Bonus:** plot the GMM on the same plot than the histogram

4 Application to the real data set [bonus: 1pts]

Ex. 15 — Adapt your code if necessary to handle the original data set dimension

References

[1] S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. Liu, G. Moore, J. Odell, D. Ollason, D. Povey *et al.*, *The HTK book*. Entropic Cambridge Research Laboratory Cambridge, 1997, vol. 2.

[2] T. Fukada, K. Tokuda, T. Kobayashi, and S. Imai, "An adaptative Algorithm for mel-cepstral analysis of speech," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 1, 1992, pp. 137–140.

Submission instructions

The following instructions are mandatory. If you are not following them, tutors can decide to not correct your exercise.

Submission architecture

You have to generate a **single ZIP file** respecting the following architecture:

```
tutorial1_<matriculation_nb1>_<matriculation_nb2>_<matriculation_nb3>
|
+--- source
|      |
|      +------- file 1
|      +------- file 2
|      +------- ...
+--- rapport.pdf
+--- README.txt
```

where

- **source** contains the source code of your project,
- **rapport.pdf** is the report where you present your solution with **the explanations (!)** and the plots,
- **README** which contains group member informations (name, matriculation numbers and emails) and a **clear** explanation about how to compile and run your source code

The ZIP filename has to be :

```
tutorial1_<matriculation_nb1>_<matriculation_nb2>_<matriculation_nb3>.zip
```

You have to choose between the following languages **python** or **matlab**. Other languages won't be accepted.

Some hints

We advice you to follow the following guidelines in order to avoid problems :

- Avoid building complex systems. The exercises are simple enough.
- Do not include any executables in your submission, as this will cause the e-mail server to reject it.

Grading

Send your assignment to the tutor who is responsible of your group:

- Gerrit Großmann gerritgr@gmail.com
- Sébastien Le Maguer slemaguer@coli.uni-saarland.de
- Kata Naszádi b.naszadi@gmail.com

The email subject should start with `[PSR TUTORIAL 5]`