# Digital Signal Processing: Speaker Recognition Stage Report

Xinyu Zhou, Yuxin Wu, and Tiezheng Li
Tsinghua University

## 1 Introduction

In this project, we are to build a **speaker recognition** system with high accuracy, along with a graphic user interface.

Here, we defined the problem of **speaker recognition** as follows, which comprised of several steps:

- **Enrollment**
  A person to be recognized later must be enrolled first. The enrollment is conducted by recording serveral seconds of a person's voice. The system will then do further process to model the person's characteristics.
- **Recognition**
  A short utterance (typically 3s-5s) of the person to be recognized should be recorded. The utterance is feed to the system, and the system will identify the person who is speaking, and give the result.

## 2 Approach

In this section we will present our aproach to tackle the speaker recognition problem.

### 2.1 Erollment

An continuous speech of a user is collected during enrollment of that user. Depending on the number of users we plan to enroll and the accuracy of recognition system, duration of the utterance we need to collect may vary from 15 seconds to 30 seconds.

Further processing of the utterance follows following steps:

1. **Voice Activity Detection (VAD)**
   Due to the channel difference and background noise may occur, which may degrade the performance of the system, non-speech part of the speech are removed from the utterance. This may lead to information loss (such as person's voice pause style), which in turns introduced the trade-off between loss of information and noise been modeled.

   As the corpus provided by teacher is nearly noise-free, we use a simple energy-based approach to remove the silence part.

2. **Extract MFCC Feature**

   The utterance is then transformed to MFCC features with $20ms$ frame-duration and $10ms$ frame-shift. Further more, we tuned the parameters of MFCC to yield better result. 40 filter banks and 19 cepstral coefficent are parameters found to perform best.

   We have extensively examined MFCC feature. We found that, the distribution of a person's MFCC feature is truly a good representation of its vocal characteristics. Here are some plots of the first three dimensions of a spearker's MFCC feature.

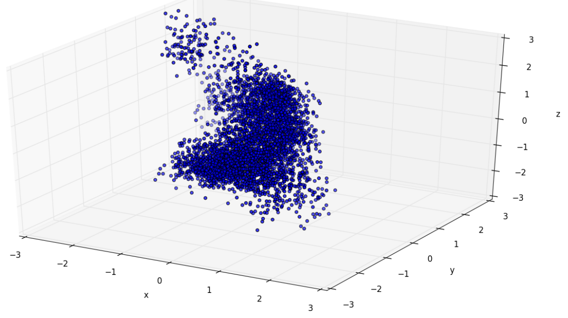

Figure 1: The first three dimension of a man's MFCC feature

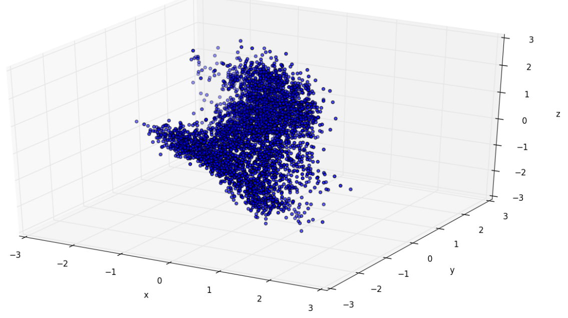

Figure 2: The first three dimension of another man's MFCC feature

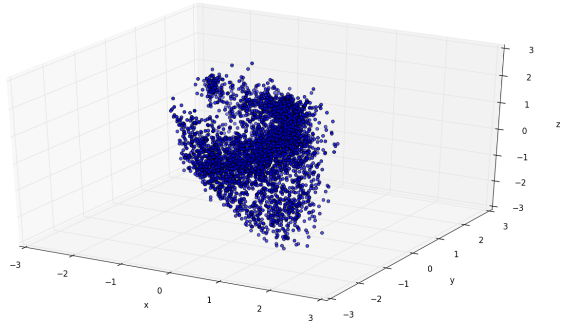

Figure 3: The first three dimension of a woman's MFCC feature

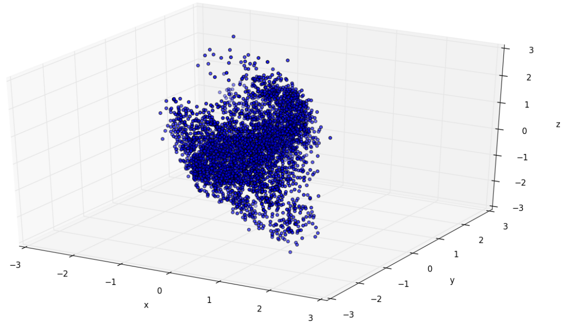

Figure 4: The first three dimension of another woman's MFCC feature

   It is clearly showed in the figure that, women's MFCC feature distribution tends to have nice ecliptical shapes in first three-dimension, as women's voice pitch is higher than man. And the two men's MFCC figure shows great difference on density over space. It shows high correspondency of the theoretical analysis.

3. **Gaussian Mixture Model**

   As the MFCC feature of a utterance of a user is a high-dimensional point set, which comprised a distribution, we model a user by modeling its MFCC feature distribution. The generative model we have primally adopted is Gaussian Mixture Model(GMM).

   In order to conduct a proof-of-concept experiments, we consequently conducted experiments using GMM from *scikit-learn*[7] and *PyPR*[9], but it turns out that these implementations are neither of efficiency, nor of decent accuracy. Therefore, we implemented our own GMM model with following improvements:

- As GMM is very sensitive to initialization, we use K-Means to initialize the configuration prior to GMM training.
- Further more, we use an improved algorithm of K-means$++$[1] named K-meansII[2], which outperforms both naive K-means and K-means++ in both efficiency and accuracy. For detailed algorithm description, please consult the orignal paper.
- All above algorithms (K-means, K-means$++$, K-meansII and GMM) are implemented utilizing multi-core CPUs in thread level. Experiments shows that our implementation scales nearly linearly with number of cores.
- In order to futher speed up the training process, we examined the time consuming part of GMM algorithm, which lies in extensive use of exponential function. Therefore, we rewrite exponential function using 5-order polynomial approximation using **SSE** instructions, which only introduced an numerical error less than $10^{-7}$. The running time of the training process cut by half when using our optimization.

The efficiency we achieved enables us to build a GUI system with more powerful features, if embeded with our GMM training techniques.

Further more, our implementation of GMM outperforms the GMM library aforementioned in terms of recognition accuracy.

For detailed efficiency and perfomance benchmark, see Section.3

4. **Continuous restricted Boltzmann Machine(CRBM)**

Restricted Boltzmann Machine(RBM) is generative stochastic neural network that can learn a probability distribution over its set of inputs[8]. But original version of RBM can only deal with binary inputs. [3] proposed and extension called Continuous restricted Boltzmann Machine, which can deal with real-valued inputs. RBM is a two-layer neural network, in which one layer called visible layer (the input), and another called hidden layer. The neurons in hidden layer controls the model complexity and the performance of the network. The Gibbs sampling of hidden layer can be seen as a representation of the original data. Therefore RBMs can be used as an auto feature-extractor.

Both RBM and CRBM can be trained using Contrastive Divergence learning.

RBM has a ability to, given an input(visible layer), reconstruct a visible layer that is similar to the input. This demonstrates the modeling essence of RBM. Fig.7 illustrate original MFCC data and the sampled output of reconstructed data from CRBM.

It turns out that, although CRBM has good modeling ability, but due to the inconsiderate of the time-varing property of sequential signal, although experiments we conducted using CRBM indicate considerable performance on speaker recognition task, using CRBM naively as a substitution of GMM did not yield better result till now.

5. **JFA**:

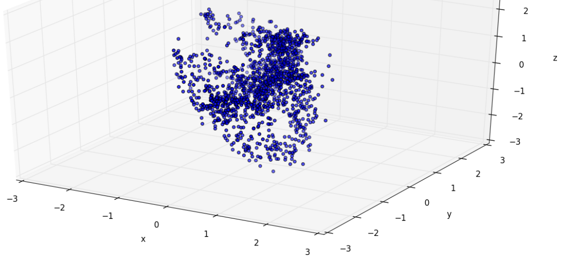

Figure 5: The first three dimension of a woman's MFCC feature

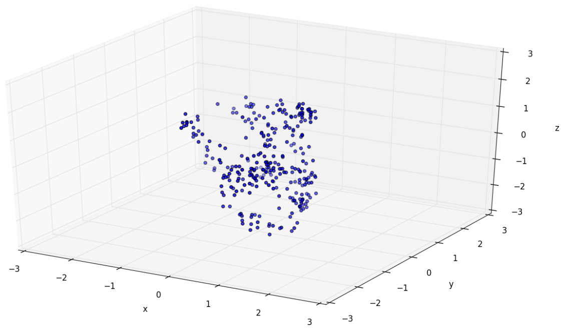

Figure 6: The first three dimension of the same woman's MFCC feature recontructed by a CRBM with 50-neuron hidden layer. We can see that, the density of these two distributions are alike

Figure 7

**Factor Analysis** is a typical method which behave very well in classification problems, due to its ability to account for different types of variability in training data. Within all the factor analysis methods, Joint Factor Analysis (JFA)[6, 4] was proved to outperform other method in the task of Speaker Recognition.

JFA models the user by "supervector" , i.e. a $C \times F$ dimension vector, where $C$ is the number of components in the Universal Background Model, trained by GMM on all the training data, and $F$ is the dimension of the acoustic feature vector. The supervector of an utterance is obtained by concatenate all the $C$ means vectors in the trained GMM model. The basic assumption of JFA on describing a supervector is:

$$\vec{M} = \vec{m} + vy + dz + ux,$$

where $m$ is a supervector usually selected to be the one trained from UBM, $v$ is a $CF \times R_s$ dimension matrix, $u$ is a $CF \times R_c$ dimension matrix, and $d$ is a diagonal matrix. This four variables are considered independent of all kinds of variabilities and remain constant after training, and $x, y, z$ are matrixes computed for each utterance sample. In this formulation, $m + vy + dz$ is commonly believed to account for the "Inter-Speaker Variability", and $ux$ accounts for the "Inter-Channel Variability". The parameter $R_s$ and $R_c$, also referred to as "Speaker Rank" and "Channel Rank", are two emprical constant selected as first. The training of JFA is to calculate the best $u, v, d$ to fit all the training data.

After our investigation, we found that the original algorithm [4] for training JFA model is of too much complication and hard to implement. Therefore, we use the simpler algorithm presented in [5] to train the JFA model. However, from the result, JFA does not seem to outperform our enhanced MFCC and GMM algorithms (but do outperform our old algorithms). It is suspected that the training of a JFA model needs more data than we have provided, since JFA needs data from various source to account for different types of variabilities. Therefore, we might need to add extra data on the training of JFA, but keep the same data scale in the stage of enrollment, to get a better result.

It is also worth mentioning that the training of JFA will take much longer time than our old method, since the estimation process of $u, v, d$ does not converge quickly. As a result, it might not be practical to add JFA approach to our GUI system. But we will still test further on the performance of it, compared to other methods.

## 2.2 Recognition

Recognition procedure follows steps below:

1. Record a short utterance of the speaker (typically 5 seconds)
2. Preprocess the utterance using first two steps described in Section.2.1, e.g, VAD and MFCC feature extraction.
3. Compute the 'score' of the MFCC featuresextraced for each model of persons enrolled, and adopt person corresponding the model which gives highest score to be the recognition result.

   A typical form of score is log-likelihood (or enery in RBM case).

## 3 Result

We have tested our method on a corpus provided by teacher Xu. For detailed description of the corpus, please see former report.

All the tests are conducted serval times (depending on computation cost, vary from 5 to 20) with random selected training and testing speakers. The average over these tests are considered as the final result.

### 3.1 Efficiency Test of our GMM

We have extensively examined the efficiency of our implementation of GMM compared to scikit-learn version. Test is conducted using real MFCC data with 13 dimensions. We consider the scenario when training a UBM with 256 mixtures. We examine the time used for ten iteration. For comparable results, we diabled the K-means initialization process of both scikit-learn GMM implementation and ours. Time used for ten iterations under different data size and concurrency is recorded.

From Fig.8, we can immediately infer that our method is much-much more efficient than the widely used version of GMM provided by scikit-learn when the data size grows sufficiently large.

We shall analyze in two aspect:

- No concurrency
  - When the number of MFCC features is below 6000, which is a typical number of features generated by 60 seconds utterances (1ms frame shift), our method is slightly slower; but this is trivial since 1 minute utterance is too small.

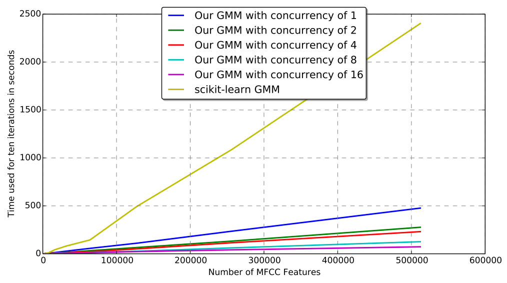

Figure 8: Comparison on efficiency

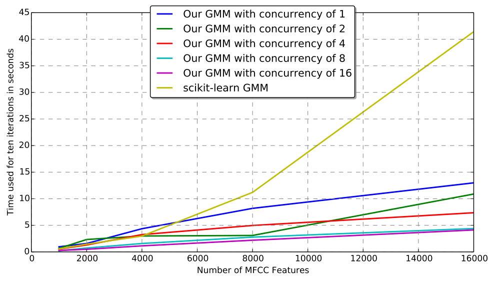

Figure 9: Comparison on efficiency when number of MFCC features is small

  - When the number of MFCC features grows sufficiently large, our method shows great improvement. When training 512,000 features, our method is 5 times faster than comparing method.

- With concurrency

  Our method shows considerable concurrency scalability that the running time is approximately lineary to the number of cores using.

  When using 8-cores, our method is 19 **times** faster than comparing method.

## 3.2 Effect Of Number Of Mixtures

We examined our GMM compared to GMM from scikit-learn. Test is conducted on 30-speaker corpus, 30 seconds training utterance and 100 random sampled 5 seconds test utterance for each speaker.

As Fig. 10 illustrates, when number of mixtures is small, our GMM outperforms scikit-learn version by 10%, which indicates our GMM models the distribution more accurately. The maximum accuracy happens when the number of mixtures is around 32, reaching 0.965. As the number of mixtures increases, the decrease in accuracy

## 4 Effect On Number Of Speakers

An apparent trade-off in speaker recognition task is the number of speakers enrolled and the accuracy of recognizing a person. We've conducted experiments examining the effect of number of speakers enrolled on the performance of the system.

The configurations of the test is as followed:

- Number of mixtures is set to 32, the optimal number we found previously
- GMM from scikit-learn, compared to our GMM.
- 30s training utterance and 5s test utterance
- 100 sampled test utterance for each user

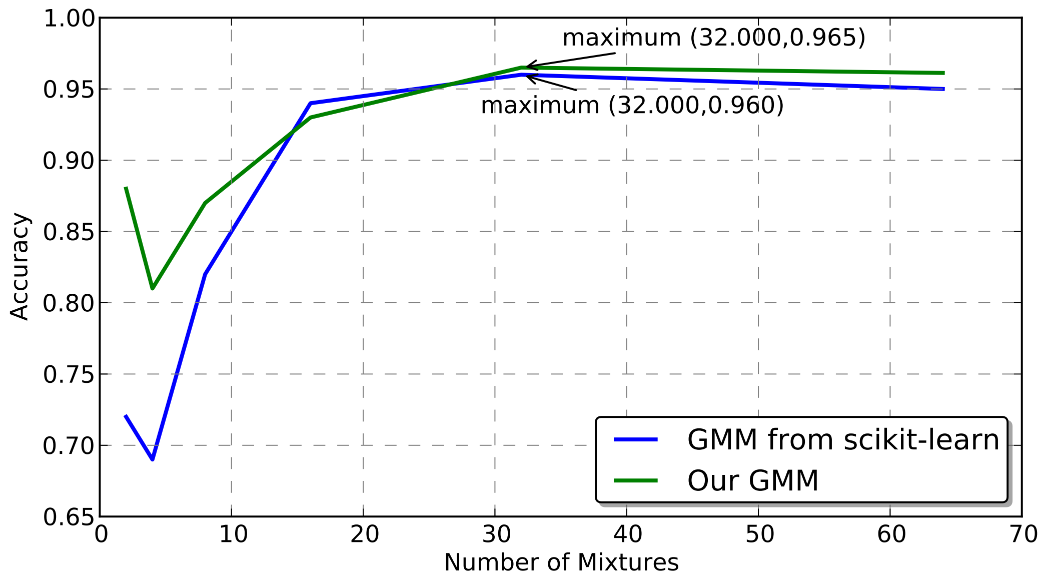

Figure 10: Accuracy curve on different number of mixtures

Scrunitizing Fig.11 we would see that, our GMM performs better than scikit GMM in general. When number of speakers is small, due to the the random selection, the variance of the tests is significantly high, as we can see from the curve fluctuants. When number of speakers increases, it is clear that the accuracy of our GMM is above scikit version. As the more speaker, the more difficult the recognition task will be, this result suggests that our optimization on GMM takes effect.

## 5 GUI

This section describes the design of Graphic User Interface to accomodate our system.

### 5.1 Functional Requirements

The GUI consists of following functions:

- **Enrollment of speaker**

  A new user can be dynamically add to this system. when enrollment starts, the user is prompted to input his identifications, e.g, name, age, sex, a photo would be better, and so on.

  Then he is required to record a piece of utterance. An article will be prompted to the user in case he/she is introrse to say something, he/she can read the article.

  There are two ways to enroll a user:

  - **Enroll by speaking** A progress indicator of enrollment process will be present to user. When enough utterance is collected, the user will be given message about the success of

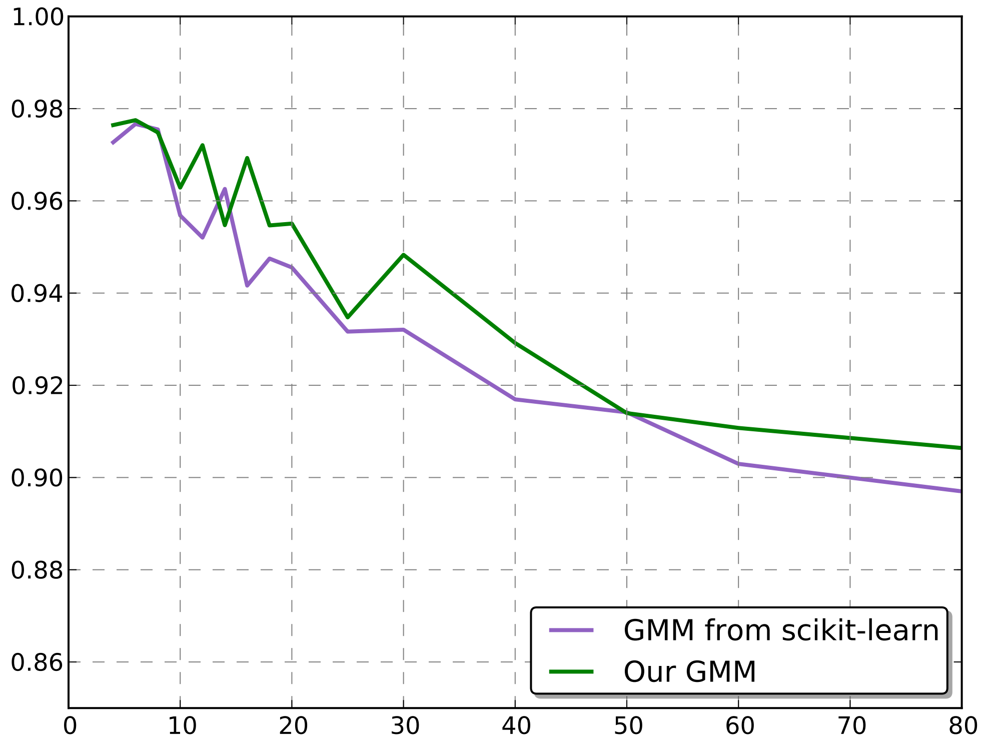

Figure 11: Accuracy curve on different number of speakers enrolled

enrollment.

  - **Enroll by pre-recorded voice** User can upload a pre-recorded voice of a speaker. The system accepts the voice given and the enrollment of a speaker is done.

- **Recognition of a user**
  A enrolled user present and record a piece of utterance, the system tells who the person is.

- **Conversation Recognition mode**
  When the system turn into Conversation Recognition mode, it will continuously collect voice data, and determine who is speaking right now. If a photo is given in previous enrollment process, current speaker's photo will show up in screen; otherwise the name will be shown.

- **Verification mode**
  A user first claim its identity, then the system required the user to record a piece of utterance. The system will then display whether the speaker is the speaker that claimed or an impostor.

  The availability of this function is still under our investigation.

## 5.2 Non-Functional Requirements

- **Enrollment Efficiency**

  The enrollment procedure involves training GMM or RBM model, which is a time consuming process. A continuous flow of user enrollment should not be interupted.

- **Platform-Independency**

  As we program using Python, our program should run on platform that supports Python.

- **Recognition Accuracy**

  The performance of the system should be good enough to make this system could be carried out in practical use.

## References

[1] David Arthur and Sergei Vassilvitskii. "k-means++: The advantages of careful seeding". In: *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics. 2007, pp. 1027–1035.

[2] Bahman Bahmani et al. "Scalable k-means++". In: *Proceedings of the VLDB Endowment* 5.7 (2012), pp. 622–633.

[3] Hsin Chen and Alan F Murray. "Continuous restricted Boltzmann machine with an implementable training algorithm". In: *Vision, Image and Signal Processing, IEE Proceedings-*. Vol. 150. 3. IET. 2003, pp. 153–158.

[4] Patrick Kenny. "Joint factor analysis of speaker and session variability: Theory and algorithms". In: *CRIM, Montreal,(Report) CRIM-06/08-13* (2005).

[5] Patrick Kenny et al. "A study of interspeaker variability in speaker verification". In: *Audio, Speech, and Language Processing, IEEE Transactions on* 16.5 (2008), pp. 980–988.

[6] Patrick Kenny et al. "Joint factor analysis versus eigenchannels in speaker recognition". In: *Audio, Speech, and Language Processing, IEEE Transactions on* 15.4 (2007), pp. 1435–1447.

[7] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[8] *Restricted Boltzmann machine - Wikipedia, the free encyclopedia*. URL: http://en.wikipedia.org/wiki/Restricted_Boltzmann_machine.

[9] *Welcome to PyPR' s documentation! – PyPR v0.1rc3 documentation*. URL: http://pypr.sourceforge.net/.