Digital Signal Processing: Speaker Recognition
Final Report
Concise Version

Xinyu Zhou, Yuxin Wu, and Tiezheng Li
Tsinghua University

1 Dataset

The dataset provided by teacher comprised of 102 speaker, in which 60 are females and the rest are males, with three different speaking style: Spontaneous, Reading and Whisper. A statistic is as follows:

	Spontaneous	Reading	Whisper
Average Duration	202s	205s	221s
Female Average Duration	205s	202s	217s
Male Average Duration	200s	203s	223s

2 Approach

In this section we will present our aproach to tackle the speaker recognition problem. There're two steps in a complete speaker recognition system: enrollment and recognition.

2.1 Erollment

An utterance of a user is collected during enrollment procedure. Further processing of the utterance follows following steps:

1. **VAD**
   Signals must be first filtered to rule out the silence part, otherwise the training might be seriously biased. Therefore **Voice Activity Detection** must be first performed. An observation found is that, the corpus provided is nearly noise-free. Therefore we use a simple energy-based approach to remove the silence part, by simply remove the frames that the average energy is below 0.01 times the average energy of the whole utterance.

   This energy-based method is found to work well on database, but not on GUI. We use LTSD(Long-Term Spectral Divergence) [9] algorithm on GUI, as well as noise reduction technique from SOX[12] to gain better result.

2. **MFCC and LPC Features**
   We extract **Mel-frequency cepstral coefficients** and **Linear Predictive Coding**

features using following parameter are found to be optimal, according to our experiments in Section.3:

- Common parameters:
  - Frame size: 32ms
  - Frame shift: 16ms
  - Preemphasis coefficient: 0.95
- MFCC parameters:
  - number of cepstral coefficient: 15
  - number of filter banks: 55
  - maximal frequency of the filter bank: 6000
- LPC Parameters:
  - number of coefficient: 23

and then concatenate the two feature vectors of the same frame forming a larger feature vector of $15 + 23 = 38$ dimension.

3. **GMM**

   We use **Gaussian Mixture Model** modeling a speaker. Some improvements made:

   - Performance:
     We investigate the effect of initialization of GMM during training. We implemented GMM with K-meansII[2], which is an improved version of K-means++[1] to initialize the mean vector of GMM. Results shows improvements compared to GMM provided by **scikit-learn**[8].
   - Efficiency:
     - We provide a parallel version of GMM, especially optimized to train large Universal Background Model(UBM).
     - We further improve efficiency by utilizing SSE instruction in computing exponential function using polynomial approximation. This can speed up the training procedure by a factor of two.

4. **UBM**

   As we are providing continuous speech close-set diarization function in GUI, we adopt **Universal Background Model** as imposter model, and use likelihood ratio test to make reject decisions.[11]

   When using conversation mode in GUI (will be present later), GMM model of each user is adapted from a pre-trained UBM using method described in [11].

5. **CRBM**

   Restricted Boltzmann Machine is generative stochastic two-layer neural network (see Figure.**??**) that can learn a probability distribution over its set of binary inputs[10]. **Continuous restricted Boltzmann Machine(CRBM)**[3] extends its ability to real-valued inputs. RBM has a ability to, given an input(visible layer), reconstruct a visible layer that is similar to the input. Figure. 3 illustrate original MFCC data and the sampled output of reconstructed data from CRBM.

   Previous working using neural network largely focused on speech recognition, such as [4] [**mohamed20111deep**], only a few ([]) on classification task.

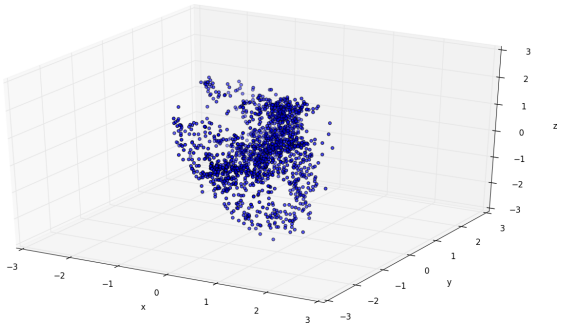


Figure 1: The first three dimension of a woman's MFCC feature

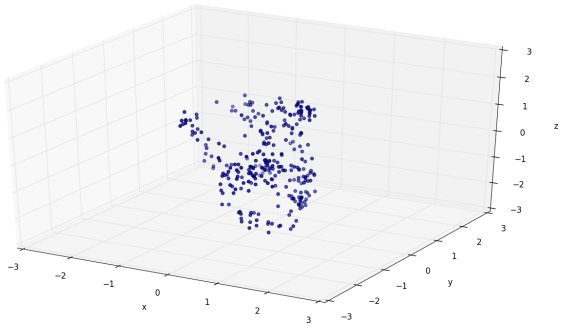


Figure 2: The first three dimension of the same woman's MFCC feature recontructed by a CRBM with 50-neuron hidden layer. We can see that, the density of these two distributions are alike

Figure 3

   Here use CRBM as a substitution of GMM, rather than an feature extractor. We train a CRBM per speaker, and estimate reconstruction error without sampling (which is stable). The person corresponds to the lowest reconstruction error CRBM is adopted as recognition result.

6. **JFA**:

   Joint Factor Analysis [7, 5] was generally considered to perform better than other method in the task of Speaker Recognition, by modeling different types of variabilities in the training data, including session variability and speaker variability.

   Therefore, we use a simpler algorithm presented in [6] to train the JFA model. However, the result shows that JFA does not seem to outperform GMM. We suspected that the training of a JFA model needs more data than we provided, since JFA needs data from various source to account for different types of variabilities. To get a higher accuracy in JFA, We might need to add extra data for training.

2.2 Recognition

Recognition procedure follows steps below:

1. Record a short utterance of the speaker (typically less than 5 seconds)

2. Preprocess the utterance using first two steps described in Section.2.1, e.g, VAD, MFCC and LPC feature extraction.

3. Compute the 'score' of each person 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 energy in RBM case).

3 Performance

We have tested our approaches under various parameters, based on a corpus provided by teacher Xu. For detailed description of the corpus, please see former report.

All the tests in this section have been conducted serval times (depending on computation cost, vary from 10 to 30) with random selected training and testing speakers. The average over these tests are considered as confidential 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, 20ms frame length. 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.

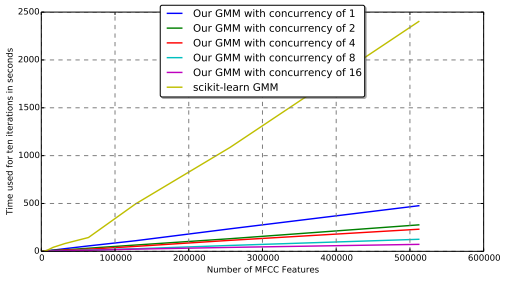


Figure 4: Comparison on efficiency

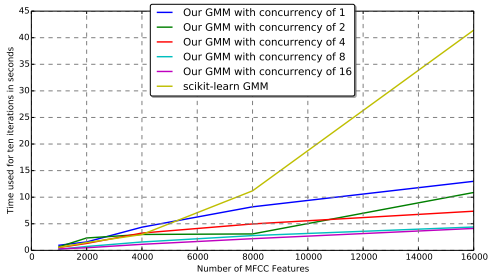


Figure 5: Comparison on efficiency when number of MFCC features is small

From Figure.4, 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 grows sufficiently large, our method shows great improvement. When training 512,000 features, our method is 5 times faster than comparing method.

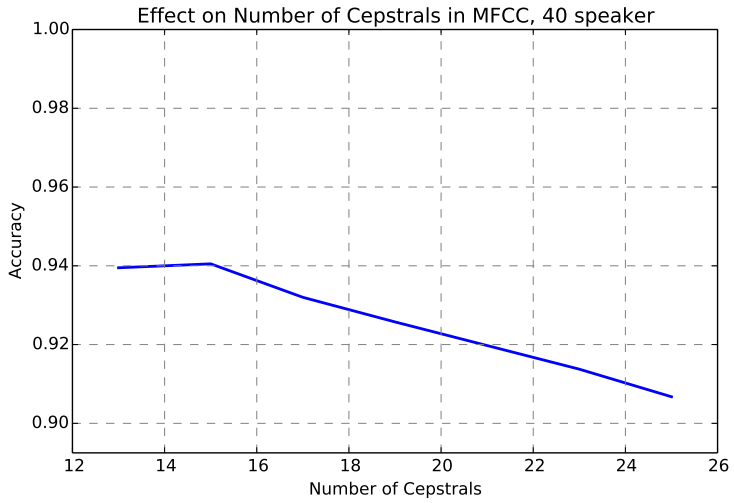
- With concurrency
  Our method shows considerable concurrency scalability that the running time is approximately lineary to the number of cores using.

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

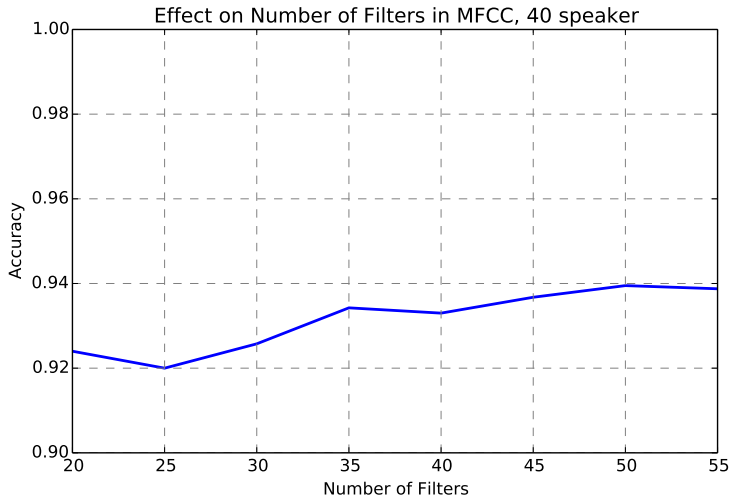
3.2 Change in MFCC Parameters

The following tests reveal the effect of MFCC parameters on the final accuracy. The tests were all performed on "Style-Reading" corpus with 40 speakers, each with 20 seconds for enrollment and 5 seconds for recognition.

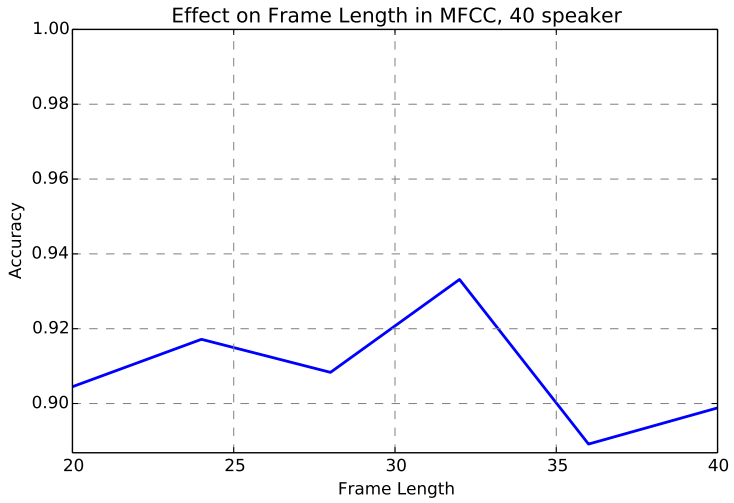
1. Different Number of Cepstrums



2. Different Number of Filterbanks



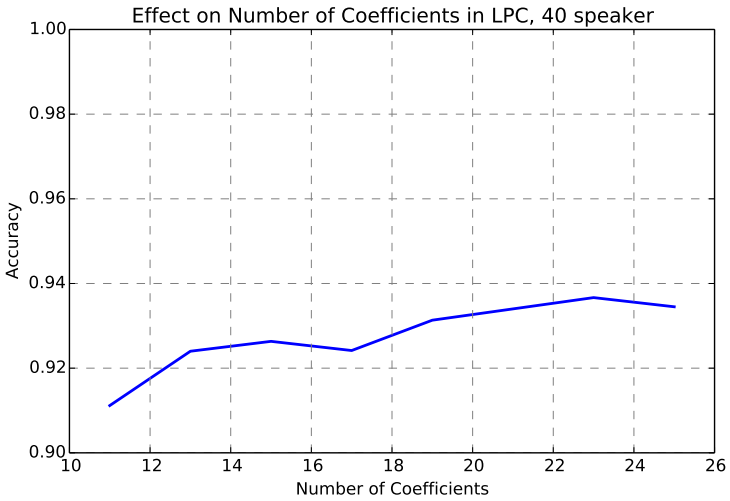
3. Different Size of Frame



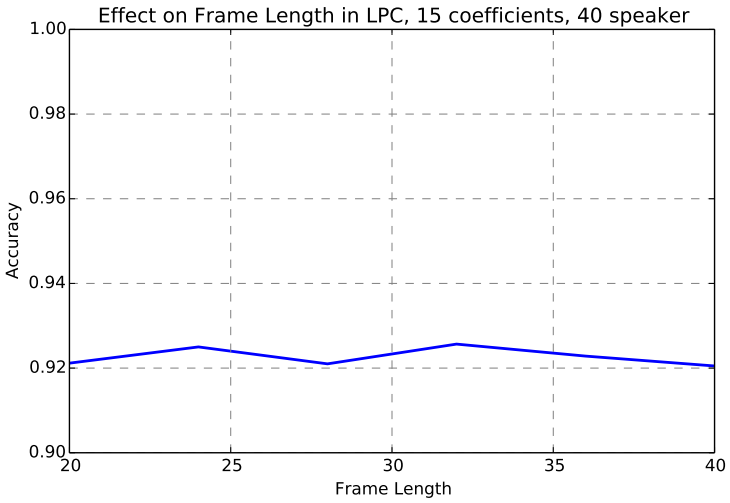
3.3 Change in LPC Parameters

The following tests display the effect of LPC parameters on the final accuracy. The tests were performed on "Style-Reading" with 40 speakers, each with 20 seconds for enrollment and 5 seconds for recognition.

1. Different Number of Coefficient

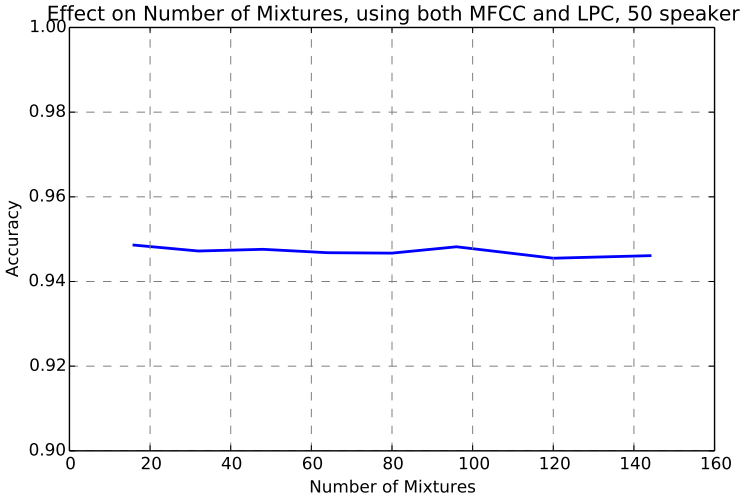


2. Different Size of Frame



3.4 Change in GMM Components

We experimented on the effect of GMM Components. We found that the number of components have slight effect on the accuracy, but a GMM with higher order might take significantly longer time to train. Therefore we still use GMM with 32 components in our system.



3.5 Different GMM Algorithms

We compare our implementation of GMM to GMM in scikits-learn.

The configurations of the test is as followed:

- Only MFCC: frame size is $20ms$, 19 cepstrums, 40 filterbanks
- 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

From this graph we could see that, our GMM performs better than GMM from scikit-learn in general. Due to the random selection of test data, the variance of the test can be high when the number of speakers is small, as is also the case in the next experiment. But this result still shows that our optimization on GMM takes effect.

3.6 Accuracy Curve on Different Number of Speakers

An apparent trade-off in speaker recognition task is the number of speakers enrolled and the accuracy on recognization. Also, the duration of signal for enrollment and test can have significant effect on the accuracy. We've conducted test using well-tuned parameters for feature extraction as well as GMM, on dataset with various number of people and with various test duration.

The configurations of this experiment is as followed:

- Database: "Style-Reading"
- MFCC: frame size is $32ms$, 19 cepstrums, 55 filterbanks

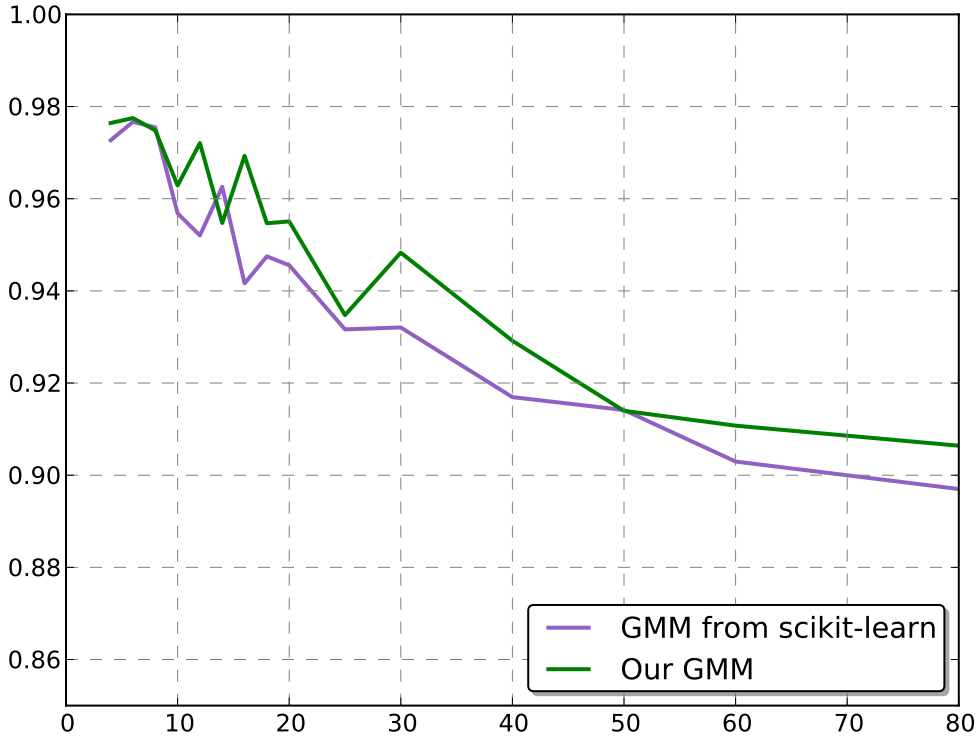
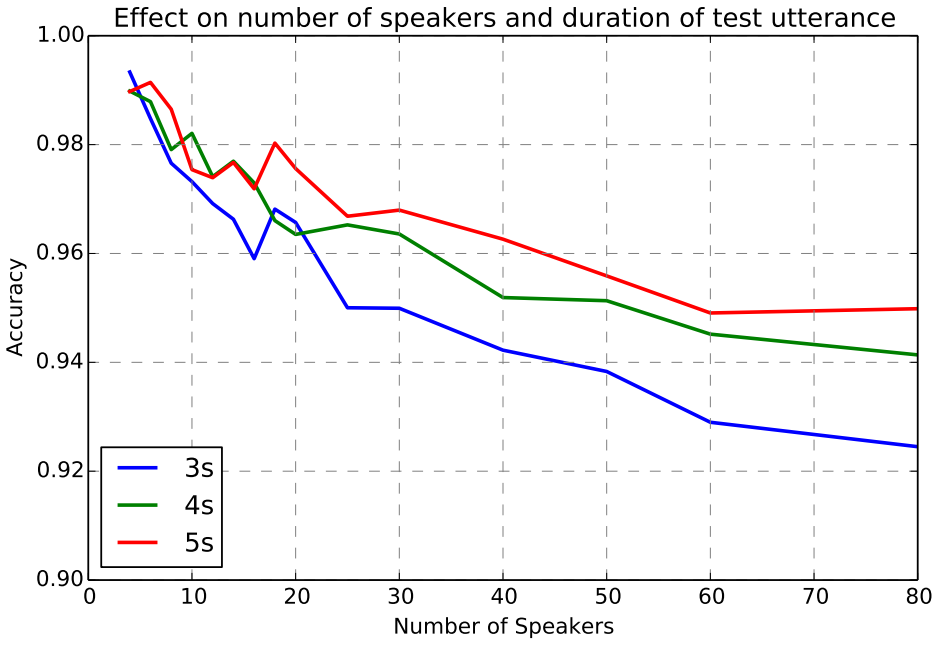


Figure 6: Accuracy curve for two GMM

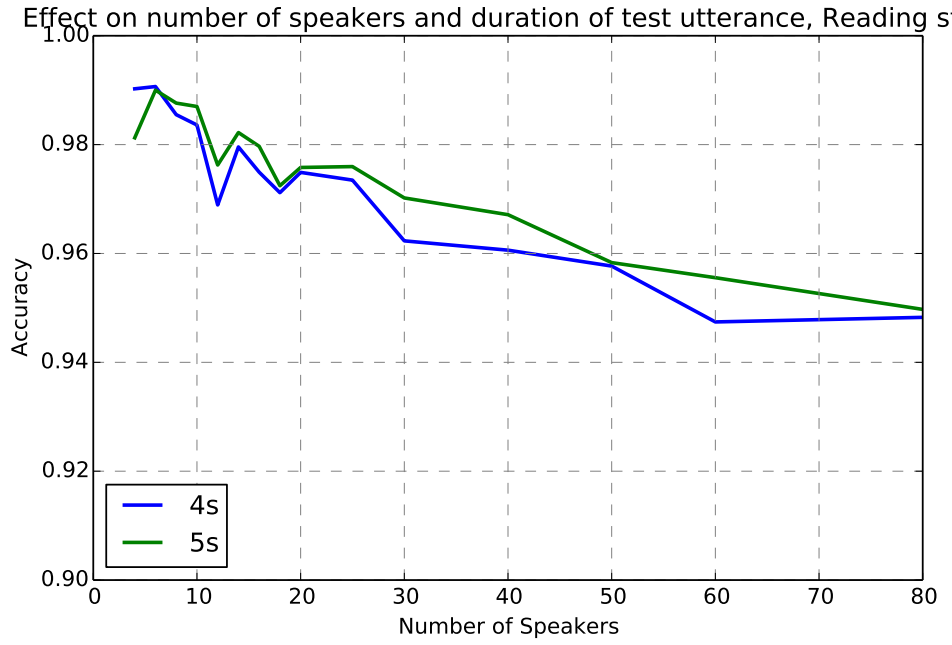
- LPC: frame size is $32ms$, 15 coefficients
- GMM from scikit-learn, number of mixtures is 32
- 20s utterance for enrollment
- 50 sampled test utterance for each user

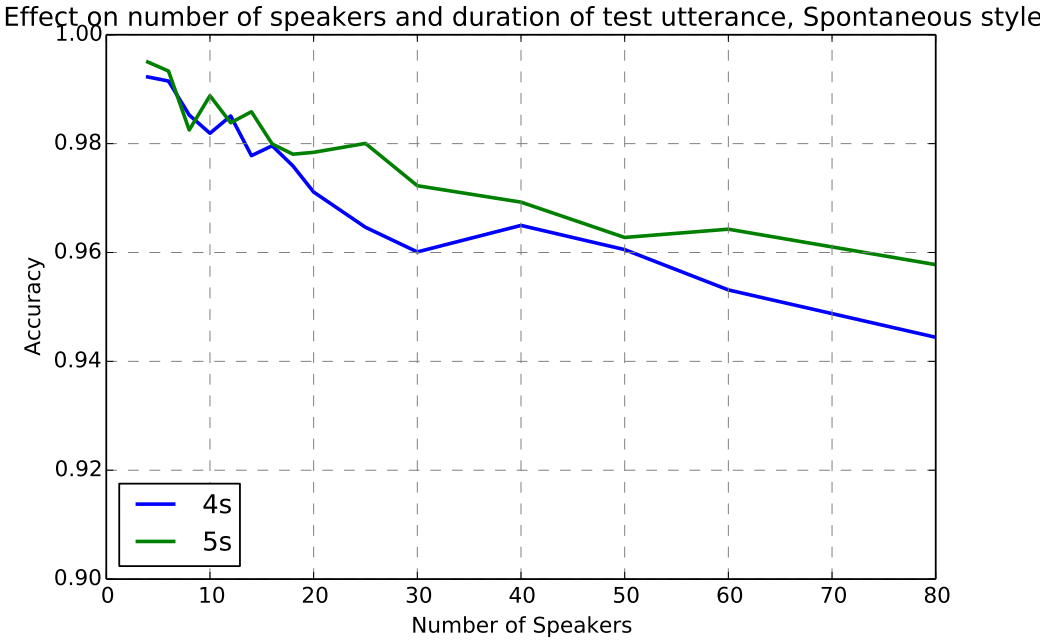


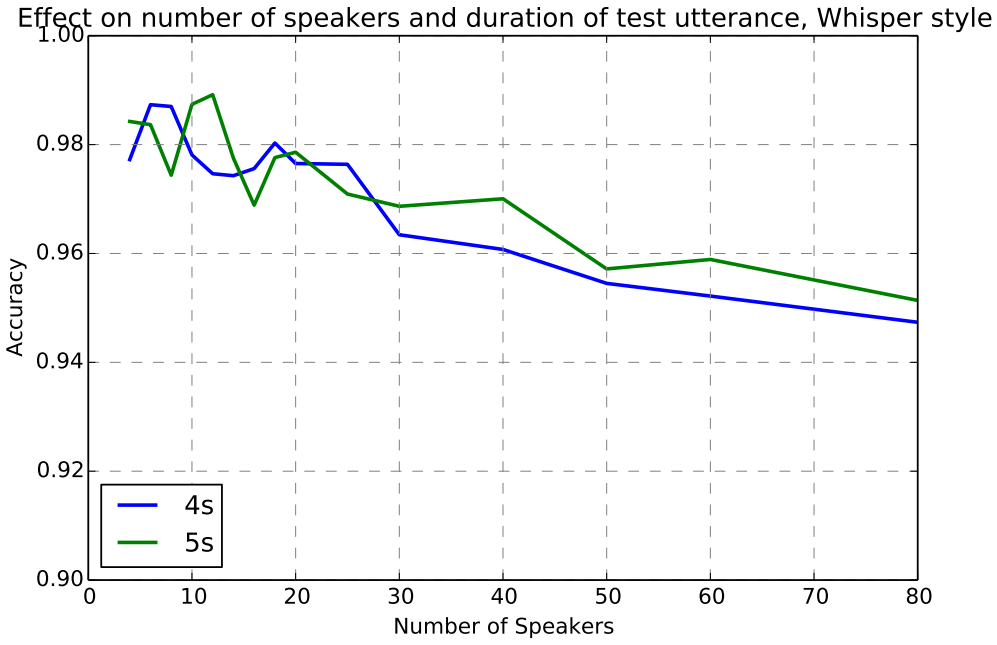
We also conducted experiments on different style of corpus. The configurations of this experiment is as followed:

- MFCC: frame size is $32ms$, 15 cepstrums, 55 filterbanks
- LPC: frame size is $32ms$, 23 coefficients
- GMM from scikit-learn, number of mixtures is 32
- 20s utterance for enrollment
- 50 sampled test utterance for each user

The result is shown below. Note that each point in the graph is an average value of 20 independent test with random sampled speakers .


Effect on number of speakers and duration of test utterance, Reading style
Accuracy
Number of Speakers
4s
5s
1.00
0.98
0.96
0.94
0.92
0.90
0
10
20
30
40
50
60
70
80


Effect on number of speakers and duration of test utterance, Spontaneous style
Accuracy
Number of Speakers
4s
5s
1.00
0.98
0.96
0.94
0.92
0.90
0
10
20
30
40
50
60
70
80



3.7 CRBM Performance Test

We also tested RBM using following configuration:

- MFCC: frame size is $32ms$, 15 cepstrums, 55 filterbanks
- LPC: frame size is $32ms$, 23 coefficients
- CRBM with 32 hidden units.
- 50 sampled test utterance for each user
- 5s test utterance

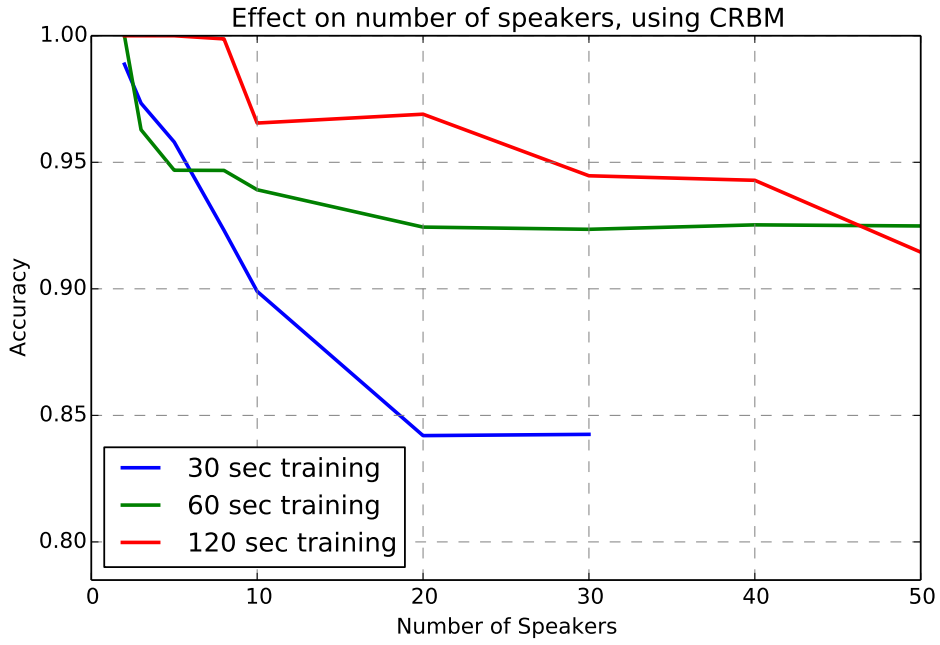


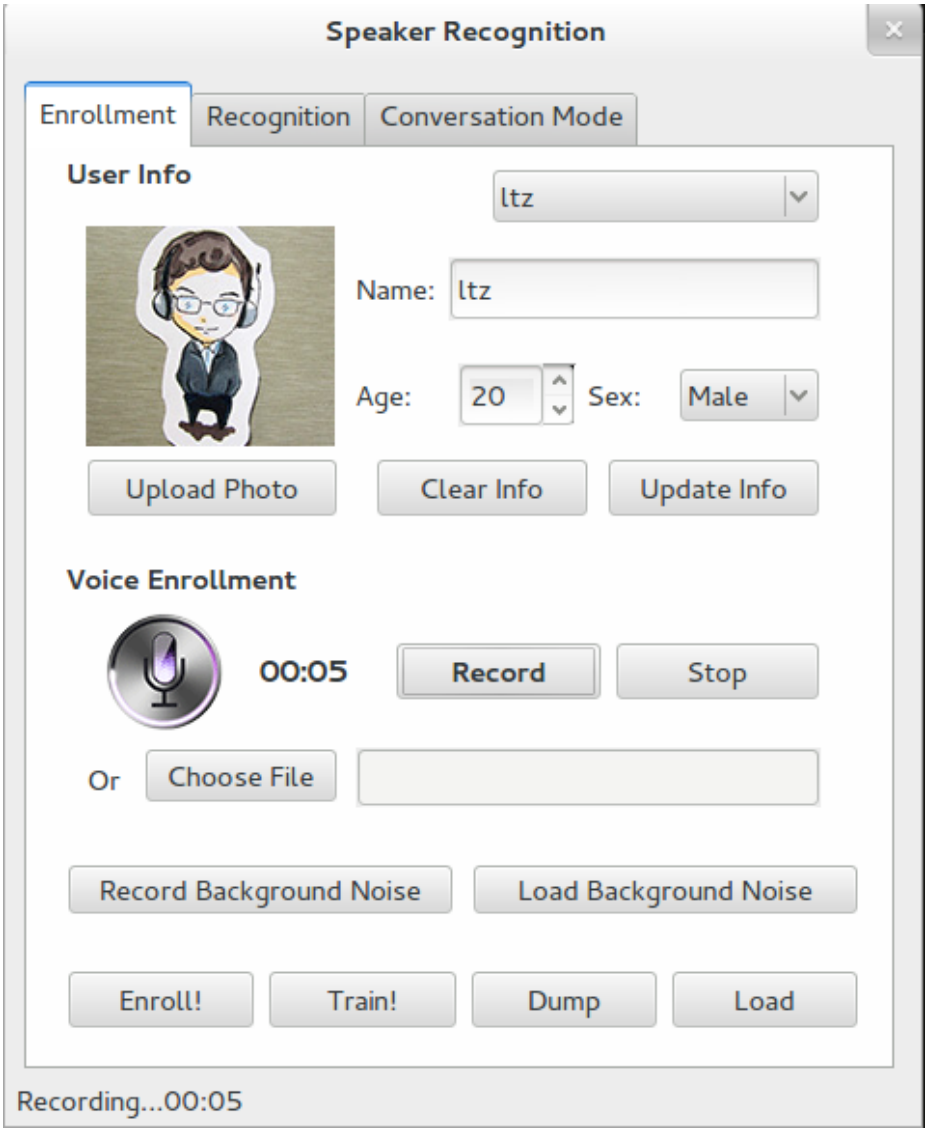
Figure 7

Result shown in Figure.**??** indicates that, although CRBM have generic modeling ability, applying it on signal features does not fit our expectation. To achieve similar results, the training utterance should be twice as large as GMM used. Further investigation on using RBM to process signal features need to be conducted.

4 GUI

The GUI contains following tabs:

- **Enrollment**



A new user may start his or her first step by clicking the tab Enrollment. New users could provide personal information such as name, sex, and age. then upload personal avatar to build up their own data. Experienced users can choose from the userlist and update their infomation.

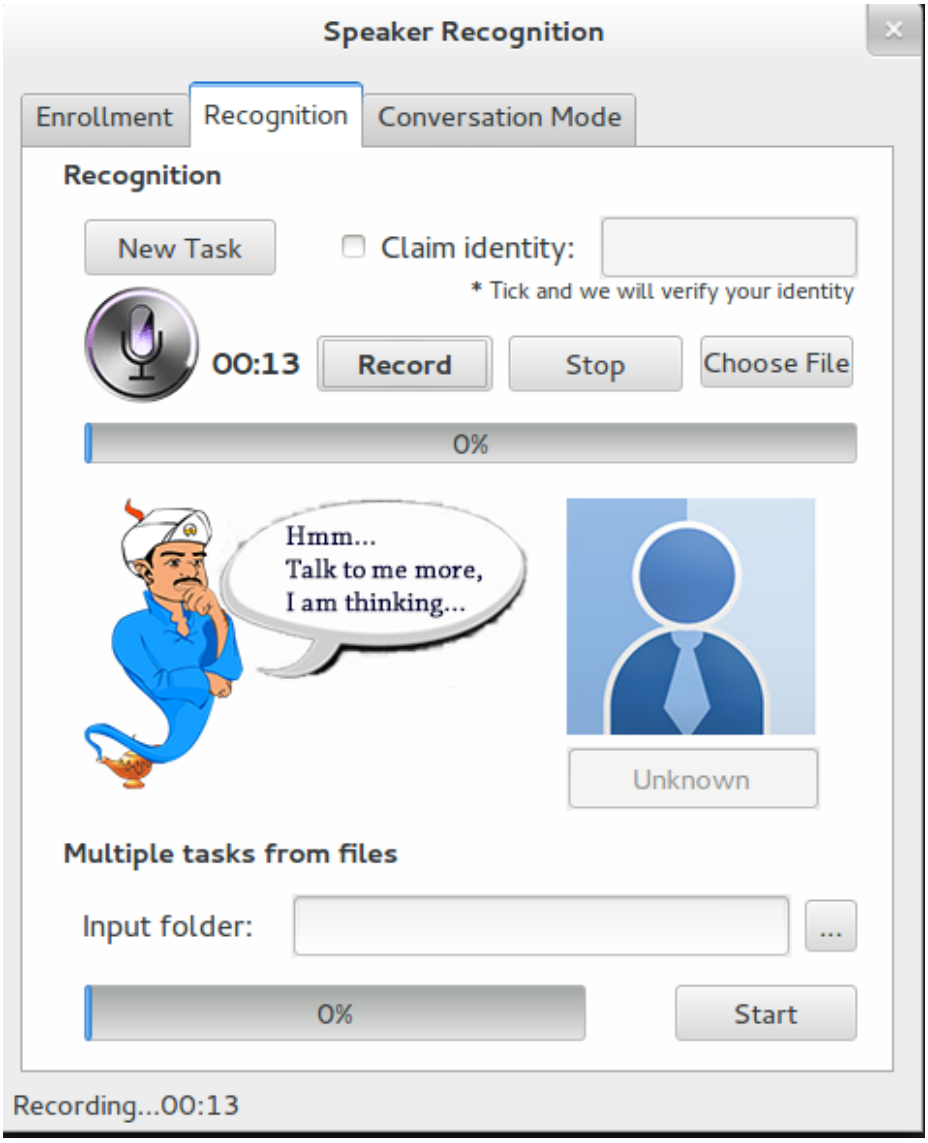
Next the user needs to provide a piece of utterance for the enrollment and training process.

There are two ways to enroll a user:

- **Enroll by Recording** Click Record and start talking while click Stop to stop and save.There is no limit of the content of the utterance, whileit is highly recommended that the user speaks long enough to provide sufficient message for the enrollment.
- **Enroll from Wav Files** User can upload a pre-recorded voice of a speaker. (*.wav recommended) The systemaccepts the voice given and the enrollment of a speaker is done.

The user can train, dump or load his/her voice features after enrollment.

- **Recognition of a user**



A enrolled user present or record a piece of utterance, the system tells who the person is and show user's avatar. Recognition of multiple pre-recorded files can be done as well.

- **Conversation Recognition Mode**

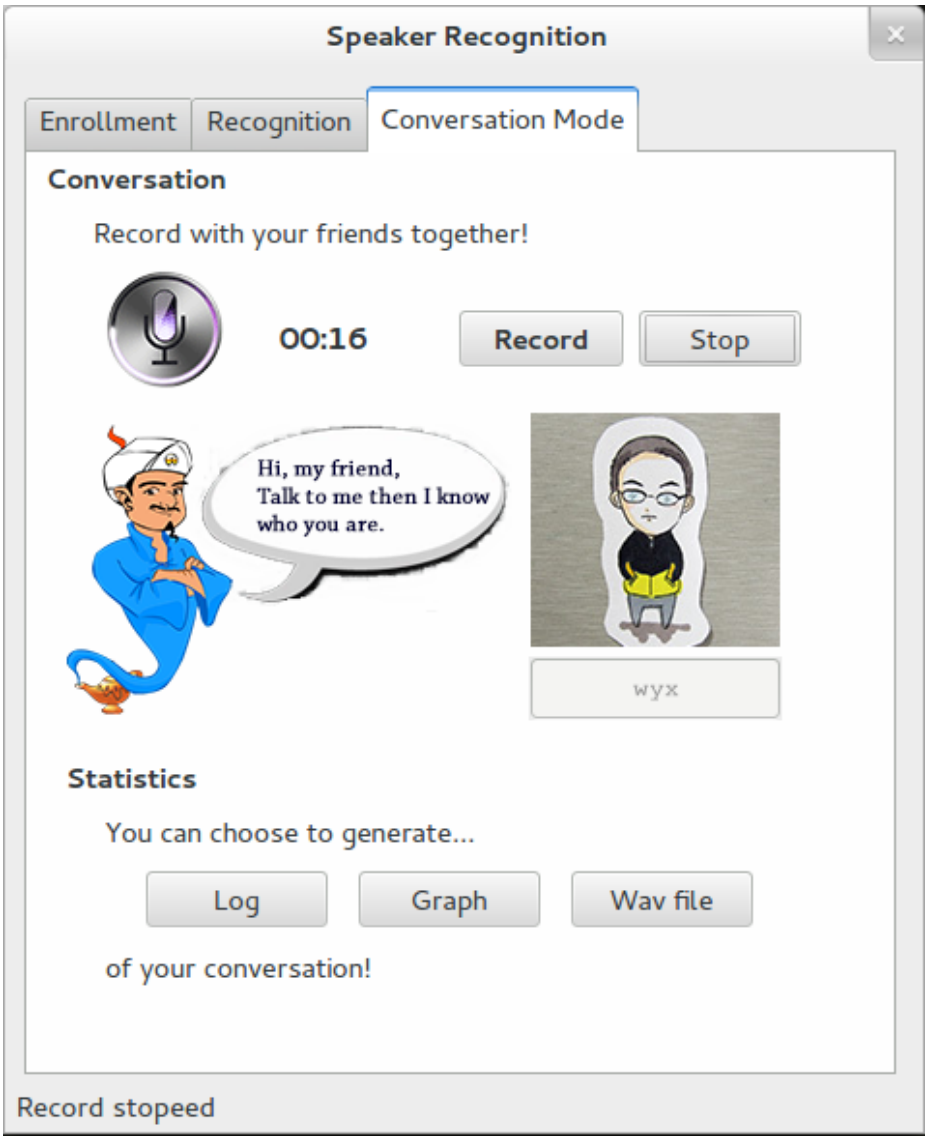
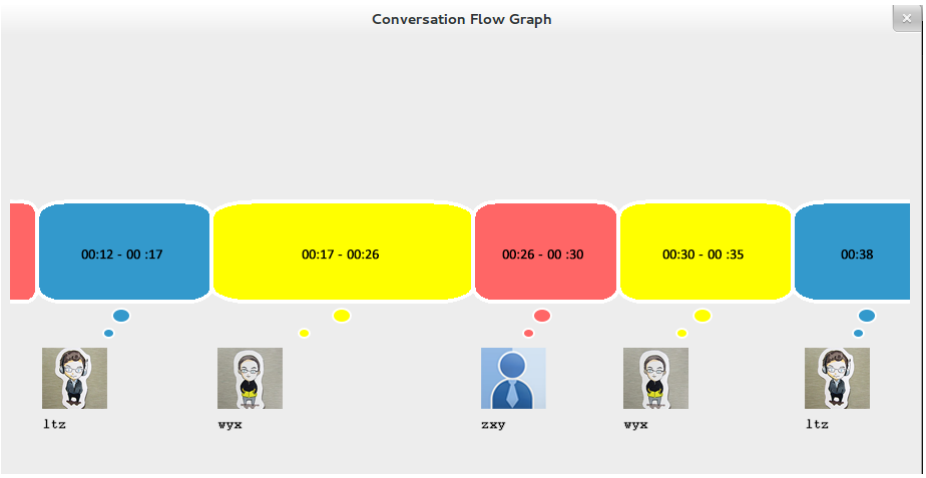


Figure 8

In Conversation Recognition mode, multiple users can have conversations together near the microphone. Same recording procedure as above. The system will continuously collect voice data, and determine who is speaking right now. Current speaker's anvatar will show up in screen; otherwise the name will be shown. The conversation audio can be downloaded and saved. There are some ways to visualize the speaker-distribution in the conversation.

- **Conversation log** A detailed log, including start time, stop time, current speaker of each period is generated.
- **Conversation flow graph**



A timeline of the conversation will be shown by a number of talking-clouds joining together, with start time, stop time and users' avatars labeled. Different users are presented with different colors.The timeline will flow to the left dynamically just as time elapses. The visualization of the conversation is done in this way. This functionality is still under development.

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] George E Dahl et al. "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition". In: *Audio, Speech, and Language Processing, IEEE Transactions on* 20.1 (2012), pp. 30–42.

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

[6] 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.

[7] 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.

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

[9] Javier Ramırez et al. "Efficient voice activity detection algorithms using long-term speech information". In: *Speech communication* 42.3 (2004), pp. 271–287.

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

[11] Douglas A Reynolds, Thomas F Quatieri, and Robert B Dunn. "Speaker verification using adapted Gaussian mixture models". In: *Digital signal processing* 10.1 (2000), pp. 19–41.

[12] *SoX - Sound eXchange*. URL: http://sox.sourceforge.net/.