speaker recognition using gmm-ubm

semester project presentation
OBJECTIVES OF THE PROJECT

• study the GMM-UBM speaker recognition system

• implement this system with matlab

• document the code and how it interfaces with the rest of toolkit

• perform validation with different data (channel, content, etc)
Steps in speaker recognition using GMM

- feature extraction (transform the original signal in frequency domain)
- training (models the original voice using gaussian parameters)
- testing (calculate the “statistical distance” with each of the original models)
training phase
test phase
LIKELIHOOD RATIO DETECTOR

• given a segment of speech Y

• implicit assumption : Y contains speech from only one speaker

• given a hypothesized speaker claimed identity S
Decision rule of likelihood ratio

• we take the hypothesis such as
• $H_0 : Y$ is from the hypothesized speaker $S$
• $H_1 : Y$ is not from the hypothesized speaker $S$
• we take $H_0$ if $P(Y \mid H_0)/P(Y \mid H_1) > \theta$
log likelihood

- many advantages to take log likelihood
- a division become a subtraction
- a multiplication become an addition
- the equation before will be represent like
- $\log(P(Y | H0)) - \log(P(Y | H1)) > \log \theta$
GMM-UBM

- text independent speaker recognition
- read sound wave files
- melcepst extracts features
- training models with GMM
- training models with UBM
- decision, recognition based on likelihood ratio
Gaussian mixture model (GMM)

- text independent verification
- not so expensive computation like HMM
- big phoneme or vocabulary database no needed
- HMM doesn’t shown advantage over GMM
- GMM training
GMM and EM algorithm
estimation step

compute the probability $P(q(k)|x(n), \theta)$ for each data point $x(n)$ to belong the mixture $q(k)$

$$P(q_k | x_n, \theta) = \frac{P(q_k | \theta) \cdot p(x_n | q_k, \theta)}{p(x_n | \theta)} = \frac{P(q_k | \theta) \cdot p(x_n | u_k, \sum_k)}{\sum_j P(q_j | \theta) \cdot p(x_n | u_k, \sum_k)}$$

in the algorithm:

$c(k) = P(q_k | \theta)$,

$lBM(n, k) = \log p(x_n | q_k, \theta)$,

$lB(k) = \log p(x_n | \theta)$,

$gamm(n, k) = P(q_k | x_n, \theta)$
maximization step

update the mean

$$u_k^{new} = \frac{\sum_{n=1}^{T} x_n P(q_k \mid x_n, \theta)}{\sum_{n=1}^{T} P(q_k \mid x_n, \theta)}$$

update the sigma

$$\sum_k^{new} = \frac{\sum_{n=1}^{T} P(q_k \mid x_n, \theta)(x_n - u_k)(x_n - u_k)^T}{\sum_{n=1}^{T} P(q_k \mid x_n, \theta)}$$

update weight

$$P(q_k^{new} \mid \theta^{(new)}) = \frac{1}{T} \sum_{n=1}^{T} P(q_k \mid x_n, \theta)$$
• it’s a iterative algorithm

• input is feature \( x \) and \( M \) for number of mixture

• the probabilities follow gaussian distribution

• we evaluate the values of \( u, \sigma \) and weight
UBM speaker verification procedure

- training UBM model (computation of $\theta$)
- adapt each training feature to this model
- compute the log-likelihood ($p(x|\theta_{ubm})$)
- decision is based on the max likelihood
UBM training

• The UBM is a large GMM trained to represent the speaker-independent distribution of features

• The idea of using UBM is in order to capture the general characteristics of a population

• and then adapting it to the individual speaker

• Training uses EM algorithm
UBM adaptation

for mixture $i$, we compute

$$\Pr(i \mid x_t) = \frac{w_i p_i(x_t)}{\sum_{j=1}^{M} w_j p_j(x_t)}$$

we then use it to compute weight, mean and variance

$$n_i = \sum_{t=1}^{T} \Pr(i \mid x_t)x_t$$

$$E_i(x) = \frac{1}{n_i} \sum_{t=1}^{T} \Pr(i \mid x_t)x_t$$

$$E_i(x^2) = \frac{1}{n_i} \sum_{t=1}^{T} \Pr(i \mid x_t)x_t^2$$
the new coefficients are

\[ w_i^{\text{new}} = [\alpha_i^w n_i / T + (1 - \alpha_i^{\text{new}}) w_i] \gamma \]

\[ u_i^{\text{new}} = \alpha_i^m E(x)_i + (1 - \alpha_i^m) u_i \]

\[ \sigma_i^{2\text{new}} = \alpha_i^v E(x^2)_i + (1 - \alpha_i^v)(\sigma_i^2 + u_i) - u_i^{\text{new}} \]

In the GMM-UBM system we use a single adaptation coefficient for all parameters \((\alpha = n(i)/(n(i) + r))\) with a relevance factor of \(r = 16\)

all above formula are took directly from Reynolds notes
the algorithm is valid

• the following experiments shows this algorithm is valid
the database

- IPSC03
- 73 speakers
- 3 wave files each speakers
TESTS

• UBM model composed by 14 people
• Comparisons with GMM
• 33 people detection
• we didn’t use any of these 33 people to train UBM parameters
different training size

- the gmm training and ubm adaption
- we split files on different size: 1/5, 1/10, 1/15, 1/25, 1/35
results

adapted UBM error rate

<table>
<thead>
<tr>
<th>Split rate</th>
<th>1</th>
<th>1/5</th>
<th>1/10</th>
<th>1/15</th>
<th>1/25</th>
<th>1/35</th>
</tr>
</thead>
<tbody>
<tr>
<td>FA</td>
<td>0.0066</td>
<td>0.0095</td>
<td>0.0142</td>
<td>0.0170</td>
<td>0.0303</td>
<td>0.0369</td>
</tr>
<tr>
<td>FR</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0303</td>
<td>0.0303</td>
<td>0.0303</td>
</tr>
<tr>
<td>EER</td>
<td>0.0033</td>
<td>0.0047</td>
<td>0.0071</td>
<td>0.0237</td>
<td>0.0303</td>
<td>0.0336</td>
</tr>
</tbody>
</table>

GMM classic error rate

<table>
<thead>
<tr>
<th>Split rate</th>
<th>1</th>
<th>1/5</th>
<th>1/10</th>
<th>1/15</th>
<th>1/25</th>
<th>1/35</th>
</tr>
</thead>
<tbody>
<tr>
<td>FA</td>
<td>0.0085</td>
<td>0.0189</td>
<td>0.0170</td>
<td>0.0379</td>
<td>0.0909</td>
<td>0.1847</td>
</tr>
<tr>
<td>FR</td>
<td>0</td>
<td>0.0303</td>
<td>0.0303</td>
<td>0.0303</td>
<td>0.0909</td>
<td>0.1818</td>
</tr>
<tr>
<td>EER</td>
<td>0.0043</td>
<td>0.0246</td>
<td>0.0237</td>
<td>0.0341</td>
<td>0.0909</td>
<td>0.1832</td>
</tr>
</tbody>
</table>
some DET curves

- UBM is blue color
- GMM is red color

6s training

5s training
data length and EER

ubm
gmm
some images

ubm 160s

gmm 160s
results analysis

• the UBM seems better in our tests

• maybe the single channel, the population, and the speech content make the UBM more preferment in these tests
conclusion

• UBM adaptation is much more fast than GMM training

• the quality of UBM is much better than GMM when we have small training segments

• the detection computation time of UBM may longer than GMM