Speaker Identification Using i-vectors

Umang Patel
The Fu Foundation School of Engineering and Computer Science
Computer Science Department
Columbia University
New York , New York-10027
Email: ujp2001@columbia.edu

***Abstract*—This paper presents an overview of a speaker identification system using GMM , which uses i-vector to perform speaker identification . The advantage of using i-vector is that it is computationally efficient . This papers intends to compare the results obtained by normal GMM process and GMM process with i-vectors .**

I. Motivation

The Joint Factor Analysis (JFA)approach has become the state of the art in the field of speaker identification during the last three years. This modeling proposes powerful tools for address- ing the problem of speaker and channel variability in Gaussian Mixture Models (GMM) framework. In this paper we compare a two channel compensation technique namely LDA (Linear Discriminant Analysis) and HLDA (Heteroscedastic Linear Discriminant Analysis).

II. Problem Statement

Here we intend to do a text independent speaker identification . Given a set of training speakers (N training speaker of the form (x_n, s_n) , x_n is the training sample , s_n is the speaker id) and a test speaker of the form (x_t, s_t), the task would be to find if x_t is from s_t.

III. Scope of the problem

Here we intend to perform , text independent closed set speaker identification system on NIST 2006 speaker dataset .

IV. Approach

We attack the problem in two parts . The first being Front-End Processing where given input signals we extract features using MFCC's. The second being Back-End where the input will be the MFCC features and the output is trained model.

A. *Front-End Processing*

[1] Here we intend to do the spectral analysis and obtain MFCC features from audio signals using the Direct Method (Moving average) .The method is as follows:

- Framing - We intend to sample at 20ms frame size with 10ms overlap across frame
- Windowing - We intend to use Hamming window on each frame to reduce the effect of frequency folding and aliasing.

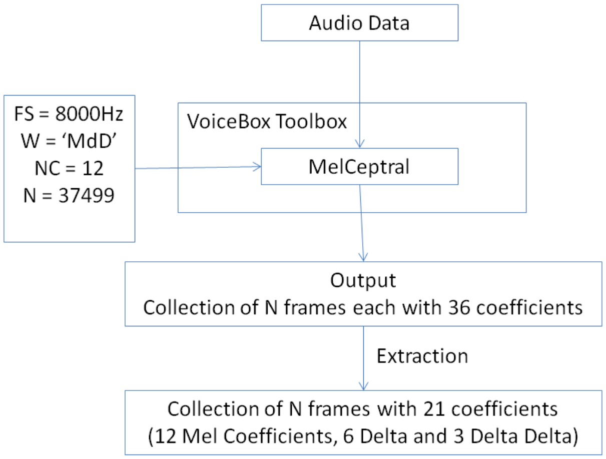
Audio Data
FS = 8000Hz
W = 'MdD'
NC = 12
N = 37499
VoiceBox Toolbox
MelCeptral
Output
Collection of N frames each with 36 coefficients
Extraction
Collection of N frames with 21 coefficients
(12 Mel Coefficients, 6 Delta and 3 Delta Delta)

Fig. 1. Front End Processing

- DFT(Discrete Fourier Transform) - We use DFT to do spectral estimation of the frames formed.
- Frequency and Magnitude Warping - We use the Mel scale to do the frequency warping . Magnitude warping is done to compensate for the perceived magnitude.
- Computation - Here we intend to do Mel Frequency Cepstral Coefficient Computation (MFCC) and Mel Frequency Cepstral Dynamics - Delta and Delta Delta Cepstra.

The flowchart is shown in above figure .
Tool Used :- Here we use **VOICEBOX**, a Matlab toolbox for speech processing to extract features from audio data.[2]

B. *Back-End Processing*

Once we have the MFCC Features for each speaker , we have to train the models for each speaker.

Speaker Verification: [3][4][5]

Here a Gaussian mixture is a weighted sum of C component densities given by equation :

$$p(x|\lambda) = \sum_{i=1}^{C} p_i b_i(x)$$

where x is the input feature vector ,$b_i(x)$ is the density component and p_i is the mixture weights. Here each $b_i(x)$ is a Gaussian Model given by equation

$$b_i(x) = \frac{1}{(2\pi)^{F/2}|\sum_i|^{\frac{1}{2}}} exp(\frac{1}{2}(x_i-\mu_i)^T\Sigma^{-1}(x_i-\mu_i))$$

where μ_i is the mean, Σ_i covariance matrix for i^{th} speaker. Each speaker in the GMM is represented by its own model λ_i .

$$\lambda_i = (p_i, \mu_i, \Sigma_i), i = 1, ..., C$$

The above model is trained using speaker vectors by the EM (Expectation Maximization) algorithm . In EM algorithm , in the E step we estimate p_i assuming means μ_i and Σ_i , and in the M step we estimate μ_i and Σ_i assuming p_i.

For the speaker verification task , the objective is to find out if x_t is from s_t.

Writing in terms of hypothesis test:
H_0 : x_t is from s_t
H_1 : x_t is NOT from s_t
Test: $\frac{p(x_t|H_0)}{p(x_t|H_1)}$, if $\geq \theta$ then accept H_0 , else accept H_1.

Here , after the EM algorithm is applied to the training voice samples , in order to model the speaker and within-speaker variability in low dimension we use i-vectors . The low dimensional subspace of the GMM super vectors space is called total variability space which represents both speaker and channel variabilities . The vectors in low dimension are called i-vectors.

We assume C to be the number of components of the GMM and F be the dimension of the feature vector.

We model the GMM super vectors as follows :
$\mathbf{s_i{=}m{+}Tw_i}$ [6]

where **m** - is the speaker and channel independent super vector . It is obtained by taking sample mean of all the features of the given speakers and stacking the sample means of all the speaker . Dimension of this vector is CF * 1.[7][8]

$\mathbf{s_i}$ - is the i^{th} speaker and channel dependent super vector . For each speaker it is obtained by stacking the mean vector of all speakers obtained from the regular GMM training and the expectation maximization process . Dimension of this vector is CF * 1.[7][8]

T - is the Total variability matrix . Its dimension is CF * d and it is a low rank matrix obtained by training the speaker .[7]

$\mathbf{w_i}$ - i^{th} speaker i-vector, it is obtained by $T^{\dagger}(s_i - m_i)$. Its dimensions are d*1 . $T^{\dagger}$ is the psudo inverse of T.[7][8]

So using the above equation we get w_{target}. w_{test} is obtained in the same way.

After getting w_{target} and w_{test} by above mentioned process , to compensate for different input medium (Channel Compensation), we intend to perform LDA (Linear Discriminant Analysis) and Bayesian HLDA (Bayesian Heteroscedastic Linear Discriminant Analysis).

Training of T (Total Variability Matrix) [6]

Here indirectly to find T , we are doing EM algorithm.The EM is done in 2 steps.

1) For each speaker s , we use the current estimate of T and $\sum$(covariance from UBM model) to find the supervector s_i which maximizes the likelihood of speakers training data $X(s)$.

$$y(s) = \underset{w_i}{a}rgmaxP(X(s)|s_{i,0} + Tw_i, \sum)$$

where s_i is assumed to come from Normal distribution having mean $s_{i,o}$.

2) Update T by maximizing

$$\prod_s P(X(s)|s_{i,0} + Tw_i, \sum)$$

It turns out that after solving the equations of EM algorithm, the algorithm has close form solution and is presented in the algorithm below .

Algorithm:(Repeat steps 1-6 for approximately 20 iterations)
1) Accumulate the 0^{th},1^{st} and 2^{nd} order statics for each speaker (s) and Gaussian mixture component (c)

$$(0^{th} orderstatics)N_c(s) = \sum_{t\in s}\gamma_t(c)$$

$$(1^{st} orderstatics)F_c(s) = \sum_{t\in s}\gamma_t(c)Y_t$$

$$(2^{nd} orderstatics)S_c(s) = diag(\sum_{t\in s}\gamma_t(c)Y_tY_t^*)$$

$\gamma_t(c)$-posterior of Gaussian component c for observation t of speaker s
$*$-denotes the Hermitian transpose of vector or matrix .
diag()-denotes taking only the diagonal elements of the

matrix.
Y_t-are acoustic feature vectors.

The dimensions of $N_c(s)$−1*1 , $F_c(s)$−F*1 ,$S_c(s)$−F*F.

2) Center the 1^{st} and 2^{nd} order statics.

$$\overline{F_c}(s) = F_c(s) - N_c(s)m_c$$

$$\overline{S_c}(s) = S_c(s) - diag(F_c(S)m_c^* + m_cF_c(s)^* - N_c(s)m_cm_c^*)$$

m_c-UBM mean of mixture component c.

The dimensions of $\overline{F}_c(s)$−F*1 ,$\overline{S}_c(s)$−F*F.

3)Expand the statistics into matrices

$$NN(s) = \begin{bmatrix} N_1(s) * I & & \\ & \ddots & \\ & & N_c(s) * I \end{bmatrix}$$

$$SS(s) = \begin{bmatrix} \overline{S}_1(s) & & \\ & \ddots & \\ & & \overline{S}_c(s) \end{bmatrix}$$

$$FF(s) = \begin{bmatrix} \overline{F}_1(s) \\ \vdots \\ \overline{F}_c(s) \end{bmatrix}$$

The dimensions of NN(s)-CF*CF, SS(s)-CF*CF, FF(s)-CF*1

4) Initial estimate of the speaker factors y

$$l_V(s) = I + (T^*)\sum^{-1} NN(s)T$$

$$\overline{y}(s) = l_V^{-1}(s)(T^*)\sum^{-1} FF(s)$$

T- randomly initialize T for first iteration.
$\sum$- is the covariance of the UBM model.
The dimensions of $l_V(s)$−d*d , $\overline{y}(s)$−d*1 , T- CF*d where d is the dimension of the i-vectors we want.

5) Accumulating some additional statics across speakers

$$A_c = \sum_s N_c(s) l_v^{-1}(s)$$

$$C = \sum_s FF(s)(l_V^{-1} T^* \sum^{-1} FF(s))^*$$

The dimensions of A_c−d*d , C-CF*d.
6) Compute T

$$T = \begin{bmatrix} T_1 \\ \vdots \\ T_c \end{bmatrix} = \begin{bmatrix} A_1^{-1}C_1 \\ \vdots \\ A_c^{-1}C_c \end{bmatrix} C = \begin{bmatrix} C_1 \\ \vdots \\ C_c \end{bmatrix}$$

The dimensions of C_i-F*d

Explanation of All the steps [9]

Step 1:
Here we calculate the Baum-Welch statistics in the usual way.
Step 2:
Here we center the statics .
Step 3:
Here we convert it into matrices.
Step 4:
For each speaker s , $l_V(s)$ is the hidden variable.$\overline{y}(s)$ is the posterior expectation over all such $y(s)$(i.e. y(s) obtained from each iteration).
Step 5:
Calculating the additional statistics across all speakers.

Once the w_i are obtained , i-vectors are subjects to LDA(Linear Discriminant Analysis) and Bayesian HLDA (Bayesian Heteroscedastic Linear Discriminant Analysis).

LDA (Linear Discriminant Analysis) [10]
Here the LDA is done to minimize the intra class variance and maximize the inter class variance . This is done as below, Let there be k classes .

$w_{i,j} = j^{th}$ i-vector in the i^{th} class.

N_i= number of data points in i^{th} class.

N= total number of data points in all classes.

$\overline{w_i} = \frac{1}{N_i}\sum_{j=1}^{N_i} w_{i,j}$ (Class Sample Mean)

$\overline{w} = \frac{1}{N}\sum_{i=1}^{k}\sum_{j=1}^{N_i} w_{i,j}$ (Complete Sample Mean)

$S_b = \sum_{i=1}^{k} N_i(\overline{w_i} - \overline{w})(\overline{w_i} - \overline{w})^T$(Between Class Matrix)

$S_w = \sum_{i=1}^{k}\sum_{j=1}^{N_i}(w_{i,j} - \overline{w_i})(w_{i,j} - \overline{w_i})^T$(Within Class Matrix)

We can get transformed i-vector(w) by below function:

Taking A=top eigen vectors of $(S_w^{-1}S_b)$
$\implies w_{transformed}$ = $(A).w_{input}$
'.'- stands for multiplication.

Here depending upon the number of eigen vectors selected we can construct the A matrix . If needed we can even reduce the dimensions of i-vector further by taking appropriate top eigen vectors in construction of A.

The draw back of LDA is it assumes that all the classes have same covariance , so it neglects the difference in covariance among different classes . To solve that problem we go for Bayesian HLDA (Bayesian Heteroscedastic Linear Discriminant Analysis) which takes into consideration that different class have different covariance matrix.

Bayesian HLDA (Bayesian Heteroscedastic Linear Discriminant Analysis) [11]

Here instead of having a single S_w for all classes , each class will be having its own S_w , renaming it as :-

W_i - is the within class covariance matrix for the i^{th} class.

$x_{\{j\}}$ - observation belonging to j^{th} class.

μ_i - mean of all observations belonging to the i^{th} class.

x_i - i^{th} observation.

N- total number of observations

$$W_i = \frac{1}{N_i}\sum_{j=1}^{N_i}(x_{\{j\}} - \mu_i)(x_{\{j\}} - \mu_i)^T$$

$$T^{'} = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)(x_i - \mu)^T \text{ (Total Covariance)}$$

Here,

$$A = \begin{bmatrix} A_p \\ A_{n-p} \end{bmatrix}$$

ie within A matrix we can control the p dimension in which we need to retain the information .

To solve this , as the covariance matrix is different for each class , the direct maximum of likelihood is not possible , so we have to use and iterative algorithm as below.

Taking $(\bar{A})$ to be K dimensional identity matrix as each dimension of feature space is covered.

Algorithm:

1) Start with some random initialization of A(A=$(\bar{A})$) .
2) While not converged
3) For each row r=1...n in A
4)

$$G_r = \begin{cases} \sum_{i=1}^{k} \frac{N_i}{N} \frac{W_i}{a_r^T W_i a_r}, & \text{if } r \leq p \\ \frac{T^{'}}{a_r^T T a_r}, & \text{otherwise} \end{cases}$$

5) $\alpha_r = (a_r^T c_r)^{-1}$
6) $\beta = det(Gr)^{\frac{1}{K}}$ where K is the dimension of Gr.

$$a_r = \begin{cases} (G_r + \beta(I))^{-1}(\alpha_r c_r + \beta I \overline{a_r}), & \text{if } r \leq p \\ (G_r)^{-1}(\alpha_r c_r), & \text{otherwise} \end{cases}$$

Once A is obtained , we can get $w_{transformed}$
$\Longrightarrow$ $w_{transformed}$ = $(A).w_{input}$.

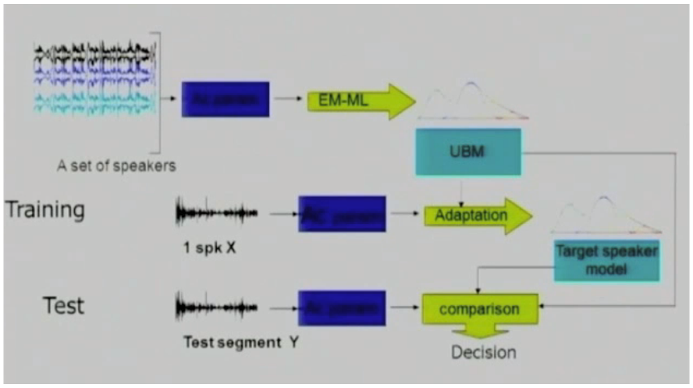

Fig. 2. Overall Process for GMM-UBM Model[13]

. - stands for multiplication.

So likewise we get all the $w_{transformed}$, we can then use CDS as below.
Here we use Cosine Distance Scoring to compare the similarity among the testing i-vectors and enrollment i-vectors.

Renaming $w_{transformed}$ to $w_{enrollment}$.
Cosine Distance Scoring

After getting $w_{enrollment}$ and w_{test} from applying LDA and Bayesian HLDA, we intend to use Cosine Distance Scoring (CDS) for identifying the speaker using below equation .

$$score(w_{enrollment}, w_{test}) = \frac{w_{enrollment}.w_{test}}{\|w_{enrollment}\|.\|w_{test}\|}.$$

Likewise for each w_{test} CDS is calculated against all $w_{enrollment}$ and the $w_{enrollment}$ which gives the highest CDS , its speaker id is taken.[12]
'.' - stands for multiplication.

All the above mentioned steps are shown in a Figure 2 and Figure 3.

V. DATASET USED

NIST 2006 Speaker Recognition Evaluation Plan Dataset is used. In total, there are 600 files of audio data.The dataset contains data corresponding to 100 males and 100 females, also for each of the 200 speakers there are 3 channels of audio data provided. Each file is 5 minutes long with single-speaker conversation. The audio data is mostly English (but also included audio data in other languages like Arabic, Bengali, Chinese, Farsi, Hindi, Korean, Russian, Spanish, Thai, Urdu, Yue Chinese). The speech files are stored as 8-bit u-law speech signals in separate WAV files.

VI. EVALUATION

The evaluation proceeds in the following way,

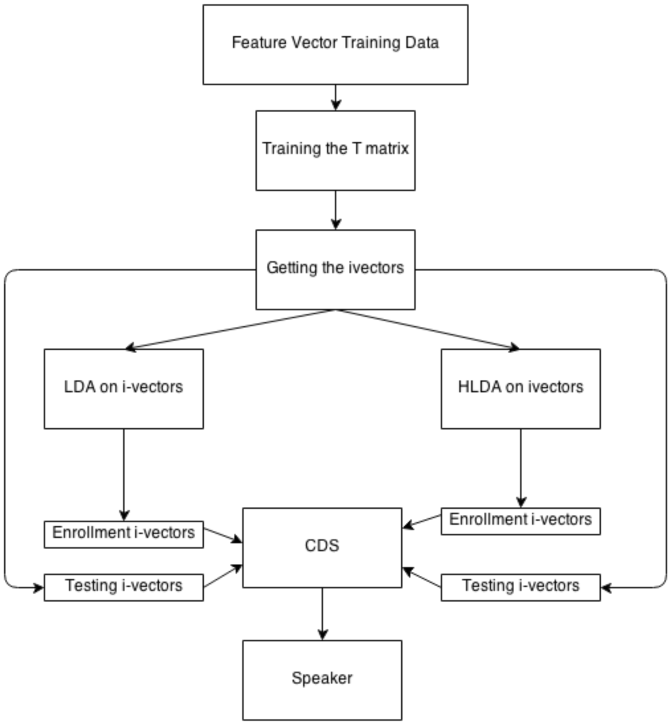

Fig. 3. Speaker Identification System Block Diagram

- BS - Baseline Score - It is the accuracy($\frac{correctmatch}{total}$) obtained by training GMM on the enrollment data and testing against test data.
- METHOD 1 - Reducing the dimension(d=30,40) while training T and again projecting it using LDA and HLDA
- METHOD 2 - Reducing the dimension(d=30,40) while training T only and NOT projecting it using LDA and HLDA
- METHOD 3 - Reducing the dimension to quarter (d=84) while training T only and NOT projecting it using LDA and HLDA

The result we expect is that METHOD3 $>$ METHOD2 $>$ METHOD1 $>$ BS, since reducing the dimension lead to less capturing of information and in turn leading to less accuracy.

VII. EXPERIMENTS

For conducting the experiments, the following front-end processing and back-processing steps were considered.

- The total number of speakers for these experiments were 200 (100 male and 100 female).
- The final mel feature vector is 21 dimensional for each frame in the audio (12 mel coefficients, 6 delta coefficients and 3 delta-delta coefficients)
- The number of components in each GMM was set to 16 components and the GMMs trained were constrained to have diagonal covariance to save on computation time.
- The whole audio data was split into training (80%), enrollment (6.7%) and testing (13.3%).
- It is assumed that the extracted training data is clean (free of noise) and non-stationary noise was added to the golden testing data (to get noisy testing data)
- Training of T matrix, A matrix for LDA and HLDA is done on both enrollment data and training data
- Using the trained T matrix, i-vectors for enrollment and testing data are obtained
- CDS comparing is done by comparing the i-vector of test data with each i-vector of enrollment data

VIII. Practical Issues

- Initialization - We know that when we are using EM algorithm , some initialization has to be given. EM has a tendency to converge to local minima instead of global one . To avoid it random restart has to be used.

- Number of Components in the GMM - Choosing the number of components of GMM is a huge problem. Theoretically there is no fixed method to select right number of components of GMM but many papers suggest to take it as 32,64,128 etc and so on.

- Memory - Here in the NIST 2006 data set we have each audio file of 5 minutes. Extracting features from full 5 minute file make the extracted feature file of more than 1 GB. To reduce the file size , features were extracted from 1 minute audio file.

IX. RESULTS

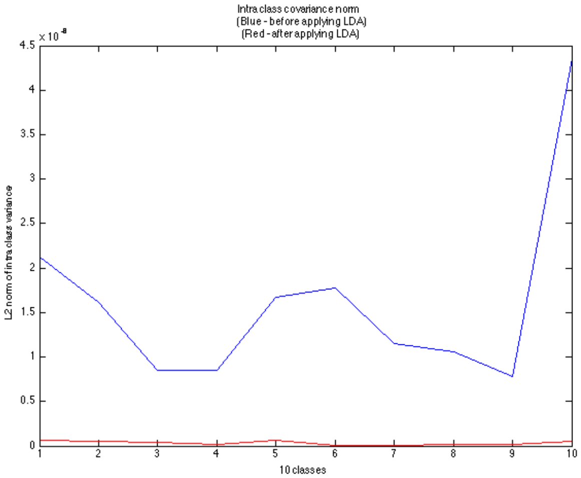

Fig. 4. Interclass Covariance for 10 classes

As shown in figure 4, the blue line shows L2 norm of intra class covariance before applying LDA and red line shows L2 norm of intra class covariance after applying LDA . As expected , the intra class covariance after applying LDA reduces.

As once can see in figure 5 and figure 6 GMM baseline is giving an accuracy of 70%. As shown in figure 5, as the d increases the accuracy moderately increases. In figure 5,d=40

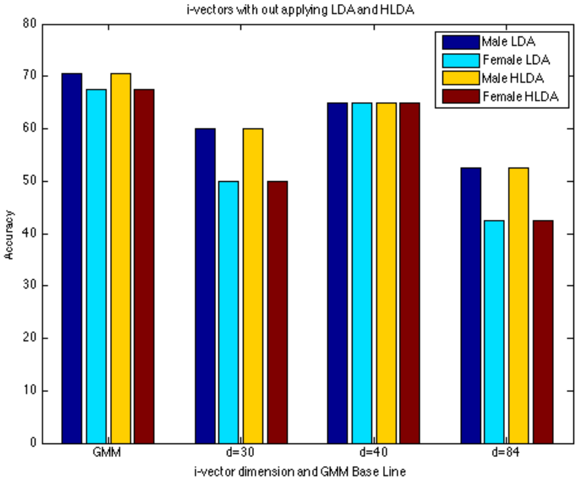

Fig. 5. i-vector without applying LDA and HLDA

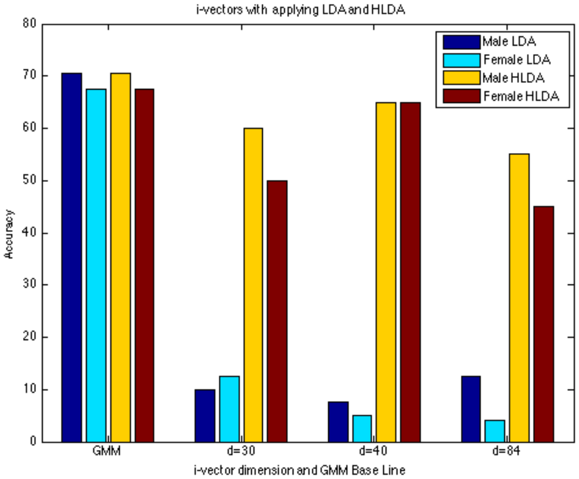

Fig. 6. i-vector after applying LDA and HLDA

is giving the highest accuracy as more information is retained in higher dimension. The same trend is seen in figure 6 but the overall accuracy of HLDA in figure 6 is more than HLDA in figure 5. The LDA in figure 6 is performing worse than LDA in figure 5. This is because in figure 6 when we do LDA , it takes same within class covariance matrix , while classes actually have different within class covariance . Also as once can see Baseline of GMM is best in both the figures. The ivector approach using HLDA is perfoming equally well as baseline which implies that by selecting appropriate within class covariance , we can do equally well as normal GMM with lesser dimension.

Figure 7 and 8 shows the maximum occurring speaker id on ivectors without applying LDA and on ivectors with applying LDA respectively .

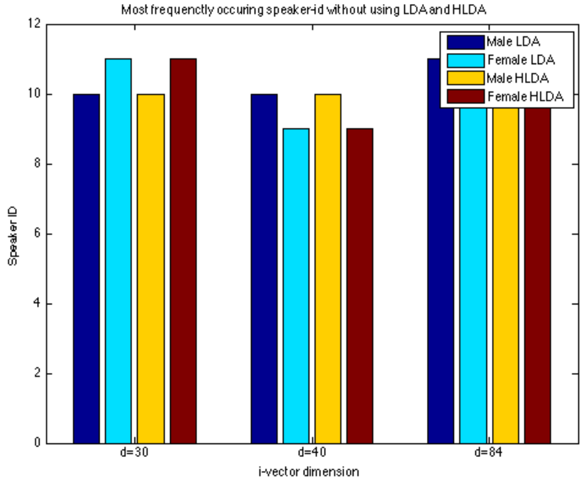

Fig. 7. Maximum occurring speaker id for i-vector without applying LDA and HLDA

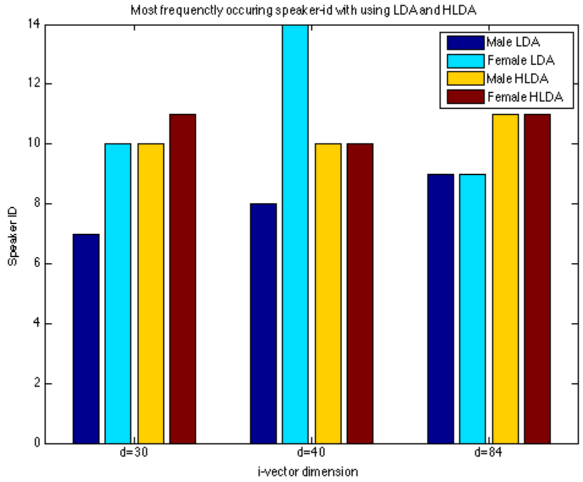

Fig. 8. Maximum occurring speaker id for i-vector with applying LDA and HLDA

Dimension of i vectors(d)	Time to Run in seconds
GMM	303.509
30	1219.62
40	1229.67
84	5714.1

TABLE I
RUN TIME TO COMPUTE I-VECTORS WITHOUT LDA AND HLDA

Dimension of i vectors(d)	Time to Run in seconds
GMM	310.57
30	1326.6214
40	1330.255
84	5769.53

TABLE II
RUN TIME TO COMPUTE I-VECTORS WITH LDA AND HLDA

As shown in table I and II the run time increases as the dimension of the i-vector increases . Also applying LDA and

HLDA takes more time so run times of table II are greater than table I .

X. Conclusion

In this paper , we compare two channel compensation techniques namely LDA (Linear Discriminant Analysis) and Bayesian HLDA (Bayesian Heteroscedastic Linear Discriminant Analysis) and the effect of increasing the dimensions of ivectors. Here LDA uses same within class covariance matrix for each class while HLDA uses different within class covariance matrix for each class . HLDA gives better accuracy than LDA. Also as the dimension of ivector increases , the accuracy generally increases.

References

[1] H. Beigi. *Fundamentals of Speaker Recognition*. SpringerLink : Bücher. Springer, 2011.

[2] SoftSound Limited Tony Robinson. VoiceBox:Speech Processing Toolbox for MATLAB.

[3] Balakrishnan Narayanaswamy, R Reddy, and Richard Stern. Improved Text-Independent Speaker Recognition using Gaussian Mixture Probabilities. 2005.

[4] S Memon. Automatic Speaker Recognition: Modelling, Feature Extraction and Effects of Clinical Environment. (June), 2010.

[5] Douglas Reynolds. Universal background models. *Encyclopedia of Biometrics*, pages 1–3, 2009.

[6] Howard Lei. Joint factor analysis (jfa) and i-vector tutorial.

[7] Mohammed Senoussaoui, Patrick Kenny, Najim Dehak, and Pierre Dumouchel. An i-vector extractor suitable for speaker recognition with both microphone and telephone speech. In *Odyssey*, page 6. ISCA, 2010.

[8] P. Kenny, P. Ouellet, N. Dehak, V. Gupta, and P. Dumouchel. A study of interspeaker variability in speaker verification. *Audio, Speech, and Language Processing, IEEE Transactions on*, 16(5):980–988, July 2008.

[9] Patrick Kenny, Gilles Boulianne, and Pierre Dumouchel. Eigenvoice modeling with sparse training data. *IEEE Transactions on Speech and Audio Processing*, 13(3):345–354, 2005.

[10] Duncan Fyfe Gillies. Intelligent data analysis and probabilistic inference lecture 16.

[11] Hakan Erdogan. Regularizing linear discriminant analysis for speech recognition. In *INTERSPEECH*, pages 3021–3024. ISCA, 2005.

[12] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet. Front-end factor analysis for speaker verification. *Audio, Speech, and Language Processing, IEEE Transactions on*, 19(4):788–798, May 2011.

[13] Bonastre Jean. Workshop: Advances in speech technologies.