Senone I-Vectors for Robust Speaker Verification

Zhili Tan\textsuperscript{1}, Yingke Zhu\textsuperscript{2}, Manwai Mak\textsuperscript{1}, Brian Mak\textsuperscript{2}

\textsuperscript{1}The Hong Kong Polytechnic University
\textsuperscript{2}The Hong Kong University of Science and Technology
Contents

1. I-Vector for Speaker Recognition
2. Motivation of Work
3. Conventional I-Vectors
4. DNN I-Vectors
5. Experiments on NIST 2012 Evaluation Set
6. Conclusions and Future Work
I-Vector Based Speaker Verification

- An i-vector is a low dimensional representation of a whole utterance, captured speaker- and channel-dependent characteristics.
Motivation

• Integrates phonetic information into i-vectors by DNN
  – Extracts bottleneck (BN) features
  – Estimates senone posteriors

• Denoises the MFCC vectors through the deoising autoencoder

• This architecture allows us to extract BN features and estimates senone posteriors given noisy MFCCs as input, resulting in robust senone i-vectors.
Sufficient Statistics

• Given the observed acoustic feature vectors of speaker \( i \),
\( O_i = \{o_1, \ldots, o_{iT_i}\} \), we can calculate the sufficient statistics corresponding to the Gaussian mixture \( c \):

\[
\gamma_c(o_{it}) = \frac{\lambda_c^{(b)} \mathcal{N}(o_{it} | \mu_c^{(b)}, \Sigma_c^{(b)})}{\sum_{j=1}^{C} \lambda_j^{(b)} \mathcal{N}(o_{it} | \mu_j^{(b)}, \Sigma_j^{(b)})}
\]

0th order statistic
\[
N_{ic} = \sum_t \gamma_c(o_{it})
\]

1st order statistic
\[
\tilde{f}_{ic} = \sum_t \gamma_c(o_{it})(o_{it} - \mu_c^{(b)})
\]

2nd order statistic
\[
S_{ic} = \sum_t \gamma_c(o_{it})(o_{it} - \mu_c)(o_{it} - \mu_c)^T
\]
Total Variability Matrix Training

- I-vector model: \( \mu_i = \mu^{(b)} + T w_i + \epsilon_i \)
- E-step:
  \[
  \langle w_i | O_i \rangle = L_i^{-1} \sum_c T_c^T (\Sigma_c^{(b)})^{-1} \tilde{f}_{ic}
  \]
  \[
  \langle w_i w_i^T | O_i \rangle = L_i^{-1} + \langle w_i | O_i \rangle \langle w_i | O_i \rangle^T
  \]
  \[
  L_i = I + T^T (\Sigma^{(b)})^{-1} N_i T
  \]
- M-step:
  \[
  T_c = \left[ \sum_i \tilde{f}_{ic} \langle w_i | O_i \rangle^T \right] \left[ \sum_i N_{ic} \langle w_i w_i^T | O_i \rangle \right]^{-1}
  \]

All we need:

\[
N_{ic} \\
\tilde{f}_{ic} \\
\Sigma_c^{(b)}
\]
Mean Vector and Covariance Matrix

- In most systems, \( \{ \mu_c \} \) and \( \{ \Sigma_c \} \) are obtained from the UBM.
- However, they can also be obtained using the sufficient statistics:

\[
\mu_c = \frac{\sum_i \sum_t \gamma_c(o_{it}) o_{it}}{\sum_i \sum_t \gamma_c(o_{it})}
\]

\[
\Sigma_c = \frac{\sum_i \sum_t \gamma_c(o_{it})(o_{it} - \mu_c)(o_{it} - \mu_c)^T}{\sum_i \sum_t \gamma_c(o_{it})}
\]
General Type of I-Vector

- Only the acoustic vectors $o_{it}$ and the mixture posteriors $\gamma_c(o_{it})$ are necessary for i-vector extraction.

- Given the speech signal of the $t$-th frame in the $i$-th utterance $s_{it}$
  - The MFCC could be replaced by other types of acoustic features:
    \[ o_{it} = f(s_{it}) \]
  - The mixture posteriors could be estimated from other model rather than GMM:
    \[ \gamma_c(s_{it}) = P(c|s_{it}) \]
Senone I-Vector

• The general type of i-vector allows the integration of supervised signal, with the standard backends remaining unchanged.
  – The supervised information is brought by $f(\bullet)$ and $c$.

• For example, a deep neural network (DNN) trained for ASR can help to integrate the phonetic information into i-vectors.
  – The BN features replaces MFCCs as acoustic features;
  – The posteriors of senones replaces the GMM mixture posteriors.

• Furthermore, a denoising autoencoder integrated into i-vector extraction may help to improve the noise robustness.
DNN I-Vectors

- BN feature vectors: \( o_{it} = \text{BN}(s_{it}) \)
- Senone posteriors: \( \gamma_c(s_{it}) = P_{DNN}(c|s_{it}) \)
  - The output of the \( c \)-th node in the softmax output layer.
- Baum-Welch statistics:

\[
N_{ic} = \sum_t P_{DNN}(c|s_{it}) \\
\tilde{f}_{ic} = \sum_t P_{DNN}(c|s_{it})(\text{BN}(s_{it}) - \mu_c) \\
S_{ic} = \sum_t P_{DNN}(c|s_{it})(\text{BN}(s_{it}) - \mu_c)(\text{BN}(s_{it}) - \mu_c)^T
\]

where:
\[
\mu_c = \frac{\sum_i \sum_t \gamma_c(s_{it})\text{BN}(s_{it})}{\sum_i N_{ic}}
\]
Denoising Classifier Training

Denoising Deep Autoencoder

Input Layer

Hidden Layer 1

$w_1 + \varepsilon_1$

Hidden Layer 2

$w_2 + \varepsilon_2$

Hidden Layer 3

$w_2^T + \varepsilon_3$

Hidden Layer 4

$w_4^T + \varepsilon_4$

Output Layer (Hidden Layer 4)

$w_1^T + \varepsilon_4$

Target

Clean MFCC

FaNT - 0dB SNR

Noisy Speech

Clean Speech

Noisy and Clean MFCC

DNN with DAE

Input Layer

Hidden Layer 1

$w_1 + \varepsilon_1$

Hidden Layer 2

$w_2 + \varepsilon_2$

Hidden Layer 3

$w_2^T + \varepsilon_3$

Hidden Layer 4

$w_4^T + \varepsilon_4$

Output Layer (Hidden Layer 4)

$w_1^T + \varepsilon_4$

Target

Clean MFCC

FaNT - 0dB SNR

Noisy Speech

Clean Speech

Noisy and Clean MFCC

Senones

BN Layer

Hidden Layer 5

$w_3 + \varepsilon_5$

Hidden Layer 6

$w_6$

Denoising Classifier Training
Denoising Senone I-vector Extraction

Speech

MFCC

DNN with DAE

BN Layer

Hidden Layer 6

Hidden Layer 5

Hidden Layer 4

Hidden Layer 3

Hidden Layer 2

Hidden Layer 1

Input Layer

Senones

$w_6$

$w_5 + \varepsilon_7$

$w_4 + \varepsilon_6$

$w_3 + \varepsilon_5$

$w_2^T + \varepsilon_4$

$w_2 + \varepsilon_3$

$w_1^T + \varepsilon_4$

$w_1 + \varepsilon_1$

BN Features

PCA Whitening

1st Order Sufficient Statistics

0th Order Sufficient Statistics

I-Vector Extractor

Senone I-Vector
Experimental Setup

• Evaluation dataset: NIST 2012 SRE CC4
  – Add babble noise at SNR of 15dB, 6dB and 0dB
• Baseline:
  – Acoustic features: 19 MFCCs together with energy plus their 1st and 2nd derivatives \( \rightarrow \) 60-Dim with feature warping
  – Posteriors: from GMM with 1024 mixtures
• T-matrix
  – 500-dimensional subspace trained by clean data
• PLDA
  – 150 latent variables
SNR Distribution of Evaluation Set
DBN Setup

- **Structure:** $\mathcal{D}-256-256-256-\mathcal{D}-256-256-60($BN$)-2000($Senone$)$
  - $\mathcal{D}$ represents the dimension of the input vectors
  - RBM pretraining: two Gaussian-Bernoulli RBMs and one Bernoulli-Gaussian RBM
  - BP fine-tuning: two linear activated layers

- **Input of DNN:**
  - 11 frames of 20-Dim MFCC without Feature Warping
  - Normalization by z-norm

- **Decorrelation for BN features:**
  - PCA whitening
  - GMM-UBM with diagonal covariance matrix
Result – Power of BN Features

- Posteriors from GMM-UBM are used
- Acoustic features are (1) MFCC, and (2) BN features

With posteriors from GMM-UBM, BN features outperform MFCC under noisy conditions
Result – Power of Senone Posteriors

- BN features are used
- Posteriors are obtained from (1) UBM, and (2) DNN

With BN features, DNN posteriors outperform UBM posteriors
Result – Power of Denoising AE

• To verify the power of Denoising Autoencoder, another DNN without DAE training was built.
• Posteriors from DNN without DAE training
• BN features are (1) with DAE training, and (2) without DAE training

• The denoising autoencoder improves noise robustness
Conclusions and Future Work

• The senone i-vectors outperforms the conventional i-vectors under all of the SNR conditions.

• The senone information benefits i-vector extraction.

• The denoising autoencoder improves noise robustness.

• The original NIST 12 CC4 evaluation set already contains noisy speech. Experiments on only clean speech are necessary in the future.
THANKS!

Q & A
# Full Result Table

<table>
<thead>
<tr>
<th>Acoustic Features</th>
<th>Postiers from</th>
<th>Original CC4 EER(%)</th>
<th>minDCF</th>
<th>15dB CC4 EER(%)</th>
<th>minDCF</th>
<th>6dB CC4 EER(%)</th>
<th>minDCF</th>
<th>0dB CC4 EER(%)</th>
<th>minDCF</th>
</tr>
</thead>
<tbody>
<tr>
<td>MFCC</td>
<td>UBM</td>
<td>2.664</td>
<td>0.2830</td>
<td>3.600</td>
<td>0.3633</td>
<td>4.034</td>
<td>0.4412</td>
<td>7.522</td>
<td>0.7313</td>
</tr>
<tr>
<td>BN with DAE</td>
<td>UBM</td>
<td>2.945</td>
<td>0.3352</td>
<td>4.167</td>
<td>0.3595</td>
<td>3.999</td>
<td>0.4279</td>
<td>5.691</td>
<td>0.6722</td>
</tr>
<tr>
<td>BN with DAE</td>
<td>DNN with DAE</td>
<td>1.537</td>
<td>0.2468</td>
<td>2.616</td>
<td>0.2387</td>
<td>2.591</td>
<td>0.3545</td>
<td>5.404</td>
<td>0.7529</td>
</tr>
<tr>
<td>BN with DAE</td>
<td>DNN without DAE</td>
<td>1.476</td>
<td>0.2345</td>
<td>2.369</td>
<td>0.2289</td>
<td><strong>2.370</strong></td>
<td>0.3481</td>
<td>5.297</td>
<td>0.7465</td>
</tr>
<tr>
<td>BN without DAE</td>
<td>DNN with DAE</td>
<td><strong>1.330</strong></td>
<td>0.2319</td>
<td><strong>2.305</strong></td>
<td>0.2171</td>
<td>2.522</td>
<td><strong>0.3372</strong></td>
<td>5.423</td>
<td>0.7495</td>
</tr>
<tr>
<td>BN without DAE</td>
<td>DNN without DAE</td>
<td>1.506</td>
<td><strong>0.2219</strong></td>
<td>2.440</td>
<td>0.2264</td>
<td>2.446</td>
<td>0.3573</td>
<td>5.531</td>
<td>0.7575</td>
</tr>
</tbody>
</table>