A Small Footprint i-Vector Extractor

Patrick Kenny

Odyssey Speaker and Language Recognition Workshop

June 25, 2012
Review of the i-vector representation

Exact posterior calculation (eigenvoice MAP) is necessary, at least for training

Both memory and computational requirements are quadratic in the i-vector dimension

A variational Bayes implementation of the probability model enables the posterior calculation to be done exactly in linear time and memory

Experiments with i-vectors of dimension up to 1600
The i-vector representation

- Given a recording of a single speaker, the i-vector representation of the recording is a vector of fixed dimension e.g. 400
- Speaker recognition reduces to a traditional pattern recognition problem like face recognition
- Classical techniques like LDA, cosine distance scoring applicable
- JFA reduces to PLDA: simpler, easier to implement and performs better
- New approaches to language recognition and diarization (based on representing speaker turns by i-vectors — works for broadcast news as well as telephone speech)
The idea

- Implicitly each utterance is represented by a (speaker and channel dependent) GMM
- **If the GMM mean supervectors were observable** you could apply probabilistic principle components analysis (PPCA)
- Supervectors assumed to lie in a low dimensional subspace
  - basis of this subspace known as *eigenvoices*
  - components of a supervector with respect to this basis constitute the *i-vector representation*
- i-vector components represent speaker characteristics, room impulse response and other types of non-phonetic variability
But GMM supervectors are *not* observable ...

- Think of the GMM supervector for an utterance as a perturbation of the UBM supervector
- An utterance is given as a sequence of frames
  \[ X = X_1, \ldots, X_T \]

The Baum-Welch statistics of orders 0 and 1 can be thought of as summarizing an incomplete observation of the utterance supervector
- For each mixture component \( c \), \( N_c \) is the number of frames accounted for by the mixture component, and \( F_c \) the sum of these frames (\( F_c \) for ‘first order’)

Think of the part of the supervector corresponding to the mixture component $c$ as being observed $N_c$ times.

This differs from the usual situation where each component of an observation vector is observed exactly once.
The probability model

\[ m = m^{\text{UBM}} + V y \]

where

- \( m \) = utterance dependent supervector
- \( m^{\text{UBM}} \) = UBM supervector
- \( V \) = matrix whose columns are the eigenvoices
- \( y \) = i-vector, assumed to have a standard normal prior
In terms of mixture components rather than supervectors:

For each mixture component $c$,

$$m_c = m_c^{\text{UBM}} + V_c y$$

- The covariance matrix associated with the mixture component $c$ (which may or may not be diagonal) is denoted by $\Sigma_c$
- $\Sigma_c$ is not utterance dependent
In most implementations, $\Sigma_c$ is copied from the UBM rather than estimated in the PPCA framework.

Surprisingly enough, this leads to slightly improved results (see later).

Fixing $\Sigma_c$ in this way, the Baum-Welch statistics can be prewhitened so that we can take $m_c^{UBM} = 0$ and $\Sigma_c = I$.

Of course this changes the estimate of $V_c$ but there is no net effect on the i-vector calculation.

We will assume $m_c^{UBM} = 0$ and $\Sigma_c = I$ as this simplifies the implementation and full covariance UBMs can be handled.
The posterior of $y$ given $X$

$$\text{Cov}(y, y) = \left( I + \sum_c N_c V_c^* V_c \right)^{-1}$$

$$\langle y \rangle = \text{Cov}(y, y) \sum_c V_c^* F_c$$

- In the case $c = 1$ and $N_c = 1$ this is the same as the PPCA posterior calculation (Bishop)
- Accumulate the matrix $\sum_c N_c V_c^* V_c$ for each utterance
- The matrices $V_c^* V_c$ are typically precomputed
- For a typical UBM configuration and 1000-dimensional i-vectors, the run-time memory overhead is 8.8 Gb (multiply by 2 for training)
The posterior computation is essential for training as well as for i-vector extraction. For example, for maximum likelihood training,

\[
V_c = \left( \sum_s \langle y(s) \rangle F_c^*(s) \right) \left( \sum_s N_c(s) \langle y(s)y^*(s) \rangle \right)^{-1}
\]

\[
\langle y(s)y^*(s) \rangle = \text{Cov}(y(s), y(s)) + \langle y(s) \rangle \langle y^*(s) \rangle
\]

where \( s \) ranges over speakers in the training set

Calculating \( \text{Cov}(y(s), y(s)) \) requires a matrix inversion (rather than merely solving a linear system)
Diagonalizing the posterior

Assume provisionally that the posterior covariance of $y$ given $X$ matrix is approximately diagonal, so that the posterior is approximately factorial

$$P(y|X) \approx Q(y^1) \ldots Q(y^R)$$

The variational lower bound, defined as

$$\mathcal{L} = E \left[ \ln \frac{P(y, X)}{Q(y)} \right],$$

increases on each variational Bayes update

$$\ln Q(y^r) = E_{y \setminus y^r} [\ln P(y, X)] + \text{constant}$$

This is an iterative method: cycle over the i-vector components
Since $P(X|y)$ and $P(y)$ are Gaussian, $\ln P(y, X)$ is quadratic in $y$.

Thus $\ln Q(y^r)$ is quadratic in $y^r$.

Thus the variational posterior of $y^r$ is Gaussian and its mean and variance can be read off by collecting first and second order terms.

Just as the calculation of the full posterior can be facilitated by pre-computing the matrices $V_c^* V_c$, the calculation of the variational posterior can be facilitated by pre-computing the diagonals $\text{diag}(V_c^* V_c)$.

The memory overhead is negligible.
If it is run until convergence, the variational Bayes method is guaranteed to find the point estimate of the i-vector exactly.

The posterior mean is calculated exactly but not the posterior covariance matrix (because of the diagonal approximation).

The algorithm is an instance of the Jacobi method of solving linear systems:
- The Jacobi method does not always converge.
- Convergence is guaranteed in this case because of variational Bayes.

Efficiency of the method depends on how accurate the diagonal approximation is.
Why is the diagonal approximation reasonable?

- If $O$ is any orthogonal matrix then the probability model is unchanged under the transformations

$$
y \leftarrow Oy
V \leftarrow VO^*
$$

(since $O^*O = I$)

- A good approximation to the full posterior precision matrix is

$$I + N \sum_c w_c V_c^* V_c$$

where $N$ is the total number of frames and $w_c$ is the mixture weight for component $c$, so that $N_c \approx Nw_c$

[Glembek, ICASSP 2011]
Diagonalizing the matrix \( \sum_c w_c V_c^* V_c \) produces a basis of the i-vector space with respect to which all the posterior covariance matrices are approximately diagonal.

Doing the calculations in this basis ensures that the variational Bayes algorithm converges very quickly, typically in 3 iterations, irrespective of the i-vector dimension.

- “iteration” refers to a pass over all the i-vector components.

Thus the computational requirements of variational Bayes are linear in the i-vector dimension.

If an i-vector extractor is trained with variational Bayes, the preferred basis changes from one iteration to the next.
The variational lower bound

- A proxy for the evidence i.e. the marginal likelihood of the data
- To evaluate it, see the paper
- Increases on successive iterations of variational Bayes so can be used for troubleshooting and monitoring convergence
- Likewise, it can be used to monitor convergence when training i-vector extractors with variational Bayes
- The exact evidence (which can only be evaluated using the full posterior covariances) cannot be used as a convergence criterion in this situation
Experimental questions

- How accurate is variational Bayes?
  - No issue here at run time (i-vectors are calculated exactly)
  - But using diagonal posterior covariance matrices in training i-vector extractors could conceivably be harmful

- Is it OK to copy the UBM covariance matrices?

- How efficient is variational Bayes?

- Variational Bayes makes it possible to train very high dimensional i-vector extractors. Is there anything to be gained from this?
Testbed

- Female det 2 trials (normal telephone speech), extended core condition, NIST 2010 speaker recognition evaluation
- Standard front end and diagonal UBM (2048 Gaussians) trained on the usual data sets (see paper)
- 400 dimensional i-vectors, except where otherwise indicated
- 100 dimensional LDA
- Heavy-tailed PLDA classifier
- Metrics: EERs and the 2008 and 2010 normalized detection cost functions (NDCF)
Accuracy of variational Bayes

- Benchmark obtained with JFA executables: EER = 3.1%, 2010 NDCF = 0.50 on NIST 2010 extended core, female data
  - i-vector extractor trained by modifying the script so as to ignore speaker/channel distinctions
  - covariance matrices re-estimated rather than copied from the UBM
  - voice activity detection later found wanting
Marginally better results obtained with variational Bayes: EER = 3.0\%, 2010 NDCF = 0.49
- covariance matrices copied from the UBM
- variances are overestimated by about 5%

Copying the covariance matrices was originally motivated by efficiency (the Baum-Welch statistics can be whitened) but it helps with accuracy

A much more important advantage is that full covariance matrices can be accommodated
Efficiency of variational Bayes

For the standard (full covariance) method the time taken to extract a 400 dimensional i-vector is $\sim 0.5$ sec
- Almost all of the time is spent in BLAS routines e.g. accumulating the precision matrices accounts for 75% of the computation
- An estimate of 0.25 sec per i-vector is given [Glembek, ICASSP 2011]
- Suggests that compiler optimization may be helpful
For the variational Bayes method the time taken to extract a 400 dimensional i-vector is $\sim 0.9$ sec

- Slower but overhead is negligible compared to the cost of extracting Baum-Welch statistics
- Number of variational Bayes iterations fixed at 5, although the average number needed in practice is about 3
- Just as for the memory requirements, the computational cost of the standard method scales quadratically whereas the the cost of the variational Bayes method scales linearly in the i-vector dimension
- High dimensional i-vector extractors can be built
High dimensional i-vectors

Table: EER / NDCF 2010 NIST extended core condition, female data. 100 dimensional LDA. Diagonal UBM, 2048 Gaussians.

<table>
<thead>
<tr>
<th>i-vector</th>
<th>EER</th>
<th>2008 NDCF</th>
<th>2010 NDCF</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>2.5%</td>
<td>0.13</td>
<td>0.45</td>
</tr>
<tr>
<td>800</td>
<td>2.7%</td>
<td>0.13</td>
<td>0.44</td>
</tr>
<tr>
<td>1200</td>
<td>2.7%</td>
<td>0.13</td>
<td>0.42</td>
</tr>
<tr>
<td>1600</td>
<td>2.8%</td>
<td>0.14</td>
<td>0.43</td>
</tr>
</tbody>
</table>
A variational Bayes implementation of the i-vector probability model enjoys several advantages

- The memory required is scarcely greater than that needed to store the eigenvoices
- The computational overhead is modest and the computation scales linearly rather than quadratically
- It can be used in training i-vector extractors as well as at run time without any compromise in recognition accuracies
- It enables i-vector extractors of very high dimension to be trained although this only produces modest improvements in speaker recognition accuracies