Odyssey 2016
21-24 Jun 2016, Bilbao, Spain

Compensation for phonetic nuisance variability in speaker recognition using DNNs

Themos Stafylakis[1], *Patrick Kenny*[1], *Vishwa Gupta*[1], *Jahangir Alam*[1], *and Marcel Kockmann*[2]

[1]Centre de Recherche Informatique de Montréal (CRIM), Canada
[2]VoiceTrust, Germany

themos.stafylakis@crim.ca

Abstract

In this paper, a new way of using phonetic DNN in text-independent speaker recognition is examined. Inspired by the Subspace GMM approach to speech recognition, we try to extract i-vectors that are invariant to the phonetic content of the utterance. We overcome the assumption of Gaussian distributed senones by combining DNN with UBM posteriors and we form a complete EM algorithm for training and extracting phonetic content compensated i-vectors. A simplified version of the model is also presented, where the phonetic content and speaker subspaces are learned in a decoupled way. Covariance adaptation is also examined, where the covariance matrices are reestimated rather than copied from the UBM. A set of primary experimental results is reported on NIST-SRE 2010, with modest improvement when fused with the standard i-vectors.

1. Introduction

The use of phonetic DNN posteriors as assignment probabilities for extracting Baum-Welch statistics was one of the major breakthroughs in speaker recognition during the last years [1], [2], [3]. It enabled us to focus on the way each speaker pronounces specific senones, rather than on spectral regions that are arbitrarily defined by applying unsupervised clustering techniques (i.e. a UBM trained with the EM algorithm). Moreover, the use of DNN-based bottleneck features as well as the successful deployment of Long Short Term Memory networks demonstrate the great potential of this direction of research [4].

A criticism on the DNN-based approach presented in [1] and [2] is based on the underlying Gaussian assumption about the distribution of senones. Although the mean value of a Gaussian distribution was proved to be a powerful feature for speaker recognition, the underlying distribution is more complex and multimodal. Experiments is speech recognition show that senone-conditional distributions are so wide that they can hardly be modelled with small-size GMMs [5]. Given a typical UBM, each of the S senones is distributed across potentially all mixture components of the UBM.

A way to overcome the Gaussian assumptions can be inspired by the Subspace GMM (SGMM) [5]. SGMM is a generative model that was initially proposed for speech recognition and its main assumptions are the following: (a) Each senone occupies potentially all UBM components. (b) Each senone introduces an offset to the UBM supervector, that is low-dimensional. (c) Each speaker-channel combination introduces another offset from the UBM supervector, that is also low-dimensional, additive and independent to the senone offsets.

The low-dimensional vector by which the speaker offset is encoded has similar properties to an i-vector, with the additional property of modelling characteristics of speakers that are invariant of the phonetic content. Recently, it was shown that the particular vector performs very well in text-independent speaker recognition, especially when the SGMM is trained with DNN alignments ("system B" in [6]). However, the use of DNN merely for estimating alignments in transcribed datasets (e.g. Fisher) prevented the authors from using untranscribed datasets e.g. the Mixer corpus. Moreover, the use of SGMMs instead of DNNs for frame-posterior estimations is suboptimal, at least from a speech recognition perspective.

A purpose of the present paper is to incorporate the main concepts of SGMM in the usual probabilistic framework of Joint Factor Analysis (JFA) and i-vectors. We do so by placing priors to the random variables and by deriving a Variational Bayes algorithm for training the model. The main assumption regarding the generative model is the decomposition of the frame-level supervector into two statistically independent low-dimensional random variables: A speaker variable, which captures speaker (and channel) information (like an i-vector), and a phonetic variables that captures the phonetic content of the particular frame. The role of the observation-dependent nuisance variables is to model the phonetic variability and subsequently remove it from the Baum-Welch statistics, leading to i-vectors that are less sensitive to the phonetic content.

An analogy can be drawn with the way JFA treats channel effect as nuisance variables, especially when focusing of the effect they have to the Baum-Welch statistics [7]. However, there are two main differences between the two methods. First, channel effects are assumed stationary throughout an utterance, while the phonetic effects vary from frame to frame. Second, channel effects are assumed unknown in JFA, while phonetic effects can be estimated using DNN-based state-of-the-art methods. Therefore, DNNs offer an additional degree of supervision to the model, by providing estimates of the nuisance variables.

The rest of the paper is organized as follows. In Sect. 2, the coupled approach is presented, where a unique EM algorithm is used to train the model. In Sect. 3, we introduce a decoupled approach for training the model, together with covariance adaptation. Finally, experimental results are shown and discussed in Sect. 4.

2. Coupled approach

In this section, we motivate and describe the proposed model, using a unified training algorithm. Although the derivation of the algorithm is not fully analyzed, we mention that we use Variational Bayes approximate inference on the E-step, maximum likelihood estimation for the model parameters, while

doi: 10.21437/Odyssey.2016-49

minimum divergence estimation is added to the M-step, to make the distribution of the hidden variables standardized multivariate Gaussian. We do not present covariance adaptation in this section, to keep the presentation of the algorithm simpler, although it is straightforward to incorporate it (see Sect. 3).

2.1. Generative model

We assume that for each file f there is an i-vector $\boldsymbol{i}_f$ and for each senone s a j-vector, $\boldsymbol{j}_s$. For each senone s and mixture component c, acoustic observations $\boldsymbol{O}_t$ associated with the triple (s, c, f) are assumed to be distributed with mean

$$\boldsymbol{m}_c + \boldsymbol{T}_c \boldsymbol{i}_f + \boldsymbol{U}_c \boldsymbol{j}_s \tag{1}$$

and covariance matrix $\boldsymbol{\Sigma}_c$. It is convenient to write this in the form $\boldsymbol{m}_c + \boldsymbol{W}_c \boldsymbol{k}_{fs}$ where

$$\boldsymbol{W}_c = \begin{pmatrix} \boldsymbol{T}_c & \boldsymbol{U}_c \end{pmatrix} \tag{2}$$

$$\boldsymbol{k}_{fs} = \begin{pmatrix} \boldsymbol{i}_f \\ \boldsymbol{j}_s \end{pmatrix}. \tag{3}$$

Note the difference between the generative model of JFA and the one presented here. While in JFA the channel effects are stationary throughout the utterance, the phonetic effects are observation-dependent. This is why we begin by defining the generative model of a supervector that corresponds to a single observation. Note also that in text-independent speaker recognition the senone labels are unknown, contrary to the text-dependent case, where hard assignments can be obtained using forced-alignment, [8].

Initially, we assume that $\boldsymbol{m}_c$ and $\boldsymbol{\Sigma}_c$ are copied from a UBM and that the priors on the i-vectors and j-vectors, $P(\boldsymbol{i}_f)$ and $P(\boldsymbol{j}_s)$, are standard normal. Covariance adaptation will be discussed in Sect. 3.

2.2. Baum-Welch Statistics

In order to extract Baum-Welch statistics, we require two probabilistic assignments; one vector of assignment probabilities given by the DNN and a second one given by the UBM. Focusing an a single observation at time t in the file f. The (s, c) element of the outer product of the two vectors corresponds to the probability of the combined event that the observation is accounted for by a senone s and mixture component c. We denote this posterior probability by $Q_t(s, c, f)$. Note that Q_t is a sparse array, especially when a threshold (equal to 0.01 in our experiments) is used for both UBM and DNN assignment probabilities.

The Baum-Welch statistics are defined in the following way

$$N_{s,c,f} = \sum_t Q_t(s, c, f) \tag{4}$$

$$\boldsymbol{F}_{s,c,f} = \sum_t Q_t(s, c, f) \left(\boldsymbol{O}_t - \boldsymbol{m}_c\right) \tag{5}$$

$$\tilde{\boldsymbol{F}}_{s,c,f} = \boldsymbol{L}_c^{-1} \boldsymbol{F}_{s,c,f} \tag{6}$$

where $\boldsymbol{L}_c \boldsymbol{L}_c^\top = \boldsymbol{\Sigma}_c$, i.e the Cholesky decomposition of $\boldsymbol{\Sigma}_c$.

Note that the overall dimensionality of $\boldsymbol{F}_{s,c,f}$ is huge. However only $\boldsymbol{F}_{\cdot,c,f} = \sum_s \boldsymbol{F}_{s,c,f}$ and $\boldsymbol{F}_{s,c,\cdot} = \sum_f \boldsymbol{F}_{s,c,f}$ are required to be computed and stored. Contrarily, $N_{s,c,f}$ is required in its non-accumulated form and therefore they should be stored in sparse matrix format.

As in the standard i-vector extractor, we prewhiten the Baum-Welch statistics, by setting $\tilde{\boldsymbol{U}}_c = \boldsymbol{L}_c^{-1} \boldsymbol{U}_c$ (and similarly for $\tilde{\boldsymbol{T}}_c$ and $\tilde{\boldsymbol{W}}_c$). By doing so, we can treat $\boldsymbol{\Sigma}_c$ as the identity matrix, since $\boldsymbol{U}_c^\top \boldsymbol{\Sigma}_c^{-1} \boldsymbol{U}_c = \tilde{\boldsymbol{U}}_c^\top \tilde{\boldsymbol{U}}_c$ and $\boldsymbol{U}_c^\top \boldsymbol{\Sigma}_c^{-1} \boldsymbol{F}_c = \tilde{\boldsymbol{U}}_c^\top \tilde{\boldsymbol{F}}_c$.

These are the variables that should be precomputed. We now show how the EM iteration is implemented, which consists of two E-steps (one for i-vectors and one for j-vectors) followed by a joint M-step.

2.3. E-Step: i-vectors

We begin with the i-vectors, although the order is arbitrary. For each file f, the variational posterior of $\boldsymbol{i}_f$ is updated using the zero order statistics $N_{\cdot,c,f}$ and synthetic first order statistics $\boldsymbol{G}_{c,f}$ defined by

$$\boldsymbol{G}_{c,f} = \tilde{\boldsymbol{F}}_{\cdot,c,f} - \sum_s N_{s,c,f} \tilde{\boldsymbol{U}}_c \langle \boldsymbol{j}_s \rangle. \tag{7}$$

From eq. (7), we observe that $\boldsymbol{G}_{c,f}$ corresponds to the regular first-order statistics, compensated for the phonetic content by the term $\sum_s N_{s,c,f} \tilde{\boldsymbol{U}}_c \langle \boldsymbol{j}_s \rangle$. We note that a similar operation takes place in JFA, where the posterior expectation of the channel factors (weighted by the zero-order statistics) is subtracted from the first-order statistics. Moreover, like in JFA, only the point estimates of the nuisance parameters are needed in order to calculate the synthetic statistics $\boldsymbol{G}_{c,f}$. We also mention that once the model is trained (are therefore $\tilde{\boldsymbol{U}}_c$ is fixed), the compensation term is completely defined by the DNN through $N_{s,c,f}$. This is not the case with JFA, where the channel factors are unknown and therefore Variational Bayes iterations are required for estimating the channel compensation term. Finally, despite the fact that the nuisance variables are observation-dependent, Variational Bayes makes them collapse into a single vector, which is the posterior expectations of all senones $\langle \boldsymbol{j}_s \rangle$, weighted by $N_{s,c,f}$ and projected onto the supervector space by $\tilde{\boldsymbol{U}}_c$.

Once the synthetic Baum-Welch statistics are calculated, the posterior distribution of $\boldsymbol{i}_f$ is defined in the usual way. Denoting the posterior covariance by $\mathrm{Cov}\left(\boldsymbol{i}_f, \boldsymbol{i}_f\right)$ and the posterior expectation by $\langle \boldsymbol{i}_f \rangle$, we obtain

$$\mathrm{Cov}\left(\boldsymbol{i}_f, \boldsymbol{i}_f\right) = \left(\boldsymbol{I} + \sum_c N_{\cdot,c,f} \tilde{\boldsymbol{T}}_c^\top \tilde{\boldsymbol{T}}_c \right)^{-1} \tag{8}$$

$$\langle \boldsymbol{i}_f \rangle = \mathrm{Cov}\left(\boldsymbol{i}_f, \boldsymbol{i}_f\right) \sum_c \tilde{\boldsymbol{T}}_c^\top \boldsymbol{G}_{c,f}. \tag{9}$$

The correlation terms that will be used in the M-step are given below:

$$\langle \boldsymbol{i}_f \boldsymbol{i}_f^\top \rangle = \langle \boldsymbol{i}_f \rangle \langle \boldsymbol{i}_f^\top \rangle + \mathrm{Cov}\left(\boldsymbol{i}_f, \boldsymbol{i}_f\right). \tag{10}$$

Finally, the Kullback-Leibler (KL) Divergence is simplified to

$$D\left(Q(\boldsymbol{i}_f) \| P(\boldsymbol{i}_f)\right) = -\frac{R}{2} - \frac{1}{2} \ln \left| \mathrm{Cov}\left(\boldsymbol{i}_f, \boldsymbol{i}_f\right) \right| + \frac{1}{2} \mathrm{tr}\left(\langle \boldsymbol{i}_f \boldsymbol{i}_f^\top \rangle \right) \tag{11}$$

since $P(\boldsymbol{i}_f)$ is standard normal. KL Divergence is required for estimating the variational lower bound, which is very useful for tracking algorithmic convergence and for debugging the code.

2.4. E-Step: j-vectors

The j-vector posteriors can only be updated after all files have been read. This operation needs to be performed in the course of training but not at run time, since the set of j-vectors will be fixed after training.

For each senone s, the variational posterior of $\boldsymbol{j}_s$ is updated using the zero order statistics $N_{s,c,\cdot}$ and synthetic first order statistics $\boldsymbol{H}_{s,c}$ defined by

$$\boldsymbol{H}_{s,c} = \tilde{\boldsymbol{F}}_{s,c,\cdot} - \sum_f N_{s,c,f} \tilde{\boldsymbol{T}}_c \langle \boldsymbol{i}_f \rangle. \tag{12}$$

Note again that only point estimates of the i-vectors are needed in order to collect these statistics.

Denoting the posterior covariance by $\mathrm{Cov}\,(\boldsymbol{j}_s, \boldsymbol{j}_s)$ and the posterior expectation by $\langle \boldsymbol{j}_s \rangle$,

$$\mathrm{Cov}\,(\boldsymbol{j}_s, \boldsymbol{j}_s) = \left(\boldsymbol{I} + \sum_c N_{s,c,\cdot} \tilde{\boldsymbol{U}}_c^\top \tilde{\boldsymbol{U}}_c \right)^{-1} \tag{13}$$

$$\langle \boldsymbol{j}_s \rangle = \mathrm{Cov}\,(\boldsymbol{j}_s, \boldsymbol{j}_s) \sum_c \tilde{\boldsymbol{U}}_c^\top \tilde{\boldsymbol{H}}_{s,c}. \tag{14}$$

We also need the following formulas for updating the subspace in the M-step

$$\langle \boldsymbol{j}_s \boldsymbol{j}_s^\top \rangle = \langle \boldsymbol{j}_s \rangle \langle \boldsymbol{j}_s^\top \rangle + \mathrm{Cov}\,(\boldsymbol{j}_s, \boldsymbol{j}_s)\,. \tag{15}$$

The KL Divergence is as follows

$$D\left(Q(\boldsymbol{j}_s) \| P(\boldsymbol{j}_s)\right) = -\frac{\Lambda}{2} - \frac{1}{2} \ln |\,\mathrm{Cov}\,(\boldsymbol{j}_s, \boldsymbol{j}_s)\,| + \frac{1}{2} \mathrm{tr}\left(\langle \boldsymbol{j}_s \boldsymbol{j}_s^\top \rangle \right) \tag{16}$$

where Λ is the j-vector dimension.

2.5. Variational Lower Bound

The E-step updates are derived by the standard variational Bayes argument which leads to

$$\ln Q(\underline{\boldsymbol{i}}) \equiv E_{Q(\underline{\boldsymbol{s}},\underline{\boldsymbol{c}})Q(\underline{\boldsymbol{j}})} \left[\ln P(\underline{\boldsymbol{s}}, \underline{\boldsymbol{c}}, \underline{\boldsymbol{i}}, \underline{\boldsymbol{j}}, \underline{\boldsymbol{O}}) \right] \tag{17}$$

$$\ln Q(\underline{\boldsymbol{j}}) \equiv E_{Q(\underline{\boldsymbol{s}},\underline{\boldsymbol{c}})Q(\underline{\boldsymbol{i}})} \left[\ln P(\underline{\boldsymbol{s}}, \underline{\boldsymbol{c}}, \underline{\boldsymbol{i}}, \underline{\boldsymbol{j}}, \underline{\boldsymbol{O}}) \right]. \tag{18}$$

Here and elsewhere we use the symbol $\equiv$ to indicate equality up to an irrelevant additive constant. We use $\underline{\boldsymbol{i}}$ to indicate the collection of all i-vectors and similarly for $\underline{\boldsymbol{s}}, \underline{\boldsymbol{c}}, \underline{\boldsymbol{j}}$ and $\underline{\boldsymbol{O}}$.

The i-vector posterior updates, the j-vector posterior updates, maximum likelihood estimation and minimum divergence estimation are all guaranteed to increase the value of the variational lower bound, defined by

$$\mathcal{L} = E_{Q(\underline{\boldsymbol{s}},\underline{\boldsymbol{c}})Q(\underline{\boldsymbol{i}})Q(\underline{\boldsymbol{j}})} \left[\ln \frac{P(\underline{\boldsymbol{s}}, \underline{\boldsymbol{c}}, \underline{\boldsymbol{i}}, \underline{\boldsymbol{j}}, \underline{\boldsymbol{O}})}{Q(\underline{\boldsymbol{s}}, \underline{\boldsymbol{c}}) Q(\underline{\boldsymbol{i}}) Q(\underline{\boldsymbol{j}})} \right] \tag{19}$$

It is convenient to write this in the form

$$\mathcal{L} = E_{Q(\underline{\boldsymbol{s}},\underline{\boldsymbol{c}})Q(\underline{\boldsymbol{i}})Q(\underline{\boldsymbol{j}})} \left[\ln P(\underline{\boldsymbol{O}} | \underline{\boldsymbol{s}}, \underline{\boldsymbol{c}}, \underline{\boldsymbol{i}}, \underline{\boldsymbol{j}}) \right] - D\left(Q(\underline{\boldsymbol{i}}) \| P(\underline{\boldsymbol{i}})\right) - D\left(Q(\underline{\boldsymbol{j}}) \| P(\underline{\boldsymbol{j}})\right) - D\left(Q(\underline{\boldsymbol{s}}, \underline{\boldsymbol{c}}) \| P(\underline{\boldsymbol{s}}, \underline{\boldsymbol{c}})\right). \tag{20}$$

The first term here is referred to as the reconstruction error. The last term is fixed so we ignore it.

Up to an irrelevant additive constant,

$$\mathcal{L} \equiv \sum_c \mathrm{tr}\left(\tilde{\boldsymbol{W}}_c \boldsymbol{B}_c^\top - \frac{1}{2} \tilde{\boldsymbol{W}}_c^\top \tilde{\boldsymbol{W}}_c \boldsymbol{A}_c \right) - \sum_f D\left(Q(\boldsymbol{i}_f) \| P(\boldsymbol{i}_f)\right) - \sum_s D\left(Q(\boldsymbol{j}_s) \| P(\boldsymbol{j}_s)\right) \tag{21}$$

where

$$\boldsymbol{A}_c = \begin{pmatrix} \sum_f N_{\cdot,c,f} \langle \boldsymbol{i}_f \boldsymbol{i}_f^\top \rangle & \sum_{s,f} N_{s,c,f} \langle \boldsymbol{j}_s \rangle \langle \boldsymbol{i}_f^\top \rangle \\ \sum_{s,f} N_{s,c,f} \langle \boldsymbol{i}_f \rangle \langle \boldsymbol{j}_s^\top \rangle & \sum_s N_{s,c,\cdot} \langle \boldsymbol{j}_s \boldsymbol{j}_s^\top \rangle \end{pmatrix} \tag{22}$$

$$\boldsymbol{B}_c = \begin{pmatrix} \sum_f \tilde{\boldsymbol{F}}_{\cdot,c,f} \langle \boldsymbol{i}_f^\top \rangle & \sum_s \tilde{\boldsymbol{F}}_{s,c,\cdot} \langle \boldsymbol{j}_s^\top \rangle \end{pmatrix}. \tag{23}$$

2.6. Maximum Likelihood Estimation

For each component c, the maximum likelihood estimates of $\tilde{\boldsymbol{T}}_c$ and $\tilde{\boldsymbol{U}}_c$ are given by

$$\begin{pmatrix} \tilde{\boldsymbol{T}}_c & \tilde{\boldsymbol{U}}_c \end{pmatrix} = \boldsymbol{B}_c \boldsymbol{A}_c^{-1}. \tag{24}$$

We derive this expression by setting the derivative of $\mathcal{L}$ with respect to $\tilde{\boldsymbol{W}}_c$ equal to zero:

$$d\mathcal{L} = \mathrm{tr}\left(\left\{ \boldsymbol{B}_c^\top - \boldsymbol{A}_c \tilde{\boldsymbol{W}}_c^\top \right\} d\tilde{\boldsymbol{W}}_c \right) \tag{25}$$

so

$$\frac{\partial}{\partial \tilde{\boldsymbol{W}}_c} \mathcal{L} = \boldsymbol{B}_c^\top - \boldsymbol{A}_c \tilde{\boldsymbol{W}}_c^\top. \tag{26}$$

Setting this to $\boldsymbol{0}$ gives

$$\tilde{\boldsymbol{W}}_c = \boldsymbol{B}_c \boldsymbol{A}_c^{-1}. \tag{27}$$

It is worth mentioning that the phonetic variability, quantified by $\sum_c \mathrm{tr}\left(\tilde{\boldsymbol{U}}_c \tilde{\boldsymbol{U}}_c^\top \right)$ is approximately twice the speaker-channel variability $\sum_c \mathrm{tr}\left(\tilde{\boldsymbol{T}}_c \tilde{\boldsymbol{T}}_c^\top \right)$. This is in contrast to JFA, where the speaker variability is higher that the channel variability.

2.7. Minimum Divergence Estimation

We find a non-standard normal prior $P'(\boldsymbol{j}_s)$ with mean zero which minimizes $\sum_s D\left(Q(\boldsymbol{j}_s) \| P'(\boldsymbol{j}_s)\right)$ by setting the covariance matrix $\boldsymbol{C}$ to

$$\frac{1}{S} \sum_s \left\langle \boldsymbol{j}_s \boldsymbol{j}_s^\top \right\rangle \tag{28}$$

where S is the total number of senones. Let $\boldsymbol{M}$ be such that $\boldsymbol{M}\boldsymbol{M}^\top = \boldsymbol{C}$ so that $\boldsymbol{M}^{-1}\boldsymbol{C}\boldsymbol{M}^{-\top} = \boldsymbol{I}$. Then the transformations

$$\boldsymbol{j}_s \to \boldsymbol{M}^{-1} \boldsymbol{j}_s \tag{29}$$

and

$$\tilde{\boldsymbol{T}}_c \to \tilde{\boldsymbol{T}}_c \boldsymbol{M} \tag{30}$$

leave the reconstruction error fixed and map $P'(\boldsymbol{j}_s)$ onto the standard normal prior. Thus the net effect of this transformations is to increase the value of $\mathcal{L}$.

2.8. Notes on EM training

Some useful notes regarding the EM algorithm are given below.

1. If both minimum divergence and maximum likelihood estimation are performed, then maximum likelihood estimation has to be performed first, in order to ensure that the value of $\mathcal{L}$ increases.

2. Assuming that in the E-step the i-vectors are estimated first, the synthetic Baum-Welch statistics in eq. (7) should be calculated using the properly transformed j-vectors. The transformation is required due to minimum divergence estimation. To transform the j-vectors (i.e. the first order moments of their variational posterior) we apply $\langle \boldsymbol{j}_s \rangle \rightarrow \boldsymbol{M}^{-1} \langle \boldsymbol{j}_s \rangle$. On the other hand, in the second part of the E-step (i.e. for j-vectors) no such transformation is required to be applied to the i-vectors, as they are extracted during the current EM iteration.
3. Minimum divergence estimation of $\tilde{\boldsymbol{T}}_c$ is performed in the same way as for $\tilde{\boldsymbol{U}}_c$.

3. Decoupled approach

The coupled method described above was too slow for experimentation. The computation bottleneck is the estimation of the cross-correlation terms between i- and j-vectors in eq. (22). We therefore describe a simplified way of training the model, where we first train the phonetic subspace $\boldsymbol{U}_c$ and then the speaker subspace $\boldsymbol{T}_c$, keeping $\boldsymbol{U}_c$ fixed. The reason for starting with the phonetic subspace is the fact that it exhibits higher variability than the speaker subspace (see Sect. 2.6) and therefore the error introduced by using regular instead of synthetic Baum-Welch statistics is smaller.

Note that a similar decoupling appeared in the development of JFA, where it was shown that the two subspaces could be trained one after the other, leading to a much faster training algorithm [9]. It is also worth mentioning that in JFA the speaker subspace is trained first since the speaker variability is much higher compared to channel variability.

In the decoupled approach we include covariance adaptation, where the covariance matrices $\boldsymbol{\Sigma}_c$ are treated as model parameters and are therefore reestimated during EM algorithm, rather that copied from the UBM. Clearly, covariance adaptation can also be applied to the coupled method. The results using SGMM in speech recognition show that covariance adaptation is beneficial and hence is worth attempting it [5].

3.1. Training the phonetic subspace

The first step is to train a j-vector extractor with covariance adaptation. The equations are similar to those given above, although some modifications are required, mainly due to covariance adaptation. First of all, there are no synthetic Baum-Welch statistics $\boldsymbol{H}_{s,c}$ but instead, $\tilde{\boldsymbol{F}}_{s,c,\cdot}$ are used. Second, the following matrices should be calculated, which will be used for covariance adaptation.

$$\boldsymbol{S}_c = \sum_{t,s,f} Q_t(s,c,f) \left(\boldsymbol{O}_t - \boldsymbol{m}_c\right) \left(\boldsymbol{O}_t - \boldsymbol{m}_c\right)^\top . \tag{31}$$

Moreover, since the covariance matrices $\boldsymbol{\Sigma}_c$ are updated after every EM iteration, the prewhitened Baum-Welch statistics $\tilde{\boldsymbol{F}}_{s,c,\cdot}$ should be updated as well. Therefore, the prewhitening step

$$\tilde{\boldsymbol{F}}_{s,c,f} = \boldsymbol{L}_c^{-1} \boldsymbol{F}_{s,c,f} \tag{32}$$

where $\boldsymbol{L}_c \boldsymbol{L}_c^\top = \boldsymbol{\Sigma}_c$ should be applied in the beginning of every E-step, using the updated covariance matrix $\boldsymbol{\Sigma}_c$.

The E-step is as follows

$$\langle \boldsymbol{j}_s \rangle = \operatorname{Cov}\left(\boldsymbol{j}_s, \boldsymbol{j}_s\right) \sum_c \tilde{\boldsymbol{U}}_c^\top \tilde{\boldsymbol{F}}_{s,c} \tag{33}$$

and the other remaining equations are identical to eq. (13), (15) and (16).

In the M-step, we first perform the update of $\tilde{\boldsymbol{U}}_c$ using the correlation terms defined below

$$\boldsymbol{A}_c = \sum_s N_{s,c,\cdot} \langle \boldsymbol{j}_s \boldsymbol{j}_s^\top \rangle \tag{34}$$

and

$$\boldsymbol{B}_c = \sum_s \tilde{\boldsymbol{F}}_{s,c,\cdot} \langle \boldsymbol{j}_s^\top \rangle . \tag{35}$$

The ML estimate of the subspace is performed as follows

$$\tilde{\boldsymbol{U}}_c = \boldsymbol{B}_c \boldsymbol{A}_c^{-1} . \tag{36}$$

3.2. Covariance adaptation

The next step is the covariance adaptation. The updated covariance matrices are given by the following equation

$$\boldsymbol{\Sigma}_c = \frac{1}{N_{\cdot,c,\cdot}} \left\{ \boldsymbol{S}_c - \boldsymbol{L}_c \boldsymbol{B}_c \tilde{\boldsymbol{U}}_c^\top \boldsymbol{L}_c^\top \right\} . \tag{37}$$

The update rule is derived by setting the derivative of $\mathcal{L}$ with respect to the inverse $\boldsymbol{\Sigma}_c$ equal to zero and using eq. (36). Note that after updating $\boldsymbol{\Sigma}_c$, the subspace $\tilde{\boldsymbol{U}}_c$ as well as $\boldsymbol{B}_c$ should be rotated, since they are defined with respect to whitened Baum-Welch statistics. To do so, we define the following set of matrices:

$$\boldsymbol{Z}_c = \boldsymbol{L}_{c,\tau+1}^{-1} \boldsymbol{L}_{c,\tau} \tag{38}$$

where $\boldsymbol{L}_{c,\tau+1} \boldsymbol{L}_{c,\tau+1}^\top = \boldsymbol{\Sigma}_c$ is derived from the updated covariance matrices in eq. (37) and $\boldsymbol{L}_{c,\tau}$ from the covariance matrices used to prewritten the Baum-Welch statistics in the current iteration. Then, the updated matrices are:

$$\tilde{\boldsymbol{U}}_c \rightarrow \boldsymbol{Z}_c \tilde{\boldsymbol{U}}_c , \;\; \boldsymbol{B}_c \rightarrow \boldsymbol{Z}_c \boldsymbol{B}_c \tag{39}$$

The variational lower bound is as follows

$$\begin{aligned} \mathcal{L} = E_{Q(\underline{\boldsymbol{s}},\underline{\boldsymbol{c}})Q(\underline{\boldsymbol{j}})} \left[\ln P(\underline{\boldsymbol{O}} | \underline{\boldsymbol{s}}, \underline{\boldsymbol{c}}, \underline{\boldsymbol{j}}) \right] - D\left(Q(\underline{\boldsymbol{j}}) \| P(\underline{\boldsymbol{j}})\right) \\ - D\left(Q(\underline{\boldsymbol{s}},\underline{\boldsymbol{c}}) \| P(\underline{\boldsymbol{s}},\underline{\boldsymbol{c}})\right) . \end{aligned} \tag{40}$$

The first term here is the reconstruction error while the last term is fixed so we ignore it. Up to an irrelevant additive constant,

$$\begin{aligned} \mathcal{L} = -\frac{1}{2} \sum_c N_{\cdot,c,\cdot} \left\{ \ln |\boldsymbol{\Sigma}_c| + \operatorname{tr}\left(\boldsymbol{\Sigma}_c^{-1} \boldsymbol{S}_c\right) \right\} \\ + \sum_c \operatorname{tr}\left(\tilde{\boldsymbol{U}}_c^\top \boldsymbol{B}_c - \frac{1}{2} \tilde{\boldsymbol{U}}_c^\top \tilde{\boldsymbol{U}}_c \boldsymbol{A}_c \right) \\ - \sum_s D\left(Q(\boldsymbol{j}_s) \| P(\boldsymbol{j}_s)\right) \end{aligned} \tag{41}$$

and should be estimated after covariance adaptation.

Finally, we apply minimum divergence estimation to ensure that the covariance matrix

$$\frac{1}{S} \sum_s \left\langle \boldsymbol{j}_s \boldsymbol{j}_s^\top \right\rangle \tag{42}$$

is equal to the identity matrix.

3.3. Training the speaker subspace

To train the speaker subspace, we need to implement a regular i-vector extractor. The only difference is the use of synthetic Baum-Welch statistics, i.e.

$$\boldsymbol{G}_{c,f} = \tilde{\boldsymbol{F}}_{\cdot,c,f} - \sum_s N_{s,c,f}\tilde{\boldsymbol{U}}_c\langle \boldsymbol{j}_s \rangle. \quad (43)$$

Note that prewhitening should be applied using the covariance matrices estimated during the j-vector extractor training. We do not update the covariance matrices when training the i-vector extractor.

4. Experimental Results

4.1. Set-up of the experiment

The results are based on NIST-SRE 2010 speaker recognition benchmark. At this stage we only focus on female, telephone speech (condition 5). We trained a diagonal 2048-component UBM (i.e. $C = 2048$) and all the subspace models on the female utterances of Fisher, Mixer ('04, '05, '06 and '08) and Switchboard (about 42K utterances overall). As front-end features we used 60-dimensional Gaussianized MFCC (i.e. $F = 60$), and for Voice Activity Detector we used the two-GMM approach described in [2].

We trained a feed-forward DNN on about 1300 hours of speech (English telephone speech from Switchboard, Callhome and Fisher) using Kaldi [10]. The DNN is a 5-hidden layer sigmoid network with 368-dimensional input, 1000 neurons per hidden layer and $S = 3925$ output nodes. The input features are known as TRAP and they are computed as follows: For each frame we compute 23-dimensional filter-bank features and for each filter-bank dimension we apply the cosine transform over 31 frames (current frame +/- 15 frames). By keeping only the first 16 coefficients, we end up with 23*16 = 368 dimensional features.

All i-vectors are 400-dimensional (i.e. $R = 400$) while j-vectors are 50-dimensional (i.e. $\Lambda = 50$). Length normalization is applied to the i-vectors with WCCN. For backend, the standard PLDA is used with full residual covariance and 150-dimensional speaker factors.

4.2. Experimental Results

The methods we compare are (a) i-vectors with diagonal UBM (line 1), (b) i-vectors with full UBM and covariance adaptation (line 2), (c) the coupled approach presented in Section II without covariance adaptation (line 3), and (d) the decoupled approach presented in Section III with covariance adaptation (line 4). The results on the NIST-SRE 2010 (cond. 5 - core-extended, female part) are given in Table 1

Model	Cov	EER (%)	minDCF$_{08}$	minDCF$_{10}$
i	fixed	**2.32**	0.131	0.449
i	adapt	2.46	**0.129**	0.455
ij-c	fixed	2.53	0.137	**0.438**
ij-d	adapt	2.65	0.134	0.462

Table 1: *Comparison between regular i-vectors and the two proposed methods using full-length utterances (female). We denote the proposed coupled and decoupled methods by ij-c and ij-d, respectively.*

Clearly, all systems perform very similarly, meaning that the proposed methods did not succeed in improving the i-vector benchmark results. We repeated the same set of experiments, but this time with truncated test utterances (randomly between 10 and 30 sec). No truncation was applied to the enrollment utterances. The results are given in Table 2.

Model	Cov	EER (%)	minDCF$_{08}$	minDCF$_{10}$
i	fixed	3.84	0.214	0.666
i	adapt	**3.62**	**0.206**	**0.629**
ij-c	fixed	3.79	0.214	0.649
ij-d	adapt	3.82	0.209	0.658

Table 2: *Comparison between regular i-vectors and the two proposed methods using truncated test utterances (female).*

We observe again that the proposed method did not manage to produce better results than the standard i-vectors. It is interesting to note that the use of covariance adaptation in the standard i-vectors yields slightly but consistently better results compared to the other methods.

As a final set of experiment, we fuse the two proposed methods with the i-vector system with covariance adaptation. The fusion is a simple averaging (with equal weights) of the Log-Likelihood Ratios (LLRs), after standardizing their distributions. The results are given in Table 3.

Model	Cov	EER (%)	minDCF$_{08}$	minDCF$_{10}$
ij-c	fixed	3.43	0.196	0.617
ij-d	adapt	3.48	0.199	0.635

Table 3: *Fusion results between the two proposed methods and regular i-vectors using truncated test utterances (female).*

A modest improvement can be observed in both cases, with the coupled method attaining slightly better results.

5. Conclusions

In this paper, an alternative way of using DNN posteriors was explored. Inspired by the SGMM framework of speech recognition, we developed a similar approach that is in line with the successful probabilistic framework of Joint Factor Analysis and i-vector extractor. Our motivation was to treat phonetic variability in a similar way the channel variability is treated in JFA, and extract i-vectors that are less sensitive to the phonetic content. The first algorithm that we developed is based on the joint estimation of the two subspaces, it assumes a single generative model and it makes use of a single optimization criterion (variational lower bound). As the training of the model was rather slow for experimentation, we developed a decoupled approach, where the phonetic subspace was trained first (using regular Baum-Welch statistics) followed by the training of the speaker subspace. The decoupling was inspired by JFA, where a similar decoupling has been successfully applied. Finally, covariance adaptation was included, which is a part of the original recipe of both JFA and SGMM.

We have also tried to overcome some limitations that a more straightforward application of SGMMs is speaker recognition exhibits [6]. Therefore, apart from the probabilistic framework, the proposed method allows us to make use of both transcribed and untranscribed speaker recognition resources for training

(e.g. Mixer corpus). Moreover, it enables us to work with state-of-the-art DNN frame-posteriors in both training and evaluation sets, rather than with those derived by the (suboptimal in terms of speech recognition) SGMM.

The experiments that we conducted on NIST-2010 were clearly unsuccessful. Neither of the methods we attempted managed to demonstrate its capacity and no gains over the regular i-vectors were attained. Furthermore, the LLRs that we obtained were so correlated to those of the regular i-vectors that score fusion led to minor improvements. However, based on the good results demonstrated in [6] using SGMM and the fact that our experimentation was far from being exhaustive, we believe that the method has the potential to show better performance in the future.

6. References

[1] Y. Lei, N. Scheffer, L. Ferrar, and M. McLaren, "A novel scheme for speaker recognition using a phonetically aware deep neural network," in *Proc. ICASSP*, Florence, Italy, May 2014, pp. 1695–1699.

[2] P. Kenny, V. Gupta, T. Stafylakis, P. Ouellet, and J. Alam, "Deep Neural Networks for extracting Baum-Welch statistics for Speaker Recognition," in *Proc. Odyssey Speaker and Language Recognition Workshop*, Joensuu, Finland, June 2014, pp. 1–8.

[3] M. McLaren, Y. Lei, and L. Ferrer, "Advances in deep neural network approaches to speaker recognition," in *Proc. ICASSP*, Brisbane, Australia, Apr. 2015.

[4] Hao Zheng, Shanshen Zhang, and Wenju Liu, "Exploring Robustness of DNN/RNN for Extracting Speaker Baum-Welch Statistics in Mismatched Conditions," in *Proc. Interspeech*, Dresden, Germany, Sept. 2015.

[5] Daniel Povey, Lukas Burget, Mohit Agarwal, et al., "The subspace Gaussian mixture model – a structured model for speech recognition," *Computer Speech and Language*, 2011.

[6] Petr Motlicek, Subhadeep Dey, Srikanth Madikeri, and Lukas Burget, "Employment of Subspace Gaussian Mixture Models in Speaker Recognition," in *Proc. ICASSP*, Brisbane, Australia, Apr. 2015.

[7] Themos Stafylakis, Patrick Kenny, Jahangir Alam, and Marcel Kockmann, "Speaker and channel factors in text-dependent speaker recognition," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 24, no. 1, pp. 65–78, 2016.

[8] Patrick Kenny, Themos Stafylakis, Jahangir Alam, Vishwa Gupta, and Marcel Kockmann, "Uncertainty Modeling Without Subspace Methods For Text-Dependent Speaker Recognition," in *Proc. Odyssey*, Bilbao, Spain, June 2016.

[9] P. Kenny, G. Boulianne, P. Ouellet, and P. Dumouchel, "Factor analysis simplified," in *Proc. ICASSP 2005*, Philadelphia, PA, Mar. 2005, pp. 637–640.

[10] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, Jan Silovsky, Georg Stemmer, and Karel Vesely, "The kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. Dec. 2011, IEEE Signal Processing Society, IEEE Catalog No.: CFP11SRW-USB.