Non-Parallel Voice Conversion Using I-Vector PLDA: Towards Unifying Speaker Verification and Transformation

NON-PARALLEL VOICE CONVERSION USING I-VECTOR PLDA: TOWARDS UNIFYING SPEAKER VERIFICATION AND TRANSFORMATION

Tomi Kinnunen[1]*, Lauri Juvela*[2]*, Paavo Alku*[2]*, Junichi Yamagishi*[3,4]

[1]University of Eastern Finland, School of Computing, Finland
[2]Aalto University, Department of Signal Processing and Acoustics, Finland
[3]National Institute of Informatics, Japan
[4]University of Edinburgh, The Centre for Speech Technology Research, United Kingdom
mailto:tkinnu@cs.uef.fi, lauri.juvela@aalto.fi, jyamagis@nii.ac.jp

ABSTRACT

Text-independent speaker verification (recognizing speakers regardless of content) and non-parallel voice conversion (transforming voice identities without requiring content-matched training utterances) are related problems. We adopt i-vector method to voice conversion. An i-vector is a fixed-dimensional representation of a speech utterance that enables treating voice conversion in utterance domain, as opposed to frame domain. The high dimensionality (800) and small number of training utterances (24) necessitates using prior information of speakers. We adopt probabilistic linear discriminant analysis (PLDA) for voice conversion. The proposed approach requires neither parallel utterances, transcriptions nor time alignment procedures at any stage.

Index Terms— Voice conversion, i-vector, non-parallel training

1. INTRODUCTION

Voice conversion (VC) [1] and *automatic speaker verification* (ASV) [2] are both concerned with modeling individual variation in speech. Although the former is a regression task and the latter a classification task, both VC [3, 4] and ASV [5, 6] involve extensive use of Gaussian mixture models (GMMs). One of the high-level main differences, however, is that while the modeling unit in a typical VC system is the short-term speech frame, in ASV it is the full utterance. At the training stage of a typical VC system, one learns a source-to-target conversion function based on acoustically aligned feature frames. As the paired frames share the same underlying phone, the conversion function learns to transform the speaker characteristics. Similarly, the objective metrics to assess voice conversion performance typically involve computing spectral distortion between the converted and the target frames (after alignment).

In contrast to the frame-level processing, modern ASV systems make inferences of speaker identity based on a higher level modeling unit, pair of *utterances*. One utterance represents the training (or enrolment) utterance and the other one is the test utterance. Each one of them is represented using a single *i-vector* [6] or other fixed-dimensional representation. An i-vector is essentially a low-dimensional parameterization of a GMM that represents a specific speech utterance. Even though feature frames are needed in extracting the i-vector, all the subsequent processing relies on i-vectors only. The simplest approach to assess speaker similarity of a given pair of i-vectors is to compute their angle, or cosine similarity, requiring no training. Alternatively, one may train a back-end classifier using a set of development i-vectors. The generative *probabilistic linear discriminant analysis* (PLDA) [7] has been particularly successful in speaker and language recognition tasks [8, 9].

We advocate the use of i-vector representation as the basic unit for voice conversion. I-vectors have been used for speaker adaptation in deep neural network (DNN) based speech synthesis [10], but our perspective to voice conversion is new. The primary motivation is that an i-vector can be extracted in an unsupervised way regardless of the utterance duration, speaker or content, which opens up new possibilities for both non-parallel and parallel data scenarios. While parallel training data usually leads to high conversion quality and speaker similarity, the requirement of a parallel corpus severely limits the practical application scope. Even for parallel data, the quality of frame alignment is important [11] but obtaining these in practice involves coping with speaker differences, one-to-many and many-to-one mappings and varied speaking rates, to mention a few problems.

There have been a few attempts to use non-parallel data for VC. For instance, [12] proposes a GMM-based technique to learn a relationship between reference speakers in advance and using the relationship for a new speaker. Another non-parallel VC technique, similar to our i-vector approach, uses *eigenvoices* [13]. The eigenvoice approach performs two mappings: the first one is from the source speaker to an eigenvoice (or average voice) trained from reference speakers and the second one is from the eigenvoice to the target speaker. Thus, even if these approaches do not require parallel training data from the source and the target speakers, they do require parallel databases of the reference speakers (for an opposite case, see [14]). As opposed to the eigenvoice approach, our method does not require any parallel data, any stage of the process. While our method is not the first one to fulfill this property (e.g. [15]), it is inspired from ASV, including the use of conventional mel-frequency cepstral coefficients for vocoding.

2. BACKGROUND ON I-VECTORS AND PLDA

2.1. I-vectors

Let F denote the dimensionality of one spectral feature vector and let $\boldsymbol{\theta}_{\mathrm{ubm}} = \{\boldsymbol{m}_c, \boldsymbol{\Sigma}_c, w_c : c = 1, \ldots, C\}$ denote a *universal back-*

The work was funded by Academy of Finland (proj. 288558). The paper also reflects some results from the OCTAVE Project (#647850), funded by the Research European Agency (REA) of the European Commission, in its framework programme Horizon 2020. The views expressed in this paper are those of the authors and do not engage any official position of the European Commission.

ground model (UBM) [5] with C Gaussians parameterized by their mean vectors $\boldsymbol{m}_c \in \mathbb{R}^F$, covariance matrices $\boldsymbol{\Sigma}_c \in \mathbb{R}^{F\times F}$ and prior probabilities π_c, with $\sum_c \pi_c = 1$. In the i-vector model [6] one assumes that, for a given utterance U, its acoustic features aligned with the c^{th} mixture component are distributed as $\mathcal{N}(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$ with $\boldsymbol{\mu}_c = \boldsymbol{m}_c + \mathbf{T}_c\boldsymbol{w}$, where $\boldsymbol{w}$ is an R-dimensional random latent vector with a standard normal prior and $\mathbf{T}_c$ is an $F \times R$ matrix. Alternatively, $\boldsymbol{\mu} = \boldsymbol{m} + \mathbf{T}\boldsymbol{w}$. Here $\boldsymbol{\mu}$ and $\boldsymbol{m}$ are the GMM mean supervectors obtained by stacking all the C means into a $(C\times F)$-dimensional GMM mean supervector and $\mathbf{T}$ is a block-diagonal matrix consisting of $\mathbf{T}_c$'s. This factor loading matrix is estimated off-line using a large number of speech utterances. The *i-vector*, indicated here by $\boldsymbol{\phi}$, is the mean of the posterior distribution of $\boldsymbol{w}$, $\boldsymbol{\phi} = \mathbb{E}[\boldsymbol{w}|U]$ computed via the so-called Baum-Welch statistics of the utterance. For further details, we point the interested reader to [6, 16, 17].

2.2. Probabilistic LDA

Unlike the *joint factor analysis* (JFA) model [18], the i-vector model does not distinguish between speaker and other signal variations. Thus, the unwanted variation not related to speaker identity has to be compensated at back-end. To this end, *probabilistic linear discriminant analysis* (PLDA) model [7] is the usual choice in speaker and language recognition. In this study, we use PLDA to transform speaker characteristics in the i-vector space.

Besides the original formulation in [7], there are other flavors to PLDA, including *simplified* [8] and *two-covariance* [9] PLDA models (for a comparison and scalable implementations of all the three, refer to [19]). We adopt the simplified PLDA which assumes the following decomposition for the j^{th} i-vector of the i^{th} speaker, denoted by $\boldsymbol{\phi}_{i,j}$:

$$\boldsymbol{\phi}_{i,j} = \boldsymbol{b} + \mathbf{S}\boldsymbol{y}_i + \boldsymbol{\varepsilon}_{i,j}, \tag{1}$$

where $\boldsymbol{b} \in \mathbb{R}^R$ is a global mean parameter, $\mathbf{S} \in \mathbb{R}^{R\times Q}$ matrix that spans between-speaker space ($Q \leq R$), $\boldsymbol{y}_i \in \mathbb{R}^Q$ the latent speaker variable with a standard normal prior and $\boldsymbol{\varepsilon}_{i,j} \in \mathbb{R}^R$ a residual vector with prior distribution $\mathcal{N}(\mathbf{0}, \boldsymbol{\Lambda})$, with a full covariance matrix $\boldsymbol{\Lambda}$. The PLDA hyperparameters, $\boldsymbol{\theta}_{\text{plda}} = \{\boldsymbol{b}, \mathbf{S}, \Lambda\}$, are trained off-line using speakers that are disjoint from any speakers a speaker verification (or as here, voice conversion) system will observe in future.

3. VOICE CONVERSION USING I-VECTORS AND PLDA

According to (1), for a given speaker, we assume $\boldsymbol{y}_i$ to be fixed while the variations in the different utterances of that speaker are explained by the residual variability $\boldsymbol{\varepsilon}_{i,j}$. This within-speaker variation in i-vectors would be caused by a number of factors or their combinations such as differences in speech content, articulation, or microphones to name a few. From the viewpoint of voice conversion, the main concern is to convert $\boldsymbol{y}_i$ only while retaining all the other characteristics of the source recording (absorbed in the residual). To explain the process, we first detail how to estimate $\boldsymbol{y}_i$ from a set of i-vectors from a particular speaker.

3.1. Estimating speaker latent variable

Let $\boldsymbol{\Phi}_i = \{\boldsymbol{\phi}_{i,1}, \ldots \boldsymbol{\phi}_{i,N_i}\}$ denote a collection of i-vectors from speaker i (either source or target). According to the simplified PLDA model, the distribution of i-vectors from this speaker is $p(\boldsymbol{\phi}_{i,j}|\boldsymbol{y}_i) = \mathcal{N}(\boldsymbol{b}+\mathbf{S}\boldsymbol{y}_i, \boldsymbol{\Lambda})$ and we view the i-vectors in $\boldsymbol{\Phi}_i$ being independently drawn random samples from that distribution. Since in a typical voice conversion setting, the number of utterances N_i would be a few tens only while the dimension of both i-vectors and the latent vector several hundreds, robust *maximum likelihood* (ML) estimates of $\boldsymbol{y}_i$ are difficult to obtain. We therefore use *maximum a posteriori* (MAP) estimates instead. Following the simplified PLDA model, we assume a standard normal prior for $\boldsymbol{y}_i$. Thus,

$$\begin{aligned}\hat{\boldsymbol{y}}_i^{\text{MAP}} &= \arg\max_{\boldsymbol{y}_i} \prod_{j=1}^{N_i} \mathcal{N}\left(\boldsymbol{\phi}_{i,j}|\boldsymbol{b}+\mathbf{S}\boldsymbol{y}_i, \boldsymbol{\Lambda}\right)\mathcal{N}(\boldsymbol{y}_i|\mathbf{0},\mathbf{I}) \\ &= \arg\max_{\boldsymbol{y}_i} \left\{\sum_{j=1}^{N_i} \log\mathcal{N}\left(\boldsymbol{\phi}_{i,j}|\boldsymbol{b}+\mathbf{S}\boldsymbol{y}_i, \boldsymbol{\Lambda}\right) + \log\mathcal{N}(\boldsymbol{y}_i|\mathbf{0},\mathbf{I})\right\}\end{aligned}$$

Setting the partial derivatives with respect to $\boldsymbol{y}_i$ to zero and rearranging the terms leads to the following solution:

$$\hat{\boldsymbol{y}}_i^{\text{MAP}} = \left(N_i\mathbf{S}^{\mathsf{T}}\boldsymbol{\Lambda}^{-1}\mathbf{S} + \mathbf{I}\right)^{-1}\mathbf{S}^{\mathsf{T}}\boldsymbol{\Lambda}^{-1}\boldsymbol{f}_i, \tag{2}$$

where $\boldsymbol{f}_i = \sum_{j=1}^{N_i}\tilde{\boldsymbol{\phi}}_{i,j}$ is the first-order sufficient statistic vector of the centered training i-vectors, $\tilde{\boldsymbol{\phi}}_{i,j} = \boldsymbol{\phi}_{i,j} - \boldsymbol{b}$.

Note that (2) requires only the first-order sufficient statistics of the training i-vectors ($\boldsymbol{f}_i$). Further, the matrix that multiplies $\boldsymbol{f}_i$ from the left involves terms that depend only on PLDA hyperparameters or N_i, enabling pre-computation for efficient implementation. While (2) extracts a single speaker latent vector for a given i-vector collection, we may also visualize the latent vectors extracted from individual i-vectors (i.e. $N_i = 1$). Fig. 1 shows an example computed from actual data involving 25 utterances (i-vectors) per speaker. The four speakers are well-separated in the speaker latent space.

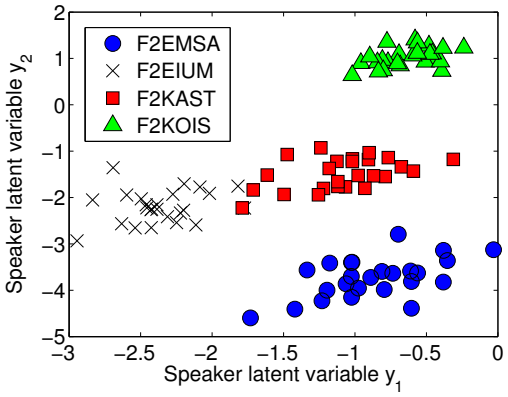

Fig. 1. Illustration of speaker latent variables from four speakers in the APP corpus. Each point corresponds to one utterance represented by 800-dimensional raw i-vector, reduced down to 2-d latent speaker variable y using simplified PLDA.

ALGORITHM 1: Voice conversion using i-vector PLDA

1. **Off-line stage:**
   (a) Train a universal background model (UBM), $\boldsymbol{\theta}_{\text{ubm}}$
   (b) Train an i-vector extractor, $\mathbf{T}$
   (c) Using $\boldsymbol{\theta}_{\text{ubm}}$ and $\mathbf{T}$, extract a set of development i-vectors, $\mathcal{D} = \{\boldsymbol{\phi}_n : n = 1, \ldots, N_{\text{dev}}\}$
   (d) Using $\mathcal{D}$, Train a probabilistic LDA (PLDA) model, $\boldsymbol{\theta}_{\text{plda}} = \{\boldsymbol{b}, \mathbf{S}, \boldsymbol{\Lambda}\}$

2. **Non-parallel source to target training:**

   (a) Using $\boldsymbol{\theta}_{\text{ubm}}$ and $\mathbf{T}$, extract source $\boldsymbol{\Phi}_{\text{src}} = \{\boldsymbol{\phi}_n : n = 1, \ldots, N_s\}$ and target $\boldsymbol{\Phi}_{\text{tar}} = \{\boldsymbol{\phi}_m : m = 1, \ldots, N_t\}$ training i-vectors, one per utterance.

   (b) Using $\boldsymbol{\Phi}_{\text{src}}$ and $\boldsymbol{\theta}_{\text{plda}}$, obtain a MAP estimate of the source speaker latent vector, $\hat{\boldsymbol{y}}_{\text{src}}$ using (2). Similarly, using $\boldsymbol{\Phi}_{\text{tar}}$ and $\boldsymbol{\theta}_{\text{plda}}$, obtain $\hat{\boldsymbol{y}}_{\text{tar}}$ independently.

   (c) The trained conversion model in the i-vector space is $f(\boldsymbol{\phi}) = \boldsymbol{\phi} + \mathbf{S}\,(\hat{\boldsymbol{y}}_{\text{tar}} - \hat{\boldsymbol{y}}_{\text{src}})$, where $\boldsymbol{\phi}$ denotes a new source speaker i-vector.

3. **Conversion stage:**

   (a) Given a source speaker utterance represented via a sequence of short-term features $\mathcal{X} = \{\boldsymbol{x}_1 \ldots, \boldsymbol{x}_T\}$, extract a single i-vector $\boldsymbol{\phi}_{\text{src}}$ and obtain an estimate of the corresponding target i-vector, $\hat{\boldsymbol{\phi}}_{\text{tar}} = f(\boldsymbol{\phi}_{\text{src}})$

   (b) Obtain the Gaussian means, $\boldsymbol{\mu}_c^{\text{src}} = \boldsymbol{m}_c + \mathbf{T}_c\boldsymbol{\phi}_{\text{src}}$ and $\boldsymbol{\mu}_c^{\text{tar}} = \boldsymbol{m}_c + \mathbf{T}_c\hat{\boldsymbol{\phi}}_{\text{tar}}$, where $c = 1, \ldots, C$

   (c) Find probabilistic alignment of each $\boldsymbol{x}_t$ against the source GMM, $P(c|\boldsymbol{x}_t) = \pi_c\mathcal{N}(\boldsymbol{x}_t|\boldsymbol{\mu}_c^{\text{src}}, \boldsymbol{\Sigma}_c)/p(\boldsymbol{x}_t)$, where $p(\boldsymbol{x}_t) = \sum_\ell \pi_\ell\mathcal{N}(\boldsymbol{x}_t|\boldsymbol{\mu}_\ell^{\text{src}}, \boldsymbol{\Sigma}_\ell)$.

   (d) Convert each source frame $t = 1, \ldots, T$ as:

$$\hat{\boldsymbol{y}}_t = \boldsymbol{x}_t + \sum_{c=1}^{C} P(c|\boldsymbol{x}_t)\left(\boldsymbol{\mu}_c^{\text{tar}} - \boldsymbol{\mu}_c^{\text{src}}\right) \quad (3)$$

3.2. Voice conversion

Our voice conversion model is conceptually extremely simple. With the tools developed above, we first find the MAP estimates of speaker latent variables from the training i-vectors of the source and the target speakers; denote them by $\hat{\boldsymbol{y}}_{\text{src}}$ and $\hat{\boldsymbol{y}}_{\text{tar}}$. Then, at the conversion stage, a new utterance produced by the source speaker decomposes according to the simplified PLDA model as $\boldsymbol{\phi}_{\text{src}} = \boldsymbol{b} + \mathbf{S}\hat{\boldsymbol{y}}_{\text{src}} + \boldsymbol{e}_{\text{src}}$ where $\hat{\boldsymbol{\phi}}_{\text{src}}, \boldsymbol{b}, \mathbf{S}$ and $\boldsymbol{y}_{\text{src}}$ are now all known, and the specific residual vector is therefore $\boldsymbol{e}_{\text{src}} = \boldsymbol{\phi}_{\text{src}} - \boldsymbol{b} - \mathbf{S}\hat{\boldsymbol{y}}_{\text{src}}$. By replacing the source latent variable with the target latent variable and using the source residual, we can now "synthesize" the corresponding target i-vector as $\hat{\boldsymbol{\phi}}_{\text{tar}} = \boldsymbol{b} + \mathbf{S}\hat{\boldsymbol{y}}_{\text{tar}} + \boldsymbol{e}_{\text{src}} = \boldsymbol{\phi}_{\text{src}} + \mathbf{S}(\hat{\boldsymbol{y}}_{\text{tar}} - \hat{\boldsymbol{y}}_{\text{src}})$. We then recover the individual Gaussian means. The original source utterance feature vectors are then aligned with respect to the source GMM (represented by the i-vector) and shifted with respect to the mean vectors. The entire process is summarized in Algorithm 1.

3.3. MFCC-based vocoding

In the current study, we want to keep a close connection to the ASV framework, where MFCCs are used to represent speech spectral information. Generally, MFCCs can not uniquely be inverted back to the discrete Fourier transform (DFT) domain and are not designed for use in speech synthesis. Nevertheless, we present here a feasible method to recover approximate spectral content of the MFCCs. The conventional MFCC vector $\boldsymbol{x}$ is given by

$$\boldsymbol{x} = \mathbf{D}^{-}\log(\mathbf{H}\boldsymbol{X}), \quad (4)$$

where $\boldsymbol{X}$ is the DFT power spectrum, $\mathbf{H}$ is the triangular mel filterbank matrix with filter centers spaced linearly on the mel-scale,

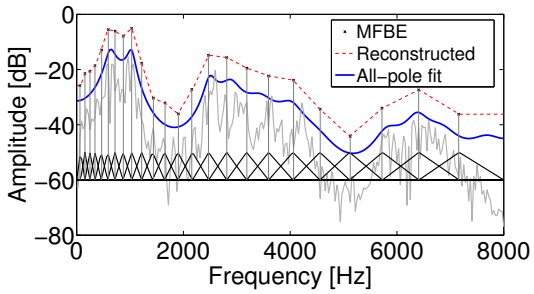

Fig. 2. The MFCC-AR fitting process. Mel filterbank energies (MFBE) are interpreted as samples of a full-length warped spectrum that is reconstructed by interpolation. The reconstruction is warped back to linear frequency domain and all-pole filter is fitted.

and $\mathbf{D}^{-}$ is a combined discrete cosine transform and liftering (truncation) operation. It is straightforward to inverse this to the mel filterbank energies (MFBE) are given by $\hat{\boldsymbol{x}} = \mathbf{H}\boldsymbol{X} = \exp(\mathbf{D}^{+}\boldsymbol{x})$, where $\mathbf{D}^{+}$ combines zero padding and inverse DCT. The MFBEs can be interpreted as a low resolution version of a full-length mel-spectrum sampled at the filterbank centers [20]. This is illustrated in Fig. 2. The reconstruction is then performed as upsampling by linear interpolation, conveniently given by a warped version of the triangular filterbank matrix $\tilde{\mathbf{H}}$. Spectrum interpolation is performed in log-domain, as suggested in [21]. Additionally, a normalization factor is applied to compensate for the non-unitarity of $\mathbf{H}$.

$$\boldsymbol{X}_j \approx \frac{1}{\sum_i(\mathbf{H}^{\mathsf{T}}\mathbf{H})_{i,j}}\exp(\tilde{\mathbf{H}}^{\mathsf{T}}\log(\hat{\boldsymbol{x}}))_j, \quad (5)$$

An all-pole filter, denoted MFCC-AR, is finally fitted to the reconstructed spectrum by calculating the autocorrelation coefficients with inverse DFT and solving the resulting normal equations with the Levinson-Durbin algorithm. In the i-vector-based conversion the MFCCs of the source utterance are replaced with converted ones. The MFCC-AR of the source utterance are used to inverse filter the signal, producing a residual excitation signal. The pitch of the MFCC-AR excitation is affine transformed so that the mean and variance of the source speaker $\log f_0$ match those of the target. Pitch-synchronous overlap-add is used for the transformation.

As a baseline method, we use *vocal tract length normalisation* (VTLN) that can also be estimated from non-parallel data. First, a bilinear transform warping that minimizes the KL-divergence between the mel-scale long term average spectra of source and target speaker is found using a grid search. Then, at conversion, the MFCC-AR spectra of the source utterance are warped using the estimated warping parameter. All transformations on the MFCC-AR excitation are done identically to the i-vector method.

4. EXPERIMENTAL SET-UP

All the experiments are carried out on a subset of APP corpus of Japanese speech. All the speakers throughout our experiments are females. Our **hyperparameter training part** defines data for training the off-line components needed before training any source-to-target conversions. We use 7552 utterances from 300 unique speakers to train the UBM and 18,447 utterances from 734 unique speakers to train the T-matrix. The same 18,447 utterances are also re-used for PLDA training. Our i-vectors are 800-dimensional and, unlike in

speaker verification, we do not apply any whitening or length normalization (the latter would not be invertible, a property that we will need in this work).

Voice conversion part, which is speaker disjoint from the hyperparameter training set, consist of the definitions of source and target speaker utterances. It is further divided into development (dev) and evaluation (eval) parts. The former serves for control parameter optimization of the PLDA parameters and contains in total 7500 conversions from 300 speaker pairs. The evaluation part, in turn, is supposed to be executed with the optimized control parameters and including production of some speech waveforms and perceptual experiments. It contains 3 speaker pairs.

Each speaker of the APP corpus has only 25 utterances. We use a leave-one-out strategy: one test i-vector is held out for testing and the i-vector mapper is trained using the remaining utterances. In **parallel training data** scenario, the contents of training utterances for the source and the target are the same whereas in **non-parallel training data** scenario, they are completely different. In both cases, however, the test utterance set is the same. Since we wish to compare the non-parallel and parallel training data scenarios, we are limited to a maximum of 12 training utterances[1], while in the matched training scenario, we can utilize up to all the 24 utterances.

To optimize the PLDA system, we measure the distance between any two GMMs implied by their corresponding i-vectors. These two GMMs correspond either to the (source, target) pair without any conversion or (converted, target) pair. The natural distance between two densities $p_1(\boldsymbol{x})$ and $p_2(\boldsymbol{x})$ is the Kullback-Leibler (KL) divergence, $\int_{\mathbb{R}^F} p_1(\boldsymbol{x}) \left[\log p_1(\boldsymbol{x}) - \log p_2(\boldsymbol{x})\right] d\boldsymbol{x}$, which cannot be computed in closed form for GMMs. A common workaround in speaker verification is to compute instead the following upper bound between the adapted GMMs [22], $D = \sum_{c=1}^{C} \pi_c \left(\boldsymbol{\mu}_c^1 - \boldsymbol{\mu}_c^2\right) \boldsymbol{\Sigma}_c^{-1} \left(\boldsymbol{\mu}_c^1 - \boldsymbol{\mu}_c^2\right)$, where $\boldsymbol{\mu}_c^j$ is the c^{th} mean vector of the j^{th} GMM. Optimizing parameters following the upper bound pulls down the KL divergence. We further normalize D by the number of Gaussians, C.

5. EXPERIMENTAL RESULTS

Our first experiment on the voice conversion dev set is summarized in Fig. 3. The *No conversion* line indicates the distance between the unconverted source and target GMMs, while the two other curves show the training and test error for the proposed method as a function of the speaker latent subspace. All the three error curves are averages of the 7500 individual test conversions. As we can see, the distances are greatly reduced compared to not doing any conversion and unsurprisingly, the training error is lower than the test error. Both errors decrease monotonically as a function of the dimensionality of $\boldsymbol{y}$, though there is not much improvement after 200 or 300 dimensions.

Our second objective analysis, shown in Fig. 4, shows the effect of training set size for both non-parallel and parallel training data conditions, for two different dimensionalities of $\boldsymbol{y}$. The distances decrease with larger number of training i-vectors, as expected. The most interesting observation, however, is that while the training error for matched-text case is lower than for non-matched case, the *test errors* in both cases are virtually identical. Thus, having or not having parallel data has no impact to test utterance conversion, on average.

Finally, the speaker similarity was evaluated by examining KL-divergences of mel-scale LTAS calculated from the voice converted

[1]From the 25 utterances, 1 is kept aside as a test case while the remaining 24 are partitioned into two content-disjoint sets in the case of non-matched training scenario.

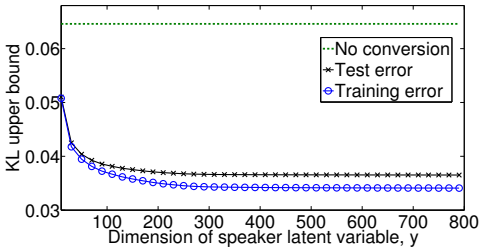

Fig. 3. Training of PLDA parameters. Here the training utterances are matched in content. The graphs are averaged results from a total of 7500 conversion pairs from 300 speaker pairs.

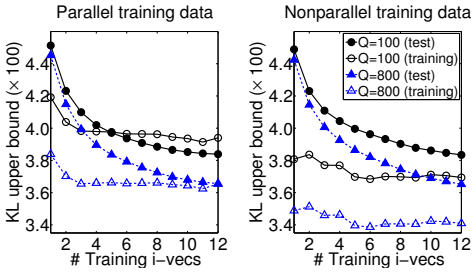

Fig. 4. Effect of the training set parameters.

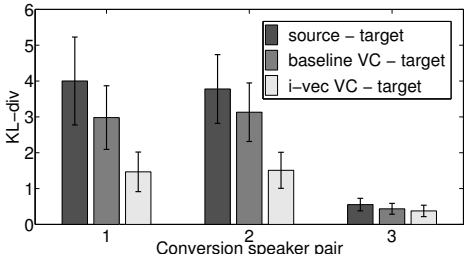

Fig. 5. Average LTAS KL-divergences to target for the baseline (T-b) and I-vector (T-i) methods, and the no conversion (T-S) case.

waveforms on the evaluation part consisting of 3 speaker pairs. For this experiment, we use parallel training set-up and increase the number of training utterances to 24 as explained above. Fig. 5 shows the average KL-divs. from the target to (1) source, (2) i-vector converted utterances and (3) baseline converted speech, where plots are grouped by speaker pairs. The baseline vocal tract normalization (baseline VC - target) somewhat reduces the divergence compared to the source - target case, while the proposed method (i-vec VC - target) consistently gives lower divergences than baseline.

6. CONCLUSION

Our i-vector based voice conversion approach was inspired from text-independent speaker verification: it requires no parallel data, transcripts or frame alignment at any stage. On our data, equivalent speaker similarity was obtained irrespectively whether the training data was parallel or not. Our near future plan includes perceptual experiments and as well as ASV experiments to evaluate speaker similarity using different approaches. There is much further scope to study links between ASV and VC systems.

7. REFERENCES

[1] Y. Stylianou, "Voice transformation: A survey," in *Proc. Int. conference on acoustics, speech, and signal processing (ICASSP 2009)*, Taipei, Taiwan, April 2009, pp. 3585–3588.

[2] T. Kinnunen and H. Li, "An overview of text-independent speaker recognition: from features to supervectors," *Speech Communication*, vol. 52, no. 1, pp. 12–40, January 2010.

[3] Y. Stylianou, O. Cappé, and E. Moulines, "Continuous probabilistic transform for voice conversion," *IEEE Trans. on Speech and Audio Processing*, vol. 6, no. 2, pp. 131–142, March 1998.

[4] D. Matrouf, J.-F. Bonastre, and C. Fredouille, "Effect of speech transformation on impostor acceptance," in *Proc. ICASSP*, Toulouse, France, May 2006, pp. 933–936.

[5] D.A. Reynolds, T.F. Quatieri, and R.B. Dunn, "Speaker verification using adapted gaussian mixture models," *Digital Signal Processing*, vol. 10, no. 1, pp. 19–41, January 2000.

[6] N. Dehak, P.J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Trans. Audio, Speech and Language Processing*, vol. 19, no. 4, pp. 788–798, May 2011.

[7] S. J. D. Prince and J. H. Elder, "Probabilistic linear discriminant analysis for inferences about identity," in *ICCV*, 2007.

[8] P. Kenny, "Bayesian speaker verification with heavy-tailed priors," in *Odyssey 2010: The Speaker and Language Recognition Workshop*, Brno, Czech Republic, June 2010, p. 14.

[9] N. Brümmer and E. de Villiers, "The speaker partitioning problem," in *Odyssey 2010: The Speaker and Language Recognition Workshop, Brno, Czech Republic, June 28 - July 1, 2010*, 2010, p. 34.

[10] Z. Wu, P. Swietojanski, C. Veaux, S. Renals, and S. King, "A study of speaker adaptation for DNN-based speech synthesis," in *Proc. Interspeech*, Dresden, Germany, 2015, pp. 879–883.

[11] E. Helander, J. Schwarz, J. Nurminen, H. Silen, and M. Gabbouj, "On the impact of alignment on voice conversion performance," 2008.

[12] A. Mouchtaris, J. Van der Spiegel, and P. Mueller, "Nonparallel training for voice conversion based on a parameter adaptation approach," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 3, pp. 952–963, May 2006.

[13] T. Toda, Y. Ohtani, and K. Shikano, "Eigenvoice conversion based on Gaussian mixture model," in *Annual Conference of the International Speech Communication Association*, 2006.

[14] Z. Wu, T. Kinnunen, E.S. Chng, and H. Li, "Mixture of factor analyzers using priors from non-parallel speech for voice conversion," *IEEE Signal Process. Lett.*, vol. 19, no. 12, pp. 914–917, 2012.

[15] T. Nakashika, T. Takiguchi, and Y. Minami, "Non-parallel training in voice conversion using an adaptive restricted boltzmann machine," *IEEE/ACM Trans. Audio, Speech & Language Processing*, vol. 24, no. 11, pp. 2032–2045, 2016.

[16] P. Kenny, "A small footprint i-vector extractor," in *Proc. Odyssey 2012: the Speaker and Language Recognition Workshop*, Singapore, June 2012.

[17] N. Brümmer, "VB calibration to improve the interface between phone recognizer and i-vector extractor," *ArXiv e-prints*, 2015.

[18] P. Kenny, "Joint factor analysis of speaker and session variability: theory and algorithms," technical report CRIM-06/08-14, 2006.

[19] A. Sizov, K.-A. Lee, and T. Kinnunen, "Unifying probabilistic linear discriminant analysis variants in biometric authentication," in *Structural, Syntactic, and Statistical Pattern Recognition - Joint IAPR International Workshop, S+SSPR 2014, Joensuu, Finland, August 20-22, 2014. Proceedings*, 2014, pp. 464–475.

[20] L. Juvela, *Perceptual spectral matching utilizing mel-scale filterbanks for statistical parametric speech synthesis with glottal excitation vocoder*, Master's thesis, Aalto University, Espoo, Finland, 2015.

[21] J. Makhoul, "Linear prediction: a tutorial review," *Proceedings of the IEEE*, vol. 64, no. 4, pp. 561–580, April 1975.

[22] W.M. Campbell, D.E. Sturim, and D.A. Reynolds, "Support vector machines using GMM supervectors for speaker verification," *IEEE Signal Process. Lett.*, vol. 13, no. 5, pp. 308–311, 2006.