POLITECNICO DI TORINO
Repository ISTITUZIONALE

Scoring heterogeneous speaker vectors using nonlinear transformations and tied PLDa models

Original
Scoring heterogeneous speaker vectors using nonlinear transformations and tied PLDa models / Cumani, S., Laface, P.. - In: IEEE/ACM TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING. - ISSN 2329-9290. - STAMPA. - 26:5(2018), pp. 995-1009. [10.1109/TASLP.2018.2806305]

Availability:
This version is available at: 11583/2707141 since: 2018-05-16T16:58:59Z

Publisher:
Institute of Electrical and Electronics Engineers Inc.

Published
DOI:10.1109/TASLP.2018.2806305

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright
IEEE postprint/Author's Accepted Manuscript

©2018 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collecting works, for resale or lists, or reuse of any copyrighted component of this work in other works.

(Article begins on next page)

28 November 2026

Scoring heterogeneous speaker vectors using non-linear transformations and Tied PLDA models

Sandro Cumani and Pietro Laface

***Abstract*—Most current state–of–the–art text–independent speaker recognition systems are based on i–vectors, and on Probabilistic Linear Discriminant Analysis (PLDA). PLDA assumes that the i–vectors of a trial are homogeneous, i.e., that they have been extracted by the same system. In other words, the enrollment and test i–vectors belong to the same class. However, it is sometimes important to score trials including "heterogeneous" i–vectors, for instance, enrollment i–vectors extracted by an old system, and test i–vectors extracted by a newer, more accurate, system.**

In this paper we introduce a PLDA model that is able to score heterogeneous i–vectors independently of their extraction approach, dimensions, and any other characteristics that make a set of i–vectors of the same speaker belong to different classes. The new model, which will be referred to as Non–Linear Tied–PLDA (NL–Tied–PLDA), is obtained by a generalization of our recently proposed Non Linear PLDA approach, which jointly estimates the PLDA parameters and the parameters of a non–linear transformation of the i–vectors. The generalization consists in estimating a class–dependent non–linear transformation of the i–vectors, with the constraint that the transformed i–vectors of the same speaker share the same speaker factor.

The resulting model is flexible and accurate, as assessed by the results of a set of experiments performed on the extended core NIST SRE 2012 evaluation. In particular, NL–Tied–PLDA provides better results on heterogeneous trials with respect to the corresponding homogeneous trials scored by the old system, and, in some configurations, it also reaches the accuracy of the new system. Similar results were obtained on the female extended core NIST SRE 2010 telephone condition.

***Index Terms*—Speaker recognition, i–vector, e–vector, PLDA, non–linear density transformations.**

I. Introduction

Most state–of–the–art speaker recognition systems are based on a compact representation of a speech segment referred to as "identity vector" (i–vector for short) [1]. I–vectors are extracted by using Factor Analysis on the statistics collected by means of the UBM/GMM [2], or by means of the hybrid DNN/GMM approach, [3]–[6]. Given a pair of i–vectors, a PLDA classifier [7]–[9] allows computing the log–likelihood ratio between the hypotheses that they belong to the same speaker or to different speakers.

PLDA obviously assumes that the i–vectors of the pair are homogeneous, i.e., that they have been extracted by the same system. However, it is sometimes important scoring "heterogeneous" trials, including enrollment and test speaker features extracted by different systems. To illustrate an example of heterogeneous i–vectors, consider a scenario in which two or more sets of i–vectors of a common set of speakers are available, but each set was extracted by a different system, and possibly from different utterances. Suppose also that, for some reasons, the speech segments corresponding to one or more sets of these i–vectors are not available, preventing the possibility of extracting homogeneous i–vectors. A similar scenario has been suggested in [10], where a technique is presented for "migrating" i–vectors generated by a system to i–vectors extracted by another "reference" system.

Examples of heterogeneous speaker vectors are i–vectors obtained by systems using different speech features. Even more heterogeneous are the e–vectors [11], [12], or other speaker specific vectors that can be extracted from the output of a Deep Neural Network layer [13]–[15]. Other i–vectors that could be considered heterogeneous, and that might not be reliably compared, are the ones extracted from speech segments largely different in duration, [16]–[24], or with different signal to noise ratio [25]–[31], or even i–vectors collected from different types of channels [32], [33].

In the following we will refer to these features simply as i–vectors even if they have been obtained by means of techniques not related to the standard i–vector extraction approach of [1].

We have recently proposed an approach that jointly estimates the distribution of the development i–vectors and the PLDA parameters, so that the i–vectors are non–linearly transformed to a new compact representation that makes PLDA classification more effective. This i–vector transformation is modeled by means of a sequence of linear and non–linear functions, the parameters of which are obtained by Maximum Likelihood (ML) estimation on the development set, as detailed in [34]–[36].

Since this model includes i–vector transformations, it is a natural basic module for designing a system that is able to compare heterogeneous i–vectors. In this work, thus, we extend and generalize the non–linear PLDA model (NL–PLDA) of [36] to deal with heterogeneous sets of i–vectors belonging to different classes. This approach is able to score heterogeneous i–vectors independently of their extraction approach, dimensions, original speech features, and any other characteristics that make a set of i–vectors of the same speaker belong to different classes. The basic assumption of this framework is that speakers can be modeled by means of a latent variable living in a common latent space.

The NL–PLDA generalization consists in estimating a class–dependent transformation of the i–vectors, with the constraint that the transformed i–vectors of the same speaker share the same speaker factor. A similar approach, excluding the non–linear transformations, was proposed as Tied–PLDA

The authors are with the Dipartimento di Automatica e Informatica, Politecnico di Torino, 10143 Torino, Italy (e-mail: sandro.cumani@polito.it, pietro.laface@polito.it).

Computational resources for this work were provided by HPC@POLITO (http://www.hpc.polito.it)

model in [37], [38] for face recognition, and used in [39] to address noisy and reverberant conditions. Thus, we refer to in the following to our model as the Non Linear Tied–PLDA (NL–Tied–PLDA) model.

In this work, beyond extending the Tied–PLDA model to embed non linear i–vector transformations, we show that training both linear and non–linear Tied–PLDA models can be performed by an Expectation–Maximization (EM) procedure that directly uses the E–step and M–step of the standard and NL–PLDA models. Furthermore, it is well known that in multi–enrollment conditions PLDA benefits from scoring with i–vector averaging [40], and so do Tied–PLDA models, but average i–vector scoring is only possible if the trial i–vectors are homogeneous. We show, instead, that it is possible to devise a scoring technique, similar to i–vector averaging, even if i–vectors are heterogeneous.

The Tied–PLDA model, like the Mixture of PLDA model [38], describes the training data as a mixture of factor analyzers. However, in the Mixture of PLDA model, the representation of identity includes the choice of the class k, whereas in our context, the class labels are known. Thus it not necessary to introduce a latent variable that estimates the assignment of an i–vector to a class.

Another approach for scoring heterogeneous i–vectors has been introduced in [23] for duration mismatch compensation, and referred to as Twin–PLDA. It uses the same assumption that i–vectors from the same speaker share identical normally distributed latent variables. An improved model has been presented in [24] by the same authors. We discuss the analogies and differences among these two models and our approach in Section V-B and VI-B.

The NL–Tied–PLDA model is flexible and accurate as assessed by the results of a set of experiments on homogeneous and heterogeneous trials on the extended core NIST SRE 2012. In particular, we consider an application scenario for a company that releases a new speaker verification system. The new system should offer the possibility that a user is recognized using her/his new test i–vectors scored against i–vectors extracted at the time of enrollment with the previous system, because the old enrollment signals are no more available. This system would allow the users of the new system to be recognized without enrolling again immediately. It is expected, of course, that the performance of the system using these heterogeneous trials is not as good as the one obtained with homogenous trials of i–vectors extracted by the new system. In this scenario, NL–Tied–PLDA provides better results on heterogeneous trials with respect to the corresponding homogeneous trials scored by the old system, and, in some configurations, it is also able to reach the accuracy of the new system.

Approximately 10% improvement is also achieved on the female extended core NIST SRE 2010 evaluation telephone–telephone condition (Condition 5) with respect to PLDA models scoring homogeneous i–vectors of the old system.

The paper is organized as follows: In Section II we recall the Gaussian PLDA model and the Non–Linear PLDA model, [36]. The generalization of the NL–PLDA model for heterogeneous i–vectors (NL–Tied–PLDA) is detailed in Section III, together with the procedures for model initialization and estimation. Section IV illustrates the relationships between proper PLDA scoring and PLDA scoring based on i–vector averaging, and extends the latter to heterogeneous i–vector scoring. In Section V our model is compared with two other approaches, and improved models are proposed in Section VI. The experimental settings and results are described Section VII. Section VIII is devoted to further comments, and conclusions are drawn in Section IX.

II. Non–Linear PLDA model

PLDA models the underlying speaker and channel variability in the i–vector space using a probabilistic framework. From the PLDA distributions it is possible to evaluate the likelihood ratio between the "same speaker" hypothesis and "different speaker" hypothesis for a set of i–vectors. In particular, the simplified Gaussian PLDA model assumes that the i–vector generation process can be described by means of a latent variable probabilistic model where an i–vector $\boldsymbol{\phi}_{i,s}$, belonging to speaker s, is represented as the sum of three factors, namely the i–vector global mean $\mathbf{m}$, a speaker factor $\mathbf{y}_s$, and a factor $\boldsymbol{\epsilon}_{i,s}$ that represents the inter–session and the residual noise, as:

$$\boldsymbol{\phi}_{i,s} = \mathbf{m} + \mathbf{U}\mathbf{y}_s + \boldsymbol{\epsilon}_{i,s} \ . \tag{1}$$

The model assumes that the speaker factor $\mathbf{y}_s$ has a standard normal prior, and that the residual term $\boldsymbol{\epsilon}_{i,s}$ is sampled from $\mathcal{N}(\mathbf{0}, \boldsymbol{\Lambda}^{-1})$, where $\boldsymbol{\Lambda}$ is the within–class precision matrix. Matrix $\mathbf{U}$ can constrain the speaker factors to be of lower dimension than the i–vectors space.

A. Gaussian PLDA posterior estimation

Let us ignore for the moment the non–linear transformation of the i–vectors. Given the current model parameters $\mathbf{m}$, $\mathbf{U}$, and $\boldsymbol{\Lambda}$, and a speaker s providing a set of n_s i–vectors $M_s = \{\boldsymbol{\phi}_{1,s}, \boldsymbol{\phi}_{2,s}, \cdots, \boldsymbol{\phi}_{n_s,s}\}$, PLDA estimates the Gaussian posterior distribution for $\mathbf{y}_s|M_s$ with mean $\boldsymbol{\mu}_{\mathbf{y},s}$ and precision matrix $\boldsymbol{\Lambda}_{\mathbf{y},s}$ given by [7]:

$$\begin{aligned} \boldsymbol{\Lambda}_{\mathbf{y},s} &= \mathbf{I} + n_s \mathbf{U}^T \boldsymbol{\Lambda} \mathbf{U} \\ \boldsymbol{\mu}_{\mathbf{y},s} &= \boldsymbol{\Lambda}_{\mathbf{y},s}^{-1} \mathbf{U}^T \boldsymbol{\Lambda} \sum_{i=1}^{n_s} (\boldsymbol{\phi}_{i,s} - \mathbf{m}) \ . \end{aligned} \tag{2}$$

The PLDA parameters can be estimated by Expectation–Maximization (EM) iterations, where the posterior distribution parameters in (2), computed using the current estimate of the model parameters, are used for the estimation of the new model parameters $\mathbf{m}$, $\mathbf{U}$, and $\boldsymbol{\Lambda}$ in the maximization step.

B. Density function transformation

In [34], [35] we have presented a method for transforming the i–vectors so that their distribution becomes Gaussian. This has been done to better fit the PLDA assumption that i–vectors are distributed according to the standard normal distribution. The non–linear i–vector transformation model proposed in those works assumes that i–vectors are independently sampled

from a standard normal distribution, and independently transformed by means of an invertible non–linear function f^{-1}:

$$\mathbf{\Phi} = f^{-1}(\mathbf{Z}) \ , \tag{3}$$

where $\mathbf{\Phi}$ is a random variable that generates i–vectors, $\mathbf{Z}$ is a standard normal random variable, and $\boldsymbol{\phi}_{i,s} \sim \mathbf{\Phi}$ is a sampled i–vector [1]. By applying function f to i–vectors $\boldsymbol{\phi}_{i,s}$, they are transformed into samples that follow a standard normal distribution. The transformed i–vectors, $f(\boldsymbol{\phi}_{i,s})$, are then used as features for training a PLDA classifier.

In particular, to fit a Gaussian distribution, in [34], [35] we make use of the affine transformation defined as:

$$f_1(\boldsymbol{\phi}, \mathbf{A}, \mathbf{b}) = \mathbf{A}\boldsymbol{\phi} + \mathbf{b} \ , \tag{4}$$

where $\mathbf{A}$ is a full–rank matrix, and $\mathbf{b}$ is an offset vector, and of a non–linear transformation, based on the sinh–arcsinh function [41], [42]:

$$\overline{f}_2(\phi, \delta, \varepsilon) = \sinh(\delta \sinh^{-1}(\phi) + \varepsilon) \ . \tag{5}$$

This latter can be generalized for N–dimensional variables as:

$$f_2(\boldsymbol{\phi}, \boldsymbol{\delta}, \boldsymbol{\varepsilon}) = \begin{bmatrix} \overline{f}_2(\phi_1, \delta_1, \varepsilon_1) \\ \vdots \\ \overline{f}_2(\phi_N, \delta_N, \varepsilon_N) \end{bmatrix} \ , \tag{6}$$

where $\delta_i > 0$ controls the tailweight of the distribution, and ε_i affects the skewness of each variable. This function has been selected because it is invertible, and flexible in mapping a distribution to another distribution. In [34] we have shown that a sequence of linear and non–linear sinh-arcsinh functions leads to more accurate transformations.

C. Joint estimation of non–linear transformations and PLDA model parameters

Although PLDA classification based on the "gaussianized" i–vectors has been successfully tested on the NIST SRE 2010 and SRE 2012 evaluation datasets, the model of [35] is approximate because it estimates the non–linear transformation assuming, as is usually done in i–vector extraction, that each training i–vector belongs to a different speaker. In [36] we proposed a more accurate model, where we jointly estimate the PLDA parameters and the parameters of the non–linear transformation of the i–vectors. In this new model, the parameters of the non–linear transformation intrinsically account for the speaker information associated to each i–vector, an information that was ignored in the previous approach.

The new generative NL–PLDA model is given by:

$$\begin{aligned} \mathbf{z}_{i,s} &= \mathbf{U}\mathbf{y}_s + \boldsymbol{\epsilon}_{i,s} \\ \boldsymbol{\phi}_{i,s} &= f^{-1}(\mathbf{z}_{i,s}) \ , \end{aligned} \tag{7}$$

where $\mathbf{y}_s$, $\boldsymbol{\epsilon}_{i,s}$ and $\mathbf{U}$ have been defined in (1). It is worth noting that $\mathbf{m}$ does not appear in (7) because it can be accounted for by the non–linear transformation.

[1] In [35] $\mathbf{\Phi}$ and $\mathbf{Z}$ were denoted as $\mathbf{X}$ and $\mathbf{Y}$, respectively. Since $\mathbf{x}$ and $\mathbf{y}$ are traditionally used for the channel and speaker factors of PLDA, we changed the notation for the i–vectors and transformed i–vectors that has been used in [36].

Given model (7), we have shown that the posterior distribution $P(\mathbf{y}_s|M_s)$ is Gaussian, with mean $\boldsymbol{\mu}_{\mathbf{y},s}$ and precision matrix $\mathbf{\Lambda}_{\mathbf{y},s}$ given by:

$$\begin{aligned} \mathbf{\Lambda}_{\mathbf{y},s} &= \mathbf{I} + n_s \mathbf{U}^T \mathbf{\Lambda} \mathbf{U} \\ \boldsymbol{\mu}_{\mathbf{y},s} &= \mathbf{\Lambda}_{\mathbf{y},s}^{-1} \mathbf{U}^T \mathbf{\Lambda} \sum_{i=1}^{n_s} f(\boldsymbol{\phi}_{i,s}) \ . \end{aligned} \tag{8}$$

These parameters are similar to the standard PLDA ones in (2), the only difference is that the posterior mean is computed summing the transformed i–vectors $f(\boldsymbol{\phi}_{i,s})$ rather than the original i–vectors $\boldsymbol{\phi}_{i,s}$.

III. Non–Linear Tied PLDA model

In this section we extend the NL–PLDA approach to deal with heterogeneous trials, which may include i–vectors extracted by means of different techniques, or extracted by means of the same technique, but conceptually belonging to different classes.

Let $M_{k,s} = \{\boldsymbol{\phi}_{ki,s}\}$, $i = 1, \ldots, n_{k,s}$ be the set of $n_{k,s}$ i–vectors belonging to class k, associated to a given speaker s. It is worth noting that the i–vectors of the same speaker, but belonging to different classes, are possibly extracted from different sets of utterances.

The NL–Tied–PLDA model for the i–vectors of K classes, sharing the same speaker factor $\mathbf{y}_s$, can be written as:

$$\begin{aligned} \boldsymbol{\phi}_{1i,s} &= f_1^{-1}(\mathbf{U}_1\mathbf{y}_s + \epsilon_{1i,s}) \ , i = 1, \ldots, n_{1,s} \\ \boldsymbol{\phi}_{2i,s} &= f_2^{-1}(\mathbf{U}_2\mathbf{y}_s + \epsilon_{2i,s}) \ , i = 1, \ldots, n_{2,s} \\ &\cdots \\ \boldsymbol{\phi}_{Ki,s} &= f_2^{-1}(\mathbf{U}_K\mathbf{y}_s + \epsilon_{Ki,s}) \ , i = 1, \ldots, n_{K,s} \ . \end{aligned} \tag{9}$$

where $\epsilon_{ki,s} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Lambda}_k^{-1})$.

It is worth noting that i–vectors of different classed might have different dimension. If this is the case, matrices $\mathbf{U}_k$ will have the same number of columns, but different number of rows.

The parameters of this model can be trained by means of an EM algorithm, as it is done for a single–class NL–PLDA model. Using the same notation of [36], we denote the set of model parameters for class k by $\boldsymbol{\theta}_k = [\boldsymbol{\vartheta}_k, \mathbf{U}_k, \mathbf{\Lambda}_k]$, where $\boldsymbol{\vartheta}_k$ are the parameters of the non–linear transformation f_k. We also denote the complete set of the model parameters by $\boldsymbol{\theta} = [\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_K]$.

The EM objective function for NL–Tied–PLDA is given by:

$$Q(\boldsymbol{\theta}, \widetilde{\boldsymbol{\theta}}) = \sum_{s=1}^{S} \mathbb{E}_{\mathbf{y}_s|M_s,\widetilde{\boldsymbol{\theta}}} \log P(M_s|\mathbf{y}_s, \boldsymbol{\theta}) \ , \tag{10}$$

where S is the number of speakers, $\widetilde{\boldsymbol{\theta}}$ are the model parameters that allow computing, in the E-step, the posterior distribution $P(\mathbf{y}_s|M_{1,s}, \ldots, M_{K,s}, \widetilde{\boldsymbol{\theta}})$, and M_s is the set of all i–vectors for speaker s, $M_s = M_{1,s} \cup M_{2,s} \cup \cdots \cup M_{K,s}$.

The NL–Tied–PLDA model assumes that i–vectors are independent given the speaker variable, thus:

$$\log P(M_{1,s}, \ldots, M_{K,s}|\mathbf{y}_s, \boldsymbol{\theta}) = \sum_{k=1}^{K} \sum_{i=1}^{n_{k,s}} \log P(\phi_{ki,s}|\mathbf{y}_s, \boldsymbol{\theta}) \ . \tag{11}$$

Since the conditional distribution of an i–vector only depends on the model parameters of its class, (11) can be rewritten as:

$$\log P(M_{1,s},\ldots,M_{K,s}|\mathbf{y}_s,\boldsymbol{\theta}) = \sum_{k=1}^{K}\sum_{i=1}^{n_{k,s}} \log P(\phi_{ki,s}|\mathbf{y}_s,\boldsymbol{\theta}_k) \ , \tag{12}$$

and the EM objective function as:

$$Q(\boldsymbol{\theta},\widetilde{\boldsymbol{\theta}}) = \sum_{s=1}^{S} \mathbb{E}_{\mathbf{y}_s|M_s,\widetilde{\boldsymbol{\theta}}} \sum_{k=1}^{K}\sum_{i=1}^{n_{k,s}} \log P(\phi_{ki,s}|\mathbf{y}_s,\boldsymbol{\theta}_k) \ . \tag{13}$$

Changing the order of the summations, we obtain:

$$\begin{aligned} Q(\boldsymbol{\theta},\widetilde{\boldsymbol{\theta}}) &= \sum_{k=1}^{K}\sum_{s=1}^{S} \mathbb{E}_{\mathbf{y}_s|M_s,\widetilde{\boldsymbol{\theta}}} \sum_{i=1}^{n_{k,s}} \log P(\phi_{ki,s}|\mathbf{y}_s,\boldsymbol{\theta}_k) \\ &= \sum_{k=1}^{K}\sum_{s=1}^{S} \mathbb{E}_{\mathbf{y}_s|M_s,\widetilde{\boldsymbol{\theta}}} \log P(M_{k,s}|\mathbf{y}_s,\boldsymbol{\theta}_k) \\ &= \sum_{k=1}^{K} Q_k(\boldsymbol{\theta}_k,\widetilde{\boldsymbol{\theta}}) \ , \end{aligned} \tag{14}$$

where

$$Q_k(\boldsymbol{\theta}_k,\widetilde{\boldsymbol{\theta}}) = \sum_{s=1}^{S} \mathbb{E}_{\mathbf{y}_s|M_s,\widetilde{\boldsymbol{\theta}}} \log P(M_{k,s}|\mathbf{y}_s,\boldsymbol{\theta}_k) \ . \tag{15}$$

A. M–step

Since each objective function Q_k does not share any optimization variable with the objective functions of the other classes, the global objective function of (14) can be maximized by independently maximizing each of its terms Q_k, as:

$$\begin{aligned} \boldsymbol{\theta}_{new} &= \operatorname*{argmax}_{\boldsymbol{\theta}} Q(\boldsymbol{\theta},\widetilde{\boldsymbol{\theta}}) = \\ &= \left[\operatorname*{argmax}_{\boldsymbol{\theta}_1} Q_1(\boldsymbol{\theta}_1,\widetilde{\boldsymbol{\theta}}),\ldots,\operatorname*{argmax}_{\boldsymbol{\theta}_K} Q_K(\boldsymbol{\theta}_K,\widetilde{\boldsymbol{\theta}})\right] \end{aligned} \tag{16}$$

Let $Q_k^{NL}(\boldsymbol{\theta}_k,\widetilde{\boldsymbol{\theta}}_k)$ denote the single–class NL–PLDA objective function for class k, i.e., the EM objective that would be optimized for a NL–PLDA model with all i–vectors belonging to class k. From equations (17) and (18) of [36], Q_k^{NL} can be written as:

$$Q_k^{NL}(\boldsymbol{\theta}_k,\widetilde{\boldsymbol{\theta}}_k) = \sum_{s=1}^{S} \mathbb{E}_{\mathbf{y}_s|M_{k,s},\widetilde{\boldsymbol{\theta}}} \log P(M_{k,s}|\mathbf{y}_s,\boldsymbol{\theta}_k) \ . \tag{17}$$

Comparing (15) and (17) we can observe that the two objectives only differ in the posterior distribution used to compute the expectation. Thus, each Q_k can be optimized by means of the same M–step algorithm of the single–class NL–PLDA using only the i–vectors that belong to class k, but computing the expectations with respect to $\mathbf{y}_s$ from $P(\mathbf{y}_s|M_s,\widetilde{\boldsymbol{\theta}})$, rather than from $P(\mathbf{y}_s|M_{k,s},\widetilde{\boldsymbol{\theta}}_k)$. Please notice that $P(\mathbf{y}_s|M_s,\widetilde{\boldsymbol{\theta}})$ depends on all the i–vectors of speaker s, while $P(\mathbf{y}_s|M_{k,s},\widetilde{\boldsymbol{\theta}}_k)$ depends only on the i–vectors of class k of speaker s. The M–step can be performed, thus, as a set of K independent optimizations.

This also means that the M–step for the Tied–PLDA model does not depend on the implementation of the M–step for the single–class models. It only needs that each single–class model provides the update of its own parameters from from the i–vectors of that class, and the posterior distribution of $\mathbf{y}_s$.

In particular, for NL–Tied–PLDA it is possible to exploit the M–step of single–class NL–PLDA described in Section IV of [36], which requires computing the gradients of Q_k with respect to $\mathbf{U}_k$ and $\boldsymbol{\vartheta}_k$, and is summarized in the following. The re-estimation of the residual noise covariances $\boldsymbol{\Lambda}_k$ can be avoided, as long as each non–linear function is the composition of an affine and of an invertible function.

The first gradient can be computed as in equation (35) of [36]:

$$\begin{aligned} &\nabla_{\mathbf{U}_k} Q_k(\boldsymbol{\theta}_k,\widetilde{\boldsymbol{\theta}}) = \\ &\sum_{s} \boldsymbol{\Lambda}_k \left(\sum_{i=1}^{n_{k,s}} f_k(\boldsymbol{\phi}_{ki,s},\boldsymbol{\vartheta}_k) \mathbb{E}\left[\mathbf{y}_s^T\right] - n_{k,s}\mathbf{U}\mathbb{E}\left[\mathbf{y}_s\mathbf{y}_s^T\right] \right) \ , \end{aligned} \tag{18}$$

where the expectation is taken with respect to $P(\mathbf{y}_s|M_s,\widetilde{\boldsymbol{\theta}})$. The gradient with respect to $\boldsymbol{\vartheta}_k$ can be obtained replacing the backward procedure initialization of equation (38) in [36]

$$\mathbf{b}_n = \boldsymbol{\Lambda}\left(\mathbf{U}\mathbb{E}\left[\mathbf{y}_s\right] - \mathbf{x}_n\right) \ , \tag{19}$$

by:

$$\mathbf{b}_n = \boldsymbol{\Lambda}_k\left(\mathbf{U}_k\mathbb{E}\left[\mathbf{y}_s\right] - \mathbf{x}_n\right) \ , \tag{20}$$

where $\mathbf{x}_n = f_k(\boldsymbol{\phi}_{i,s},\boldsymbol{\vartheta}_k)$, i.e., replacing the initialization $\mathbf{b}_n = -\mathbf{x}_n$ in equation (20) of Algorithm 1 of [35] by (20).

B. E–step

As for the M–step, also the E–step can be implemented exploiting the E–step used for estimating the parameters of single–class PLDA models. In particular, the posterior distributions required by the M–step can be directly computed from the class–dependent posterior distributions $P(\mathbf{y}_s|M_{k,s})$, computed using single–class models.

Let us consider, without loss of generality, only two classes and a single speaker, thus dropping the subscript s. Let also, for the sake of notation simplicity, redefine the sets $M_{1,s}$ and $M_{2,s}$ as M_1 and M_2, respectively.

The posterior probability of $\mathbf{y}$, given the sets M_1 and M_2, can be computed, applying Bayes' rules, as:

$$\begin{aligned} P(\mathbf{y}|M_1,M_2) &= \frac{P(M_1|\mathbf{y})\cdot P(M_2|\mathbf{y})\cdot P(\mathbf{y})}{P(M_1,M_2)} \\ &= \frac{P(\mathbf{y})}{P(M_1,M_2)} \cdot \frac{P(\mathbf{y}|M_1)P(M_1)}{P(\mathbf{y})} \cdot \frac{P(\mathbf{y}|M_2)P(M_2)}{P(\mathbf{y})} \end{aligned} \tag{21}$$

Rearranging the terms in (21), and collecting the terms that do not depend on $\mathbf{y}$ in a single term c, we get:

$$\begin{aligned} P(\mathbf{y}|M_1,M_2) &= \frac{P(M_1)P(M_2)}{P(M_1,M_2)} \cdot \frac{P(\mathbf{y}|M_1)}{P(\mathbf{y})} \cdot \frac{P(\mathbf{y}|M_2)}{P(\mathbf{y})} \cdot P(\mathbf{y}) \\ &= c \cdot \frac{P(\mathbf{y}|M_1)}{P(\mathbf{y})} \cdot \frac{P(\mathbf{y}|M_2)}{P(\mathbf{y})} \cdot P(\mathbf{y}) \end{aligned} \tag{22}$$

Although the posterior $P(\mathbf{y}|M_1,M_2)$ can be obtained by explicitly writing the joint likelihood of M_1, M_2 and $\mathbf{y}$, equation (22) allows us to directly relate $P(\mathbf{y}|M_1,M_2)$ to

the posterior distributions $P(\mathbf{y}|M_1)$ and $P(\mathbf{y}|M_2)$, which are computed using the model parameters of the corresponding classes, defined in (8). Since $P(\mathbf{y}|M_1)$, $P(\mathbf{y}|M_2)$ and $P(\mathbf{y})$ are Gaussians, it is possible to rewrite their log–pdfs in terms of their natural parameters [43] , η_0, η_1 and η_2, as:

$$\begin{aligned}\log P(\mathbf{y}) &= \eta_0^T T(\mathbf{y}) - k(\eta_0)\\ \log P(\mathbf{y}|M_1) &= \eta_1^T T(\mathbf{y}) - k(\eta_1)\\ \log P(\mathbf{y}|M_2) &= \eta_2^T T(\mathbf{y}) - k(\eta_2)\ ,\end{aligned} \tag{23}$$

where[2]

$$T(\mathbf{y}) = \begin{bmatrix} \mathbf{y} \\ -\frac{1}{2}\,\mathrm{vec}(\mathbf{y}\mathbf{y}^T) \end{bmatrix}\ , \tag{24}$$

$\mathrm{vec}(\cdot)$ is the operator that stacks the columns of a matrix into a vector, and k is the log–partition function.
Recalling that, for a Gaussian distribution with mean $\mathbf{m}$ and covariance matrix $\mathbf{\Sigma}$ the natural parameters are given by:

$$\eta(\mathbf{m}, \mathbf{\Sigma}) = \begin{bmatrix} \mathbf{\Sigma}^{-1}\mathbf{m} \\ \mathrm{vec}(\mathbf{\Sigma}^{-1}) \end{bmatrix}\ , \tag{25}$$

the natural parameters of $P(\mathbf{y}|M_1)$ and $P(\mathbf{y}|M_2)$ are:

$$\eta_1 = \begin{bmatrix} \mathbf{\Lambda}_{\mathbf{y}_1}\boldsymbol{\mu}_{\mathbf{y}_1} \\ \mathrm{vec}(\mathbf{\Lambda}_{\mathbf{y}_1}) \end{bmatrix}\ ,\quad \eta_2 = \begin{bmatrix} \mathbf{\Lambda}_{\mathbf{y}_2}\boldsymbol{\mu}_{\mathbf{y}_2} \\ \mathrm{vec}(\mathbf{\Lambda}_{\mathbf{y}_2}) \end{bmatrix}\ , \tag{26}$$

and the natural parameters of $P(\mathbf{y})$ are:

$$\eta_0 = \begin{bmatrix} \mathbf{0} \\ \mathrm{vec}(\mathbf{I}) \end{bmatrix}\ . \tag{27}$$

Applying (23) to (22) we can conclude that $P(\mathbf{y}|M_1, M_2)$ is also Gaussian, with natural parameters easily obtained by simply adding and subtracting the natural parameters of the distributions that appear in the numerator and in the denominator of (22) as:

$$\eta_{12} = \eta_1 + \eta_2 - \eta_0\ . \tag{28}$$

Thus, we can finally write the distribution precision matrix and mean of $P(\mathbf{y}|M_1, M_2)$ as:

$$\begin{aligned}\mathbf{\Lambda}_{\mathbf{y}} &= \mathbf{\Lambda}_{\mathbf{y}1} + \mathbf{\Lambda}_{\mathbf{y}2} - \mathbf{I}\\ \boldsymbol{\mu}_{\mathbf{y}} &= \mathbf{\Lambda}_{\mathbf{y}}^{-1}\left(\mathbf{\Lambda}_{\mathbf{y}1}\boldsymbol{\mu}_{\mathbf{y}1} + \mathbf{\Lambda}_{\mathbf{y}2}\boldsymbol{\mu}_{\mathbf{y}2}\right)\ ,\end{aligned} \tag{29}$$

where $\boldsymbol{\mu}_{\mathbf{y}_1}$, $\mathbf{\Lambda}_{\mathbf{y}_1}$ and $\boldsymbol{\mu}_{\mathbf{y}_2}$, $\mathbf{\Lambda}_{\mathbf{y}_2}$ are the parameters of the posterior distributions $P(\mathbf{y}|M_1)$ and $P(\mathbf{y}|M_2)$, which depend only on the model parameters of class 1 and 2, respectively.

[2]It is custom in literature to represent function T(**y**), and the corresponding natural parameters, for Gaussian distributions as:

$$T(\mathbf{y}) = \begin{bmatrix} \mathbf{y} \\ \mathrm{vec}(\mathbf{y}\mathbf{y}^T) \end{bmatrix}\ ,\quad \eta(\mathbf{m}, \mathbf{\Sigma}) = \begin{bmatrix} \mathbf{\Sigma}^{-1}\mathbf{m} \\ -\frac{1}{2}\,\mathrm{vec}(\mathbf{\Sigma}^{-1}) \end{bmatrix}\ .$$

Since the constant term $-\frac{1}{2}$ can be equivalently included in function T, we prefer to adopt the definitions given in (24) and (25) to simplify the notation of the other equations appearing in our paper.

Generalizing to K classes, the precision matrix and the mean of the posterior distribution $P(\mathbf{y}|M_1, \ldots, M_K)$ are:

$$\begin{aligned}\mathbf{\Lambda}_{\mathbf{y}} &= \sum_{k=1}^{K} \mathbf{\Lambda}_{\mathbf{y}k} - (K-1)\mathbf{I}\\ \boldsymbol{\mu}_{\mathbf{y}} &= \mathbf{\Lambda}_{\mathbf{y}}^{-1}\sum_{k=1}^{K} \mathbf{\Lambda}_{\mathbf{y}k}\boldsymbol{\mu}_{\mathbf{y}k}\\ &= \mathbf{\Lambda}_{\mathbf{y}}^{-1}\sum_{k=1}^{K} \mathbf{U}_k^T\mathbf{\Lambda}_k\sum_{i=1}^{n_k} f_k(\boldsymbol{\phi}_{ki})\end{aligned} \tag{30}$$

C. NL–Tied–PLDA model initialization

The standard Gaussian PLDA is a particular case of the NL–PLDA model, where the transformation function f is the identity function. The NL–PLDA model, thus, can be initialized by setting matrices $\mathbf{U}$ and $\mathbf{\Lambda}$ to the values estimated by a PLDA model, and selecting the function parameters $\boldsymbol{\vartheta}$ such that $f(\boldsymbol{\phi}, \boldsymbol{\vartheta}) = \boldsymbol{\phi}$. By analogy, we initialize the NL–Tied–PLDA model from an heterogeneous linear PLDA model (Tied–PLDA).

A straightforward initialization of the Tied–PLDA model can be performed by training a PLDA model independently for each class. However, the EM–algorithm might experience slow convergence in its initial iterations because the speaker factor subspaces of the different PLDA models could be not well aligned. It is easy verifying, for example, that swapping two columns of the $\mathbf{U}$ matrix in a PLDA model does not change the model, but changes the speaker posterior representation.

In order to mitigate this effect, we select a reference PLDA model, k_r, e.g., the best performing system, and transform each other PLDA model k, by replacing matrix $\mathbf{U}_k$ by matrix $\mathbf{U}_k\mathbf{R}_k^T$, where $\mathbf{R}_k$ is the unitary matrix that minimizes the Euclidean distance between the MAP–point estimates of the speaker factors of the reference PLDA model and the ones of each other class k.

We look for a unitary matrix because the PLDA model for class k is invariant to a transformation $\mathbf{U}_k \longrightarrow \mathbf{U}_k\mathbf{R}_k^T$, in the sense that it provides the same likelihood for i–vectors of class k. Furthermore, it is well known that, replacing $\mathbf{U}_k$ by $\mathbf{U}_k\mathbf{R}_k^T$, the parameters of the posterior distribution of $P(\mathbf{y}|\cdot)$ are modified as follows:

$$\mathbf{\Lambda}_{\mathbf{y}k} \longrightarrow \mathbf{R}_k\mathbf{\Lambda}_{\mathbf{y}k}\mathbf{R}_k^T\ ,\quad \boldsymbol{\mu}_{\mathbf{y}k} \longrightarrow \mathbf{R}_k\boldsymbol{\mu}_{\mathbf{y}k}\ . \tag{31}$$

Let $\mathbf{Y}_k = \left[\boldsymbol{\mu}_{\mathbf{y}k,1}, \ldots, \boldsymbol{\mu}_{\mathbf{y}k,S}\right]$ be the matrix stacking the MAP–point estimates for all the speakers in the training set, computed from i–vectors of class k using the corresponding PLDA model, and let t denote the target class. Matrix $\mathbf{R}_k$ is the solution of equation:

$$\mathbf{R}_k = \underset{\mathbf{R}}{\mathrm{argmax}}\,\|\mathbf{Y}_t - \mathbf{R}\mathbf{Y}_k\|^2\ , \tag{32}$$

which is given by:

$$\mathbf{R}_k = \mathbf{V}\mathbf{W}^T\ , \tag{33}$$

where $\mathbf{V}$ and $\mathbf{W}$ are the left and right vectors of the Singular Value Decomposition of

$$\mathbf{Y}_t\mathbf{Y}_k^T = \mathbf{V}\mathbf{S}\mathbf{W}^T\ . \tag{34}$$

This solution provides the best rotation that relates two set of i–vectors [44].

An alternative and even simpler approach[3] consists in modifying the first iteration of the EM procedure by performing the E–step using only the i–vectors and PLDA model of the best system. This allows reducing the possible misalignment between the initial estimates of the speaker factor subspaces of the different PLDA models. The successive EM steps remain unchanged.

The effect of the two approaches is similar, and affects only the convergence rate of the algorithm.

IV. Scoring heterogeneous i–vectors with NL–Tied–PLDA

As will be detailed in the Section devoted to the experiments, scoring heterogeneous i–vectors may involve enrollment and test sets that contain i–vectors of possibly different utterances, belonging to different classes. We illustrate in this section a unique framework for proper PLDA scoring, and for scoring based on i–vector averaging. We then extend the latter to heterogeneous i–vector scoring.

Let $M_{k,E} = \{\boldsymbol{\phi}_{ki,E}\}$, $i = 1, \ldots, n_{k,E}$ be the set of $n_{k,E}$ enrollment i–vectors for a speaker belonging to class k, and let $M_E = \cup_{k=1}^{K} M_{k,E}$. Similarly, let $M_{k,T} = \{\boldsymbol{\phi}_{ki,T}\}$, $i = 1, \ldots, n_{k,T}$ denote the set of $n_{k,T}$ test i–vectors for class k, and $M_T = \cup_{k=1}^{K} M_{k,T}$. The likelihood ratio of the "same" and "different" speaker hypotheses is:

$$lr = \frac{P(M_T, M_E)}{P(M_T)P(M_E)} = \frac{P(\mathbf{y}|M_T)P(\mathbf{y}|M_E)}{P(\mathbf{y}|M_T, M_E)P(\mathbf{y})} \ , \qquad (35)$$

where the posterior terms $P(\mathbf{y}|\cdot)$ are computed using (30). It is worth noting that (35) holds for any value of $\mathbf{y}$ for which the denominator is non–zero. It can be, thus, written as a function of the natural parameters of the prior and posterior distributions appearing in (35).

As for standard PLDA, efficient scoring can be achieved by pre–computing the covariance matrices of the posterior distributions of $\mathbf{y}$. However, in case of PLDA or NL–PLDA, their covariance matrices depend only on the number of enrollment and test i–vectors, whereas for Tied–PLDA or NL–Tied–PLDA they depend on number of enrollment (or test) i–vectors available for each class, as can be seen in (8) and (30), respectively.

Furthermore, it is well known that computing multi–enrollment PLDA scores by simply averaging the enrollment i–vectors provides higher accuracy with respect to proper multi–session PLDA scoring [40]. Since it is not possible to average heterogeneous i–vectors, we will illustrate, in Section IV-B, how i–vector average scoring can be computed as a function of the natural parameters of the posterior distributions $P(\mathbf{y}|\cdot)$. This framework will also allow us to illustrate in Section IV-C an easily derived i–vector "averaging" scoring approach for Heterogeneous PLDA models. In the following, for the sake of notation simplicity, we will drop class indexes whenever we consider only homogeneous i–vectors sets.

[3]We thanks one of the reviewers for the suggestion.

A. Proper PLDA scoring

The posterior distributions required for PLDA scoring are $P(\mathbf{y}|M_E)$, $P(\mathbf{y}|M_T)$, and $P(\mathbf{y}|M_E, M_T)$, where M_E and M_T are sets of homogeneous i–vectors. Letting f in (8) be the identity function, we have that $P(\mathbf{y}|M_E)$, $P(\mathbf{y}|M_T)$ are Gaussian distributed, with posterior parameters:

$$\begin{aligned} \boldsymbol{\Lambda}_{\mathbf{y},E} &= \mathbf{I} + n_E \mathbf{U}^T \boldsymbol{\Lambda} \mathbf{U} \ , \quad \boldsymbol{\mu}_{\mathbf{y},E} = \boldsymbol{\Lambda}_{\mathbf{y},E}^{-1} \mathbf{U}^T \boldsymbol{\Lambda} \sum_{i=1}^{n_E} \boldsymbol{\phi}_{i,E} \ , \\ \boldsymbol{\Lambda}_{\mathbf{y},T} &= \mathbf{I} + n_T \mathbf{U}^T \boldsymbol{\Lambda} \mathbf{U} \ , \quad \boldsymbol{\mu}_{\mathbf{y},T} = \boldsymbol{\Lambda}_{T}^{-1} \mathbf{U}^T \boldsymbol{\Lambda} \sum_{i=1}^{n_T} \boldsymbol{\phi}_{i,T} \ . \end{aligned} \qquad (36)$$

Let us consider the parameters of the ratios $\frac{P(\mathbf{y}|M_E)}{P(\mathbf{y})}$ and $\frac{P(\mathbf{y}|M_T)}{P(\mathbf{y})}$, given by:

$$\overline{\eta}_E^p = \begin{bmatrix} \mathbf{U}^T \boldsymbol{\Lambda} \sum_{i=1}^{n_E} \boldsymbol{\phi}_{i,E} \\ \mathrm{vec}(n_E \mathbf{U}^T \boldsymbol{\Lambda} \mathbf{U}) \end{bmatrix} \ , \quad \overline{\eta}_T^p = \begin{bmatrix} \mathbf{U}^T \boldsymbol{\Lambda} \sum_{i=1}^{n_T} \boldsymbol{\phi}_{i,T} \\ \mathrm{vec}(n_T \mathbf{U}^T \boldsymbol{\Lambda} \mathbf{U}) \end{bmatrix} \ , \qquad (37)$$

where the superscript p refers to *proper* PLDA scoring. Using these definitions, the natural parameters of the distribution $P(\mathbf{y}|M_E)$ and $P(\mathbf{y}|M_T)$ can be written as:

$$\eta_E^p = \overline{\eta}_E^p + \eta_0 \ = \begin{bmatrix} \mathbf{U}^T \boldsymbol{\Lambda} \sum_{i=1}^{n_e} \boldsymbol{\phi}_{i,E} \\ \mathrm{vec}(\mathbf{I} + n_E \mathbf{U}^T \boldsymbol{\Lambda} \mathbf{U}) \end{bmatrix} \ . \qquad (38)$$

By analogy, η_T^p is given by:

$$\eta_T^p = \overline{\eta}_T^p + \eta_0 \ , \qquad (39)$$

and, using (28), the natural parameters of the posterior distribution $P(\mathbf{y}|M_E, M_T)$ can be obtained as:

$$\eta_{E,T}^p = \overline{\eta}_E^p + \overline{\eta}_T^p + \eta_0 = \begin{bmatrix} \mathbf{U}^T \boldsymbol{\Lambda} \left(\sum_{i=1}^{n_E} \boldsymbol{\phi}_{i,E} + \sum_{i=1}^{n_T} \boldsymbol{\phi}_{i,T} \right) \\ \mathrm{vec}(\mathbf{I} + (n_E + n_T)\, \mathbf{U}^T \boldsymbol{\Lambda} \mathbf{U}) \end{bmatrix} \qquad (40)$$

B. PLDA scoring with i–vector averaging

Assuming that the set of enrollment and test i–vectors are replaced by the corresponding i–vector means, the PLDA posteriors are given by:

$$\begin{aligned} \boldsymbol{\Lambda}_{\mathbf{y},E} &= \mathbf{I} + \mathbf{U}^T \boldsymbol{\Lambda} \mathbf{U} \ , \quad \boldsymbol{\mu}_{\mathbf{y},E} = \boldsymbol{\Lambda}_{\mathbf{y},E}^{-1} \mathbf{U}^T \boldsymbol{\Lambda} \frac{1}{n_E} \sum_{i=1}^{n_E} \boldsymbol{\phi}_{i,E} \ , \\ \boldsymbol{\Lambda}_{\mathbf{y},T} &= \mathbf{I} + \mathbf{U}^T \boldsymbol{\Lambda} \mathbf{U} \ , \quad \boldsymbol{\mu}_{\mathbf{y},T} = \boldsymbol{\Lambda}_{\mathbf{y},T}^{-1} \mathbf{U}^T \boldsymbol{\Lambda} \frac{1}{n_T} \sum_{i=1}^{n_T} \boldsymbol{\phi}_{i,T} \ . \end{aligned} \qquad (41)$$

In this case, the natural parameters of $P(\mathbf{y}|E)$ and $P(\mathbf{y}|T)$ are:

$$\eta_E^a = \overline{\eta}_E^a + \eta_0 \ , \quad \eta_T^a = \overline{\eta}_T^a + \eta_0 \ , \qquad (42)$$

where the superscript a refers to *average* PLDA scoring, and $\overline{\eta}_E^a$ and $\overline{\eta}_T^a$ are given by:

$$\overline{\eta}_E^a = \begin{bmatrix} \mathbf{U}^T\mathbf{\Lambda}\frac{1}{n_E}\sum_{i=1}^{n_E}\boldsymbol{\phi}_{i,E} \\ \text{vec}(\mathbf{U}^T\mathbf{\Lambda}\mathbf{U}) \end{bmatrix} \quad \overline{\eta}_T^a = \begin{bmatrix} \mathbf{U}^T\mathbf{\Lambda}\frac{1}{n_T}\sum_{i=1}^{n_T}\boldsymbol{\phi}_{i,T} \\ \text{vec}(\mathbf{U}^T\mathbf{\Lambda}\mathbf{U}) \ . \end{bmatrix} . \tag{43}$$

Comparing (43) and (37) we can observe that

$$\overline{\eta}_E^a = \frac{1}{n_E}\overline{\eta}_E^p \ , \quad \overline{\eta}_T^a = \frac{1}{n_T}\overline{\eta}_T^p \ . \tag{44}$$

The parameters of the distributions that are used for PLDA scoring with average i–vectors are, thus, related to the natural parameters of the proper PLDA scoring by:

$$\eta_E^a = \frac{\overline{\eta}_E^p}{n_E} + \eta_0 \ , \quad \eta_T^a = \frac{\overline{\eta}_T^p}{n_T} + \eta_0 \ . \tag{45}$$

The natural parameters of the distribution $P(\mathbf{y}|E,T)$ can again be obtained as:

$$\eta_{E,T}^a = \overline{\eta}_E^a + \overline{\eta}_T^a + \eta_0 = \eta_E^a + \eta_T^a - \eta_0 \ . \tag{46}$$

C. Heterogeneous NL–PLDA scoring with i–vector averaging

Equation (45) provides a method for extending the i–vector averaging scoring approach to Heterogeneous NL–PLDA. In this case, the parameters $\overline{\eta}_E^p$ and $\overline{\eta}_T^p$ can be computed on the basis of (30) as:

$$\overline{\eta}_E^p = \begin{bmatrix} \sum_{k=1}^{K}\mathbf{U}_k^T\mathbf{\Lambda}_k\sum_{i=1}^{n_{E,k}} f_k(\boldsymbol{\phi}_{ki,E}) \\ \text{vec}(\sum_{k=1}^{K} n_{k,E}\mathbf{U}_k^T\mathbf{\Lambda}_k\mathbf{U}_k) \end{bmatrix} ,$$

$$\overline{\eta}_T^p = \begin{bmatrix} \sum_{k=1}^{K}\mathbf{U}_k^T\mathbf{\Lambda}_k\sum_{i=1}^{n_{T,k}} f_k(\boldsymbol{\phi}_{ki,T}) \\ \text{vec}(\sum_{k=1}^{K} n_{k,T}\mathbf{U}_k^T\mathbf{\Lambda}_k\mathbf{U}_k) \end{bmatrix} , \tag{47}$$

and the natural parameters used for scoring are:

$$\eta_E^a = \frac{\overline{\eta}_E^p}{n_E} + \eta_0 = \begin{bmatrix} \frac{1}{n_E}\sum_{k=1}^{K}\mathbf{U}_k^T\mathbf{\Lambda}_k\sum_{i=1}^{n_{k,E}} f_k(\boldsymbol{\phi}_{ki,E}) \\ \text{vec}(\mathbf{I} + \frac{1}{n_E}\sum_{k=1}^{K} n_{k,E}\mathbf{U}_k^T\mathbf{\Lambda}_k\mathbf{U}_k) \end{bmatrix} ,$$

$$\eta_T^a = \frac{\overline{\eta}_T^p}{n_T} + \eta_0 = \begin{bmatrix} \frac{1}{n_T}\sum_{k=1}^{K}\mathbf{U}_k^T\mathbf{\Lambda}_k\sum_{i=1}^{n_{k,T}} f_k(\boldsymbol{\phi}_{ki,T}) \\ \text{vec}(\mathbf{I} + \frac{1}{n_T}\sum_{k=1}^{K} n_{k,T}\mathbf{U}_k^T\mathbf{\Lambda}_k\mathbf{U}_k) \end{bmatrix} , \tag{48}$$

with $n_E = \sum_{k=1}^{K} n_{k,E}$, $n_T = \sum_{k=1}^{K} n_{k,T}$, and

$$\eta_{E,T}^a = \eta_E^a + \eta_T^a - \eta_0 \ . \tag{49}$$

V. NL–Tied–PLDA versus other approaches

The NL–Tied–PLDA model can be used whenever a set of i–vectors can be characterized as belonging to different classes. For example, i–vectors extracted by the same system but belonging to classes defined according to some characteristics of the corresponding speech segments, such as their duration or their signal–to–noise level. As mentioned in the introduction section, we target a "migration" scenario, where a company releases a new speaker verification system. The new system should offer the possibility that a user is recognized using her/his new test i–vectors scored against i–vectors extracted at the time of enrollment with a previously released system.

We will analyze the performance of the NL–Tied–PLDA model on heterogeneous trials as a function of the number of enrollment i–vectors per speaker, and as a function of the availability of new enrollment speech segments. Our model will be also compared with three other techniques that will be recalled and commented in the next subsections: the "migration" approach illustrated in [10], the "Twin–PLDA" model proposed in [23], and the improved model introduced in [24], which will be referred to as the "Double–PLDA" model. Improved versions of the first and of the latter will be introduced in Section VI.

A. Migration approach

The "migration" technique of [10] allows transforming heterogeneous i–vectors so that they become compatible with the ones extracted by a new system, provided that there is a training set of i–vectors generated, for the same utterance, both by the old and the new system. The transformation can be obtained by training various topologies of Regression Neural Networks, but the results of the experiments in [10] indicate that a simple Multivariate Linear Regression ("LinRegr" for short, in the following) performs better than Neural Networks with hidden layers.

LinRegr estimates a linear matrix $\boldsymbol{A}$ independently of the classifier that will be used. It does not need the labels of the i–vectors because it estimates its transformation matrix from pairs of i–vectors of the same utterance produced by two different systems, trying to make them "similar". Training a LinRegr matrix, is simple and fast, and its performance is good in some conditions, as confirmed by our experiments illustrated in Section VII. It has, however, a main drawback with respect to our Tied–PLDA, which does not perform any i–vector migration, but allows comparing heterogeneous speaker vectors by estimating a non–linear transformation for each class, tailored to maximize the log–likelihood of the development dataset. According to (1), and considering $\mathbf{m} = \mathbf{0}$ for simplicity, homogeneous i–vectors of two classes are generated according to the two models:

$$\begin{aligned} \boldsymbol{\phi}_{1i,s} &= \mathbf{U}_1\mathbf{y}_s + \boldsymbol{\epsilon}_{1i,s} \\ \boldsymbol{\phi}_{2i,s} &= \mathbf{U}_2\mathbf{y}_s + \boldsymbol{\epsilon}_{2i,s} \ , \end{aligned} \tag{50}$$

with the residual terms $\boldsymbol{\epsilon}_{1i,s}$ and $\boldsymbol{\epsilon}_{2i,s}$ sampled from $\mathcal{N}(\mathbf{0}, \mathbf{\Lambda}_{1i}^{-1})$ and $\mathcal{N}(\mathbf{0}, \mathbf{\Lambda}_{2i}^{-1})$, respectively. Since LinReg per-

forms a linear transformation of i–vectors, the transformed i–vectors are generated by the model:

$$\boldsymbol{\phi}_{2i,s} = \mathbf{A}\boldsymbol{\phi}_{1i,s} = \mathbf{A}\mathbf{U}_1\mathbf{y}_s + \mathbf{A}\boldsymbol{\epsilon}_{1i,s} \tag{51}$$

The migration model, thus, assumes that:

$$\mathbf{U}_2 \approx \mathbf{A}\mathbf{U}_1 \tag{52}$$

$$\boldsymbol{\epsilon}_{2i,s} \approx \mathbf{A}\boldsymbol{\epsilon}_{1i,s} \ , \tag{53}$$

and it fails if these assumptions are not satisfied. Please notice that (53) follows from the assumption that the i–vectors belong to the same utterance, and implies that:

$$\boldsymbol{\Lambda}_{1i} \approx \mathbf{A}\boldsymbol{\Lambda}_{2i}\mathbf{A}^T \ . \tag{54}$$

NL–Tied–PLDA, instead, needs the speaker labels, but it explicitly models the intra–speaker and inter–speaker variability of each class of speaker vectors. Even excluding the additional parameters in non–linear transformation of NL–Tied–PLDA, the linear Heterogeneous PLDA model (Tied–PLDA) of (50) is more accurate than the LinRegr model because it estimates four different matrices: $\mathbf{U}_1$, $\boldsymbol{\Lambda}_1$, $\mathbf{U}_2$, and $\boldsymbol{\Lambda}_2$ with the unique constraint that vectors of different classes, but belonging to the same speaker, share their speaker factor $\mathbf{y}_s$.

B. Twin PLDA model

The Twin PLDA model, proposed in [23] for duration mismatch compensation, does not include non–linear transformations, but is based on the same assumption of our NL–Tied–PLDA model that i–vectors from the same speakers share identical latent variables. However, in [23] the model parameters are estimated by creating merged i–vectors, i.e., concatenating an i–vector from a long utterance and one from a short utterance of the same speaker. This choice may lead to an incorrect estimation of the log–likelihood of the development data distribution, and of the speaker factor posterior. Indeed, it considers different pairs as independent, whereas many pairs include the same i–vector. In particular, the log–likelihood of an i–vector from a long utterance of a speaker, who also provides many short utterances to the development set, would contribute more than the i–vector of a long utterance of another speaker who provides fewer short utterances. The same problem would appear for the i–vectors extracted by different systems if the number of development i–vectors provided by two speakers are different.

C. Double–PLDA models

An improved model based on the Twin–PLDA model has been proposed in [24], which, excluding the non–linear transformations, is similar to the Tied–PLDA model. However, in [24] the derivation of the posterior distribution of the speaker factors is not detailed, and the paper is mainly focused on the use of a sequence of two factor analyzers, i.e., two PLDA models. The first is a Tied–PLDA model, devoted to the extraction of a speaker factor for each i–vector. These factors represent a mapping of the original i–vectors to a common latent variable space, and a second PLDA is trained with these transformed i–vectors. We will refer to this approach in the following as the Double–PLDA model.

VI. Improved models

In this section we illustrate how the migration and the Double-PLDA approach can be improved, and how these improved models can be combined with non–linear transformations.

A. Improved migration models

In the original migration approach proposed in [10], the target PLDA model is trained only with the target development i–vectors. The original and target development i–vectors are only used to train the "migration" matrix $\mathbf{A}$. We propose, instead, that a new PLDA model is trained by adding the set of original development i–vectors, transformed according to matrix $\mathbf{A}$, to the target development i–vector set. This makes the LinRegr training procedure more similar to the one performed in our Tied–PLDA approach, which jointly estimates the linear transformation and the PLDA parameters, using the original and target development sets. We will refer to the LinRegr model with retrained PLDA as "LinRegr+Retraining". This models, of course, needs the speaker labels.

Furthermore, in [10], the original and target i–vectors are length normalized before estimating the transformation matrix $\mathbf{A}$. Since length–normalization (LN) is beneficial to PLDA [45], because it reduces the mismatch between the development and evaluation sets, but it is ineffective or even detrimental to NL–PLDA, we argue that LinRegr should be more effective if it is applied to the original i–vectors, as confirmed by the experiments on standard PLDA reported in Section VII. The transformed i–vectors should be then length–normalized for PLDA scoring, whereas an i–vector dependent scaling factor that optimizes the NL–PLDA objective function, must be used for NL–PLDA [36].

B. Improved Double–PLDA models

In the Double–PLDA model of [24], the i–vector transformation proposed in equations (8–10) does not correspond to a MAP point estimate, and is linear because it is class–independent. Since, as far as scoring is involved, a PLDA model is not affected by a linear transformations of the trial i–vectors, this transformation should not produce any benefit with respect to the use of the original i–vectors. Any performance difference with respect to PLDA, thus, can only depend on the additional length–normalization performed in [24] on the linearly transformed i–vectors.

In Section VII, we compare the performance of our linear and non–linear Tied–PLDA model with a Double–PLDA model similar to [24], but with the transformed i–vectors properly computed as:

$$\tilde{\boldsymbol{\phi}}_{ki,s} = \left[\left(\mathbf{I} + \mathbf{U}_k^T \boldsymbol{\Lambda}_k \mathbf{U}_k \right)^{-1} \mathbf{U}_k^T \boldsymbol{\Lambda}_k \right] f_k(\boldsymbol{\phi}_{ki,s}) \ , \tag{55}$$

where k is the i–vector class (original or target). It is worth noting that (55) is exactly the standard speaker factor MAP point

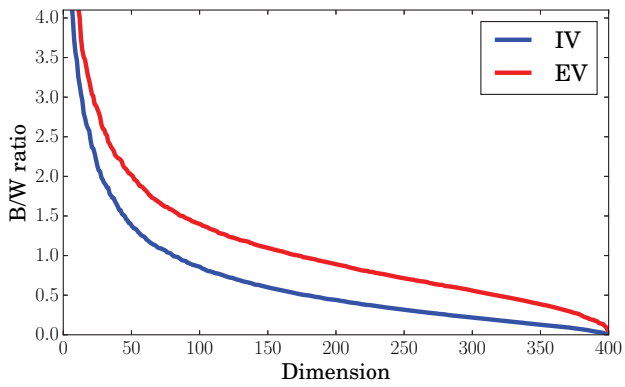

Fig. 1: Between to Within variance ratios of i—vectors and e–vectors components sorted in descending order.

estimate given by equation (8), but the estimation is performed independently for each i–vector. Our proposed improvement to the Double–PLDA approach consists in computing a class–dependent speaker factor for each i–vector, using Tied–PLDA or NL–Tied–PLDA. These speaker factors are then used for training a standard PLDA model. We will refer to these models in the following as the "Double–Tied–PLDA" or "Double–NL–Tied–PLDA".

It is worth noting that these models are an approximation of the Tied–PLDA models that allow simplifying multi–enrollment scoring when the number of enrollment i–vectors of each class is variable. Indeed, classical PLDA scoring with averaged i–vectors can be performed efficiently by pre–computing the covariance matrices of the posteriors of the speaker factors that appear in the likelihood–ratio (35). This pre–computation is possible because the posterior covariances only depend on the number of test and enrollment utterances. Using average i–vectors is like having single enrollment and test utterances, regardless of their actual number.
For the Tied–PLDA models, instead, even when the "averaging" procedure of Section IV-B is applied, the posterior covariances depend on the number of utterances per class, which can be different for each class. Therefore, the number of covariance matrices that should be pre–computed for obtaining fast scoring grows with the combination of the different number of enrollment and test utterances that are available for each class and each speaker. Thus, fast scoring is possible only when the variability of the number of utterances per class is low. The Double–PLDA models, on the other hand, extract a single speaker factor for each i–vector, mapping i–vectors from different classes to a common latent space independently. Fast scoring is thus possible because it is performed by using a standard PLDA model, and the i–vector averaging scoring procedure.

VII. Experiments

In this Section we evaluate the linear and non–linear approaches that have been illustrated in the previous Sections, comparing their performance for trials including speaker vectors obtained by different extraction procedures. Tests have been performed both on the extended core NIST SRE 2012, and on the female extended core NIST SRE 2010 evaluations.

Since every speaker in the NIST SRE 2012 dataset can be enrolled with multiple utterances (approximately 19 on average), in the first set of experiments we will consider that all the original enrollment i–vectors are available, but not the corresponding utterances, which does not allow extracting i–vectors using the new system. The second set of experiments, instead, focuses on a migration scenario from an old system to a new and better performing system, assuming that only a small subset of the old enrollment i–vectors for each speaker are available, or that a few additional enrollment utterances have been collected for the new system.

A. Acoustic features and speaker vectors

1) 45–MFCC — GMM — i–vectors for SRE 2012: The first feature set consists of 45–dimensional feature vectors obtained by stacking 18 cepstral (c_1–c_{18}), 19 delta (Δc_0–Δc_{18}) and 8 double–delta ($\Delta\Delta c_0$–$\Delta\Delta c_7$) parameters.

We trained a gender–independent i–vector extractor, based on a 2048–component full covariance UBM, estimated with data from NIST SRE 2004–2010, and additionally with the Switchboard II, Phases 2 and 3, and Switchboard Cellular, Parts 1 and 2 datasets.
The i–vector dimension was set to $d = 400$.

2) 60–PLP — DNN/GMM — e–vectors for SRE 2012: The second set of features are 60–dimensional PLP feature vectors obtained by stacking 20 RASTA Perceptual Linear Predictive (PLP) coefficients [46], and their first and second derivatives.

In this case, we trained a gender–independent e–vector extractor [11], [12]. The e–vector representation of a speech segment is similar to the speaker factors of Joint Factor Analysis (JFA) and to i–vectors. It is based on the observation that JFA estimates a more informative speaker subspace than the "total variability" i–vector subspace, because the latter is obtained by considering each training segment as belonging to a different speaker. E–vectors are obtained by replacing the "total variability" matrix $\mathbf{T}$ by a new variability matrix, which allows keeping the span of the speaker–specific eigenvoice subspace, but at the same time provides a better prior for i–vector extraction.

The e–vector extractor for these experiments is based on a gender–independent 2048–component full covariance UBM, using the hybrid DNN/GMM approach of [3]. In particular, we used the approach and the DNN described in [6], associating 8 Gaussians to each of 256 output units of the DNN. The training data for this system are the same used for training the 45–MFCC based GMM model.
The e–vector dimension was set to $d = 400$.

We focused preferably on the e–vectors as heterogeneous set of features with respect to i–vectors not only because the former have shown in [11], [12] to produce better results than i–vectors using different systems and classifiers, but also because they are more "heterogeneous" with respect to the 45–MFCC based i–vectors. Indeed, some i–vector directions convey channel information, whereas e–vector directions mainly convey speaker information. The e–vectors components have

TABLE I: Average % Equal Error Rate, min DCF08, and min $C_{primary}$ for homogeneous trials of the extended NIST SRE 2012 evaluation using different features (45–MFCC and 60–PLP) and identity vector extraction systems (i–vectors and e–vectors).

PLDA average performance				
Features	vectors	% EER	min DCF08	min $C_{primary}$
45–MFCC	i–vectors	3.26	0.134	0.301
60–PLP	i–vectors	2.94	0.116	0.262
60–PLP	e–vectors	2.52	0.107	0.236
Non–Linear PLDA average performance				
45–MFCC	i–vectors	3.30	0.120	0.264
60–PLP	e–vectors	2.58	0.098	0.208

TABLE II: Comparison of the average % Equal Error Rate, min DCF08, and min $C_{primary}$ of the Tied–PLDA and LinRegr approaches, with full enrollment data available, on heterogeneous trials of the extended NIST SRE 2012 evaluation.

System	% EER	min DCF08	min $C_{primary}$
Tied–PLDA	2.92	0.122	0.272
LinRegr towards 60–PLP	2.88	0.133	0.359
LinRegr towards 45–MFCC	3.32	0.132	0.293

a different distribution of their Between to Within variance ratio with respect to i–vectors. This is shown in Figure 1, which plots the Between to Within variance ratio of the i–vector and e–vectors components sorted by their value, i.e., according to their importance for speaker discrimination. The e–vector ratios are larger than the i–vector ones, but most important to notice is that the ratio of the plotted values along the dimension axis is not constant. Thus, the assumption of the migration approach that a linear transformation, such as (52), is sufficient to make consistent i–vectors and e–vectors is inaccurate.

3) 45–MFCC — GMM — i–vectors for SRE 2010: To further validate the heterogeneous PLDA models, we performed additional experiments on the female core extended NIST SRE 2010 evaluations. The first set of i–vectors for these experiments is based again on the 45–MFCC features, illustrated in Section VII-A1, but a gender–dependent 400–dimensional i–vector extractor was trained, based on a 2048–component diagonal covariance UBM, and trained with the SRE04–06, and the Switchboard datasets used for the SRE 2012 gender–independent i–vector extractor.

4) 45–MFCC — DNN/GMM — e–vectors for SRE 2010: The second set consists of 400–dimension gender–dependent e–vectors based on the same 45–MFCC features used for the other systems. The extractor is based on the same hybrid DNN/GMM approach used for the SRE 2012 e–vector system. The extractor was trained with the same datasets that were used for training the 45–MFCC SRE 2010 i–vector system, but with the addition of SRE 2008 and Fisher datasets, so that the number of speakers was large enough to reliably estimate the e–vector subspace.

B. Results on NIST 2012

The baseline results for homogeneous trials of the extended NIST SRE 2012 evaluation using different identity vector extraction systems and features (45–MFCC and 60–PLP) are summarized in Table I in terms of average, over Conditions 1 to 5, of the percent Equal Error Rate, min DCF08, and $C_{primary}$ defined by NIST [47]. The first frame of Table I refers to the results of standard PLDA systems, whereas the second frame reports the performance of the corresponding Non–Linear PLDA systems. Considering the average $C_{primary}$ cost function, the 60–PLP PLDA i–vector system performance is 13% better than the 45–MFCC i-vector system, and the 60–PLP e–vector system improves the average $C_{primary}$ by an additional 9%. The Non–Linear PLDA 45–MFCC i–vector system is 12% better than the corresponding PLDA system, and NL–PLDA using e–vector keeps approximately the same relative improvement with respect to the corresponding i–vector based NL–PLDA. Since the 60–PLP i–vectors systems are worse than the corresponding e–vector systems, and since the 60–PLP e–vectors are more "heterogeneous" with respect to the 45–MFCC based i–vectors, in the following the 45–MFCC ivector and 60–PLP e–vector system will be taken as reference for the original (old) and the target (new) system, respectively.

Table II compares the average performance of the LinRegr and Tied–PLDA approaches scoring heterogeneous trials, i.e., trials including enrollment i–vectors of the old system, and test e–vectors extracted by the new system. The Heterogeneous PLDA model achieves an average $C_{primary}$ of 0.272, which is 9% better than the result of the old model on homogeneous trials. The additional information provided by the e–vectors representation is, thus, beneficial to Tied–PLDA. On the contrary, the migration of the 45–MFCC i–vectors towards the 60–PLP e–vectors is detrimental to the average performance, as shown in the second row of Table II. Considering the results per condition, the second row of Table III shows that the migration of the 45–MFCC i–vectors towards the 60–PLP e–vectors slightly improves the performance with respect to the old model only for telephone Conditions 2 ad 5, but it fails on the other test conditions, including Condition 4 where noise has been added to phone calls. Overall, using LinRegr leads to a 16% loss of the average $C_{primary}$.

Last row of the Tables II and III shows the results of LinRegr migrating the 60–PLP e–vectors toward the 45–MFCC i–vectors, i.e., toward the old system. In this configuration, the achieved average $C_{primary}$ is 0.293, which is far better than the one obtained using the more reasonable configuration of the second row. It is interesting to interpret the asymmetry arising in these two conditions because this analysis allows us illustrating one of the weakness of this approach. The results depend on the different nature of the two representations that LinRegr tries to match. As already said, some i–vector directions convey channel information, whereas all e–vectors directions mainly convey speaker information. Migration towards the 60–PLP e–vector system leads to bad results because the i–vectors are transformed to e–vectors. Since e–vectors convey mainly speaker information over all their components, PLDA trained on e–vectors considers all directions important for speaker discrimination. However, some components of the 45–MFCC i–vectors mainly convey channel information. PLDA trained on e–vectors, therefore, considers some channel variability of the transformed i–vectors as speaker variability. In the migration towards the 45–MFCC system, instead, the

TABLE III: Comparison of the % Equal Error Rate, min DCF08, and min $C_{primary}$ (C_{12} for short) of the Tied–PLDA and LinRegr approaches, with full enrollment data available, on heterogeneous trials of the extended NIST SRE 2012 evaluation.

System	Cond 1 interview without added noise			Cond 2 phone call without added noise			Cond 3 interview with added noise			Cond 4 phone call with added noise			Cond 5 phone call noisy		
	% EER	min DCF08	min C_{12}	% EER	min DCF08	min C_{12}	% EER	min DCF08	min C_{12}	% EER	min DCF08	min C_{12}	% EER	min DCF08	min C_{12}
Tied–PLDA	2.91	0.112	0.240	1.83	0.091	0.244	2.75	0.112	0.200	4.81	0.184	0.389	2.29	0.113	0.285
LinRegr towards 60–PLP	2.62	0.120	0.409	1.62	0.082	0.241	2.40	0.147	0.432	5.77	0.213	0.429	2.01	0.103	0.284
LinRegr towards 45–MFCC	3.38	0.120	0.248	2.12	0.102	0.264	3.49	0.113	0.218	5.04	0.201	0.423	2.58	0.124	0.310

TABLE IV: Average % Equal Error Rate, min DCF08, and min $C_{primary}$ for trials of the extended NIST SRE 2012 evaluation: comparison of all models enrolled with the full enrollment set.

Trials	System	Each speakers enrolled with all available i–vectors and e–vectors					
		Linear			Non–Linear		
		% EER	min DCF08	min $C_{primary}$	% EER	min DCF08	min $C_{primary}$
Homogeneous	45–MFCC PLDA i–vectors	3.26	0.134	0.301	3.30	0.120	0.264
	60–PLP PLDA e–vectors	2.52	0.107	0.236	2.58	0.098	0.208
Heterogeneous	LinRegr	2.88	0.133	0.359	-	-	-
	LinRegr–Raw	3.27	0.129	0.286	3.31	0.130	0.286
	LinRegr+Retrain	2.87	0.124	0.286	-	-	-
	LinRegr–Raw+Retrain	2.88	0.120	0.270	3.15	0.119	0.256
	Tied–PLDA	2.92	0.122	0.272	3.02	0.115	0.247
	Double–Tied–PLDA	2.95	0.119	0.273	2.79	0.108	0.238

e–vectors are transformed to i–vectors. PLDA trained on i–vectors discards the directions that mainly convey channel information. For the transformed e–vectors, this corresponds to the removal of some speaker information. This roughly corresponds to a dimension reduction of the e–vectors, which is known to have much lower impact on accuracy.

In the following, we will only refer, for the LinRegr approach and its improved model, to the "natural" migration of the i–vectors of the old system towards the e–vectors of the new system. We will also report, for the sake of readability, only the average results of the experiments performed on the five conditions.

Table IV summarize the performance of the linear and non–linear systems that have been introduced in Sections V and VI. The first two rows show the average performance of the old and new systems on homogeneous trials, taken from Table I. These results are given again in this table as references to make easier their comparison with the heterogeneous systems, which aim at obtaining better results than the old system, and at reducing the performance gap with respect to the new system.

The third row reports the average performance of LinRegr. The corresponding Non–Linear fields are not filled, because NL–PLDA provides sub–optimal performance when length–normalization is applied to i–vectors. The fourth row shows the results of the best configuration of the LinRegr approach, shown in row labeled "LinRegr–Raw". This label refers to the migration approach applied on raw i–vectors, i.e., not subject to previous length–normalization. LN is applied, instead, to the migrated i–vectors, for PLDA classification, whereas the i–vector dependent scaling approach presented in [36] is used for NL–PLDA classification. It is worth noting that this approach is much more effective than the original one for standard PLDA, as can be observed looking at the $C_{primary}$, which shows a relative improvement of 17%. However, using a NL–PLDA classifier on "migrated" i–vectors, rather than a standard PLDA, both trained with the e–vector development set only, does not give any improvement, on the contrary, it gets worse performance with respect to the old system scoring homogeneous trials. This is not surprising, because NL–PLDA models the e–vector distribution more accurately than PLDA, but the transformed i–vectors typically have a different distribution. Thus, a further step towards better heterogeneous LinRegr based models consists in training a new PLDA classifier using the new e–vector development set and, in addition, the transformed i–vectors of the old development set. This improved model (LinRegr+Retraining) largely improves the performance with respect to the original LinRegr model. Even better results are obtained by the improved model LinRegr–Raw+Retraining, which not only improves the performance of the linear model, but it is even more effective if a NL–PLDA classifier is trained using the vectors produced by LinRegr–Raw. The relative improvement of $C_{primary}$ with respect to the original formulation of LinRegr–Raw is 5% and 10%, respectively.

Finally, the Tied–PLDA and the Double–Tied–PLDA models perform as well as our LinRegr–Raw+Retrain model, whereas non–linear Tied models improve $C_{primary}$ up to 7% with respect to LinRegr–Raw followed by a retrained NL–PLDA classifier. Overall, the performance of the NL–Tied–PLDA models scoring heterogeneous trials is somewhere in the middle between the old and new systems scoring their homogeneous trials.

C. Results on NIST 2012 with reduced enrollment sets

In the previous section we compared several approaches that allow comparing heterogeneous vectors assuming that all the enrollment i–vectors of the old system, but not the e–vectors of the corresponding utterances, were available. In this

TABLE V: Average % Equal Error Rate, min DCF08, and min $C_{primary}$ for trials of the extended NIST SRE 2012 evaluation: comparison of models with a reduced amount of enrollment data per speaker.

(a) Each speaker provides only two old i–vectors as enrollment

Trials	System	Linear			Non–Linear		
		% EER	min DCF08	min $C_{primary}$	% EER	min DCF08	min $C_{primary}$
Homogeneous	45–MFCC PLDA i–vectors	4.19	0.183	0.393	3.88	0.160	0.338
	60–PLP PLDA e–vectors (two old utterances)	3.22	0.141	0.302	3.58	0.139	0.275
Heterogeneous	LinRegr–Raw	4.63	0.197	0.394	5.00	0.203	0.400
	LinRegr–Raw+Retrain	4.40	0.184	0.373	4.43	0.173	0.345
	Tied–PLDA	3.88	0.171	0.357	3.92	0.159	0.326
	Double–Tied–PLDA	3.64	0.166	0.369	3.56	0.147	0.311

(b) Each speaker provides two old i–vectors and two new e–vectors as enrollment

Trials	System	Linear			Non–Linear		
		% EER	min DCF08	min $C_{primary}$	% EER	min DCF08	min $C_{primary}$
Homogeneous	60–PLP PLDA e–vectors (two new e–vectors only)	3.42	0.143	0.302	3.37	0.135	0.270
	60–PLP PLDA e–vectors (four e–vectors)	2.82	0.118	0.263	2.76	0.109	0.232
Heterogeneous	LinRegr–Raw	2.88	0.124	0.292	3.09	0.120	0.255
	LinRegr–Raw+Retrain	2.81	0.120	0.275	3.16	0.121	0.258
	Tied–PLDA	2.85	0.120	0.270	2.86	0.111	0.236
	Double–Tied–PLDA	2.82	0.121	0.272	2.76	0.109	0.235

section, we examine the effects of the availability of a reduced enrollment set on the performance of both homogeneous and heterogeneous trials with different systems.
In particular, we consider the following two scenarios:

- only two enrollment i–vectors per speaker, on average, from the old 45–MFCC system are available, but the corresponding audio segments are not,
- in addition to the two old enrollment i–vectors, the audio files of two new enrollment utterances per speaker, on average, are available.

In our experiments, the two new enrollment utterances for each speaker do not overlap with the ones represented by the two i–vectors. We decided to use only a fraction of the enrollment data per speaker to stress the effects of the usage of an incremental number of speaker vectors on the performance of the systems. The segments were chosen randomly (a single random selection was considered).

The results obtained on homogeneous and heterogeneous trials by using different systems, using only two enrollment i–vectors per speaker, are shown in Table V (a). It is not surprising to observe a relevant performance loss with respect to the use of the complete set given in Table IV.

The first row of Table V (a) reports the results for the first scenario. In this case, homogeneous trials can only be scored by the old system, which gives, thus, the "lower bound" reference results, to be compared with the ones obtained by the other systems appearing in the remaining rows of the table.
An "upper bound" reference can be reached if, rather than only the two i–vectors per speaker, we can exploit the corresponding audio segments to extract the e–vectors for the new system. These results are shown in the second row of Table V (a). Similar performance is, of course, obtained if the audio segments come from data collected for the new systems, as shown in the first row of Table V (b).

Comparing the results of the LinRegr models in rows three and four of Table V (a), we see that better performance is obtained, mostly for the Non–Linear PLDA, by retraining a classifier after the migration of the old development i–vectors. However, the corresponding results for the non–linear models are worse that the baseline results of the old system. On the contrary, both Tied–PLDA and NL–Tied–PLDA are better than the LinRegr+Retrain approach, and even more improvement is obtained by using the Double-NL–Tied–PLDA model. This latter gets a $C_{primary}$ of 0.311, which is 11% worse than the homogeneous upper bound, but 8% better than the lower bound. These results show that this non–linear model allows effectively scoring new test e–vectors against speaker enrolled with the old system only.

Collecting two new enrollment utterances per speaker allows dramatically improving system performance both for homogeneous and heterogeneous trials, as shown in Table V (b). The first and second row of the table give the lower and the upper bound reference performance of the old and new system on homogeneous trials, respectively. Four enrollment e–vectors per speaker, corresponding to the two old and two new utterances, are used for the upper bound configuration. As expected, using more utterances per speaker, the performance of all systems has a significant increase. In this scenario, and looking at the column of the non–linear models, the results of the LinRegr approaches, without and with retraining, are similar, but again 8% worse than the Heterogeneous models in terms of $C_{primary}$. The two non–linear heterogeneous models achieve similar results, with a small advantage for the Double–NL–Tied–PLDA model.

It is worth noting that NL–Tied–PLDA models (single or Double), scoring heterogeneous trials, reach almost the upper bound performance of NL–PLDA scoring homogeneous trials shown in row two of Table V (b).

TABLE VI: % Equal Error Rate , min DCF08, and min DCF10 for trials of the female extended NIST SRE 2010 evaluation: comparison of different systems.

Trials	System	Linear			Non–Linear		
		% EER	min DCF08	min DCF10	% EER	min DCF08	min DCF10
Homogeneous	45–MFCC i–vectors	2.24	0.120	0.401	2.19	0.102	0.366
	45–MFCC e–vectors	1.40	0.071	0.242	1.22	0.059	0.247
Heterogeneous	LinRegr–Raw+Retrain	2.20	0.107	0.350	2.32	0.099	0.368
	Tied–PLDA	2.09	0.105	0.348	1.89	0.095	0.329

The analysis of these results suggests that a good strategy for a smooth migration from an old to a better performing system that uses different speaker vectors, depends on the availability of old and new enrollment data. In the first phase of the migration, when a speaker has not yet provided new enrollment utterances, if his/her enrollment utterances were not saved, it is still possible to score a new test utterance using the heterogeneous approaches. If old enrollment utterances were not saved, but a few new enrollment utterances have been collected, it is advisable to still exploit the old speaker enrollment vectors (compare the first and last rows of Table V (b)), until enough new enrollment data are available.

D. Results on NIST 2010

The experiments on the female extended NIST SRE 2010 evaluation have been performed by using an "old" system, based on 45–MFCC features, 400–dimensional i–vectors, and standard GMM models, whereas the "new" system is based on 45–MFCC features, 400–dimensional e–vectors, and DNN/GMM models. It is worth recalling that in NIST 2012 SRE only a single enrollment utterance per speaker is available. We evaluated only the telephone–telephone condition (Condition 5) because the number of speakers in the training datasets is not sufficient to reliably estimate e–vectors for the microphone and interview conditions [12].

The results, both for the linear and for the non–linear PLDA classifiers, are summarized in Table VI in terms of percent Equal Error Rate, min DCF08, and min DCF10 defined by NIST [48]. The results with the "Double–Tied–PLDA" or "Double–NL–Tied–PLDA" models are not reported because they are similar to the corresponding Tied–PLDA and NL–Tied–PLDA, respectively.

The first two rows show the performance of the old and new systems on homogeneous trials. It is worth noting that due to the small number of false alarms at the DCF10 operating point, DCF08 is more significant as a performance index for NIST 2010 tests. The new system improves the min DCF08 of the old one by more than 40%. The third row reports the performance of the LinRegr–Raw+Retrain approach on heterogeneous trials, where we get a relative DCF08 improvement of 11% with respect to the old system. Using a linearized Heterogeneous PLDA classifier, the relative DCF08 improvement with respect to the old system is more than 12%, and still remain more than 6% for the NL–Tied–PLDA.

VIII. Discussion

The NL–Tied–PLDA formulation allows also addressing other tasks characterized by the necessity of scoring heterogeneous i–vectors. It could be used, for example, to address duration mismatch compensation, as the Twin–PLDA or Double–PLDA models of [23], [24] do. These works, however, target an experimental scenario with long enrollment, and short test utterances of fixed–duration. In a more realistic scenario the short utterances may have variable durations. However, Tied–PLDA models are better suited for tasks in which i–vectors can be associated to a fixed number of discrete classes; they are not appropriate for tasks in which it is difficult to define class boundaries. Our work on PLDA scoring using i–vector full posterior distributions (FPD–PLDA) [20], [21] has shown that it is reasonable to assume that duration has a direct impact on the within–class covariance of i–vectors, but the additional uncertainty is not easily summarized by a few class–dependent covariances. Indeed, in preliminary experiments scoring i–vectors extracted from short utterances of variable duration the FPD–PLDA model proposed in [20] provided better accuracy than the linear and non–linear Tied–PLDA models.

Our model assumes that all the training i–vectors are independent given the speaker factors. Although it has shown to be accurate, if the i–vectors are extracted from the same set of utterances, the independence assumption may be violated because both the phonetic and channel variability of two i–vectors of the same utterance are correlated. This might lead to worse performance with respect to a system that correctly models these correlations. It is worth noting that this issue affects also the LinRegr and Double–Tied–PLDA models. We have shown that LinRegr provides good performance if PLDA is trained using both the original and the transformed i–vectors. Also in this framework, PLDA assumes that i–vectors of the same utterances are independent. The same consideration applies to the Double–PLDA models, which consider the speaker factors extracted from the first Tied–PLDA as features for a second PLDA.

Our model can be extended to explicitly account for this issue by introducing an additional latent variable as follows:

$$\boldsymbol{\phi}_{ki,s} = \mathbf{U}_k \mathbf{y}_s + \mathbf{W}_k \mathbf{z}_{i,s} + \boldsymbol{\epsilon}_{ki,s} \ , \tag{56}$$

where, the subscripts k, i, and s refer to the class, to the i-th utterance of class k, and to the speaker, respectively. Similar to the original PLDA formulation, the term $\mathbf{z}_{i,s}$ represents the phonetic and channel variability, but it should be tied for i–vectors extracted from the same utterance.

The EM procedure can be adapted to estimate the additional matrix $\mathbf{W}_k$. However, since $\mathbf{y}_s$, and corresponding $\mathbf{z}_{i,s}$ are correlated in the posterior, the exact solution requires inverting covariance matrices of large size [7].

An approximate solution can be obtained by using Variational Bayes (VB) estimation [7]. Our EM procedure for the Tied–PLDA allows directly using the E–step and M–step of the standard PLDA models. This approach can be adapted to perform VB approximation, using a VB implementation of the single–class PLDA model. This possible extension deserves further investigation.

Finally, the heterogeneous scoring approach is not limited to PLDA models, but it can also be applied to other classifiers, such as the Pairwise Support Vector Machine (PSVM) of [49], [50]. A straightforward technique consists in replacing the second PLDA of the Double–PLDA model by a PSVM, whereas a more challenging approach aims at directly classifying heterogeneous trials by properly reformulating the PSVM parameter estimation. Both approaches are viable as assessed by the results of preliminary experiments.

IX. Conclusions

We have presented a generalization of the Non–Linear PLDA model that jointly estimates the distribution of the development i–vectors and the PLDA parameters of heterogeneous identity vectors, so that they are transformed to make PLDA classification more accurate.

We have improved a "migration" approach, and some linear models that have been proposed for duration mismatch compensation, and compared their performance with respect to the linear and non–linear Tied–PLDA.

We have shown that training Tied–PLDA models can be performed by EM, where the M–step can be implemented as a set of independent optimizations, each equivalent to the M–step and E–step of a single–class PLDA model.

Finally, we proposed an extension of the i–vector average scoring to heterogeneous trials.

Using these approaches it is possible to score heterogeneous i–vectors with consistent improvement over previous migration models, and in some tasks to reach the accuracy of the best homogeneous models.

References

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

[2] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted Gaussian Mixture Models," *Digital Signal Processing*, vol. 10, no. 1-3, pp. 31–44, 2000.

[3] Y. Lei, N. Scheffer, L. Ferrer, and M. McLaren, "A novel scheme for speaker recognition using a phonetically-aware Deep Neural Networks," in *Proceedings of ICASSP 2014*, pp. 1695–1699, 2014.

[4] D. Garcia-Romero, X. Zhang, A. McCree, and D. Povey, "Improving speaker recognition performance in the domain adaptation challenge using deep neural networks," in *Proceedings of SLT 2014*, pp. 378–383, 2014.

[5] D. Garcia-Romero and A. McCree, "Insights into Deep Neural Networks for speaker recognition," in *Proceedings of Interspeech 2015*, pp. 1141–1145, 2015.

[6] S. Cumani, P. Laface, and F. Kulsoom, "Speaker recognition by means of acoustic and phonetically informed GMMs," in *Proceedings of Interspeech 2015*, pp. 200–204, 2015.

[7] P. Kenny, "Bayesian speaker verification with Heavy–Tailed Priors," in *Keynote presentation, Odyssey 2010, The Speaker and Language Recognition Workshop*, 2010. Available at http://www.crim.ca/perso/patrick.kenny/kenny_Odyssey2010.pdf.

[8] N.Brummer, "A farewell to SVM: Bayes factor speaker detection in supervector space," 2006. Available at https://sites.google.com/site/nikobrummer/.

[9] N. Brümmer and E. de Villiers, "The speaker partitioning problem," in *Proc. Odyssey 2010*, pp. 194–201, 2010.

[10] O. Glembek, P. Matejka, O. Plchot, J. Pesán, L. Burget, and P. Schwarz, "Migrating i–vectors between speaker recognition systems using Regression Neural Networks," in *Proceedings of Interspeech 2015*, pp. 2327–2331, 2015.

[11] S. Cumani and P. Laface, "E–vectors: JFA and i–vectors revisited," in *Proceedings of ICASSP 2017*, pp. 5435–5439, 2017.

[12] S. Cumani and P. Laface, "Speaker recognition using e–vectors," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 4, pp. 736–748, 2018.

[13] E. Variani, X. Lei, E. McDermott, I. L. Moreno, and J. Gonzalez-Dominguez, "Deep Neural Networks for small footprint text-dependent speaker verification," in *Proceedings of ICASSP 2014*, pp. 4052–4056, 2014.

[14] G. Heigold, I. Moreno, S. Bengio, and N. Shazeer, "End-to-end text-dependent speaker verification," in *Proceedings of ICASSP 2016*, pp. 5115–5119, 2016.

[15] D. Snyder, P. Ghahremani, D. Povey, D. Garcia-Romero, Y. Carmiel, and S. Khudanpur, "Deep Neural Network-based speaker embeddings for end-to-end speaker verification," in *2016 IEEE Spoken Language Technology Workshop (SLT)*, pp. 165–170, 2016.

[16] A. Kanagasundaram, R. Vogt, D. Dean, S. Sridharan, and M. Mason, "i-vector based speaker recognition on short utterances," in *Proceedings of Interspeech 2011*, pp. 2341–2344, 2011.

[17] S. Cumani, O. Plchot, and P. Laface, "Probabilistic Linear Discriminant Analysis of i–vector posterior distributions," in *Proceedings of ICASSP 2013*, pp. 7644–7648, 2013.

[18] P. Kenny, T. Stafylakis, P. Ouellet, M. Alam, and P. Dumouchel, "PLDA for speaker verification with utterances of arbitrary duration," in *Proceedings of ICASSP 2013*, pp. 7649–7653, 2013.

[19] T. Hasan, R. Saeidi, J. H. L. Hansen, and D. A. van Leeuwen, "Duration mismatch compensation for i–vector based speaker recognition systems," in *Proceedings of ICASSP 2013*, pp. 7663–7667, 2013.

[20] S. Cumani, O. Plchot, and P. Laface, "On the use of i-vector posterior distributions in probabilistic linear discriminant analysis," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 4, pp. 846–857, 2014.

[21] S. Cumani, "Fast scoring of full posterior plda models," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 11, pp. 2036–2045, 2015.

[22] W. B. Kheder, D. Matrouf, M. Ajili, and J. Bonastre, "Probabilistic approach using joint long and short session i–vectors modeling to deal with short utterances for speaker recognition," in *Proceedings of Interspeech 2016*, pp. 1830–1834, 2016.

[23] J. Ma, V. Sethu, E. Ambikairajah, and K. A. Lee, "Twin model G-PLDA for duration mismatch compensation in text-independent speaker verification," in *Proceedings of Interspeech 2016*, pp. 1853–1857, 2016.

[24] J. Ma, V. Sethu, E. Ambikairajah, and K. A. Lee, "Duration compensation of i–vectors for short duration speaker verification," *Electronics Letters*, vol. 53, no. 6, pp. 405–407, 2017.

[25] Y. Lei, L. Burget, and N. Scheffer, "A noise robust ivector extractor using Vector Taylor Series for speaker recognition," in *Proceedings of ICASSP 2013*, pp. 6788–6791, 2013.

[26] Y. Lei, L. Burget, L. Ferrer, M. Graciarena, and N. Scheffer, "Towards noise-robust speaker recognition using Probabilistic Linear Discriminant Analysis," in *Proceedings of ICASSP 2012*, pp. 4253–4256, 2012.

[27] D. Martinez, L. Brget, T. Stafylakis, Y. Lei, P. Kenny, and E. Lleida, "Unscented transform for ivector-based noisy speaker recognition," in *Proceedings of ICASSP 2014*, pp. 4042–4046, 2014.

[28] W. B. Kheder, D. Matrouf, J. F. Bonastre, M. Ajili, and P. M. Bousquet, "Additive noise compensation in the i-vector space for speaker recognition," in *Proceedings of ICASSP 2015*, pp. 4190–4194, 2015.

[29] O. Plchot, L. Burget, H. Aronowitz, and P. Matjka, "Audio enhancing with DNN autoencoder for speaker recognition," in *Proceedings of ICASSP 2015*, pp. 5090–5094, 2016.

[30] W. B. Kheder, D. Matrouf, M. Ajili, and J. Bonastre, "Probabilistic approach using joint clean and noisy i–vectors modeling for speaker recognition," in *Proceedings of Interspeech 2016*, pp. 3838–3842, 2016.

[31] M. Mak, X. Pang, and J. Chien, "Mixture of PLDA for noise robust i–vector speaker verification," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 1, pp. 130–142, 2016.

[32] M.Senoussaoui, P. Kenny, N. Dehak, and P. Dumouchel, "An i–vector extractor suitable for speaker recognition with both microphone and telephone speech," in *Proceedings of Odyssey 2010*, pp. 28–33, 2010.
[33] T. Pekhovsky and A. Sizov, "Comparison between supervised and unsupervised learning of Probabilistic Linear Discriminant Analysis Mixture models for speaker verification," *Pattern Recognition Letters*, vol. 34, pp. 1307–1313, 2013.
[34] S. Cumani and P. Laface, "I–vector transformation and scaling for PLDA based speaker recognition," in *Proceedings of Odyssey 2016*, pp. 39–46, 2016.
[35] S. Cumani and P. Laface, "Non–linear i–vector transformations for PLDA based speaker recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 4, pp. 908–919, 2017.
[36] S. Cumani and P. Laface, "Joint estimation of PLDA and non–linear transformations of speaker vectors," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 10, pp. 1890–1900, 2017.
[37] S. Prince, J. Elder, J. Warrell, and F. Felisberti, "Tied factor analysis for face recognition across large pose differences," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 6, pp. 970–984, 2008.
[38] S. Prince, J. Elder, J. Warrell, and F. Felisberti, "Probabilistic models for inference about identity," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 1, pp. 144–157, 2012.
[39] D. Garcia-Romero, X. Zhou, and C. Espy-Wilson, "Multicondition training of Gaussian PLDA models in i-vector space for noise and reverberation robust speaker recognition," in *Proceedings of ICASSP 2012*, pp. 4257–4260, 2012.
[40] P. Rajan, A. Afanasyev, V. Hautamki, and T. Kinnunen, "From single to multiple enrollment i-vectors: Practical PLDA scoring variants for speaker verification," *Digital Signal Processing*, vol. 31, pp. 93–101, 2014.
[41] M. C. Jones and A. Pewsey, "Sinh–arcsinh distributions," *Biometrika*, vol. 96, no. 4, pp. 761–780, 2009.
[42] J. F. Rosco, M. C. Jones, and A. Pewsey, "Skew t distributions via the sinh-arcsinh transformation," *TEST*, vol. 20, no. 3, pp. 630–652, 2011.
[43] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.
[44] W. Kabsch, "A solution for the best rotation to relate two sets of vectors," *Acta Crystallographica Section A*, vol. 32, no. 5, pp. 922–923, 1976.
[45] D. Garcia-Romero and C. Y. Espy-Wilson, "Analysis of i–vector length normalization in speaker recognition systems," in *Proc. of Interspeech 2011*, pp. 249–252, 2011.
[46] H. Hermansky, N. Morgan, A. Bayya, and P. Kohn, "RASTA-PLP speech analysis technique," in *"Proceedings of ICASSP 1992*, pp. 121–124, 1992.
[47] "The NIST year 2012 speaker recognition evaluation plan," 2012. Available at "http://www.nist.gov/itl/iad/mig/upload/NIST_SRE12_evalplan-v17-r1.pdf.
[48] "The NIST year 2010 speaker recognition evaluation plan," 2010. Available at http://www.itl.nist.gov/iad/mig/tests/sre/2010/NIST_SRE10_evalplan.r6.pdf.
[49] S. Cumani, N. Brümmer, L. Burget, P. Laface, O. Plchot, and V. Vasilakakis, "Pairwise discriminative speaker verification in the i-vector space," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 6, pp. 1217–1227, 2013.
[50] S. Cumani and P. Laface, "Large scale training of Pairwise Support Vector Machines for speaker recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 11, pp. 1590–1600, 2014.



Sandro Cumani received the M.S. degree in Computer Engineering from the Politecnico di Torino, Torino, Italy, in 2008, and the Ph.D. degree in Computer and System Engineering of Politecnico di Torino in 2011. He worked at the Brno University of Technology, Czech Republic, and is a research fellow within the Department of Control and Computer Engineering of Politecnico di Torino. His current research interests include machine learning, speech processing and biometrics, in particular speaker and language recognition.



Pietro Laface received the M.S. degree in Electronic Engineering from the Politecnico di Torino, Torino, Italy, in 1973.
Since 1988 it has been full Professor of Computer Science at the Dipartimento di Automatica e Informatica of Politecnico di Torino, where he leads the speech technology research group. He has published over 150 papers in the area of pattern recognition, artificial intelligence, and spoken language processing. His current research interests include all aspects of automatic speech recognition and its applications, in particular speaker and spoken language recognition.