POLITECNICO DI TORINO
Repository ISTITUZIONALE

Speaker Recognition Using e–Vectors

Original
Speaker Recognition Using e–Vectors / Cumani, S., Laface, P.. - In: IEEE/ACM TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING. - ISSN 2329-9290. - STAMPA. - 26:4(2018), pp. 736-748. [10.1109/TASLP.2018.2791806]

Availability:
This version is available at: 11583/2699467 since: 2018-02-08T11:17:23Z

Publisher:
IEEE Signal Processing Society

Published
DOI:10.1109/TASLP.2018.2791806

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

1

Speaker recognition using e–vectors

Sandro Cumani and Pietro Laface

***Abstract*—Systems based on i–vectors represent the current state–of–the–art in text–independent speaker recognition. Unlike Joint Factor Analysis (JFA), which models both speaker and intersession subspaces separately, in the i–vector approach all the important variability is modeled in a single low dimensional subspace. This work 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. We propose a speaker modeling approach that extracts a compact representation of a speech segment, similar to the speaker factors of JFA and to i–vectors, referred to as "e–vector". Estimating the e–vector subspace follows a procedure similar to i–vector training, but produces a more accurate speaker subspace, as confirmed by the results of a set of tests performed on the NIST 2012 and 2010 Speaker Recognition Evaluations. Simply replacing the i–vectors with e–vectors we get approximately 10% average improvement of the $C_{primary}$ cost function, using different systems and classifiers. It is worth noting that these performance gains come without any additional memory or computational costs with respect to the standard i–vector systems.**

I. Introduction

A simple and effective model for speaker recognition has been introduced in [2], [3], where a speech segment is represented by a low–dimensional "identity vector" or i–vector. An i–vector is obtained from the statistics collected by using a Gaussian Mixture Model (GMM) trained to represent a Universal Background Model (UBM) [4]. The i–vector model constrains the utterance GMM supervectors $\mathbf{s}$, consisting of the stacked GMM means, to live in a single subspace, including both speaker and channel variability, according to:

$$\mathbf{s} = \mathbf{m} + \mathbf{T}\mathbf{w} \ , \tag{1}$$

where $\mathbf{m}$ is the Universal Background Model (UBM) mean supervector, composed of C GMM components of dimension F. $\mathbf{T}$ is a low-rank rectangular matrix spanning the subspace including important inter and intra–speaker variability in the supervector space, and $\mathbf{w}$ is a realization of a latent variable $\mathbf{W}$, of size M, having a standard normal prior distribution. A Maximum-Likelihood estimate of matrix $\mathbf{T}$ is usually obtained by minor modifications of the Joint Factor Analysis approach [5]. Given $\mathbf{T}$, and the set of τ feature vectors $\mathcal{X} = \{\mathbf{x}_1\mathbf{x}_2 \dots \mathbf{x}_\tau\}$ extracted from a speech segment, it is possible computing the likelihood of $\mathcal{X}$ given the model (1), and a value for the latent variable $\mathbf{W}$. The i–vector, which represents the segment, is computed as the Maximum a Posteriori (MAP) point estimate of the variable $\mathbf{W}$, i.e., the mode $\boldsymbol{\mu}_\mathcal{X}$ of the posterior distribution $P_{\mathbf{W}|\mathcal{X}}(\mathbf{w})$. The mode can be computed by collecting the zero-order statistics estimated on each Gaussian component of the UBM for the set of feature vectors $\mathcal{X}$, and the corresponding first-order statistics centered around the UBM means.

The main advantage of the i–vector representation is that the problem of intersession variability is deferred to a second stage, dealing with low–dimensional vectors rather than with the high–dimensional space of the GMM supervectors. This allows training better classifiers, such as PLDA [6]–[10], and Pairwise Support Vector Machine (PSVM) or Logistic Regression [11]–[17]. Furthermore, in such low dimensional space it is possible to perform transformations that are particularly suited for enhancing the classifier performance, such as compensating development and evaluation set mismatches by means of Within Class Covariance Normalization [18] and length normalization [19], or transforming i–vectors so that they better fit the classifier assumptions [20]–[22]. Finally, PLDA and PSVM models allow exploiting multiple recordings of the same speaker by simply averaging their corresponding i–vectors. This simple approach is known to be more effective than proper estimation of multi–session likelihood ratios in PLDA, see for example [23].

I–vector modeling stems from the Joint Factor Analysis (JFA) approach [24]–[26]. JFA models the speaker and channel variability of a Gaussian supervector by means of a linear combination of eigenvoice, eigenchannel and MAP adaptated supervectors. These factors can be estimated according to the iterative procedure illustrated in [24]. It has been, however, experimentally shown in [27] that the eigenchannel factors keep some correlation with the eigenvoice factors, i.e., they still convey some information about the speaker identity. This observation motivated the introduction of the i–vector approach as a feature extractor [2], [3], where speaker and channel variability are modeled in a single low–dimensional space spanned by the column vectors of a single matrix $\mathbf{T}$.

Although the i–vector subspace also includes channel variability, which is detrimental for speaker recognition, i–vectors have shown to provide a large performance boost over JFA–based methods for text–independent tasks.

On the other hand, JFA estimates a more informative speaker subspace with respect to the total variability i–vector subspace, because the latter considers each training segment as belonging to a different speaker. In this work we propose a speaker modeling approach that combines the benefits of the more informative JFA speaker subspace and of the i–vector framework. It extracts a compact representation of a speech segment, similar to the speaker factors or i–vectors. This representation better characterizes the speaker, and thus allows obtaining better performance with respect to the standard i–vectors. By analogy with i–vectors, we will refer to this

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)

This paper is an extended and revised version of ICASSP 2017 paper [1].

representation as "eigenvoice–vector", or "e–vector" for short.

The main idea of our approach is to rely on the eigenvoice space, which should be more accurate than the one represented by the total variability matrix, which has a larger variability because it includes channel information. The novelty of our proposal consists in estimating a linear transformation that allows keeping the span of the speaker–specific eigenvoice subspace, but at the same time provides a better prior for i–vector extraction. We will show that this is simply obtained by considering each training segment as belonging to a different speaker, as it is done in standard i–vector training, but applying solely Minimum Divergence Estimation (MDE) [8], [28] during the training iterations for obtaining a new variability matrix, $\mathbf{E}$.

Replacing i–vectors with e–vectors we were able to obtain approximately 10% performance improvement on the extended core NIST SRE 2012 [29] evaluation, using different extraction techniques, classifiers, and e–vector dimensions. Similar, but slightly lower, performance has been achieved on the core extended NIST SRE2010 female telephone [30] dataset.

The paper is organized as follows: Section II recalls the eigenvoice, JFA, and i–vector approaches. Section III introduces the e–vectors and their training procedure. The classifiers that have been used in our experiments are illustrated in Section IV together with two recently proposed techniques, which are particularly suited to enhance the performance of a Gaussian PLDA classifier. The experimental settings and results are presented and commented in Section V, and conclusions are drawn in Section VI.

II. Supervector representations

A model–based speaker adaptation approach was proposed in [31], which constrains the adapted model to be a linear combination of a small number of basis vectors obtained from a set of reference speakers. These "eigenvoice" vectors were estimated, in this approach, with the objective of capturing the most important components of variation among the reference speakers. The adaptation data were then used for obtaining, by means of Maximum Likelihood Eigen-Decomposition, the weights of the linear combination, leading to a low–dimensional vector representation of a new speaker in the eigenvoice space. Eigenvoice modeling, thus, aims at characterizing the speaker within the speaker subspace, and thanks to the correlations between GMM components, allows adapting also rarely observed Gaussians. This modeling approach was successively also proposed for speaker recognition in [32], and in [33], where eigenvoice MAP adaptation was introduced.

In [34] the eigenvoice approach has been applied effectively to the problem of modeling intra–speaker variability, by compensating the session (channel) variability at recognition time. Finally JFA modeling was introduced in [24], where the eigenvoice model was extended to deal with intersession speaker variability and channel mismatches between enrollment and evaluation conditions, taking care of the channel effects also in speaker enrollment.

JFA defines two subspaces: the speaker space represented by an eigenvoice matrix $\mathbf{V}$, and the channel space, represented by an eigenchannel matrix $\mathbf{U}$. In particular, JFA models the speaker and channel dependent supervector $\mathbf{s}$ for a given speech segment as:

$$\mathbf{s} = \mathbf{m} + \mathbf{V}\mathbf{y} + \mathbf{U}\mathbf{x} + \mathbf{D}\mathbf{z} \ , \tag{2}$$

where $\mathbf{m}$ is UBM supervector, which stacks the Gaussian means, $\mathbf{V}$ and $\mathbf{U}$ are rectangular low rank matrices, $\mathbf{D}$ is a diagonal matrix, and $\mathbf{y}$, $\mathbf{x}$ and $\mathbf{z}$ are the speaker, channel, and residual (or common) factors, respectively.

As recalled in the introduction section, JFA channel factors also contain information about the speaker identity [27], thus, the speaker and channel dependent supervector model (2) was simplified in the i–vector approach as in (1), collecting in a single matrix the speaker, channel, and residual noise matrices, leaving to the classifier the duty of taking care of intersession variability .

III. e–vectors

The i–vector approach allows channel compensation to be performed in a low–dimensional subspace, rather than in the much larger GMM supervector space. It is worth noting that, due to the substantial similarity of models (1) and (2), matrix $\mathbf{T}$ training can be performed similarly to eigenvoice $\mathbf{V}$ matrix training. The only difference is that in the $\mathbf{V}$ matrix estimation procedure, the segments of the same speaker are labeled as a single class, whereas all segments are considered as belonging to different classes in $\mathbf{T}$ matrix estimation.

Since the $\mathbf{T}$ matrix eigenvectors span both the speaker and channel subspace, matrix $\mathbf{T}$ does not model the speaker subspace as well as the eigenvoice matrix $\mathbf{V}$.

On the basis of these observations, we propose a speaker modeling approach that tries to take advantage of the best of the JFA and of the i–vector techniques. We keep the i–vectors framework to exploit the possibility of representing a voice segment in a low–dimensional space, but we estimate a different $\mathbf{T}$ matrix, which better accounts for the speaker space. This new matrix $\mathbf{E}$ is similar to the $\mathbf{T}$ matrix, but it is estimated with the additional constraint that it spans the same subspace represented by the eigenvoice matrix trained on the same dataset.

The new model is similar to the i–vector model:

$$\mathbf{s} = \mathbf{m} + \mathbf{E}\mathbf{w} \ , \tag{3}$$

where, as in the standard i–vector model, $\mathbf{s}$ and $\mathbf{m}$ are the GMM supervector and UBM mean supervector, respectively and $\mathbf{w}$ is a random vector, of dimension d, with standard normal prior distribution.

Since our goal is that $\mathbf{E}$ spans the same subspace of $\mathbf{V}$, we define $\mathbf{E}$ as:

$$\mathbf{E} = \mathbf{V}\mathbf{A} \ , \tag{4}$$

where $\mathbf{A}$ is a full rank $d \times d$ matrix. Matrix $\mathbf{E}$ spans the same subspace of $\mathbf{V}$ because its columns are a linear combination of the columns of $\mathbf{V}$. Thus, model (3) can be rewritten as:

$$\mathbf{s} = \mathbf{m} + \mathbf{V}\mathbf{A}\mathbf{w} \ . \tag{5}$$

Maximum Likelihood Estimation (MLE) of $\mathbf{A}$ can be performed by means of the Expectation Maximization (EM) algorithm, with a procedure similar to ML estimation of matrix $\mathbf{T}$ in standard i–vector extraction [8], [35], as illustrated in subsection III-A. However, contrary to the standard solution, the solution using this approach is cumbersome and expensive, thus, in subsection III-B we reformulate model (5) so that a very simple solution can be found.

In the next two subsections we use the following notations:

- F is the dimension of the acoustic feature vectors, and C is the number of the UBM components,
- $\mathbf{m}^{(c)}$ and $\mathbf{\Sigma}^{(c)}$ are the mean vector and the covariance matrix of the c–th UBM component, respectively,
- $\mathbf{\Sigma}$ denotes the block–diagonal matrix whose elements are the covariance matrices $\mathbf{\Sigma}^{(c)}$,
- $N_i^{(c)}$ is the zero order statistics for utterance i estimated on the c–th Gaussian component of the UBM,
- $\mathbf{f}_i^{(c)}$ is the set of first order statistics centered around the UBM mean:

$$\begin{aligned} N_i^{(c)} &= \sum_t \gamma_{i,t}^{(c)} \\ \mathbf{f}_i^{(c)} &= \sum_t \left(\gamma_{i,t}^{(c)} \mathbf{o}_{i,t}\right) - N_i^{(c)} \mathbf{m}^{(c)} \ , \end{aligned} \tag{6}$$

  where $\gamma_{i,t}^{(c)}$ is the occupation probability of the t–th feature vector $\mathbf{o}_{i,t}$ of utterance i for the c–th UBM component,
- $\mathbf{f}_i$ is the supervector stacking the first order statistics $\mathbf{f}_i^{(c)}$,
- $\mathbf{N}_i$ denotes the $CF \times CF$ diagonal matrix whose diagonal blocks are the $F \times F$ diagonal matrices $N_i^{(c)}\mathbf{I}$.

A. Estimation of matrix $\mathbf{A}$ in model $\mathbf{s} = \mathbf{m} + \mathbf{VAw}$

The joint likelihood of the set of the t_i feature vectors $\mathcal{X}_i = [\mathbf{o}_{i,1}, \ldots, \mathbf{o}_{i,t_i}]$ of utterance i and the corresponding hidden variable $\mathbf{w}_i$, given matrix $\mathbf{A}$, is:

$$\begin{aligned} \log P(\mathcal{X}_i, \mathbf{w}_i|\mathbf{A}) =& -\frac{1}{2}\mathbf{w}_i^T\mathbf{w}_i + \mathbf{w}_i^T\mathbf{E}^T\mathbf{\Sigma}^{-1}\mathbf{f}_i \\ & -\frac{1}{2}\mathbf{w}_i^T\mathbf{E}^T\mathbf{N}_i\mathbf{\Sigma}^{-1}\mathbf{E}\mathbf{w}_i + G_{\mathbf{\Sigma},i} \\ =& -\frac{1}{2}\mathbf{w}_i^T\mathbf{w}_i + \mathbf{w}_i^T\mathbf{A}^T\mathbf{V}^T\mathbf{\Sigma}^{-1}\mathbf{f}_i \\ & -\frac{1}{2}\mathbf{w}_i^T\mathbf{A}^T\mathbf{V}^T\mathbf{N}_i\mathbf{\Sigma}^{-1}\mathbf{V}\mathbf{A}\mathbf{w}_i + G_{\mathbf{\Sigma},i} \ , \end{aligned} \tag{7}$$

where $G_{\mathbf{\Sigma},i}$ collects all terms that do not dependend on $\mathbf{A}$ or $\mathbf{w}_i$. By inspection, and in analogy with the standard i–vector posterior distribution, the posterior for $\mathbf{w}_i$ is Gaussian, with mean $\boldsymbol{\mu}_{w,i}$ and covariance matrix $\mathbf{\Lambda}_{w,i}^{-1}$:

$$\begin{aligned} \mathbf{\Lambda}_{w,i}^{-1} &= \left(\mathbf{I} + \mathbf{E}^T\mathbf{N}\mathbf{\Sigma}^{-1}\mathbf{E}\right)^{-1} \\ &= \left(\mathbf{I} + \mathbf{A}^T\mathbf{V}^T\mathbf{N}_i\mathbf{\Sigma}^{-1}\mathbf{V}\mathbf{A}\right) \\ \boldsymbol{\mu}_{w,i} &= \mathbf{\Lambda}_{w,i}^{-1}\mathbf{E}^T\mathbf{\Sigma}^{-1}\mathbf{f}_i \\ &= \mathbf{\Lambda}_{w,i}^{-1}\mathbf{A}^T\mathbf{V}^T\mathbf{\Sigma}^{-1}\mathbf{f}_i \ . \end{aligned} \tag{8}$$

Given the posteriors computed using the current estimate of $\mathbf{A}$, $\mathbf{A}_{old}$, the maximization step of the EM algorithm requires optimizing the auxiliary function:

$$\begin{aligned} Q(\mathbf{A}, \mathbf{A}_{old}) =& \sum_i \mathbb{E}_{\mathbf{w}_i|\mathcal{X}_i,\mathbf{A}_{old}}\left[\log P(\mathcal{X}_i, \mathbf{w}_i|\mathbf{A})\right] \\ =& \sum_i \mathrm{Tr}\left(\mathbf{A}^T\mathbf{V}^T\mathbf{\Sigma}^{-1}\mathbf{f}_i\mathbb{E}\left[\mathbf{w}_i\right]^T\right) \\ & -\frac{1}{2}\mathrm{Tr}\left(\mathbf{A}^T\mathbf{V}^T\mathbf{N}_i\mathbf{\Sigma}^{-1}\mathbf{V}\mathbf{A}\mathbb{E}\left[\mathbf{w}_i\mathbf{w}_i^T\right]\right) + k \ , \end{aligned} \tag{9}$$

where k collects all terms that do not depend on $\mathbf{A}$.
Setting to $\mathbf{0}$ the derivative of $Q(\mathbf{A}, \mathbf{A}_{old})$ with respect to $\mathbf{A}$ we obtain:

$$\sum_i \mathbf{V}^T\mathbf{N}_i\mathbf{\Sigma}^{-1}\mathbf{V}\mathbf{A}\mathbb{E}\left[\mathbf{w}_i\mathbf{w}_i^T\right] = \mathbf{V}^T\mathbf{\Sigma}^{-1}\sum_i \mathbf{f}_i\mathbb{E}\left[\mathbf{w}_i\right]^T \ , \tag{10}$$

which can be recognized as a generalized Sylvester equation of the form $\sum_i A_i X B_i = C$, where A_i and B_i are $d \times d$ square matrices. Solving (10) is not trivial because the standard approach requires building Kronecker products of the form $A_i \otimes B_i$. Due to the large size of these Kronecker product matrices, the solution is inefficient or even infeasible.

Although one can rely on numerical optimization to directly maximize the EM auxiliary function (9), this approach is not appealing if a simpler solution is possible. We show in the next subsection that a simple solution can be obtained appropriately rewriting model (5).

B. Estimation of matrix $\mathbf{A}$ in model $\mathbf{s} = \mathbf{m} + \mathbf{Vz}$

Let's rewrite model (5) as:

$$\mathbf{s} = \mathbf{m} + \mathbf{V}\mathbf{z} \ , \tag{11}$$

where $\mathbf{z} = \mathbf{A}\mathbf{w}$. According to the properties of the normal distribution of random variables [36], if $\mathbf{x}$ is a normal distributed random vector, $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}_\mathbf{x}, \mathbf{\Sigma}_\mathbf{x})$, then the random vector resulting from an affine transformation $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{b}$ is normally distributed with mean and covariance:

$$\boldsymbol{\mu}_\mathbf{y} = \mathbf{A}\boldsymbol{\mu}_\mathbf{x} \ , \quad \mathbf{\Sigma}_\mathbf{y} = \mathbf{A}\mathbf{\Sigma}_\mathbf{x}\mathbf{A}^\mathbf{T} \ . \tag{12}$$

Since $\mathbf{w}$ has a standard normal distribution, $\mathbf{z}$ has a prior distribution $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{A}\mathbf{A}^T)$.
Models (5) and (11) are equivalent, in the sense that they provide the same marginal likelihood of the data, $P(\mathcal{X}|\mathbf{A})$. Writing the joint likelihood:

$$\begin{aligned} \log P(\mathcal{X}_i,\mathbf{z}_i|\mathbf{A}) &= \log P(\mathcal{X}_i|\mathbf{z}_i, \mathbf{A}) + \log P(\mathbf{z}_i|\mathbf{A}) \\ &= \mathbf{z}_i^T\mathbf{V}^T\mathbf{\Sigma}^{-1}\mathbf{f}_i - \frac{1}{2}\mathbf{z}_i\mathbf{V}^T\mathbf{N}_i\mathbf{\Sigma}^{-1}\mathbf{V}\mathbf{z} + G_{\mathbf{\Sigma},i} - \\ & \quad \frac{1}{2}\mathbf{z}_i^T\mathbf{A}^{-T}\mathbf{A}^{-1}\mathbf{z}_i - \frac{1}{2}\log\left|\mathbf{A}\mathbf{A}^T\right| \ , \end{aligned} \tag{13}$$

where $G_{\mathbf{\Sigma},i}$ collects all terms that do not dependent on $\mathbf{A}$ or $\mathbf{z}_i$, and noticing that its first three terms do not depend on $\mathbf{A}$, we can verify that the posterior distributions of $\mathbf{w}$ and $\mathbf{z}$ are related by the same properties (12) that apply to their prior distributions.

The posterior for $\mathbf{z}_i$ is Gaussian. Its covariance matrix $\mathbf{\Lambda}_{z,i}^{-1}$ and mean $\boldsymbol{\mu}_{z,i}$ are given by:

$$\begin{aligned}\mathbf{\Lambda}_{z,i}^{-1} &= \left(\mathbf{A}^{-T}\mathbf{A}^{-1} + \mathbf{V}^T\mathbf{N}_i\mathbf{\Sigma}^{-1}\mathbf{V}\right)^{-1} \\ &= \mathbf{A}\Big(\mathbf{I} + \mathbf{A}^T\mathbf{V}^T\mathbf{N}_i\mathbf{\Sigma}^{-1}\mathbf{V}\mathbf{A}\Big)^{-1}\mathbf{A}^T \\ &= \mathbf{A}\mathbf{\Lambda}_{w,i}^{-1}\mathbf{A}^T \ , \end{aligned} \tag{14}$$

and

$$\begin{aligned}\boldsymbol{\mu}_{z,i} &= \mathbf{\Lambda}_{z,i}^{-1}\mathbf{V}^T\mathbf{\Sigma}^{-1}\mathbf{f}_i \\ &= \mathbf{A}\mathbf{\Lambda}_{w,i}^{-1}\mathbf{A}^T\mathbf{V}^T\mathbf{\Sigma}^{-1}\mathbf{f}_i \\ &= \mathbf{A}\boldsymbol{\mu}_{w,i} \ . \end{aligned} \tag{15}$$

Please notice that the joint likelihood of (13) corresponds to the one derived for the i–vector model, just with a non–standard prior of $\mathbf{z}_i$.

The EM auxiliary function for estimating $\mathbf{A}$ takes the form:

$$\begin{aligned}Q(\mathbf{A},&\mathbf{A}_{old}) = \sum_i \mathbb{E}_{\mathbf{z}_i|\mathcal{X}_i,\mathbf{A}_{old}}\left[\log P(\mathcal{X}_i,\mathbf{z}_i|\mathbf{A})\right] \\ &= \sum_i \mathbb{E}_{\mathbf{z}_i|\mathcal{X}_i,\mathbf{A}_{old}}\left[\log P(\mathcal{X}_i|\mathbf{z}_i,\mathbf{A}) + \log P(\mathbf{z}_i|\mathbf{A})\right] \ . \end{aligned} \tag{16}$$

Since the term $\log P(\mathcal{X}_i|\mathbf{z}_i,\mathbf{A})$ does not depend on $\mathbf{A}$, the maximization of (16), reduces to the maximization of the second term:

$$\begin{aligned}\tilde{Q}(\mathbf{A},\mathbf{A}_{old}) &= \sum_i \mathbb{E}_{\mathbf{z}_i|\mathcal{X}_i,\mathbf{A}_{old}}\left[\log P(\mathbf{z}_i|\mathbf{A})\right] \\ &= \sum_i \mathbb{E}_{\mathbf{z}_i|\mathcal{X}_i,\mathbf{A}_{old}}\left[-\frac{1}{2}\mathbf{z}_i^T\mathbf{A}^{-T}\mathbf{A}^{-1}\mathbf{z}_i\right] + \\ &\quad \sum_i -\frac{1}{2}\log\left|\mathbf{A}\mathbf{A}^T\right| \ . \end{aligned} \tag{17}$$

It is worth noting that maximization of (17) corresponds to the minimization of the KL–divergence [24]:

$$\sum_i D(P(\mathbf{z}_i|\mathcal{X}_i,\mathbf{A}_{old})\|P(\mathbf{z}_i|\mathbf{A})) \ . \tag{18}$$

Indeed,

$$\begin{aligned}&\sum_i D_{KL}(P(\mathbf{z}_i|\mathcal{X}_i,\mathbf{A}_{old})\|P(\mathbf{z}_i|\mathbf{A})) = \\ &\sum_i \mathbb{E}_{\mathbf{z}_i|\mathcal{X}_i,\mathbf{A}_{old}}\log\frac{P(\mathbf{z}_i|\mathcal{X}_i,\mathbf{A}_{old})}{P(\mathbf{z}_i|\mathbf{A})} = \\ &\sum_i \mathbb{E}_{\mathbf{z}_i|\mathcal{X}_i,\mathbf{A}_{old}}\log P(\mathbf{z}_i|\mathcal{X}_i,\mathbf{A}_{old}) - \mathbb{E}_{\mathbf{z}_i|\mathcal{X}_i,\mathbf{A}_{old}}\log P(\mathbf{z}_i|\mathbf{A}) \ , \end{aligned} \tag{19}$$

but the first term is constant, and does not depend on $\mathbf{A}$.

Defining, for convenience, $\mathbf{P} = \left(\mathbf{A}\mathbf{A}^T\right)^{-1}$, the auxiliary function (17) becomes:

$$\begin{aligned}&\tilde{Q}(\mathbf{P},\mathbf{A}_{old}) \\ &= \sum_i \mathbb{E}_{\mathbf{z}_i|\mathcal{X}_i,\mathbf{A}_{old}}\left[-\frac{1}{2}\mathrm{Tr}\left(\mathbf{P}\mathbf{z}_i\mathbf{z}_i^T\right)\right] + \sum_i \frac{1}{2}\log|\mathbf{P}| \\ &= -\frac{1}{2}\mathrm{Tr}\left(\mathbf{P}\sum_i \mathbb{E}_{\mathbf{z}_i|\mathcal{X}_i,\mathbf{A}_{old}}\left[\mathbf{z}_i\mathbf{z}_i^T\right]\right) + \frac{N}{2}\log|\mathbf{P}| \ , \end{aligned} \tag{20}$$

where N is the number of training utterances.

Setting to $\mathbf{0}$ the derivative of $\tilde{Q}(\mathbf{P},\mathbf{A}_{old})$ with respect to $\mathbf{P}$, we obtain:

$$\mathbf{A}\mathbf{A}^T = \mathbf{P}^{-1} = \frac{1}{N}\sum_i \mathbb{E}_{\mathbf{z}_i|\mathcal{X}_i,\mathbf{A}_{old}}\left[\mathbf{z}_i\mathbf{z}_i^T\right] \ . \tag{21}$$

Thus, an optimal solution for $\mathbf{A}$ is given by the Cholesky decomposition of $\mathbf{P}^{-1}$.

However, $\mathbf{P}$ is given in terms of the posterior distribution of $\mathbf{z}_i$, but we are interested in an iterative estimation that is a function of $\mathbf{E}$ and $\mathbf{w}_i$.

Since the posterior distributions $\mathbf{z}|\mathcal{X}$ and $\mathbf{w}|\mathcal{X}$ are related according to (12), we can rewrite (21) as:

$$\mathbf{A}\mathbf{A}^T = \frac{1}{N}\sum_i \mathbf{A}_{old}\mathbb{E}_{\mathbf{w}_i|\mathcal{X}_i,\mathbf{A}_{old}}\left[\mathbf{w}_i\mathbf{w}_i^T\right]\mathbf{A}_{old}^T \ . \tag{22}$$

Since the distribution $\mathbf{w}|\left(\mathcal{X},\mathbf{A}_{old}\right) = \mathbf{w}|\left(\mathcal{X},\mathbf{E}_{old}\right)$, because $\mathbf{E}_{old} = \mathbf{V}\mathbf{A}_{old}$,

$$\mathbf{A}\mathbf{A}^T = \mathbf{A}_{old}\left(\frac{1}{N}\sum_i \mathbb{E}_{\mathbf{w}_i|\mathcal{X}_i,\mathbf{E}_{old}}\left[\mathbf{w}_i\mathbf{w}_i^T\right]\right)\mathbf{A}_{old}^T \ . \tag{23}$$

A solution for $\mathbf{A}$ is then given by:

$$\mathbf{A} = \mathbf{A}_{old}\ \mathbf{R} \tag{24}$$

where $\mathbf{R}$ is the Cholesky decomposition of:

$$\mathbf{R}\mathbf{R}^T = \frac{1}{N}\sum_i \mathbb{E}_{\mathbf{w}_i|\mathcal{X}_i,\mathbf{E}_{old}}\left[\mathbf{w}_i\mathbf{w}_i^T\right] \ , \tag{25}$$

and the optimal solution for $\mathbf{E}$ is:

$$\begin{aligned}\mathbf{E} &= \mathbf{V}\mathbf{A} \\ &= \mathbf{V}\mathbf{A}_{old}\ \mathbf{R} \\ &= \mathbf{E}_{old}\ \mathbf{R} \end{aligned} \tag{26}$$

Thus, given i–vectors posteriors for model (3), the new estimate for $\mathbf{E}$ is obtained by a right–multiplication of the eigenvoice matrix $\mathbf{E}$ with the Cholesky decomposition of matrix $\frac{1}{N}\sum_i \mathbb{E}[\mathbf{w}_i\mathbf{w}_i^T]$, where the sum extends over all the N training i–vectors. The EM training procedure for this model corresponds to performing only Minimum Divergence Estimation for model (3), as in [24]. MDE was originally proposed as an additional, optional, step mainly meant to speed–up EM training convergence in JFA and i–vector modeling.

C. Matrix $\mathbf{E}$ training

The step for training matrix $\mathbf{E}$ can be summarized as follows:

- First a $\mathbf{V}$ matrix is trained exactly as matrix $\mathbf{T}$ is, but assuming that the segments of a given speaker belong to a single class, i.e., accumulating the sufficient statistics per speaker, rather than per segment. In other words, we perform eigenvoice matrix $\mathbf{V}$ estimation in i–vector style, rather than according to the JFA procedure, which estimates matrix $\mathbf{V}$ together with matrices $\mathbf{U}$ and $\mathbf{D}$. We decided to train the eigenvoice matrix without relying on the more complex JFA model because no performance degradation was observed on preliminary experiments

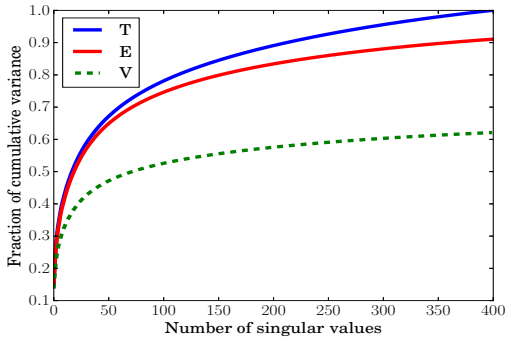


Fig. 1: Cumulative sum of the squared singular values of matrices **T**, **E**, and **V**. All values are normalized by the sum of the squared singular values of matrix **T**.

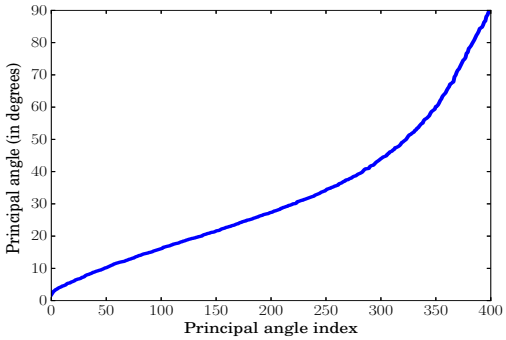


Fig. 2: Principal angles between the subspaces represented by matrices **T** and **E**.

using only the eigenvoice model, i.e., setting to **0** matrices **U** and **D**.

- In the second step, matrix **E** is initialized by **V**. Then, it is trained considering each training segment as belonging to a different speaker, as it is done for the estimation of matrix **T**, but applying only the iterations of (26), which correspond to performing only MDE. These iterations increase the data likelihood but leave unchanged the span of the original eigenvoice matrix. MDE affects the precision matrix of the posterior distribution of the e–vectors by changing the eigenvalues of matrix **E**. This makes the empirical distribution of the e–vectors conform to the standard normal prior [8], [37]. Thus, we keep the span of the eigenvoice space, but we estimate a more accurate model prior with the aim of better estimating the e–vector posterior.

The goal of these procedure is to estimate a matrix **E** that better spans the speaker variability subspace. This means that e–vectors better characterize the speakers with respect to i–vectors having the same dimensions, because some eigenvectors in matrix **T** account for the channel effects. These effects are reduced in matrix **E**, which is trained with more segments per speaker, collected from different sessions and channels.

Figure 1 shows the cumulative variance accounted by the first k squared singular values of matrices **T**, **E**, and **V**, in blue, red, and green lines, respectively. All values are normalized by the sum of the squared singular values of matrix **T**. The blue and green lines clearly show that the eigenvoice matrix **V** has smaller squared singular values with respect to matrix **T**, which means that the channel variability that is contained in the latter is largely reduced in **V**. Since matrix **E** is estimated considering all training recordings as belonging to separate classes, its cumulative variance increases, as shown by the red curve, which becomes closer to the blue one. However, since matrix **E** is estimated with the constraint of keeping the span of matrix **V**, it still discards part of the contribution of channel variability that is contained in matrix **T**, in favor of the most speaker specific directions. The difference between the subspaces spanned by matrices **T** and **E** is illustrated in Figure 2, which plots the principal angles between the two subspaces. The first few directions are almost aligned between the two subspaces, whereas they progressively differ for the others.

Since matrix **V** training is performed considering the segments of a given speaker as belonging to the same class, it is faster than matrix **T** estimation. However, the number of different speakers in the training corpora must be large enough to accurately model the speaker subspace. The effects of the dimension of the speaker population on system performance is analyzed in the Section V.

IV. Discriminative and non–linear classifiers

The comparison of the performance of the e–vectors with respect to i–vectors has been performed by using two main classifiers: the standard Gaussian PLDA (GPLDA), and the Pairwise Support Vector Machine classifier [13], [15]. The latter is briefly recalled in the Subsection IV-A.

In the following we will refer to i–vector classification, but it is taken for granted that the same techniques apply to e–vectors. Since our goal is to compare the performance of e–vectors and i–vectors with different classifiers, and using the most accurate models, we will also rely on improved i–vector extraction using the hybrid DNN/GMM approach, shortly detailed in Subsection IV-B, and we also exploit two recently proposed approaches that allows obtaining relevant improvement for PLDA classification. These approaches, based on a non–linear transformation of the i–vectors, are presented in Subsection IV-C.

A. PSVM

A successful alternative to the generative PLDA model is a discriminative SVM model, where a single Pairwise SVM (PSVM) is trained to classify a trial – composed of two i–vectors - as belonging to the “same speaker”, or to the “different speaker” class. This is in contrast with the usual “one-versus-all” framework, where an SVM model is created for each enrolled speaker, using as samples of the impostor class the i–vectors of a background cohort of speakers. The PSVM model is tightly related with the Gaussian PLDA model. However, while the estimation of the GPLDA parameters is constrained by the definiteness of their covariance matrices, the pairwise discriminative training approach does not make

TABLE I: Number of segments, and of speakers, in the datasets used for training SRE10 models with telephone female data.

Dataset	Number of segments	Number of speakers	Total segments	Total speakers
SRE 04-06 + SWB	22511	1972		
+ SRE 08	7775	753	30286	2725
+ Fisher	13292	7198	43578	9923

TABLE II: Number of segments, and of speakers, in the SRE05-08 datasets available for training SRE10 models with interview/microphone female data.

Dataset	Number of segments	Number of speakers	Total segments	Total speakers
SRE 05	1436	52		
SRE 06	1416	43	2852	95
SRE 08	5574	156	8626	251

any a priori assumption about the i–vector distribution, thus,no parameter constraints are imposed, except for the ones arising from the regularization of its optimization function.

The computation of the loss function gradients, and of the verification scores would be expensive in terms of memory and computation costs using a naïve quadratic expansion of the i–vector pairs. A solution for these issues has been given in [11]–[13]. Furthermore, since the PSVM training still remain expensive in terms of computational resources, which grow quadratically with the number of the training i–vectors, in [14], [15] we introduced a simple and effective technique for discarding the speaker i–vector pairs that are not essential for training. In particular, we theoretically proved that the number of Support Vectors of a PSVM increases linearly with respect to the number of training speakers, rather than quadratically as the number of i–vector pairs. Therefore, the number of training pairs necessary for obtaining an accurate PSVM model is a very small fraction of the total number of training pairs, which can be selected either by computing the scores of a PLDA model on the training data pairs, or by a two–stage random selection strategy [15].

Although discriminative training using the same approach can be performed by Logistic Regression (LR) [12], we found preferable training a PSVM, because it optimizes the margin separation between the classes, whereas LR minimizes the cross-entropy error function. PSVM ignores the huge number of training pairs that are far from the margins. This is more difficult in LR training, whose loss function accumulates a huge number of contributions from the different–speaker pairs, which grow quadratically with the number of i–vectors, whereas the contributions of the same–speaker pairs grow only linearly. While it is possible to reduce the number of different–speaker pairs, or to reduce their weight in the loss function, both approaches still require some heuristic decision, which is not necessary for PSVM training. On the contrary, both contributions to the PSVM loss function grow linearly.

B. I–vector extraction

For both classifiers, the i–vector extraction has been performed either by means of the standard UBM/GMM approach or by exploiting the DNN posteriors of a hybrid DNN/GMM architecture [38]–[41]. In this approach, the standard acoustic UBM is replaced by a fine–grained “phonetic” UBM obtained by associating a Gaussian to each output unit of a DNN, trained to classify the states of a set of context–dependent phonetic units. For each frame, the posterior probability of the DNN states is used as the occupation probability for computing the usual statistics that allow training the UBM parameters, and successively to extract the i–vectors. In particular, we used the hybrid DNN/GMM architecture described in [42], where we associate more than one Gaussian to each output unit of the DNN.

C. I–vector transformation and Non–Linear PLDA

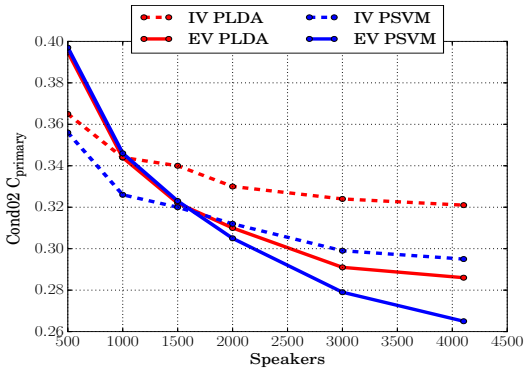
We have shown in [20], [21] that improved PLDA classification performance can be obtained if the development i–vectors are transformed so that their distribution becomes more Gaussian–like, as assumed by the Gaussian PLDA model. The Gaussian target distribution can be obtained by applying to the input i–vectors a sequence of affine and non–linear transformations, whose parameters are estimated on the development set. The evaluation i–vectors are then subject to the same transformation. Since the non–linear transformation that we use is based on the sinh-arcsinh function, we will refer to these method of classification as “PLDA SAS”. Results obtained using PLDA SAS on the NIST SRE 2012 are reported in Section V.

The assumption of PLDA SAS that the i–vectors are statistically independent and distributed according to the standard normal distribution is, however, not completely satisfactory, because i–vectors extracted from segments of the same speaker are not independent. We have, thus, recently developed an improved PLDA model [22], which will refer to as “non–linear PLDA” (NL–PLDA). This is a generative model that allows jointly estimating 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 accurate. It is interesting noting that the i–vector transformation in this enhanced PLDA model is obtained as in the PLDA SAS approach of [21], just changing the transformation target distribution, which becomes speaker–dependent.

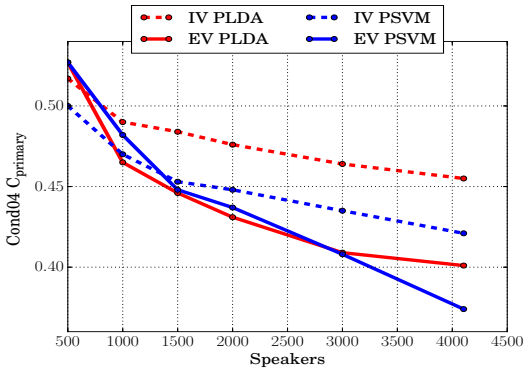
Both approaches also incorporates a technique for reducing the mismatch between the development and evaluation length distributions of the i–vectors, by estimating for each i–vector a scaling factor suited to the target i–vector distribution. Results obtained using NL–PLDA on the NIST SRE 2010 are reported in subsection V-C.

V. Experiments

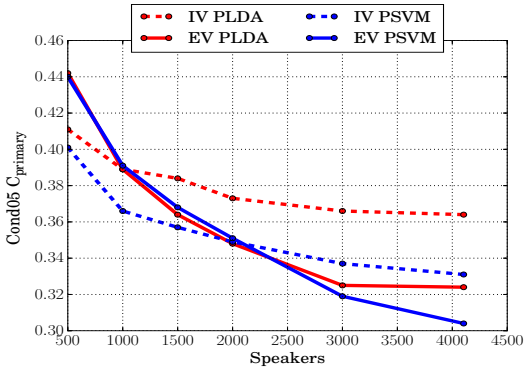
We compared the performance of i–vectors and e–vectors based systems mainly on the core extended NIST SRE 2012 tests. We will refer to the NIST SRE 2012 as SRE12 for short, and the same notation will be used for the other NIST SRE datasets. SRE12 was preferred to SRE10 as a testbed for the first set of experiments for a number of reasons:



(a) Phone call without added noise $C_{primary}$

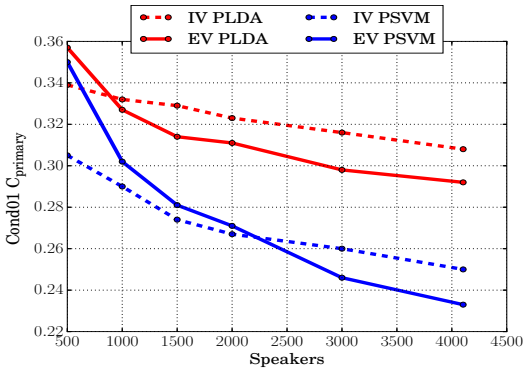


(b) Phone call with added noise $C_{primary}$

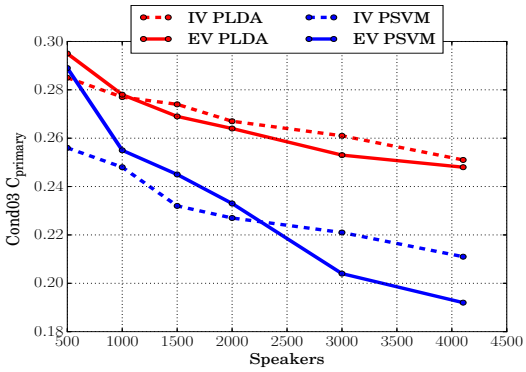


(c) Phone call noisy environment $C_{primary}$

Fig. 3: Plots of the $C_{primary}$ of a PLDA and a PSVM systems, using i–vectors or e–vectors, as a function of the number of speaker in the training set, on the telephone conditions of the core extended NIST SRE 2012 tests.



(a) Interview without added noise $C_{primary}$



(b) Interview with added noise $C_{primary}$

Fig. 4: Plots of the $C_{primary}$ of a PLDA and a PSVM systems, using i–vectors or e–vectors, as a function of the number of speaker in the training set, on the interview conditions of the core extended NIST SRE 2012 tests.

- It is known that the SRE10 male evaluation set is less difficult that the female evaluation, thus the results typically reported in literature refer to the latter.
  Our telephone female training set typically includes data from SRE04–06, and additionally the Switchboard II, Phases 2 and 3, and Switchboard Cellular, Parts 1 and 2 datasets [29], [43]. SRE08 is left out as a possible score normalization set. Score normalization, however, is not performed for all experiments described in this work. As shown in Table I, the total number of female speakers for these datasets is limited to 1972. This number increases to 2725 including SRE08.
  By adding the Fisher corpus, which includes telephone only segments, the number of speakers increases more than three times, but it is worth noting that in this dataset each speaker provides at most two segments only.
- Very few female speakers, and training segments, are available for the SRE10 interview or microphone conditions, as shown in Table II.
- The number of same–speaker and different–speaker female trials for the evaluation is small (3704 and 233077, respectively). Also the total number of speakers in evaluation is limited to 273. Thus, if the absolute error of a set of systems is small, their comparison becomes difficult because very few segments are incorrectly classified by different systems.

SRE12 does not suffer from the scarcity of training data and evaluation trials, because even excluding the Fisher dataset, exploiting the SRE10 training data, the number of training speaker and segments increases to 4103 and 79185, respectively. Moreover, the comparison of different systems becomes significant because the evaluation includes, summing all conditions, 27400 same–speaker and millions of different-speaker trials.

A possible drawback of SRE12 is that all test speakers have also contributed to the training set. However, we will show that even using only the segments of a half of the training speakers the performance of e–vector based systems (either PLDA or PSVM) improves with respect to the corresponding i–vector

TABLE III: Average %EER, DCF08, $C_{primary12}$, and relative improvement of PLDA and PSVM systems, using different systems with i–vectors or e–vectors, on the core extended NIST SRE 2012 tests. Label DNN refers to the hybrid DNN–GMM framework. Label SAS refers to the approach that post–processes the vectors by means of a cascade of affine and non–linear sinh–arcsinh transformations.

System	%EER	DCF08	$C_{primary12}$	% relative improvement		
PLDA IV	3.56	0.150	0.340			
PLDA EV	3.06	0.137	0.310	14.0%	8.7%	8.8%
PLDA SAS IV	3.75	0.148	0.312			
PLDA SAS EV	3.12	0.134	0.285	16.8%	9.5%	8.7%
PLDA DNN IV	3.39	0.134	0.295			
PLDA DNN EV	2.79	0.118	0.264	17.7%	11.9%	10.5%
PLDA DNN SAS IV	3.18	0.119	0.261			
PLDA DNN SAS EV	2.81	0.113	0.240	11.6%	5.0%	8.0%
PSVM IV	3.05	0.132	0.302			
PSVM EV	2.84	0.121	0.274	6.9 %	8.3%	9.3%
PSVM DNN IV	2.71	0.114	0.261			
PSVM DNN EV	2.41	0.104	0.234	11.1%	8.8%	10.3%

TABLE IV: % EER, DCF08, and $C_{primary}$ of PLDA and PSVM systems, using different systems with i–vectors or e–vectors, on the core extended SRE12 tests. An e–vector system performance that is worse the the corresponding i–vector system value is marked in boldface.

System	Cond 1 interview without noise			Cond 2 phone call without noise			Cond 3 interview with added noise			Cond 4 phone call with added noise			Cond 5 phone call noisy environment		
	%EER	DCF08	C_{prim12}	%EER	DCF08	C_{prim12}	%EER	DCF08	C_{prim12}	%EER	DCF08	C_{prim12}	%EER	DCF08	C_{prim12}
PLDA IV	3.77	0.149	0.308	2.38	0.121	0.321	3.59	0.120	0.251	5.19	0.217	0.455	2.88	0.143	0.364
PLDA EV	3.22	0.139	0.292	2.24	0.105	0.286	2.66	0.132	0.248	4.42	0.183	0.401	2.76	0.128	0.324
PLDA SAS IV	3.48	0.122	0.245	2.62	0.126	0.301	4.00	0.117	0.213	5.39	0.224	0.459	3.24	0.150	0.344
PLDA SAS EV	2.88	0.120	0.238	2.37	0.108	0.266	2.81	0.117	0.204	4.64	0.193	0.408	2.91	0.132	0.307
PLDA DNN IV	3.95	0.146	0.279	2.07	0.100	0.264	3.63	0.118	0.229	4.86	0.184	0.396	2.42	0.120	0.305
PLDA DNN EV	3.43	0.134	0.258	1.68	0.083	0.233	2.77	0.118	0.215	4.14	0.155	0.346	1.93	0.101	0.268
PLDA DNN SAS IV	3.54	0.115	0.230	1.79	0.090	0.231	3.84	0.109	0.192	4.59	0.177	0.387	2.16	0.106	0.265
PLDA DNN SAS EV	3.09	0.112	0.212	1.69	0.081	0.208	3.13	**0.112**	0.188	4.15	0.164	0.349	2.01	0.096	0.241
PSVM IV	3.20	0.117	0.250	2.27	0.114	0.295	2.75	0.103	0.211	4.32	0.192	0.421	2.73	0.136	0.331
PSVM EV	2.89	0.111	0.233	2.26	0.105	0.265	2.46	0.096	0.192	3.77	0.165	0.374	**2.83**	0.127	0.304
PSVM DNN IV	3.20	0.114	0.233	1.69	0.088	0.239	2.87	0.104	0.199	3.75	0.159	0.363	2.03	0.105	0.272
PSVM DNN EV	2.86	0.107	0.218	1.50	0.077	0.209	2.63	0.102	0.185	3.30	0.141	0.319	1.78	0.093	0.238

based systems. In this configuration, half of the speakers contributing to the evaluation set are unknown to the system.

Finally, the SRE12 setup is also an interesting case–study from an application perspective. Assuming that most of the customers may perform further calls, the system can be retrained after collecting a large enough set of speakers of the application domain.

Despite all the drawbacks related to the evaluation of e–vectors with SRE10, we also performed an additional set of experiments on the female telephone segments of the SRE10 core extended tests. We did not include the microphone/interview condition due to the scarcity of speakers providing microphone/interview training segments.

For all these experiments we used a set 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.

For the SRE12 experiments, we trained gender–independent i–vector and e–vector extractors, based on a 1024–component diagonal covariance UBM, estimated with data from SRE04–10, and the Switchboard datasets, for a total of 42522 utterances of 3209 speakers. The training list includes all telephone files, whereas we randomly selected a maximum of two files per interview session, eliminating very short or long duration segments, i.e., keeping segments lasting from 20 to 300 seconds.

Matrix $\mathbf{V}$ has been trained using 10 Maximum Likelihood and MDE iterations, then matrix $\mathbf{E}$ has been trained by means of 5 MDE iterations. MDE has also been applied for the 10 iterations that we used for matrix $\mathbf{T}$ training.

For the first set of experiments, the i–vector and e–vector dimensions were both set to $d = 400$, and the PLDA speaker factors were not reduced.

A. Performance using 400–dimensional vectors on SRE12

We first evaluated the performance of PLDA and PSVM classifiers using i–vector or e–vector models trained with the speech segments of an increasing number of speakers. The evaluation was performed on the core extended SRE12 dataset using the baseline PLDA and PSVM systems, and the UBM/GMM based vector extraction. All PLDA results have been obtained using whitening and length normalization.

Figures 3 and 4 plot the $C_{primary}$ cost function [29] of these systems as a function of the number of speakers in the training set. The first set of figures refers to the telephone call conditions, whereas the second set presents the results for the interview conditions. Looking at these figures, it is evident that the systems based on the eigenvoice subspace suffer from the lack of training speakers. However, the systems using e–vectors estimated with more than 2000 speakers outperform the corresponding i–vector systems. Thus, if the eigenvoice matrix $\mathbf{E}$ is trained with a large enough number of speakers, the extracted e–vectors are more informative about the speaker

identity than the corresponding i–vectors, extracted by using the total variability matrix **T**. Also evident in these figures is that a PSVM system trained with all available data is more accurate than the corresponding PLDA system for all conditions.

It is also worth noting, looking at Figures 3 and 4, that the e–vector systems are equivalent or better than the corresponding i–vector systems when trained with the segments of a half of the development speakers, so that the other half of the speakers appearing in the development set are as a matter of fact "new speakers" for the evaluation.

Table III summarizes the results of a set of experiments performed on the same evaluation dataset, using i–vectors or e–vectors. The matrices **T** and **E** trained for these experiments are the ones estimated with the full set of training data previously described. The results are given in terms of %EER, DCF08, and $C_{primary}$ cost functions [29], [44]. The last three columns show the relative improvement of e–vectors based systems with respect to the same system using i–vectors. Labels "SAS" and "DNN " refer to our non–linear i–vector transformation approach [21], and to the hybrid DNN–GMM system [42], respectively.

The first row reports the results of "PLDA IV", the baseline GMM–based PLDA system using i–vectors. The "PLDA EV" system is identical to the first one, but uses e–vectors rather than i–vectors. It improves the average decision cost functions by approximately 9% for the DCFs, and 14% for the EER. The e–vectors are also effective if they are subject to the cascade of affine and non–linear sinh–arcsinh (SAS) transformations that makes their distribution better approximate the standard normal distribution. The results of this approach are shown in the row of the tables labeled as "PLDA SAS EV". The improvement with respect to the corresponding "PLDA SAS IV" system is again of the order of 9% for the DCFs, and approximately 17% for the EER.

We also assessed the quality of e–vectors extracted by means of our hybrid DNN/GMM architecture [42]. Comparing rows "PLDA DNN EV" and "PLDA DNN IV", it can be noticed that using these more accurate models, the e–vectors provide even better improvement with respect to i–vectors.

Applying, additionally, the SAS transformation, we get further accuracy for both the "PLDA DNN SAS IV" and the "PLDA DNN SAS EV" systems. The e–vectors provide even in this case a smaller but consistent improvement with respect to i–vectors.

Finally, a third set of experiments was performed to verify that the e–vectors are better than i–vectors also using a different, and more accurate, discriminative classifier. The baseline performance of our PSVM approach is approximately 12% better than the baseline PLDA for both i–vectors and e–vectors. The effectiveness of the e–vectors with respect to the i–vectors is confirmed: the former gets approximately 9% accuracy gain both using vectors extracted by means of the standard GMM approach and exploiting the posteriors of the hybrid DNN/GMM framework.

We did not perform experiments using the SAS non–linear transformation of the i–vectors and e–vectors in the PSVM approach, because PSVM estimation is based on the optimization of the margin between the same-speaker and different-speaker classes, thus, it does not rely on the assumption that the input i–vectors are distributed according to the standard normal, or any other distribution.

Overall, using DNN–GMM and e–vectors the performance improvement with respect to the PLDA and PSVM baseline i–vector systems is approximately 29% and 22%, respectively.

It is worth noting that the MDE iterations are essential for the e–vector performance. For example, extracting e–vectors using the **V** matrix, rather than the **E** matrix, leads to a "PLDA EV" model that obtains an average $C_{primary}$ of 0.340, similar to the "PLDA IV" model, and a worse average for the % EER and DCF08 of 3.66 and 0.162, respectively. The results obtained for each condition defined by SRE12 are collected in Table IV.

B. Performance on SRE12 using vectors of other dimensions

Since some dimensions of the i–vectors subspace are wasted to take into account the channel effects, more dimensions are necessary for the i–vectors, with respect to the e–vectors, to accurately model the speaker identity. This can verified comparing the performance of the same classifiers using i–vectors or e–vectors of different dimensions. In particular we compared PLDA, PLDA SAS, and PSVM systems using vectors of dimension d equal to 300, 400, and 600, respectively.

The benefits of using e–vectors can be appreciated looking at the results shown in Table V and Figure 5. Table V reports the average SRE12 performance obtained by UBM/GMM systems using different dimension of the speaker vectors, whereas Figure 5 compares the $C_{primary}$ of different systems, using i–vectors or e–vectors, as a function of their dimensions. Looking at the average $C_{primary}$, it is possible to notice that PLDA SAS is always better than PLDA both for i–vector and e–vector based systems, and that PSVM is always better than the other classifiers.

Increasing the vector dimensions, the performance improves, and the results of the e–vector based systems are better than the corresponding i–vector based systems, with the exception of the 600–dimensional e–vectors PLDA.

The slight average $C_{primary}$ degradation observed for this system is due to a bad performance on the interview conditions, indicated in boldface in Table VI.

PLDA SAS is able to partially recover this problem, and in general it is more effective than PLDA on interview rather than in telephone conditions.

Driven by this observations, and by the desire to understand the reasons of the different behavior of large dimension e–vectors and i–vectors on interview data, we inspected the distribution of the vectors components. We observed that the distributions along the directions characterized by low values of the between class to within class covariance ratio largely deviate from a Gaussian distribution. These directions are more detrimental for e–vector PLDA than for i–vector PLDA because the former conveys in these directions speaker information, whereas the latter conveys mainly channel information. Indeed, removing just 10 directions using LDA allows significantly reducing these effects for 600–dimensional

TABLE V: Average %EER, DCF08, $C_{primary}$, and % relative improvement for PLDA, PLDA LDA, PLDA SAS, and PSVM systems on the core extended SRE12 tests, using i–vectors and e–vectors of increasing dimensions.

Features	System	% EER	DCF08	$C_{primary12}$	Relative improvement		
300 IV	PLDA	3.70	0.158	0.365			
	PLDA LDA	3.70	0.157	0.364			
	PLDA SAS	3.88	0.155	0.334			
	PSVM	3.27	0.147	0.333			
300 EV	PLDA	3.22	0.140	0.324	13.0%	11.4%	11.2%
	PLDA LDA	3.34	0.138	0.323	9.7%	12.1%	11.3%
	PLDA SAS	3.37	0.140	0.304	13.1%	9.7%	9.0%
	PSVM	3.01	0.132	0.300	8.0%	10.2%	9.9%
400 IV	PLDA	3.56	0.150	0.340			
	PLDA LDA	3.58	0.150	0.339			
	PLDA SAS	3.75	0.148	0.312			
	PSVM	3.05	0.132	0.302			
400 EV	PLDA	3.06	0.137	0.310	14.0%	8.7%	8.8%
	PLDA LDA	3.27	0.132	0.304	8.7%	12.0%	10.3%
	PLDA SAS	3.12	0.134	0.285	16.8%	9.5%	8.7%
	PSVM	2.84	0.121	0.274	6.9%	8.3%	9.3%
600 IV	PLDA	3.51	0.145	0.318			
	PLDA LDA	3.54	0.144	0.317			
	PLDA SAS	3.46	0.132	0.290			
	PSVM	2.84	0.122	0.276			
600 EV	PLDA	3.00	0.138	0.320	14.5%	4.8%	-0.6%
	PLDA LDA	3.26	0.129	0.287	7.9%	10.4%	9.5%
	PLDA SAS	3.09	0.130	0.280	10.7%	1.5%	3.4%
	PSVM	2.66	0.113	0.254	6.3%	7.4%	8.0%

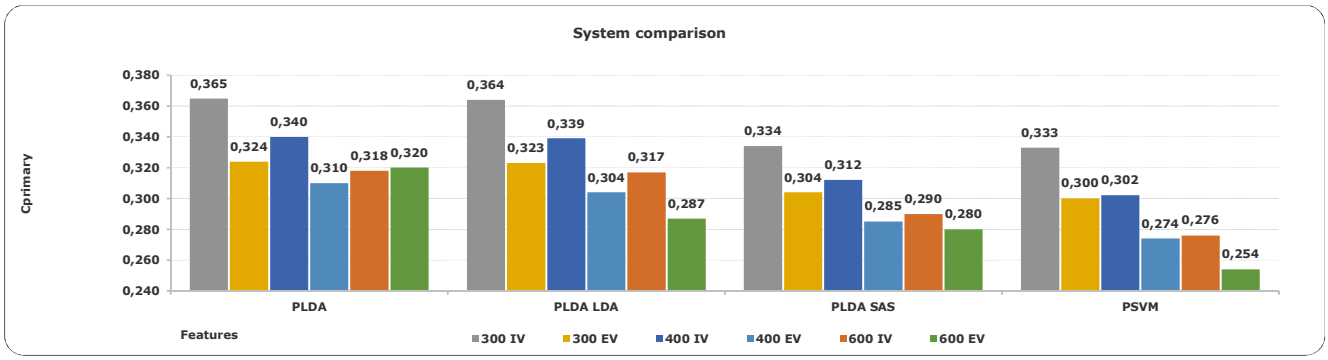


Fig. 5: Comparison of $C_{primary}$ for PLDA, PLDA LDA, PLDA SAS, and PSVM systems, using i–vectors or e–vectors, as a function of their dimensions.

TABLE VI: % EER, DCF08 and $C_{primary}$ on the core extended SRE12 tests for PLDA, PLDA LDA, PLDA SAS, and PSVM systems using i–vectors and e–vectors of 600 dimensions. An e–vector system performance that is worse than the corresponding i–vector system value is marked in boldface.

Feature	Classifier	Cond 1 interview without noise			Cond 3 phone call without noise			Cond 3 interview with added noise			Cond 4 phone call with added noise			Cond 5 phone call noisy environment		
		%EER	DCF08	C_{prim12}	%EER	DCF08	C_{prim12}	%EER	DCF08	C_{prim12}	%EER	DCF08	C_{prim12}	%EER	DCF08	C_{prim12}
600 IV	PLDA	4.05	0.152	0.297	2.43	0.117	0.300	3.39	0.117	0.232	4.78	0.196	0.418	2.91	0.141	0.342
	PLDA LDA	4.13	0.149	0.293	2.45	0.118	0.301	3.35	0.115	0.229	4.83	0.196	0.419	2.94	0.142	0.343
	PLDA SAS	3.43	0.117	0.240	2.50	0.110	0.275	3.57	0.105	0.204	4.85	0.196	0.420	2.96	0.133	0.313
	PSVM	3.07	0.109	0.226	2.16	0.105	0.269	2.61	0.100	0.198	3.78	0.170	0.382	2.58	0.124	0.305
600 EV	PLDA	3.38	0.143	**0.345**	2.23	0.103	0.268	2.61	0.149	0.309	4.00	0.168	0.370	2.78	0.128	0.310
	PLDA LDA	3.69	0.134	0.271	2.23	0.105	0.270	**3.61**	0.107	0.209	4.01	0.170	0.370	2.76	0.130	0.313
	PLDA SAS	2.81	0.114	0.237	2.46	0.107	0.257	2.86	**0.117**	**0.222**	4.33	0.182	0.382	3.01	0.132	0.303
	PSVM	2.94	0.106	0.218	2.07	0.097	0.244	2.24	0.098	0.186	3.45	0.147	0.340	2.58	0.117	0.281

TABLE VII: Comparison of the performance of i–vectors and e–vectors, in terms of % EER, DCF08 and DCF10, on the female telephone core extended SRE10 tests, using different PLDA classifiers based on 1024 or 2048 GMM components.

Features		i–vectors			e–vectors			Relative		
System	Dataset	% EER	DCF08	DCF10	% EER	DCF08	DCF10	improvement (%)		
1024 Diagonal GMM	SRE 04-06 + SWB	**2.91**	**0.144**	0.469	3.16	0.149	**0.468**	-8.6	-3.5	0.2
	+ SRE 08	3.00	0.147	0.462	**2.78**	**0.133**	**0.420**	7.3	9.5	9.1
	+ Fisher	2.59	0.137	0.449	**2.46**	**0.129**	**0.418**	5.0	5.8	6.9
2048 Diagonal GMM	SRE 04-06 + SWB	**2.43**	**0.123**	**0.404**	2.99	0.152	0.443	-23.0	-23.6	-9.6
	+ SRE 08	**2.44**	**0.121**	**0.410**	2.46	0.129	0.411	-0.8	-6.6	-0.2
	+ Fisher	2.27	0.110	0.357	**2.24**	**0.107**	**0.352**	1.3	2.7	1.4
+ Fisher in PLDA	All	2.00	0.103	0.340	**1.92**	**0.096**	**0.329**	4.0	6.8	3.2
2048 Diagonal DNN/GMM	All	1.62	0.082	0.324	**1.54**	**0.077**	**0.291**	4.9	6.1	10.2
2048 Diagonal DNN/GMM NL-PLDA	All	1.54	0.072	0.250	**1.48**	**0.068**	**0.219**	3.9	5.6	12.4
2048 Full DNN/GMM	All	1.48	0.079	0.260	**1.40**	**0.071**	**.0241**	5.41	10.1	7.3
2048 Full DNN/GMM NL-PLDA	All	1.46	0.068	**0.221**	**1.27**	**0.059**	0.230	13.0	13.2	-4.1

TABLE VIII: Comparison of the performance in terms of DCF08 and DCF10 of the 2048 Full DNN/GMM NL-PLDA i–vector and e–vector systems. The corresponding number of errors are given in parentheses.

System	DCF10	P_{miss}	P_{fa}	DCF08	P_{miss}	P_{fa}
2048 Full DNN/GMM NL-PLDA IV	0.221	18.3% (677)	3.9*10^{-3}% (9)	0.068	4.78% (177)	0.21% (480)
2048 Full DNN/GMM NL-PLDA EV	0.230	20.0% (740)	3.0*10^{-3}% (7)	0.059	3.72% (138)	0.22% (514)

e–vectors, whereas it does not sensibly affects the i–vector results.

Keeping all components, but applying the SAS transformation to e–vector, we are able to compensate these effects, and to improve e–vector PLDA performance. In particular, for vector of dimensions 300 and 400, PLDA SAS is more effective than LDA because it is able to make the e–vector distribution more Gaussian, whereas LDA just removes some directions. For 600–dimensional e–vectors, the improvement is less evident, but this is mainly due to the conditions characterized by artificial added noise, which was not seen in training. For conditions with real noise, PLDA SAS outperforms LDA also for 600–dimensional e–vectors.

It is interesting noting that the 300–dimensional e–vector PLDA system is almost equivalent to PLDA using 600–dimensional i–vectors.

PSVM, which does not rely on gaussianity assumptions, is able to keep the gain of e–vectors over i–vectors also for the 600–dimensional vectors, without any further processing, for all conditions

C. Performance on SRE10

Additional experiments on the female telephone segments of the core extended of SRE10 confirm that e–vectors are a good alternative to i–vectors when the number of speakers and segments is large enough.

We trained gender–dependent i–vector and e–vector extractors, again based on a 1024–component diagonal covariance UBM, but estimated with an increasing amount of data, beginning with the SRE04–06, and from Switchboard datasets, and incrementally adding the SRE08 and Fisher datasets. We did not include the microphone/interview conditions due the scarcity of speakers providing microphone/interview training segments. Training of matrices $\mathbf{E}$ and $\mathbf{T}$ has been performed exactly as we did for the the SRE12 experiments. The i–vector and e–vector dimensions were both set to $d = 400$, but the PLDA speaker factors were reduced to 150.

A summary of the results of these experiments, using PLDA classifiers, is given in Table VII.

As expected, training matrices $\mathbf{T}$ and $\mathbf{E}$ only with SRE04-06 and Switchboard (1972 speakers overall) does not allow a 1024 UBM/GMM e–vector system to outperform the corresponding i–vector system. However, adding to training the data of the 753 speakers of SRE08, the e–vector system gives better results, which become even better if the training list also includes the Fisher dataset.

The same behavior is obtained by using larger, more accurate, 2048 Gaussian models. Adding only SRE08 data is not sufficient for e–vectors to obtain better results with respect to i–vectors. However, including also the Fisher dataset to the training set, the models and the absolute performance of both i–vector and e–vector based systems improve, and we get a small advantage for the e–vectors with respect to i–vectors. Further improvement can be achieved for both i–vectors and e–vectors using the Fisher dataset also in PLDA estimation, as shown by the results given in the table row labeled "+ Fisher in PLDA".

The last two frames of Table VII show the result obtained by models trained using all available training data, and also including the Fisher dataset in PLDA training. The e–vectors are effective also in the hybrid DNN/GMM approach, both using 2048 diagonal or full covariance GMMs.

Finally, exploiting the posterior probability of the DNN in combination with the non–linear PLDA approach [22], we achieve our best results, where e–vector system keeps a 13% relative advantage with respect to the corresponding i–vector system for EER and DCF08. We do not consider statistically significant the 4% loss for DCF10 because setting the classification threshold to get the minimum DCF10, leads to very small false alarm errors - less than 10 - as shown in Table VIII. The number of errors corresponding to the the minimum DCF08 threshold is, instead, statistically more significant.

Using a PSVM in these experiments does not achieve better

results that the corresponding PLDA classifier because it does not take much advantage of the Fisher data. If these data are not included in the training set, the amount of training data is not sufficient for PSVM. Their inclusion is beneficial for PLDA, but it is not helpful for PSVM because Fisher data provide relatively few additional same–speaker pairs, compared to the huge amount of mostly useless different–speaker pairs.

VI. Conclusions

In this work we have shown that the eigenvoice space has more information about the speakers than the "total variability" i–vector subspace, because the latter includes more channel effects. To exploit this information, we have proposed a simple procedure for training an i–vector extractor that spans the same subspace of the JFA eigenvoice matrix. This leads to a compact representation of speech segments, similar to i–vectors, which we named e–vectors.

E–vectors have shown to be a good replacement of i–vectors, consistently providing better performance with different systems and classifiers. The best results were obtained on SRE12 using 600–dimensional e–vectors, particularly using a PSVM classifier. These performance gains come without additional memory or computational costs, on the contrary, 300–dimensional e–vector PLDA systems are almost equivalent to PLDA systems using 600–dimensional i–vectors.

Experiments performed on SRE10 female telephone data demonstrate that it is important that the training corpus contains enough speakers and multiple recordings to accurately model the speaker subspace.

References

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

[2] N. Dehak, R. Dehak, P. Kenny, N. Brümmer, and P. Ouellet, "Support Vector Machines versus fast scoring in the low-dimensional total variability space for speaker verification," in *Proceedings of Interspeech 2009*, pp. 1559–1562, 2009.

[3] 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.

[4] 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.

[5] P. Kenny, "Joint factor analysis of speaker and session variability: Theory and algorithms," in *Technical report CRIM-06/08-13*, 2005.

[6] S. Ioffe, "Probabilistic Linear Discriminant Analysis," in *Proceedings of the 9th European Conference on Computer Vision - Volume Part IV*, ECCV'06, pp. 531–542, 2006.

[7] S. J. D. Prince and J. H. Elder, "Probabilistic Linear Discriminant Analysis for inferences about identity," in *Proceedings of 11th International Conference on Computer Vision*, pp. 1–8, 2007.

[8] 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.

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

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

[11] S. Cumani, N. Brümmer, L. Burget, and P. Laface, "Fast discriminative speaker verification in the i–vector space," in *Proceedings of ICASSP 2011*, pp. 4852–4855, 2011.

[12] L. Burget, O. Plchot, S. Cumani, O. Glembek, P. Matějka, and N. Brümmer, "Discriminatively trained Probabilistic Linear Discriminant Analysis for speaker verification," in *Proceedings of ICASSP 2011*, pp. 4832–4835, 2011.

[13] 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.

[14] S. Cumani and P. Laface, "Training pairwise support vector machines with large scale datasets," in *ICASSP 2014*, 2014.

[15] 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.

[16] J. Rohdin, S. Biswas, and K. Shinoda, "Robust discriminative training against data insufficiency in PLDA-based speaker verification," *Computer Speech & Language*, vol. 35, pp. 32–7, 2016.

[17] A. Sholokhov, T. Kinnunen, and S. Cumani, "Discriminative multi-domain PLDA for speaker verification," in *Proceedings of ICASSP 2016*, pp. 5030–5034, 2016.

[18] A. Hatch, S. Kajarekar, and A. Stolcke, "Within-class covariance normalization for SVM-based speaker recognition," in *Proceedings of ICSLP 2006*, pp. 1471–1474, 2006.

[19] 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.

[20] S. Cumani and P. Laface, "I–vector transformation and scaling for PLDA based speaker recognition," in *Proceedings of Odyssey 2016*, pp. 39–46, 2016.

[21] 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.

[22] S. Cumani and P. Laface, "Joint estimation of PLDA and non–linear transformations of speaker vectors," 2017.

[23] 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.

[24] P. Kenny, G. Boulianne, P. Ouellet, and P. Dumouchel, "Joint Factor Analysis versus eigenchannels in speaker recognition," *IEEE Transactions on Audio Speech and Language Processing*, vol. 15, no. 4, pp. 1435–1447, 2007.

[25] P. Kenny, G. Boulianne, P. Ouellet, and P. Dumouchel, "Speaker and session variability in GMM-based speaker verification," *IEEE Transactions on Audio Speech and Language Processing*, vol. 15, no. 4, pp. 1448–1460, 2007.

[26] P. Kenny, P. Ouellet, N. Dehak, V. Gupta, and P. Dumouchel, "A study of interspeaker variability in speaker verification," *IEEE Transactions on Audio Speech and Language Processing*, vol. 5, no. 16, pp. 980–988, 2008.

[27] N. Dehak, *Discriminative and generative approaches for long- and short-term speaker characteristics modeling: Application to speaker verification*. PhD thesis, Ècole de Technologie Supérieure, Université du Québec, Montreal, Canada, 2009.

[28] N. Brummer, "EM for probabilistic LDA," 2010. Technical report, Agnitio Research, Cape Town, Available at sites.google.com/site/nikobrummer/.

[29] "The NIST year 2012 speaker recognition evaluation plan." Available at "http://www.nist.gov/itl/iad/mig/upload/NIST_SRE12_evalplan-v17-r1.pdf.

[30] "The NIST year 2010 speaker recognition evaluation plan." Available at http://www.itl.nist.gov/iad/mig/tests/sre/2010/NIST_SRE10_evalplan.r6.pdf.

[31] R. Kuhn, J. Junqua, P. Nguyen, and N. Niedzielski, "Rapid speaker adaptation in eigenvoice space," *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 6, pp. 695–707, 2000.

[32] O. Thyes, R. Kuhn, P. Nguyen, and J. Junqua, "Speaker identification and verification using eigenvoices," in *Proceedings of ICSLP 2000*, pp. 242–245, 2000.

[33] S. Lucey and T. Chen, "Improved speaker verification through probabilistic subspace adaptation," in *Proceedings of EUROSPEECH 2003*, pp. 2021–2024, 2003.

[34] P. Kenny, M. Mihoubi, and P. Dumouchel, "New map estimators for speaker recognition," in *Proceedings of EUROSPEECH 2003*, pp. 2964–2967, 2003.

[35] P. Kenny, G. Boulianne, and P. Dumouchel, "Eigenvoice modeling with sparse training data," *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 3, pp. 345–354, 2005.
[36] M. Taboga, "Lectures on probability theory and mathematical statistics," 2012. Available at https://www.statlect.com/probability-distributions/normal-distribution-linear-combinations.
[37] P. Kenny, "A small footprint i-vector extractor," in *Proceedings of Odyssey 2012*, pp. 1–6, 2012.
[38] 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.
[39] P. Kenny, V. Gupta, T. Stafylakis, P. Ouellet, and J. Alam, "Deep Neural Networks for extracting Baum-Welch statistics for speaker recognition," in *Proceedings of Odyssey 2014*, pp. 293–298, 2014.
[40] D. Garcia-Romero and A. McCree, "Insights into Deep Neural Networks for speaker recognition," in *Proceedings of Interspeech 2015*, pp. 1141–1145, 2015.
[41] F. Richardson, D. A. Reynolds, and N. Dehak, "A unified Deep Neural Network for speaker and language recognition," in *Proceedings of Interspeech 2015*, pp. 1146–1150, 2015.
[42] 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.
[43] Available at http://www.ldc.upenn.edu/Catalog/.
[44] NIST, "The NIST year 2008 and 2010 speaker recognition evaluation plans." http://www.itl.nist.gov/iad/mig/tests/sre.



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.