Discriminatively Trained i-vector Extractor for Speaker Verification

Ondřej Glembek[1], Lukáš Burget[1], Niko Brümmer[2], Oldřich Plchot[1], Pavel Matějka[1]

[1]Speech@FIT group, Brno University of Technology, Czech Republic
[2]Agnitio, South Africa
{glembek,burget,matejkap}@fit.vutbr.cz,
niko.brummer@gmail.com

10.21437/Interspeech.2011-49

Abstract

We propose a strategy for discriminative training of the i-vector extractor in speaker recognition. The original i-vector extractor training was based on the maximum-likelihood generative modeling, where the EM algorithm was used. In our approach, the i-vector extractor parameters are numerically optimized to minimize the discriminative cross-entropy error function. Two versions of the i-vector extraction are studied—the original approach as defined for Joint Factor Analysis, and the simplified version, where orthogonalization of the i-vector extractor matrix is performed.

Index Terms: speaker verification, i-vectors, PLDA, discriminative training

1. Introduction

Recently, systems based on i-vectors [1, 2] (extracted from cepstral features) have provided superior performance in speaker verification. The so-called i-vector is an information-rich low-dimensional fixed-length vector extracted from the feature sequence representing a speech segment (see Section 2 for details on i-vector extraction). A speaker verification score is produced by comparing two i-vectors corresponding to the segments in the verification trial. The function taking two i-vectors as an input and producing the corresponding verification score is designed to give the log-likelihood ratio between the "same-speaker" and "different-speaker" hypotheses. Best performance is currently obtained with Probabilistic Linear Discriminant Analysis (PLDA) [2]—a generative model that models i-vector distributions allowing for direct evaluation of the desired log-likelihood ratio verification score (see Section 2.4 for details).

In [3], it was shown that discriminatively training the PLDA parameters can lead to improvement in recognition performance. In this paper, we go deeper in the speaker recognition chain and we show that a similar discriminative training framework can be adopted for training the parameters of the i-vector extractor. We apply this technique in two kinds of i-vector extractor. In the first case, the traditional extraction—as proposed in [1]—is studied. It will be further referred to as the *full i-vector extractor*. Its parameters are given by a single matrix $\mathbf{T}$. In the second case, the simplified extraction (referred to as "Simplification 2" in [4]) is addressed. Its parameters are given by three matrices—$\mathbf{T}$, $\mathbf{G}$, and $\mathbf{V}$. It will be further referred to as the *simplified i-vector extractor*.

This paper is organized as follows: Section 2 introduces a theoretical background of the individual parts of the speaker recognition chain, Section 3 introduces the technique of discriminative training, Section 4 describes the experimental setup and results, and Section 5 concludes the paper.

BUT researchers carrying on this work were funded by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), through the Army Research Laboratory (ARL). All statements of fact, opinion or conclusions contained herein are those of the authors and should not be construed as representing the official views or policies of IARPA, the ODNI, or the U. S. Government.

2. Theoretical background

The i-vectors provide an elegant way of reducing large-dimensional input data to a small-dimensional feature vector while retaining most of the relevant information. The technique was originally inspired by Joint Factor Analysis (JFA) framework introduced in [5, 6].

The main idea is that the speaker- and channel-dependent Gaussian Mixture Model (GMM) supervector $\mathbf{s}$ can be modeled as:

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

where $\mathbf{m}$ is the Universal Background Model (UBM) GMM mean supervector, $\mathbf{T}$ is a low-rank matrix representing M bases spanning subspace with important variability in the mean supervector space, and $\mathbf{w}$ is a latent variable of size M with standard normal distribution.

For each observation $\mathcal{X}$, the aim is to compute the parameters of the posterior probability of $\mathbf{w}$:

$$\mathrm{p}(\mathbf{w}|\mathcal{X}) = \mathcal{N}(\mathbf{w}; \mathbf{w}_{\mathcal{X}}, \mathbf{L}_{\mathcal{X}}^{-1}) \tag{2}$$

The i-vector ϕ is the Maximum a Posteriori (MAP) point estimate of the variable $\mathbf{w}$, i.e., the mean $\mathbf{w}_{\mathcal{X}}$ of the posterior distribution $\mathrm{p}(\mathbf{w}|\mathcal{X})$. It maps most of the relevant information from a variable-length observation $\mathcal{X}$ to a fixed- (small-) dimensional vector. $\mathbf{L}_{\mathcal{X}}$ is the precision of the posterior distribution.

2.1. Sufficient statistics

The input data for the observation $\mathcal{X}$ is given as a set of *zero-* and *first-order statistics* — $\mathbf{n}_{\mathcal{X}}$ and $\mathbf{f}_{\mathcal{X}}$. These are extracted from F dimensional features using a GMM UBM with C mixture components, defined by a mean supervector $\mathbf{m}$, component covariance matrices $\mathbf{\Sigma}^{(c)}$, and a vector of mixture weights $\boldsymbol{\omega}$. For each Gaussian component c, the statistics are given respectively as

$$N_{\mathcal{X}}^{(c)} = \sum_t \gamma_t^{(c)} \tag{3}$$

$$\mathbf{f}_{\mathcal{X}}^{(c)} = \sum_t \gamma_t^{(c)} \mathbf{o}_t \tag{4}$$

Copyright © 2011 ISCA

where $\mathbf{o}_t$ is the feature vector in time t, and $\gamma_t^{(c)}$ is its occupation probability. The complete zero- and first-order statistics supervectors are $\mathbf{f}_\mathcal{X} = \left(\mathbf{f}_\mathcal{X}^{(1)\prime}, \ldots, \mathbf{f}_\mathcal{X}^{(C)\prime}\right)'$, and $\mathbf{n}_\mathcal{X} = \left(N_\mathcal{X}^{(1)}, \ldots, N_\mathcal{X}^{(C)}\right)'$.

For convenience, we *center* the first-order statistics around the UBM means, which allows us to treat the UBM means effectively as a vector of zeros:

$$\begin{aligned} \mathbf{f}_\mathcal{X}^{(c)} &\leftarrow \mathbf{f}_\mathcal{X}^{(c)} - N_\mathcal{X}^{(c)}\mathbf{m}^{(c)} \\ \mathbf{m}^{(c)} &\leftarrow \mathbf{0} \end{aligned}$$

Similarly, we "normalize" the first-order statistics and the matrix $\mathbf{T}$ by the UBM covariances, which again allows us to treat the UBM covariances as an identity matrix:[1]

$$\begin{aligned} \mathbf{f}_\mathcal{X}^{(c)} &\leftarrow \mathbf{\Sigma}^{(c)-\frac{1}{2}}\mathbf{f}_\mathcal{X}^{(c)} \\ \mathbf{T}^{(c)} &\leftarrow \mathbf{\Sigma}^{(c)-\frac{1}{2}}\mathbf{T}^{(c)} \\ \mathbf{\Sigma}^{(c)} &\leftarrow \mathbf{I} \end{aligned}$$

where $\mathbf{\Sigma}^{(c)-\frac{1}{2}}$ is a Cholesky decomposition of an inverse of $\mathbf{\Sigma}^{(c)}$, and $\mathbf{T}^{(c)}$ is an $F \times M$ submatrix of $\mathbf{T}$ corresponding to the c mixture component such that $\mathbf{T} = \left(\mathbf{T}^{(1)\prime}, \ldots, \mathbf{T}^{(C)\prime}\right)'$.

2.2. i-vector extraction

As described in [5] and with the data transforms from the previous section, for an observation $\mathcal{X}$, the corresponding i-vector is computed as a point estimate:

$$\boldsymbol{\phi}_\mathcal{X} = \mathbf{L}_\mathcal{X}^{-1}\mathbf{T}'\mathbf{f}_\mathcal{X} \tag{5}$$

where $\mathbf{L}$ is the precision matrix of the posterior distribution, computed as

$$\mathbf{L}_\mathcal{X} = \mathbf{I} + \sum_{c=1}^{C} N_\mathcal{X}^{(c)}\mathbf{T}^{(c)\prime}\mathbf{T}^{(c)} \tag{6}$$

2.3. i-vector extraction—simplified version

According to [4], the i-vector extraction can be simplified to reduce the computation complexity. Assuming there is a linear (orthogonal) transformation $\mathbf{G}$ that would orthogonalize all individual per-component submatrices $\mathbf{T}^{(c)}$, the i-vector extraction can be expressed as

$$\hat{\boldsymbol{\phi}}_\mathcal{X} = \mathbf{G}\hat{\mathbf{L}}_\mathcal{X}^{-1}\mathbf{G}'\mathbf{T}'\mathbf{f}_\mathcal{X} \tag{7}$$

where

$$\hat{\mathbf{L}}_\mathcal{X} = \mathrm{Diag}\left(\mathbf{I} + \mathbf{V}\mathbf{n}_\mathcal{X}\right) \tag{8}$$

where $\mathbf{V}$ is an $M \times C$ matrix whose cth column is $\mathrm{diag}(\mathbf{G}'\mathbf{T}^{(c)\prime}\mathbf{T}^{(c)}\mathbf{G})$. $\mathrm{Diag}(\cdot)$ maps a vector to a diagonal matrix.

2.4. PLDA

To facilitate comparison of i-vectors in a verification trial, we use a Probabilistic Linear Discriminant Analysis (PLDA) model [7, 2]. It can be seen as a special case of JFA with a single Gaussian component. Given a pair of i-vectors, PLDA allows to compute the log-likelihood for the same-speaker hypothesis and for the different-speaker hypothesis. One can directly evaluate the log-likelihood ratio of the same-speaker and different-speaker trial using

$$\begin{aligned} s(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2) &= \boldsymbol{\phi}_1^T\mathbf{\Lambda}\boldsymbol{\phi}_2 + \boldsymbol{\phi}_2^T\mathbf{\Lambda}\boldsymbol{\phi}_1 + \boldsymbol{\phi}_1^T\mathbf{\Gamma}\boldsymbol{\phi}_1 + \boldsymbol{\phi}_2^T\mathbf{\Gamma}\boldsymbol{\phi}_2 \\ &+ (\boldsymbol{\phi}_1 + \boldsymbol{\phi}_2)^T\mathbf{c} + k, \end{aligned} \tag{9}$$

where $\mathbf{\Lambda}$, $\mathbf{\Gamma}$, $\mathbf{c}$, k are derived from the parameters of PLDA as in [3].

2.5. i-vector length normalization

PLDA assumes that the input i-vectors are normally distributed. However, in earlier studies ([2]), it has been shown that this assumption is not met.

Length normalization [1, 8] of the i-vectors forces them to lie on a unity sphere, which brings them closer to the Gaussian distribution shell where most of the probability density mass is concentrated. The transformation is given as

$$\bar{\boldsymbol{\phi}} = \frac{\boldsymbol{\phi}}{\|\boldsymbol{\phi}\|} = \frac{\boldsymbol{\phi}}{\sqrt{\boldsymbol{\phi}'\boldsymbol{\phi}}} \tag{10}$$

3. Discriminative classifier

We describe how we train the i-vector extractor parameters $\boldsymbol{\theta}$ in order to discriminate between same-speaker and different-speaker trials, without having to explicitly model the distributions of i-vectors.

The set of training examples, which we continue referring to as training trials, comprises both different-speaker, and same-speaker trials. Let us use the coding scheme $t \in \{-1, 1\}$ to represent labels for the different-speaker, and same-speaker trials, respectively. Assigning each trial a log-likelihood ratio s and the correct label t, the log probability of recognizing the trial correctly can be expressed as

$$\log p(t|\boldsymbol{\phi}_1, \boldsymbol{\phi}_2) = -\log(1 + \exp(-st)). \tag{11}$$

In the case of logistic regression, the objective function to be maximized is the log probability of correctly classifying all training examples, i.e., the sum of expressions (11) evaluated for all training trials. Equivalently, this can be expressed by minimizing the cross-entropy error function, which is a sum over all training trials

$$E(\boldsymbol{\theta}) = \sum_{n=1}^{N} \alpha_n E_{LR}(t_n s_n) + \frac{\lambda}{2}\|\boldsymbol{\theta} - \boldsymbol{\theta}_{\mathrm{ML}}\|^2, \tag{12}$$

where the logistic regression loss function

$$E_{LR}(ts) = \log(1 + \exp(-ts)) \tag{13}$$

is simply the negative log probability (11) of correctly recognizing a trial. We have also added the regularization term $\frac{\lambda}{2}\|\boldsymbol{\theta} - \boldsymbol{\theta}_{\mathrm{ML}}\|^2$, where λ is a constant controlling the trade-off between the error function and the regularizer, and $\boldsymbol{\theta}_{\mathrm{ML}}$ is the original maximum-likelihood estimate of the given parameter. This kind of regularization is similar to the sum-of-squares penalty; however, it controls the distance from the original parameters rather than the parameter range itself. This way, optimizing the error function fine tunes the already good parameters.

The coefficients α_n allow us to weight individual trials. Specifically, we use them to assign different weights to same-speaker and different-speaker trials. This allows us to select

[1] Part of the factor computation is the evaluation of $\mathbf{T}'\mathbf{\Sigma}^{-1}\mathbf{f}$, where the decomposed $\mathbf{\Sigma}^{-1}$ can be projected to the neighboring terms, see [5] for detailed formulae.

a particular operating point, around which we want to optimize the performance of our system without relying on the proportion of same- and different-speaker trials in the training set. The advantage of using the cross-entropy objective for training is that it reflects performance of the system over a wide range of operating points (around the selected point).

3.1. Gradient evaluation

In order to numerically optimize the parameters $\boldsymbol{\theta}$, we want to express the gradient of the error function

$$\nabla E(\boldsymbol{\theta}) = \sum_{n=1}^{N} \alpha_n \frac{\partial E_{LR}(t_n s_n)}{\partial \boldsymbol{\theta}} + \lambda(\boldsymbol{\theta} - \boldsymbol{\theta}_{\mathrm{ML}}). \quad (14)$$

We see that the loss function $E_{LR}(t_n s_n)$ is not directly dependent on $\boldsymbol{\theta}$; therefore, the chain rule must be subsequently applied.

Let us start by deriving the loss function w.r.t. the direct parameters of E_{LR}

$$\frac{\partial E_{LR}}{\partial \boldsymbol{\theta}} = \frac{\partial E_{LR}}{\partial s}\frac{\partial s}{\partial \boldsymbol{\theta}} \quad (15)$$

The first r.h.s. fraction of (15) is defined as

$$\frac{\partial E_{LR}(ts)}{\partial s} = -t\sigma(-ts), \quad (16)$$

where $\sigma(\cdot)$ is the logistic function. Noting that the score s is a function of a length-normalized i-vector pair

$$s = s(\bar{\boldsymbol{\phi}}_1, \bar{\boldsymbol{\phi}}_2),$$

we get

$$\frac{\partial s_n}{\partial \boldsymbol{\theta}} = \frac{s(\bar{\boldsymbol{\phi}}_1, \bar{\boldsymbol{\phi}}_2)}{\partial \bar{\boldsymbol{\phi}}_1}\frac{\partial \bar{\boldsymbol{\phi}}_1}{\partial \boldsymbol{\theta}} + \frac{s(\bar{\boldsymbol{\phi}}_1, \bar{\boldsymbol{\phi}}_2)}{\partial \bar{\boldsymbol{\phi}}_2}\frac{\partial \bar{\boldsymbol{\phi}}_2}{\partial \boldsymbol{\theta}} \quad (17)$$

From (9), knowing that $\boldsymbol{\Lambda}$ and $\boldsymbol{\Gamma}$ are symmetrical, we can derive

$$\frac{s(\bar{\boldsymbol{\phi}}_1, \bar{\boldsymbol{\phi}}_2)}{\partial \bar{\boldsymbol{\phi}}_1} = 2\boldsymbol{\phi}_2'\boldsymbol{\Lambda} + 2\boldsymbol{\phi}_1'\boldsymbol{\Gamma} + \mathbf{c} \quad (18)$$

Note that the two sides of the trial can be swapped so that an analogous equation applies when deriving w.r.t. $\boldsymbol{\phi}_2$. Again, we apply the chain rule to derive through the length normalization:

$$\frac{\partial \bar{\boldsymbol{\phi}}}{\partial \boldsymbol{\theta}} = \frac{\partial \bar{\boldsymbol{\phi}}}{\partial \boldsymbol{\phi}}\frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{\theta}} \quad (19)$$

where

$$\frac{\partial \bar{\boldsymbol{\phi}}}{\partial \boldsymbol{\phi}} = \frac{1}{\|\boldsymbol{\phi}\|}\left(\mathbf{I} - (\bar{\boldsymbol{\phi}}\bar{\boldsymbol{\phi}}')\right). \quad (20)$$

At this point, it is trivial to express the cross-entropy E as a function of some arbitrary set of M i-vectors $\boldsymbol{\Phi} = (\boldsymbol{\phi}_1, \cdots, \boldsymbol{\phi}_M)$. With the given formulas for derivatives, it is also straightforward to express the gradient $\frac{\partial E(\boldsymbol{\Phi})}{\partial \boldsymbol{\Phi}}$. To derive through the i-vector extractor, we will make use of the chain rule for differentials, where the following holds:

$$dE = \sum_{ij}\frac{\partial E}{\partial \phi_{ij}}d\phi_{ij} = \sum_{k}\frac{\partial E}{\partial \theta_k}d\theta_k. \quad (21)$$

By making use of the matrix differentials, we can express $d\boldsymbol{\Phi}$ as a function of $d\boldsymbol{\theta}$. For the full i-vector extractor, the differential for j-th column of $d\boldsymbol{\Phi}$ is given as

$$d\boldsymbol{\phi}_j = -\mathbf{L}_j^{-1}d\mathbf{L}_j\mathbf{L}_j^{-1}\mathbf{T}'\mathbf{f}_j + \mathbf{L}_j^{-1}d\mathbf{T}'\mathbf{f}_j \quad (22)$$

$$d\mathbf{L}_j = \sum_c N_j^{(c)}\left(d\mathbf{T}^{(c)\prime}\mathbf{T}^{(c)} + \mathbf{T}^{(c)\prime}d\mathbf{T}^{(c)}\right) \quad (23)$$

In the case of the simplified i-vector extractor, the corresponding differentials w.r.t. the matrices $\mathbf{T}$, $\mathbf{G}$, and $\mathbf{V}$ are given respectively as

$$d\boldsymbol{\phi}_{\mathbf{T}j} = \mathbf{G}\hat{\mathbf{L}}_j^{-1}\mathbf{G}'d\mathbf{T}'\mathbf{f}_j \quad (24)$$

$$d\boldsymbol{\phi}_{\mathbf{G}j} = \left(d\mathbf{G}\hat{\mathbf{L}}_j^{-1}\mathbf{G}' + \mathbf{G}\hat{\mathbf{L}}_j^{-1}d\mathbf{G}'\right)\mathbf{T}'\mathbf{f}_j \quad (25)$$

$$d\boldsymbol{\phi}_{\mathbf{V}j} = -\mathbf{G}\hat{\mathbf{L}}_j^{-1}\mathrm{Diag}(d\mathbf{V}\mathbf{n}_j)\hat{\mathbf{L}}_j^{-1}\mathbf{G}'\mathbf{T}'\mathbf{f}_j \quad (26)$$

where $\hat{\mathbf{L}}$ is defined in (8). Substituting one of the $d\boldsymbol{\phi}$ from the above catalogue to (21), we can find the gradient $\frac{\partial E(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$.

In the case of the full i-vector extractor, the derivative can be expressed as

$$\frac{\partial E(\mathbf{T})}{\partial \mathbf{T}} = \sum_{j=1}^{M} -\left(\mathbf{L}_j^{-1}\frac{\partial E}{\partial \boldsymbol{\phi}_j'}\boldsymbol{\phi}_j' + \boldsymbol{\phi}_j\frac{\partial E}{\partial \boldsymbol{\phi}_j}\mathbf{L}_j^{-1}\right)\mathbf{T}'\mathbf{N}_j + \mathbf{L}_j^{-1}\frac{\partial E}{\partial \boldsymbol{\phi}_j}\mathbf{f}_j, \quad (27)$$

where $\mathbf{N}_j$ is a diagonal matrix, whose entries are $(N_j^{(1)}, \cdots, N_j^{(1)}, N_j^{(2)}, \cdots, N_j^{(2)}, \cdots)$, where every $N_j^{(i)}$ of $\mathbf{n}_j$ is expanded to match the feature dimensionality.

For the simplified i-vector extraction, the derivatives of the parameters are

$$\frac{\partial E(\mathbf{T})}{\partial \mathbf{T}} = \sum_{j=1}^{M}\mathbf{f}_j\frac{\partial E}{\partial \boldsymbol{\phi}_j}\mathbf{G}\hat{\mathbf{L}}_j^{-1}\mathbf{G}' \quad (28)$$

$$\frac{\partial E(\mathbf{G})}{\partial \mathbf{G}} = \sum_{j=1}^{M}\hat{\mathbf{L}}_j^{-1}\mathbf{G}'\left(\mathbf{T}'\mathbf{f}_j\frac{\partial E}{\partial \boldsymbol{\phi}_j} + \frac{\partial E}{\partial \boldsymbol{\phi}_j'}\mathbf{f}_j'\mathbf{T}\right) \quad (29)$$

$$\frac{\partial E(\mathbf{V})}{\partial \mathbf{V}} = \sum_{j=1}^{M} -\mathbf{n}_j\left(\frac{\partial E}{\partial \boldsymbol{\phi}_j}\mathbf{G}' \circ \mathbf{f}_j'\mathbf{T}\mathbf{G}\hat{\mathbf{L}}_j^{-2}\right) \quad (30)$$

where the $\circ$ stands for the Hadamard product.

4. Experiments

4.1. Test setup

The results of our experiments are reported on the female part of Condition 5 of the NIST 2010 speaker recognition evaluation (SRE) dataset [9]. The recognition accuracy is given as a set of equal error rate (EER), and the normalized detection cost function (DCF) as defined in both the NIST 2010 SRE task ($\mathrm{DCF}_{\mathrm{new}}$) and the previous SRE evaluations ($\mathrm{DCF}_{\mathrm{old}}$).

4.2. Feature extraction

In our experiments, we used cepstral features, extracted using a 25 ms Hamming window. 19 mel frequency cepstral coefficients together with log energy were calculated every 10 ms. This 20-dimensional feature vector was subjected to short time gaussianization [10] using a 3 s sliding window. Delta and double delta coefficients were then calculated using a five-frame window giving a 60-dimensional feature vector.

Segmentation was based on the Brno University of Technology (BUT) Hungarian phoneme recognizer and relative average energy thresholding. Also, short segments were pruned out, after which the speech segments were merged.

4.3. System Setup

One gender-independent UBM was represented as a diagonal covariance, 64-component GMM. It was trained using LDC releases of Switchboard II Phases 2 and 3, Switchboard Cellular Parts 1 and 2, and NIST 2004-2005 SRE.

The initial i-vector extractor $\mathbf{T}$ was trained on the female portion of the following telephone data: NIST SRE 2004, NIST SRE 2005, NIST SRE 2006, Switchboard II Phases 2 and 3, Switchboard Cellular Parts 1 and 2, Fisher English Parts 1 and 2, giving 8396 female speakers in 1463 hours of speech. The dimensionality of the i-vectors was set to 400. The initial orthogonalization matrix $\mathbf{G}$ was estimated using heteroscedastic linear discriminant analysis (HLDA), as described in [4].

As described in Section 2.5, length normalization was applied after i-vector extraction.

PLDA was trained using the same data set as the $\mathbf{T}$ matrix. Only the Fisher portion was trimmed off, reducing the amount of data by approximately 50%. The across-class covariance matrix (eigen-voices) was of rank 90, and the within-class covariance matrix (eigen-channels) was full-rank.

The training dataset for the discriminative training was identical to the dataset of PLDA. The cross-entropy function was evaluated on the complete trial set, i.e., all training samples were scored against each other, giving 378387 same-speaker trials, and over 468 million different-speaker trials.

4.4. Numerical optimization

The numerical optimization of the parameters was performed in matlab using the optimization and differentiation tools in the BOSARIS Toolkit [11]. It uses the trust region Newton conjugate gradient method, as described in [12, 13]. In addition to the first derivatives as given in Section 3.1, this method needs to evaluate the second order Hessian-vector product [14], which can be effectively computed via the 'complex step differentiation' [15].

Different values for the regularization coefficient λ were tested. Good convergence and stability were observed when setting it to 0.2 for the full i-vector extractor parameters, and 0.8 for the simplified version. In the case of the simplified version, the matrices $\mathbf{G}$ and $\mathbf{T}$ were optimized subsequently. It was found, however, that even though optimizing $\mathbf{V}$ kept on decreasing the error function, it would always decrease the recognition performance on the test set. Different regularizers were also tested; however, it turned out that together with good initialization, the discriminative training works only as a "fine-tuner" of the initial parameters.

Table 1 shows the situation when training the full i-vector extractor. There is only a slight improvement in performance. In the case of the simplified i-vector extractor, the improvement

Table 1: *Comparison of ML and discriminatively trained full i-vector extractors in terms of normalized DCFs and EER*

	$\mathrm{DCF}_{\mathrm{new}}$	$\mathrm{DCF}_{\mathrm{old}}$	EER
ML	0.6678	0.2200	4.74
discriminative	0.6478	0.2144	4.41

is more apparent—see Table 2 for results. We see that the simplified system is still worse than the full one; however, discriminative training has shown its potential.

Table 2: *Comparison of ML and discriminatively trained simplified i-vector extractors in terms of norm. DCFs and EER*

	$\mathrm{DCF}_{\mathrm{new}}$	$\mathrm{DCF}_{\mathrm{old}}$	EER
ML	0.7496	0.2710	6.18
discriminative	0.6691	0.2403	5.41

5. Conclusions

We have proposed a technique for discriminative training of the i-vector extractor parameters using cross-entropy as the error function. We have applied the technique both to the original i-vector extractor and to its simplified version. In both cases, the discriminative training was effective, giving higher relative improvement in the simplified case.

6. 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. PP, no. 99, pp. 1 –1, 2010.

[2] Patrick Kenny, "Bayesian speaker verification with heavy–tailed priors," in *Proc. of Odyssey 2010*, Brno, Czech Republic, June 2010, http://www.crim.ca/perso/patrick.kenny, keynote presentation.

[3] L. Burget, O. Plchot, S. Cumani, O. Glembek, P. Matějka, and N. Brümmer, "Discriminatively trained probabilistic linear discriminant analysis for speaker verification," in *Proc. of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Prague, CZ, May 2011.

[4] O. Glembek, P. Matějka, and L. Burget, "Simplification and optimization of i-vector extraction," in *Proc. of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Prague, CZ, May 2011.

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

[6] P. Kenny, G. Boulianne, P. Oullet, and P. Dumouchel, "Joint factor analysis versus eigenchannels in speaker recognition," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 15, no. 7, pp. 2072–2084, 2007.

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

[8] Daniel Garcia-Romero, "Analysis of i-vector length normalization in Gaussian-PLDA speaker recognition systems," in *Proc. of the International Conference on Spoken Language Processing (ICSLP)*, 2011.

[9] "National institute of standards and technology," http://www.nist.gov/speech/tests/spk/index.htm.

[10] J. Pelecanos and S. Sridharan, "Feature warping for robust speaker verification," in *Proceedings of Odyssey 2006: The Speaker and Language Recognition Workshop*, 2006, pp. 213–218.

[11] Niko Brümmer and Edwards de Villiers, "The BOSARIS toolkit," http://sites.google.com/site/bosaristoolkit/.

[12] Chih-Jen Lin, Ruby C. Weng, and S. Sathiya Keerthi, "Trust region Newton method for large-scale logistic regression," *Journal of Machine Learning Research*, Sept. 2008.

[13] Jorge Nocedal and Stephen J. Wright, *Numerical Optimization*, Springer, 2nd edition, 2006.

[14] Barak A. Pearlmutter, "Fast exact multiplication by the Hessian," *Neural Computation*, vol. 6, pp. 147–160, 1994.

[15] Lawrence F. Shampine, "Accurate numerical derivatives in MATLAB," *ACM Trans. Math. Softw.*, pp. –1–1, 2007.