FAST DISCRIMINATIVE SPEAKER VERIFICATION IN THE I–VECTOR SPACE

Sandro Cumani [*], *Niko Brümmer* [+], *Lukáš Burget* [o], *Pietro Laface* [*]

[*] Politecnico di Torino, Italy, Sandro.Cumani, Pietro.Laface@polito.it
[o] Brno University of Technology, Czech Republic, burget@fit.vutbr.cz
[+] AGNITIO, South Africa, niko.brummer@gmail.com

ABSTRACT

This work presents a new approach to discriminative speaker verification. Rather than estimating speaker models, or a model that discriminates between a speaker class and the class of all the other speakers, we directly solve the problem of classifying pairs of utterances as belonging to the same speaker or not.

The paper illustrates the development of a suitable Support Vector Machine kernel from a state–of–the–art generative formulation, and proposes an efficient approach to train discriminative models.

The results of the experiments performed on the tel–tel extended core condition of the NIST 2010 Speaker Recognition Evaluation are competitive or better, in terms of normalized Decision Cost Function and Equal Error Rate, compared to the more expensive generative models.

***Index Terms*—** Discriminative Training,Two–covariance Kernel, Support Vector Machines, i–vectors

1. INTRODUCTION

Recent trends in speaker recognition have seen the development of Bayesian generative models. This has been made possible by advances in the representation of speech segments by means of low dimensional feature vectors referred to as i–vectors [1]. These techniques aim at modeling the i–vectors by decomposing them into a speaker and a channel component whose underlying distributions are then estimated using expectation–maximization. The most effective current flavours of these approaches are the Gaussian (G–PLDA) or Heavy–Tailed Probabilistic Linear Discriminant Analysis (HT–PLDA) [2] and the Two–covariance model, a linear–Gaussian generative model introduced in [3]. The advantage of a Bayesian approach in speaker recognition is that, in principle, it produces likelihood ratios that do not need to be normalized [4]. In [2] this has been confirmed in the case of telephone speech, for heavy–tailed distributions, whereas normalization was needed for Gaussian distributions. A complete symmetry of the train and test segments is another interesting characteristic of these approaches.

In this work, we illustrate a fast discriminative training procedure for a linear–Gaussian model. In this new approach, we do not model speaker classes anymore, but we build a binary classifier which simply classifies a pair of utterances as either target (same speaker) or non–target (different speakers) [5]. Training is performed by means of Support Vector Machines (SVMs), using a suitable kernel derived from the two–covariance generative model. The advantage of this approach is that while training is more expensive compared with the Gaussian PLDA approach, testing is extremely fast (as in G–PLDA) and results are comparable with those provided by HT–PLDA.

The paper is organized as follows: Section 2 describes the SVM classifier, focusing on the properties that the training algorithm should have in order to make our task feasible. Section 3 briefly summarizes the Two–covariance and the PLDA generative models. The steps necessary to derive a discriminative solution for the former model by means of an appropriate expansion of i–vector pairs are given in Section 4 together with the procedure to efficiently train the SVM. The experimental results comparing the performance of the discriminative and generative models are given in Section 5 and conclusions are drawn in Section 6.

2. SVM

A Support Vector Machine is a two–class classifier which looks for the hyperplane that best discriminates two given classes of patterns according to a maximum separation margin criterion.

The separation hyperplane is obtained by solving an unconstrained *regularized risk minimization* problem

$$\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|^2 + C \cdot \sum_{i=1}^{n} \max(0, 1 - \zeta_i \mathbf{w}^T \mathbf{x}_i) \qquad (1)$$

where vector $\mathbf{w}$ is the vector representing the hyperplane and the second term is the (L1–)loss function

$$l_{L1}(\mathbf{w}, \mathbf{x}, \zeta) = \max(0, 1 - \zeta \mathbf{w}^T \mathbf{x}) \qquad (2)$$

evaluated on training patterns $\mathbf{x}_i \in \mathbb{R}^d$ with associated class label $\zeta_i \in \{-1, +1\}$.

Non–linear classification can be obtained by expanding the feature patterns into a high dimensionality space where linear classification is carried out. The kernel trick allows solving the SVM problem without explicitly constructing the expanded features, provided that the dot–products in the expanded space can be evaluated.

Many algorithms exist that solve problem 1, providing both primal and dual solutions [6]. The scale of the classification problem we address in this paper does not allow for an explicit construction of the kernel matrix, which would have a size $O(n^4)$, n being the number of i–vectors in the training set. However, we will show that, by using an appropriate feature expansion, the loss function and its gradient with respect to the hyperplane parameters in the expanded space of the i–vector pairs can be evaluated without explicitly expanding the i–vector pairs. Deriving a formulation for the dot–product in the expanded space we can efficiently train our models by using a primal SVM solver such as the one proposed in [7].

3. GENERATIVE MODELS

I–vectors are a recently proposed compact representation of speaker segments which boosted the study of Bayesian generative models [1, 3]. The procedure for extracting i–vectors has been described and effectively used in [8, 9].

3.1. Two–covariance model

We need to briefly recall in this subsection the two–covariance modeling of [3] because we derive our expression for the SVM dot–product in the expanded space from its formulation. The i–vectors are assumed to be features produced by a linear–Gaussian generative model $\mathcal{M}$. In particular, an i–vector ϕ can be decomposed into a speaker y and a Gaussian distributed channel component z:

$$\phi = y + z \tag{3}$$
$$P(\phi|y,\mathcal{M}) = \mathcal{N}(\phi|y, W^{-1}) \tag{4}$$

where W^{-1} is the within–speaker covariance matrix.
If we assume the prior for y is Gaussian distributed

$$P(y|\mathcal{M}) = \mathcal{N}(y|\mu, B^{-1}) \tag{5}$$

with (between–speaker) covariance matrix B^{-1} then the posterior, given a set S of n i–vectors associated to speaker identity y, is also normal:

$$P(y|S,\mathcal{M}) = \mathcal{N}(y|L^{-1}\gamma, L^{-1}) \tag{6}$$

$$L = B + nW \qquad \gamma = B\mu + W\textstyle\sum_{\phi\in S}\phi \tag{7}$$

Since our problem is to decide whether two spoken segments belong to the same or to different speakers, we have three sets, S_1, S_2 if the two i–vectors are in different sets; otherwise both belong to set $S_{1,2} = S_1 \cup S_2$.

The resulting formulation for the speaker detection log–likelihood was given in [3]

$$\begin{aligned}\log l = {} & \frac{1}{2}(\log|B| - \mu^T B\mu + \log|\tilde{\Lambda}| + \gamma_{1,2}^T\tilde{\Lambda}\gamma_{1,2}) \\ & - \frac{1}{2}(2\log|B| - 2\mu^T B\mu + 2\log|\tilde{\Gamma}| \\ & + {\gamma_1}^T\tilde{\Gamma}\gamma_1 + {\gamma_2}^T\tilde{\Gamma}\gamma_2)\end{aligned} \tag{8}$$

where

$$\begin{aligned}\tilde{\Lambda} &= (B+2W)^{-1} & \tilde{\Gamma} &= (B+W)^{-1} \\ \gamma_{1,2} &= B\mu + W(\phi_1+\phi_2) & \gamma_i &= B\mu + W\phi_i\end{aligned}$$

3.2. PLDA model

The two covariance model can be seen as a particular case of the more general framework of Probabilistic Linear Discriminant Analysis [4, 2], where an i–vector is represented as

$$\phi = U_1 y + U_2 x + z \tag{9}$$

where x represents “channel factors” and z is the residual error. The matrices U_1 and U_2 constrain the speaker and channel spaces to be of lower dimension than the i–vector space.

4. DISCRIMINATIVE MODEL

We are not interested in exactly evaluating (8) to perform discriminative training — we derive instead a formally equivalent expression that can be transformed into a valid dot–product.
By dropping the $\frac{1}{2}$ factor and collecting in a constant $\tilde{k}$ all the i–vector independent terms in the sum, (8) can be rewritten as:

$$\log l = \tilde{k} + \gamma_{1,2}^T\tilde{\Lambda}\gamma_{1,2} - {\gamma_1}^T\tilde{\Gamma}\gamma_1 - {\gamma_2}^T\tilde{\Gamma}\gamma_2 \tag{10}$$

Replacing (7) in (10) we obtain:

$$\begin{aligned}\log l &= (B\mu + W(\phi_1+\phi_2))^T\tilde{\Lambda}(B\mu + W(\phi_1+\phi_2)) \\ &- (B\mu + W\phi_1)^T\tilde{\Gamma}(B\mu + W\phi_1) \\ &- (B\mu + W\phi_2)^T\tilde{\Gamma}(B\mu + W\phi_2) + \tilde{k}\end{aligned} \tag{11}$$

which we rewrite as:

$$\begin{aligned}\log l &= \phi_1^T\Lambda\phi_2 + \phi_2^T\Lambda\phi_1 + \phi_1^T\Gamma\phi_1 + \phi_2^T\Gamma\phi_2 \\ &+ (\phi_1+\phi_2)^T c + k\end{aligned} \tag{12}$$

with

$$\begin{aligned}\Lambda &= W^T\tilde{\Lambda}W & \Gamma &= W^T(\tilde{\Lambda}-\tilde{\Gamma})W \\ c &= 2W^T(\tilde{\Lambda}-\tilde{\Gamma})B\mu & k &= \tilde{k} + (B\mu)^T(\tilde{\Lambda}-2\tilde{\Gamma})B\mu\end{aligned}$$

To demonstrate that (12) is a dot–product in some i–vector pairs expanded space, we recall that the computation of a bilinear form $x^T Ay$ can be expressed in terms of the Frobenius inner product as $x^T A y = \langle A, xy^T \rangle = vec(A)^T vec(xy^T)$, where the operator $vec(A)$ is the operator that stacks the columns of A into a column vector. Hence, the expression for the speaker detection log–likelihood can be rewritten as

$$\log l = \langle \Lambda, \phi_1 \phi_2^T + \phi_2 \phi_1^T \rangle + \langle \Gamma, \phi_1 \phi_1^T + \phi_2 \phi_2^T \rangle + c^T(\phi_1 + \phi_2) + k \quad (13)$$

Thus, if we stack the parameters as:

$$w = \begin{bmatrix} vec(\Lambda) \\ vec(\Gamma) \\ c \\ k \end{bmatrix} = \begin{bmatrix} w_\Lambda \\ w_\Gamma \\ w_c \\ w_k \end{bmatrix} \quad (14)$$

and we expand the i–vector pairs as

$$\varphi(\phi_1, \phi_2) = \begin{bmatrix} vec(\phi_1 \phi_2^T + \phi_2 \phi_1^T) \\ vec(\phi_1 \phi_1^T + \phi_2 \phi_2^T) \\ \phi_1 + \phi_2 \\ 1 \end{bmatrix} = \begin{bmatrix} \varphi_\Lambda(\phi_1, \phi_2) \\ \varphi_\Gamma(\phi_1, \phi_2) \\ \varphi_c(\phi_1, \phi_2) \\ \varphi_k(\phi_1, \phi_2) \end{bmatrix} \quad (15)$$

the scoring given by (12) can be expressed as a dot–product as

$$\begin{aligned} S(\phi_1, \phi_2) &= \log l \\ &= S_\Lambda(\phi_1, \phi_2) + S_\Gamma(\phi_1, \phi_2) \\ &\quad + S_c(\phi_1, \phi_2) + S_k(\phi_1, \phi_2) \\ &= w_\Lambda^T \varphi_\Lambda(\phi_1, \phi_2) + w_\Gamma^T \varphi_\Gamma(\phi_1, \phi_2) \\ &\quad + w_c^T \varphi_c(\phi_1, \phi_2) + w_k^T \varphi_k(\phi_1, \phi_2) \\ &= w^T \varphi(\phi_1, \phi_2) \end{aligned} \quad (16)$$

The terms $S_\Lambda, S_\Gamma, S_c, S_k$ represent the contributions of the different terms of w to the final score.

4.1. Fast scoring

Since the number of i–vector pairs is of the order of hundreds of millions in our experiments, the evaluation of a Gram matrix would be clearly unfeasible. However, if we use a primal SVM solver, we need only to evaluate the SVM loss function and its gradient with respect to the hyperplane. Both evaluations require, in principle, a sum over all the i–vector pairs, but in the next two subsections we show that given the dot–product in (16) the loss function and the gradient evaluations can be done without an explicit expansion of all the i–vector pairs.

4.2. Loss function evaluation

Let us denote by D the matrix of all stacked i–vectors ϕ_i

$$D = [\phi_1 \phi_2 \dots \phi_n]$$

Let $\Theta \in \{\Lambda, \Gamma, c, k\}$ be a component of the hyperplane, and let S_Θ, the score matrix of training patterns due to component Θ, be defined as: $S_{\Theta i,j} = S_\Theta(\phi_i, \phi_j)$. From (16) and (12) the score matrices can be evaluated as:

$$S_\Lambda(\phi_1, \phi_2) = \phi_1^T \Lambda \phi_2 + \phi_2^T \Lambda \phi_1 \Rightarrow S_\Lambda = 2 D^T \Lambda D \quad (17)$$

$$S_\Gamma(\phi_1, \phi_2) = \phi_1^T \Gamma \phi_1 + \phi_2^T \Gamma \phi_2 \Rightarrow S_\Gamma = \tilde{S_\Gamma} + \tilde{S_\Gamma}^T \quad (18)$$

$$S_c(\phi_1, \phi_2) = c^T(\phi_1 + \phi_2) \Rightarrow S_c = \tilde{S_c} + \tilde{S_c}^T \quad (19)$$

$$S_k(\phi_1, \phi_2) = k \Rightarrow S_k = k \cdot \mathbf{1} \quad (20)$$

where

$$\tilde{S_\Gamma} = [\underbrace{d_\Gamma \dots d_\Gamma}_{n}], \qquad d_\Gamma = \mathbf{diag}\,(D^T \Gamma D),$$

$$\tilde{S_c} = [\underbrace{d_c \dots d_c}_{n}], \qquad d_c = D^T c$$

diag is the operator that returns the diagonal of a matrix as a column vector, $\mathbf{1}$ is an $n \times n$ matrix of ones.

Denoting by S the sum of these partial score matrices, the SVM loss function can be summarized as:

$$\begin{aligned} L(D, Z) &= C \sum_{i,j} \max(0, 1 - \zeta_{i,j} w^T \varphi(\phi_i, \phi_j)) \\ &= C \langle \mathbf{1}, \max(\mathbf{0}, \mathbf{1} - (Z \circ S)) \rangle \end{aligned} \quad (21)$$

where $\mathbf{0}$ is an $n \times n$ matrix of all zeros, Z is the $n \times n$ matrix of labels for trials (ϕ_i, ϕ_j), $Z_{i,j} = \zeta_{i,j} \in \{-1, +1\}$, and $\circ$ is the element–wise matrix multiplication operator.

4.3. Gradient Evaluation

The gradient of the loss function can be evaluated from its derivative with respect to the m–th dimension of w as

$$\begin{aligned} \frac{\partial L}{\partial w_m} &= \sum_{i,j} \frac{\partial l_{L1}(w, (\phi_i, \phi_j), \zeta_{i,j})}{\partial (w^T \varphi(\phi_j, \phi_j))} \frac{\partial w^T \varphi(\phi_j, \phi_j)}{\partial w_m} \\ &= \sum_{i,j} g_{i,j} \frac{\partial S_{i,j}}{\partial w_m} = \sum_{i,j} g_{i,j} \varphi(\phi_i, \phi_j)_m \end{aligned} \quad (22)$$

where $g_{i,j}$ is the derivative of the loss function with respect to the dot product

$$g_{i,j} = \begin{cases} 0 & \text{if } S_{i,j} \zeta_{i,j} \geq 1 \\ -\zeta_{i,j} & \text{otherwise} \end{cases}$$

Let G be the matrix $G_{i,j} = g_{i,j}$, then

$$\begin{aligned} \nabla L = \begin{bmatrix} \nabla_\Lambda L \\ \nabla_\Gamma L \\ \nabla_c L \\ \nabla_k L \end{bmatrix} &= \begin{bmatrix} vec\left(\sum_{i,j} g_{i,j} \left(\phi_i \phi_j^T + \phi_j \phi_i^T \right) \right) \\ vec\left(\sum_{i,j} g_{i,j} \left(\phi_i \phi_i^T + \phi_j \phi_j^T \right) \right) \\ \sum_{i,j} g_{i,j} \left(\phi_i + \phi_j \right) \\ \sum_{i,j} g_{i,j} \end{bmatrix} \\ &= \begin{bmatrix} 2 \cdot vec\left(D G D^T \right) \\ 2 \cdot vec\left(\left[D \circ (\mathbf{1}_{\boldsymbol{A}} G) \right] D^T \right) \\ 2 \left[D \circ (\mathbf{1}_{\boldsymbol{A}} G) \right] \mathbf{1}_{\boldsymbol{B}} \\ \mathbf{1}_{\boldsymbol{B}}^T G \mathbf{1}_{\boldsymbol{B}} \end{bmatrix} \end{aligned} \quad (23)$$

where $\mathbf{1}_{\boldsymbol{A}}$ is a $d \times n$ matrix of ones and $\mathbf{1}_{\boldsymbol{B}}$ is a size n column vector of ones. Again, no explicit expansion of i–vectors is necessary for this evaluation.

Table 1. EER, DCF as defined in SRE 2008 (oldDCF), minimum DCF (minDCF) and actual DCF (actDCF) as defined in SRE 2010 for the extended tel–tel core condition (condition 5) of NIST SRE10

Male Set

System	EER	oldDCF	minDCF	actDCF
G–PLDA	3.82%	0.165	0.401	0.442
G–PLDA+AT–norm	2.11%	0.106	0.309	0.374
HT–PLDA	1.55%	0.082	0.313	0.364
2C–SVM	1.50%	0.074	0.308	0.355

Female Set

System	EER	oldDCF	minDCF	actDCF
G–PLDA	4.08%	0.179	0.448	0.531
G–PLDA+AT–norm	2.54%	0.122	0.438	0.454
HT–PLDA	2.29%	0.118	0.412	0.415
2C–SVM	2.35%	0.108	0.394	0.398

All

System	EER	oldDCF	minDCF	actDCF
G–PLDA	4.21%	0.183	0.470	0.498
G–PLDA+AT–norm	2.39%	0.118	0.420	0.422
HT–PLDA	1.98%	0.102	0.379	0.393
2C–SVM	1.94%	0.095	0.373	0.378

5. EXPERIMENTS

Three systems were trained on SRE pre–2010 data and tested on the extended tel–tel core condition (condition 5) of SRE10 [10]: a Gaussian PLDA (G–PLDA), a Heavy–Tailed PLDA (HT–PLDA) and the discriminative Two–covariance SVM system (2C–SVM). The 2C–SVM is compared with G–PLDA because PLDA is a more general framework than the two–covariance model, from which the discriminative approach has been derived, and both rely on Gaussian distribution of i–vectors and noise. Moreover, we compare 2C–SVM with HT–PLDA, which assumes heavy–tailed distributions for the priors, and has shown impressive performance improvement with respect to G–PLDA [2].

Even if the expression given in (12) can be directly used to train an SVM, the lack of normalization of the i–vector dimensions results in poor classification performance, due to the presence of the SVM regularizer term. Thus, the SVM is trained by centering the i–vectors and scaling the i–vector space to whiten the within–speaker covariance matrix. Class balancing is then performed to optimize for an operating point near the 2008 SRE DCF one by artificially lowering the contribution to the loss function of miss–classified targets.

The results are given in terms of EER and normalized minimum and actual Decision Cost Functions as defined by NIST for SRE08 and SRE10 [10]. The scores were calibrated on the SRE08 data [11]. Both PLDA systems were trained with 200 speaker factors and 400 channel factors. 400–dimensional i–vectors were extracted via a 60–dimensional features, full–covariance, 2048 Gaussians UBM [9].

Table 1 summarizes the results obtained for the female and male speakers separately, and pooled together.

As pointed out in [2], G–PLDA requires score normalization, which has been performed in our experiments by means of Adaptive T–norm [12], whereas no normalization is required for heavy–tailed PLDA and for the 2C–SVM systems.

Discriminative training not only performs better than generative modeling under the assumption of Gaussian distributed i–vectors, but its performance is even slightly better than that of the Heavy–Tailed PLDA.

As far as training complexity is concerned, less than 3 hours was needed to train the female system (21663 utterances — more than 450 million trials) and even less for the male system (16969 utterances, approximately 290 millions of trials) on a HP DS160G5 server equipped with two Xeon X5472 3 GHz quad–core processors and 32 GB of DDR2–800 RAM. Testing all test segments against all the other test segments is done in less than 2 seconds.

6. CONCLUSIONS

A fast discriminative training approach for speaker verification based on i–vectors has been presented. On NIST telephone evaluation data, the resulting models perform better, without the need for normalization techniques, than the generative ones, even compared with heavy–tailed models.

7. ACKNOWLEDGMENTS

We would like to thank the organizers and the participants to the BOSARIS Workshop, in particular Ondřej Glembek, Pavel Matějka and Oldřich Plchot from Brno University of Technology, for providing us with the BUT i–vectors, Fabio Castaldo from Loquendo, for giving us the PLDA results we used as baseline, and Edward de Villiers from Agnitio, for his support with calibration tools.

8. REFERENCES

[1] N. Dehak, P. Kenny, et al., “Front–end factor analysis for speaker verification,” in *IEEE Trans. on Audio, Speech and Lang. Process.*, 2010.

[2] P. Kenny, “Bayesian speaker verification with Heavy–Tailed Priors,” in *keynote presentation, Odyssey 2010*, 2010.

[3] N. Brümmer and E. de Villiers, “The speaker partitioning problem,” in *Proc. of Odyssey 2010*, 2010, pp. 194–201.

[4] 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.

[5] L. Burget et al., “Robust speaker recognition over varying channels,” in *Johns Hopkins University CLSP Summer Workshop Report*, 2008, www.clsp.jhu.edu/workshops/ws08/documents/jhu_report_main.pdf.

[6] S. Cumani, F. Castaldo, P. Laface, D. Colibro, and C. Vair, “Comparison of large-scale SVM training algorithms for language recognition,” in *Proc. of Odyssey 2010*, 2010, pp. 222–229.

[7] C.H. Teo, A. Smola, et al., “A scalable modular convex solver for regularized risk minimization,” in *Proc. of KDD*, 2007, pp. 727–736.

[8] N. Dehak et al., “Support Vector Machines versus fast scoring in the low-dimensional total variability space for speaker verification,” in *Proc. of Interspeech 2009*, 2009, pp. 1559–1562.

[9] N. Brummer, L. Burget, P. Kenny, et al., “ABC system description for NIST SRE 2010,” in *Proc. NIST 2010 Speaker Recognition Evaluation*, 2010.

[10] NIST, “The NIST Year 2008 and 2010 Speaker Recognition Evaluation plans,” http://www.itl.nist.gov/iad/mig/tests/sre.

[11] N. Brümmer and J. A. du Preez, “Application-independent evaluation of speaker detection,” *Computer Speech & Language*, vol. 20, no. 2-3, pp. 230–275, 2006.

[12] D. Sturim and D. A. Reynolds, “Speaker adaptive cohort selection for tnorm in text–independent speaker recognition,” in *Proc. of ICASSP 2005*, 2005.