AN EFFICIENT IMPLEMENTATION OF PROBABILISTIC LINEAR DISCRIMINANT ANALYSIS

Lukáš Machlica, Zbyněk Zajíc

Department of Cybernetics, Faculty of Applied Sciences, University of West Bohemia,
Pilsen, Czech Republic

ABSTRACT

Probabilistic Linear Discriminant Analysis (PLDA), used particularly in image and speech processing for face and speaker recognition, respectively, is a generative model requesting lots of data to be trained. In the paper several enhancements concerning the implementation of the estimation algorithm of PLDA are proposed providing substantial computational savings. At first, an inverse of a huge matrix is replaced by an inversion of two significantly smaller matrices. Subsequently, it is shown how to avoid the need to process the whole data set in each iteration of the estimation algorithm. Supplementary results are presented on NIST SRE 2008.

Index Terms— PLDA, implementation, latent variables, generative model

1. INTRODUCTION

Probabilistic Linear Discriminant Analysis (PLDA) was proposed by Prince and Elder [1] for the task of face recognition. It is a statistical model used to decompose the feature space to a person/individual's specific invariable part and a channel part describing the variation in distinct realizations of an individual. PLDA can be also seen as a special case of Joint Factor Analysis (JFA) [2], which is working with Gaussian Mixture Model (GMM) based supervectors, and was developed independently of PLDA by Patrick Kenny. JFA and closely related concept of i-vectors [3] form today the core of most of the state-of-the-art Speaker Recognition (SR) systems [4, 5]. PLDA is often used in the verification phase of a SR system either directly with supervectors [6, 7] or it is utilized to build a generative model in the i-vectors space [8, 9] decomposing the space to two subspaces responsible for speaker and channel variabilities, respectively.

Since the estimation of PLDA parameters requests several iterations until the convergence is reached, and a large development data set is needed to train a reliable PLDA model, implementation enhancements are of significant importance to ensure robustness, speed and proper data/memory management. In the beginning of the following section the concept and existing implementation of the PLDA estimation algorithm are described. In Section 2.2 and Section 2.3 novel implementation enhancements are proposed. At the end, some supplementary experiments are provided in Section 3.

This research was supported by the Grant Agency of the Czech Republic, project No. GAČR P103/12/G084.

2. PROBABILISTIC LDA

PLDA [1] is a generative statistical model of the form

$$\boldsymbol{x}_{ij} = \boldsymbol{\mu} + \boldsymbol{F}\boldsymbol{h}_i + \boldsymbol{G}\boldsymbol{w}_{ij} + \boldsymbol{\epsilon}_{ij} \tag{1}$$

where $\boldsymbol{X} = \{\boldsymbol{x}_{i1}, \ldots, \boldsymbol{x}_{iJ_i}\}_{i=1}^{I}$ is the set of I individuals represented as D_x dimensional observable vectors $\boldsymbol{x}_{ij}$, J_i is the count of distinct representations/observations of each individual, $N = \sum_{i=1}^{I} J_i$ is the number of vectors in $\boldsymbol{X}$, and $\boldsymbol{\mu} = \mathrm{E}[\boldsymbol{x}_{ij}]$ is the mean value of vectors in $\boldsymbol{X}$. Let denote $\boldsymbol{\Lambda}_i = \{\boldsymbol{x}_{ij}\}_{j=1}^{J_i}$ the set of distinct representations of one individual. Columns of the matrix $\boldsymbol{F}$ span the between individual subspace, $\boldsymbol{h}_i$ is a D_h dimensional latent vector of coordinates in this space and represents the mutual information shared between vectors in $\boldsymbol{\Lambda}_i$. Columns of the matrix $\boldsymbol{G}$ span the within individual subspace of the space formed by vectors in $\boldsymbol{X}$, and $\boldsymbol{w}_{ij}$ is a D_w dimensional vector of coordinates in this space. It is assumed that both $\boldsymbol{h}_i$ and $\boldsymbol{w}_{ij}$ follow standard normal distribution $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. The term $\boldsymbol{\epsilon}_{ij}$ represents the residual noise factor having normal distribution $\mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma})$ with diagonal covariance matrix $\boldsymbol{\Sigma}$. Thus, one can identify the identity component $\boldsymbol{\mu}+\boldsymbol{F}\boldsymbol{h}_i$ and the noise/channel component $\boldsymbol{G}\boldsymbol{w}_{ij}+\boldsymbol{\epsilon}_{ij}$ of each vector $\boldsymbol{x}_{ij}$. Note that the distribution of $\boldsymbol{x}_{ij}$ is normal $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{F}\boldsymbol{F}^{\mathrm{T}} + \boldsymbol{G}\boldsymbol{G}^{\mathrm{T}} + \boldsymbol{\Sigma})$.

2.1. Training

In the training phase the parameters $\theta = \{\boldsymbol{\mu}, \boldsymbol{F}, \boldsymbol{G}, \boldsymbol{\Sigma}\}$ have to be trained. Mean $\boldsymbol{\mu}$ is estimated as the mean of all vectors $\boldsymbol{x}_{ij}$ from the development set $\boldsymbol{X}$, and to facilitate subsequent formulas let subtract $\boldsymbol{\mu}$ from all $\boldsymbol{x}_{ij}$ beforehand. In [1] a system of equations is formed

$$\begin{bmatrix} \boldsymbol{x}_{i1} \\ \vdots \\ \boldsymbol{x}_{iJ_i} \end{bmatrix} = \begin{bmatrix} \boldsymbol{F} & \boldsymbol{G} & \ldots & \boldsymbol{0} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{F} & \boldsymbol{0} & \ldots & \boldsymbol{G} \end{bmatrix} \begin{bmatrix} \boldsymbol{h}_i \\ \boldsymbol{w}_{i1} \\ \vdots \\ \boldsymbol{w}_{iJ_i} \end{bmatrix} + \begin{bmatrix} \boldsymbol{\epsilon}_1 \\ \boldsymbol{\epsilon}_2 \\ \vdots \\ \boldsymbol{\epsilon}_{J_i} \end{bmatrix},$$

$$\hat{\boldsymbol{\Sigma}}_i = \begin{bmatrix} \boldsymbol{\Sigma} & \dots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \dots & \boldsymbol{\Sigma} \end{bmatrix}, \tag{2}$$

what can be written in a compact form as

$$\hat{\boldsymbol{x}}_i = \boldsymbol{A}_i \hat{\boldsymbol{y}}_i + \hat{\boldsymbol{\epsilon}}, \tag{3}$$

where the distribution of $\hat{\boldsymbol{\epsilon}}$ follows $\mathcal{N}(\mathbf{0}, \hat{\boldsymbol{\Sigma}}_i)$. Matrices $\boldsymbol{A}_i, \hat{\boldsymbol{\Sigma}}_i$ depend on i through the number of their row- and column-blocks, which are given by the number of vectors in $\boldsymbol{\Lambda}_i$. Note that the joint probability of vectors in $\boldsymbol{\Lambda}_i$ given θ equals to

$$p(\boldsymbol{\Lambda}_i|\theta) = \mathcal{N}(\hat{\boldsymbol{x}}_i|\mathbf{0}, \boldsymbol{A}_i\boldsymbol{A}_i^{\mathrm{T}} + \hat{\boldsymbol{\Sigma}}_i). \tag{4}$$

This formulation is equivalent to the formulation of Factor Analysis (FA) [10] and can be solved by the same estimation procedure based on maximization of (4). At first $\boldsymbol{h}_i, \boldsymbol{w}_{i1}, \dots, \boldsymbol{w}_{iJ_i}$ are extracted (in fact only their MAP estimates are obtained) utilizing matrices $\boldsymbol{A}_i$, $\hat{\boldsymbol{\Sigma}}_i$, hence

$$\hat{\boldsymbol{y}}_i = \left(\boldsymbol{A}_i^{\mathrm{T}}\hat{\boldsymbol{\Sigma}}_i^{-1}\boldsymbol{A}_i + \boldsymbol{I}\right)^{-1} \boldsymbol{A}_i^{\mathrm{T}}\hat{\boldsymbol{\Sigma}}_i^{-1}\hat{\boldsymbol{x}}_i, \tag{5}$$

and then $\hat{\boldsymbol{y}}_i$ is decomposed to $\boldsymbol{z}_{ij} = [\boldsymbol{h}_i^{\mathrm{T}}, \boldsymbol{w}_{ij}^{\mathrm{T}}]^{\mathrm{T}}, j = 1, \dots, J_i$. Finally, couples $(\boldsymbol{x}_{ij}, \boldsymbol{z}_{ij})$ are used to train $\boldsymbol{B} = [\boldsymbol{F}, \boldsymbol{G}]$ and $\boldsymbol{\Sigma}$. Update formulas are

$$\boldsymbol{Z} = \left(\boldsymbol{B}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{B} + \boldsymbol{I}\right)^{-1}, \tag{6}$$

$$\boldsymbol{B}^* = \left(\sum_{i,j} \boldsymbol{x}_{ij}\boldsymbol{z}_{ij}^{\mathrm{T}}\right)\left(\sum_{i,j} \boldsymbol{Z} + \boldsymbol{z}_{ij}\boldsymbol{z}_{ij}^{\mathrm{T}}\right)^{-1}, \tag{7}$$

$$\boldsymbol{\Sigma} = \frac{1}{N}\sum_{ij} \mathrm{diag}\left(\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^{\mathrm{T}} - \boldsymbol{B}^*\boldsymbol{z}_{ij}\boldsymbol{x}_{ij}^{\mathrm{T}}\right), \tag{8}$$

where $\boldsymbol{B}^*$ is the new estimate of $\boldsymbol{B}$, and the function diag() zeros the non-diagonal elements. The estimation is iterative, steps (5)-(8) have to be repeated until the convergence of (4) is reached.

2.2. Training revisited I

The problem associated with the training procedure described in the previous section is that the matrix $\boldsymbol{A}_i^{\mathrm{T}}\hat{\boldsymbol{\Sigma}}_i^{-1}\boldsymbol{A}_i + \boldsymbol{I}$ has to be reassembled and inverted whenever the number of vectors in $\boldsymbol{\Lambda}_i$ changes in order to evaluate (5). If vectors $\boldsymbol{x}_{ij}$ are of significantly high dimension and/or the number J_i of representations of an individual is high, the inversion of the matrix $\boldsymbol{A}_i^{\mathrm{T}}\boldsymbol{\Sigma}_i^{-1}\boldsymbol{A}_i + \boldsymbol{I}$ can become intractable and/or ill-conditioned. Even if a reliable inversion can be computed, the memory management will become expensive. Now we will show how to invert $\boldsymbol{A}_i^{\mathrm{T}}\boldsymbol{\Sigma}_i^{-1}\boldsymbol{A}_i + \boldsymbol{I}$ and adjust (5) leading to a much faster and easier implementation. We have to find a decomposition

$$\left(\boldsymbol{A}_i^{\mathrm{T}}\boldsymbol{\Sigma}_i^{-1}\boldsymbol{A}_i + \boldsymbol{I}\right)^{-1} = \begin{bmatrix} \boldsymbol{\Omega}_1 & \boldsymbol{\Omega}_3 \\ \boldsymbol{\Omega}_3^{\mathrm{T}} & \boldsymbol{\Omega}_2 \end{bmatrix}^{-1} = \begin{bmatrix} \tilde{\boldsymbol{\Omega}}_1 & \tilde{\boldsymbol{\Omega}}_3 \\ \tilde{\boldsymbol{\Omega}}_3^{\mathrm{T}} & \tilde{\boldsymbol{\Omega}}_2 \end{bmatrix}.$$

Choosing

$$\begin{aligned} \boldsymbol{\Omega}_1 &= J_i\boldsymbol{F}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{F} + \boldsymbol{I}, D_h \times D_h, \\ \boldsymbol{\Omega}_2 &= \begin{bmatrix} \boldsymbol{K} & \dots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \dots & \boldsymbol{K} \end{bmatrix}, J_iD_w \times J_iD_w, \\ \boldsymbol{\Omega}_3 &= \begin{bmatrix} \boldsymbol{F}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{G} & \dots & \boldsymbol{F}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{G} \end{bmatrix}, D_h \times J_iD_w, \end{aligned} \tag{9}$$

where $\boldsymbol{K} = \boldsymbol{G}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{G} + \boldsymbol{I}$, and using the formulas for block inverses

$$\begin{aligned} \tilde{\boldsymbol{\Omega}}_1 &= (\boldsymbol{\Omega}_1 - \boldsymbol{\Omega}_3\boldsymbol{\Omega}_2^{-1}\boldsymbol{\Omega}_3^{\mathrm{T}})^{-1}, \\ \tilde{\boldsymbol{\Omega}}_3 &= -\tilde{\boldsymbol{\Omega}}_1\boldsymbol{\Omega}_3\boldsymbol{\Omega}_2^{-1}, \\ \tilde{\boldsymbol{\Omega}}_2 &= \boldsymbol{\Omega}_2^{-1} + \boldsymbol{\Omega}_2^{-1}\boldsymbol{\Omega}_3^{\mathrm{T}}\tilde{\boldsymbol{\Omega}}_1\boldsymbol{\Omega}_3\boldsymbol{\Omega}_2^{-1}, \end{aligned} \tag{10}$$

we can note that the sizes of corresponding blocks (e.g. sizes of $\tilde{\boldsymbol{\Omega}}_1$ and $\boldsymbol{\Omega}_1$) remain the same. Since blocks in $\boldsymbol{\Omega}_3$ and $\boldsymbol{\Omega}_2$ repeat we can state that $\tilde{\boldsymbol{\Omega}}_3$ will contain J_i identical block-matrices $\tilde{\boldsymbol{\Omega}}_{3\mathrm{S}}$ each of size $D_h \times D_w$, and since $\boldsymbol{\Omega}_3^{\mathrm{T}}\tilde{\boldsymbol{\Omega}}_1\boldsymbol{\Omega}_3$ is full and contains identical $D_w \times D_w$ blocks and $\boldsymbol{\Omega}_2$ is block-diagonal, $\tilde{\boldsymbol{\Omega}}_2$ will contain two kind of block-matrices: $\tilde{\boldsymbol{\Omega}}_{2\mathrm{D}}$ will be repeated on the diagonal, and $\tilde{\boldsymbol{\Omega}}_{2\mathrm{S}}$ will fill the non-diagonal blocks (for more details see [11]). For example, for $J_i = 3$ we get

$$\begin{aligned} \hat{\boldsymbol{y}}_i &= (\boldsymbol{A}_i^{\mathrm{T}}\hat{\boldsymbol{\Sigma}}_i^{-1}\boldsymbol{A}_i^{\mathrm{T}} + \boldsymbol{I})^{-1}\boldsymbol{A}_i^{\mathrm{T}}\hat{\boldsymbol{\Sigma}}_i^{-1}\hat{\boldsymbol{x}}_i = \\ &= \begin{bmatrix} \tilde{\boldsymbol{\Omega}}_1 & \tilde{\boldsymbol{\Omega}}_{3\mathrm{S}} & \tilde{\boldsymbol{\Omega}}_{3\mathrm{S}} & \tilde{\boldsymbol{\Omega}}_{3\mathrm{S}} \\ \tilde{\boldsymbol{\Omega}}_{3\mathrm{S}}^{\mathrm{T}} & \tilde{\boldsymbol{\Omega}}_{2\mathrm{D}} & \tilde{\boldsymbol{\Omega}}_{2\mathrm{S}} & \tilde{\boldsymbol{\Omega}}_{2\mathrm{S}} \\ \tilde{\boldsymbol{\Omega}}_{3\mathrm{S}}^{\mathrm{T}} & \tilde{\boldsymbol{\Omega}}_{2\mathrm{S}} & \tilde{\boldsymbol{\Omega}}_{2\mathrm{D}} & \tilde{\boldsymbol{\Omega}}_{2\mathrm{S}} \\ \tilde{\boldsymbol{\Omega}}_{3\mathrm{S}}^{\mathrm{T}} & \tilde{\boldsymbol{\Omega}}_{2\mathrm{S}} & \tilde{\boldsymbol{\Omega}}_{2\mathrm{S}} & \tilde{\boldsymbol{\Omega}}_{2\mathrm{D}} \end{bmatrix} \begin{bmatrix} \boldsymbol{F}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\sum_{j=1}^{3}\boldsymbol{x}_{ij} \\ \boldsymbol{G}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{x}_{i1} \\ \boldsymbol{G}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{x}_{i2} \\ \boldsymbol{G}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{x}_{i3} \end{bmatrix}. \end{aligned} \tag{11}$$

Assigning $\boldsymbol{\Sigma}_{\mathrm{N}} = \boldsymbol{\Sigma} + \boldsymbol{G}\boldsymbol{G}^{\mathrm{T}}$ and $\boldsymbol{\mu}_i = 1/J_i\sum_{j=1}^{J_i}\boldsymbol{x}_{ij}$, and utilizing the Sherman-Morrison-Woodbury (SMW) matrix inverse identity following formulas can be derived:

$$\begin{aligned} \tilde{\boldsymbol{\Omega}}_1 &= 1/J_i\,(\boldsymbol{F}^{\mathrm{T}}\boldsymbol{\Sigma}_{\mathrm{N}}^{-1}\boldsymbol{F} + 1/J_i\,\boldsymbol{I})^{-1}, \\ \tilde{\boldsymbol{\Omega}}_{2\mathrm{S}} &= \boldsymbol{G}^{\mathrm{T}}\boldsymbol{\Sigma}_{\mathrm{N}}^{-1}\boldsymbol{F}\tilde{\boldsymbol{\Omega}}_1\boldsymbol{F}^{\mathrm{T}}\boldsymbol{\Sigma}_{\mathrm{N}}^{-1}\boldsymbol{G}, \\ \tilde{\boldsymbol{\Omega}}_{2\mathrm{D}} &= \tilde{\boldsymbol{\Omega}}_{2\mathrm{S}} + \boldsymbol{K}^{-1}, \\ \tilde{\boldsymbol{\Omega}}_{3\mathrm{S}} &= -\tilde{\boldsymbol{\Omega}}_1\boldsymbol{F}^{\mathrm{T}}\boldsymbol{\Sigma}_{\mathrm{N}}^{-1}\boldsymbol{G}. \end{aligned} \tag{12}$$

Combining these results with (11) and using again the SMW identities we get formulas for MAP estimates of $\boldsymbol{h}_i$ and $\boldsymbol{w}_{ij}$

$$\begin{aligned} \boldsymbol{h}_i &= \tilde{\boldsymbol{\Omega}}_1\boldsymbol{F}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\sum_{j=1}^{J_i}\boldsymbol{x}_{ij} + \tilde{\boldsymbol{\Omega}}_{3\mathrm{S}}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\sum_{j=1}^{J_i}\boldsymbol{x}_{ij} \\ &= \left(\boldsymbol{F}^{\mathrm{T}}\boldsymbol{\Sigma}_{\mathrm{N}}^{-1}\boldsymbol{F} + 1/J_i\,\boldsymbol{I}\right)^{-1}\boldsymbol{F}^{\mathrm{T}}\boldsymbol{\Sigma}_{\mathrm{N}}^{-1}\boldsymbol{\mu}_i, \end{aligned} \tag{13}$$

$$\begin{aligned} \boldsymbol{w}_{ij} &= \tilde{\boldsymbol{\Omega}}_{3\mathrm{S}}^{\mathrm{T}}\boldsymbol{F}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\sum_{j=1}^{J_i}\boldsymbol{x}_{ij} + \tilde{\boldsymbol{\Omega}}_{2\mathrm{D}}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{x}_{ij} + \\ &+ \tilde{\boldsymbol{\Omega}}_{2\mathrm{S}}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\sum_{k=1,k\neq j}^{J_i}\boldsymbol{x}_{ik} = \\ &= \left(\boldsymbol{G}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{G} + \boldsymbol{I}\right)^{-1}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\left(\boldsymbol{x}_{ij} - \boldsymbol{F}\boldsymbol{h}_i\right). \end{aligned} \tag{14}$$

For detailed derivations of previous formulas see [12].

The latent variable $\boldsymbol{h}_i$, which represents the mutual information, is the projected mean of all the given representations of an individual i assuming full noise covariance $\boldsymbol{\Sigma}_{\mathrm{N}}$. In addition, the more representations are given the lesser the influence of the $\boldsymbol{h}_i$'s prior $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ – the term $1/J_i\boldsymbol{I}$ in (13). The representation dependent latent variable $\boldsymbol{w}_{ij}$ is the projection of the residual $(\boldsymbol{x}_{ij} - \boldsymbol{F}\boldsymbol{h}_i)$ on the space formed by columns of $\boldsymbol{G}$ assuming only the unexplained variance $\boldsymbol{\Sigma}$, however since only one vector is used at a time the prior is fixed. It is also obvious that $\boldsymbol{h}_i$ depends not only on one particular $\boldsymbol{x}_{ij} \in \boldsymbol{\Lambda}_i$, but on the whole set $\boldsymbol{\Lambda}_i$, whereas $\boldsymbol{w}_{ij}$ depends on $\boldsymbol{x}_{ij} \in \boldsymbol{\Lambda}_i$ and even on $\boldsymbol{h}_i$.

The *size of the matrices to be inverted* is now $D_h \times D_h$ and $D_w \times D_w$ instead of one huge matrix $(D_h + J_i D w) \times (D_h + J_i D w)$. In cases where the number of representations in $\boldsymbol{\Lambda}_i$ is high the term $1/J_i\boldsymbol{I}$ in (13) can be left out, or fixed to an arbitrary small number to handle ill-conditioned situations. Otherwise, $\left(\boldsymbol{F}^{\mathrm{T}}\boldsymbol{\Sigma}_{\mathrm{N}}^{-1}\boldsymbol{F} + 1/J_i\boldsymbol{I}\right)^{-1}\boldsymbol{F}^{\mathrm{T}}\boldsymbol{\Sigma}_{\mathrm{N}}^{-1}$ has to be still recomputed for each distinct value of J_i. If $\rho = 1/J_i$ is fixed, one can compute the transformation matrices

$$\boldsymbol{T}_F = \left(\boldsymbol{F}^{\mathrm{T}}\boldsymbol{\Sigma}_{\mathrm{N}}^{-1}\boldsymbol{F} + \rho\boldsymbol{I}\right)^{-1}\boldsymbol{F}^{\mathrm{T}}\boldsymbol{\Sigma}_{\mathrm{N}}^{-1}, \tag{15}$$

$$\boldsymbol{T}_G = \left(\boldsymbol{G}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{G} + \boldsymbol{I}\right)^{-1}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1} \tag{16}$$

(for some small ρ) before each iteration, and iterate through all the data in $\boldsymbol{X}$ without the need to reestimate/invert any of the matrices when the amount of data in any of the sets $\boldsymbol{\Lambda}_i$ changes. Hence, the difference from the training procedure described in Section 2.1 stands in the use of $\boldsymbol{T}_F$ and $\boldsymbol{T}_G$ instead of using (5) when estimating $\boldsymbol{z}_{ij} = [\boldsymbol{h}_i^{\mathrm{T}}, \boldsymbol{w}_{ij}^{\mathrm{T}}]^{\mathrm{T}}$, steps (6)-(8) remain the same.

2.3. Training revisited II

The goal of the previous section was to facilitate evaluations of latent variables $\boldsymbol{h}_i$ and $\boldsymbol{w}_{ij}$, now we will focus on the accumulation process in the PLDA training, more precisely on summation terms (note that $\boldsymbol{z}_{ij} = [\boldsymbol{h}_i^{\mathrm{T}}, \boldsymbol{w}_{ij}^{\mathrm{T}}]^{\mathrm{T}}$)

$$\sum_{i,j} \boldsymbol{x}_{ij}\left[\boldsymbol{h}_i^{\mathrm{T}}, \boldsymbol{w}_{ij}^{\mathrm{T}}\right], \ \sum_{i,j}\begin{bmatrix}\boldsymbol{h}_i \\ \boldsymbol{w}_{ij}\end{bmatrix}\left[\boldsymbol{h}_i^{\mathrm{T}}, \boldsymbol{w}_{ij}^{\mathrm{T}}\right] \tag{17}$$

from (7) and (8). We will show how to make the iteration process in the PLDA training independent of the size of the input feature vector set. Let

$$\boldsymbol{C}_X = 1/N \sum_{i,j} \boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^{\mathrm{T}}, \tag{18}$$

$$\boldsymbol{C}_B = 1/N \sum_i J_i \boldsymbol{\mu}_i \boldsymbol{\mu}_i^{\mathrm{T}} \tag{19}$$

be the data and the between covariance matrices, respectively (assuming the data have been normalized to zero mean beforehand). Let substitute for $\boldsymbol{h}_i = \boldsymbol{T}_F\boldsymbol{\mu}_i$ and for $\boldsymbol{w}_{ij} = \boldsymbol{T}_G(\boldsymbol{x}_{ij} - \boldsymbol{F}\boldsymbol{h}_i)$ in (17) and decompose yielding

$$\boldsymbol{E}_{xh} = \sum_{ij} \boldsymbol{x}_{ij}\boldsymbol{h}_i^{\mathrm{T}} = N\boldsymbol{C}_B\boldsymbol{T}_F^{\mathrm{T}}, \tag{20}$$

$$\boldsymbol{E}_{xw} = \sum_{ij} \boldsymbol{x}_{ij}\boldsymbol{w}_{ij}^{\mathrm{T}} = N\boldsymbol{\Sigma}_{/F}\boldsymbol{T}_G^{\mathrm{T}}, \tag{21}$$

$$\boldsymbol{E}_{hh} = \sum_{ij} \boldsymbol{h}_i\boldsymbol{h}_i^{\mathrm{T}} = \boldsymbol{T}_F\boldsymbol{E}_{xh}, \tag{22}$$

$$\boldsymbol{E}_{wh} = \sum_{ij} \boldsymbol{w}_{ij}\boldsymbol{h}_i^{\mathrm{T}} = \boldsymbol{T}_G(\boldsymbol{E}_{xh} - \boldsymbol{F}\boldsymbol{E}_{hh}), \tag{23}$$

$$\boldsymbol{E}_{ww} = \sum_{ij} \boldsymbol{w}_{ij}\boldsymbol{w}_{ij}^{\mathrm{T}} = \boldsymbol{T}_G(\boldsymbol{E}_{xw} - \boldsymbol{F}\boldsymbol{E}_{hw}), \tag{24}$$

where $\boldsymbol{\Sigma}_{/F} = \boldsymbol{C}_X - \frac{1}{N}(\boldsymbol{F}\boldsymbol{E}_{xh}^{\mathrm{T}})^{\mathrm{T}}$ is the residual covariance not captured by $\boldsymbol{F}\boldsymbol{F}^{\mathrm{T}}$. Now the update formula (7) is the solution of the system of equations

$$\left(N\boldsymbol{Z} + \begin{bmatrix}\boldsymbol{E}_{hh} & \boldsymbol{E}_{wh}^{\mathrm{T}} \\ \boldsymbol{E}_{wh} & \boldsymbol{E}_{ww}\end{bmatrix}\right)\begin{bmatrix}\boldsymbol{F}^{\mathrm{T}} \\ \boldsymbol{G}^{\mathrm{T}}\end{bmatrix} = \begin{bmatrix}\boldsymbol{E}_{xh}^{\mathrm{T}} \\ \boldsymbol{E}_{xw}^{\mathrm{T}}\end{bmatrix}, \tag{25}$$

solved for $\boldsymbol{F}$ and $\boldsymbol{G}$ (a system of equation is solved rather than to compute the inverse of a matrix), and $\boldsymbol{Z}$ was defined in (6). Let $\boldsymbol{F}^*$ and $\boldsymbol{G}^*$ be the solutions of (25). Using the already accumulated $\boldsymbol{E}_{xh}$ and $\boldsymbol{E}_{xw}$ update formula (8) changes to

$$\boldsymbol{\Sigma} = \mathrm{diag}\left(\boldsymbol{C}_X - \frac{1}{N}\boldsymbol{F}^*\boldsymbol{E}_{xh}^{\mathrm{T}} - \frac{1}{N}\boldsymbol{G}^*\boldsymbol{E}_{xw}^{\mathrm{T}}\right). \tag{26}$$

Now the time to train PLDA *does not depend* on the size of the dataset. Once the matrices $\boldsymbol{C}_X$ and $\boldsymbol{C}_B$ have been estimated, the data set is no longer needed. Since the estimation process is iterative (data had to be seen/processed several times) significant computational savings may be acquired for large datasets.

In this section we have assumed an approximation that $\rho = 1/J_i$ is fixed. In the exact case the matrix $\boldsymbol{T}_F$ has to be computed for each distinct J_i. Notice that this will affect only $\boldsymbol{E}_{xh}$ and $\boldsymbol{E}_{hh}$. Several covariances $\boldsymbol{C}_B(J_i) = J_i \sum_{j\in\Phi_i} \boldsymbol{\mu}_j\boldsymbol{\mu}_j^{\mathrm{T}}$ will have to be stored in the memory (Φ_i is the set of indexes of those individuals i who contain the same number of representations J_i), and several $\boldsymbol{T}_F(J_i)$ will have to be computed. However, if the number of distinct J_is is small, the increase in the computational cost is negligible. Otherwise, it is useful to fix J_i to some small value, or e.g. cluster J_is into a few clusters and fix one J_i for each cluster.

2.4. Training summary

The covariance matrices $\boldsymbol{C}_X$ and $\boldsymbol{C}_B$ are estimated, and the number of iterations is set by the user. In each iteration matrices (15) and (16) are computed and matrices (20)-(24) are evaluated. New estimates of $\boldsymbol{F}$, $\boldsymbol{G}$ and $\boldsymbol{\Sigma}$ are acquired according to (25) and (26), respectively. Now, these new estimates are used to repeat the process until specified number of iterations is reached. The latent variables may be then obtained through (13), (14).

2.5. Verification

In the verification phase two hypotheses are tested [1], namely: hypotheses $\mathcal{H}_s$ that two vectors $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ share the same identity, and hypotheses $\mathcal{H}_d$ that the identity of two vectors $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ differs. The Log-Likelihood Ratio (LLR) can be written as

$$\begin{aligned}\mathrm{LLR}(\boldsymbol{x}_1,\boldsymbol{x}_2) &= \log\frac{p(\boldsymbol{x}_1,\boldsymbol{x}_2|\mathcal{H}_s)}{p(\boldsymbol{x}_1|\mathcal{H}_d)p(\boldsymbol{x}_2|\mathcal{H}_d)} = \\ &= \log\mathcal{N}\left(\begin{bmatrix}\boldsymbol{x}_1\\ \boldsymbol{x}_2\end{bmatrix};\begin{bmatrix}\boldsymbol{\mu}\\ \boldsymbol{\mu}\end{bmatrix},\begin{bmatrix}\hat{\boldsymbol{C}}_X & \boldsymbol{C}_F\\ \boldsymbol{C}_F & \hat{\boldsymbol{C}}_X\end{bmatrix}\right) - \\ &- \log\mathcal{N}\left(\begin{bmatrix}\boldsymbol{x}_1\\ \boldsymbol{x}_2\end{bmatrix};\begin{bmatrix}\boldsymbol{\mu}\\ \boldsymbol{\mu}\end{bmatrix},\begin{bmatrix}\hat{\boldsymbol{C}}_X & \boldsymbol{0}\\ \boldsymbol{0} & \hat{\boldsymbol{C}}_X\end{bmatrix}\right),\end{aligned}$$

where $\hat{\boldsymbol{C}}_X = \boldsymbol{F}\boldsymbol{F}^{\mathrm{T}} + \boldsymbol{G}\boldsymbol{G}^{\mathrm{T}} + \boldsymbol{\Sigma}$ and $\boldsymbol{C}_F = \boldsymbol{F}\boldsymbol{F}^{\mathrm{T}}$, for more details see [13]. Note that in this verification scenario we do not care about the decomposition of $\boldsymbol{x}_1$ or $\boldsymbol{x}_2$, the question is whether two vectors share the same identity given the subspaces generated by $\boldsymbol{F}$ and $\boldsymbol{G}$.

3. EXPERIMENTS

To evaluate the time consumption of the algorithm and demonstrate the impact of fixing the weight ρ of the prior in $\boldsymbol{T}_F$ a Speaker Recognition (SR) system based on i-vectors [3] was used, and PLDA model was estimated in the i-vector space [8]. Development set consisted of corpora NIST SRE 2004, 2005, 2006, Switchboard 1 Release 2 and Switchboard 2 Phase 3, only male speakers who had more than 4 recorded sessions were utilized. Maximum number of sessions was 32, number of distinct J_i was 26 and the median was $J_i^{\mathrm{med}} = 8$. Overall 8312 recordings of 892 males were used, each of the recordings had approximately 5 minutes in duration including the silence (approx. 700 hours of speech). All the data were used to train both an Universal Background Model (UBM) containing 1024 Gaussians and an i-vector extractor. The dimension of i-vectors was set to $D_x = 400$, the dimension of the speaker identity space in the PLDA model was set to $D_h = 300$, and the dimension of the session/channel space was set to $D_w = 100$, 50 iterations were carried out to get estimates of PLDA parameters. Note that for each of the 8312 recordings one i-vector was extracted and PLDA model was trained from all of them.

The feature extraction was based on Linear Frequency Cepstral Coefficients (LFCCs), Hamming window of length 25 ms was used and shifted each 10 ms, 20 LFCCs were extracted, Δ-coefficients were added leading to 40 dimensional feature vectors. Voice activity detection was carried out to discard the non-speech frames, and at the end feature warping was applied utilizing a sliding window of length 3 seconds.

Time durations needed to process 10 iterations of the PLDA estimation algorithm on the development set are given in Table 1. Algorithms were implemented in MATLAB® R2011b (7.13.0.564), computer with Intel Core2 Quad CPU 2.83GHz and 8GB RAM was used, and only one thread was run. The naive algorithm follows the implementation from [1] described in Section 2.1 utilizing some optimizations when constructing new $\boldsymbol{A}_i^{\mathrm{T}}\boldsymbol{\Sigma}_i^{-1}\boldsymbol{A}_i + \boldsymbol{I}$ from the previous one (used for lower number of observations J_i). Note that for each distinct J_i the matrix $\boldsymbol{A}_i^{\mathrm{T}}\boldsymbol{\Sigma}_i^{-1}\boldsymbol{A}_i + \boldsymbol{I}$ was inverted in each iteration only once and stored for the future use. Method denoted $\mathrm{T_{F,G}}$ uses only the enhancement described in Section 2.2 with $\rho = 1/J_i$ fixed, and methods denoted "exact" (ρ recomputed for each distinct J_i) and "ρ fixed" use in addition also enhancements from Section 2.3.

Table 1. *Time durations in seconds needed to process 10 iterations on the development set.*

algorithm	naive	$\mathrm{T_{F,G}}$	exact	ρ fixed
t [s]	4820.23	46.48	12.82	1.92

Table 2. *Results obtained on NIST SRE 2008.*

ρ	exact	1/2	1/4	1/8	1/16
EER [%]	7.69	7.84	7.43	7.58	8.05

In the naive algorithm in each iteration 26 assembles and inversions of matrices of sizes ranging from $(300+4*100)\times(300+4*100) = 700\times700$ to $(300+32*100)\times(300+32*100) = 3500\times3500$ had to be done (for details see the discussion preceding (15), (16)), whereas in the case of $\mathrm{T_{F,G}}$ in each iteration only one inversion of a 300×300 matrix and one of a 100×100 matrix is performed. From Table 1 it is obvious that the time savings are huge. Moreover, if the required data covariances $\boldsymbol{C}_X$ and $\boldsymbol{C}_B$ where precomputed in advance, thus there was no need to iterate through the whole input set containing 8312 i-vectors in each iteration, additional speed up was acquired even if $\boldsymbol{T}_F$ had to be recomputed for each distinct value of J_i ("exact" case). And as expected, fixing ρ and using all the enhancements gives us the fastest algorithm. To show the impact of fixed ρ on the verification performance of the SR system experiments were carried out on "short2-short3 trials" from NIST SRE 2008 [14], only the male telephone speech was used (648 target speakers and 1535 test speakers) yielding 16968 trials in total. Error rates are given in Table 2, note that the effect on the performance of the SR system is negligible.

4. CONCLUSION

A novel implementation of the estimation algorithm of parameters of the PLDA model was described. Main contributions are the computational savings related to the inversion of a huge dense matrix. It was also shown how to make the estimation independent of the size of the development set. And moreover, greater insight into the method was provided.

5. REFERENCES

[1] Simon J D Prince and J H Elder, "Probabilistic Linear Discriminant Analysis for Inferences About Identity," *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pp. 1–8, 2007.

[2] Patrick Kenny, "Joint Factor Analysis of Speaker and Session Variability: Theory and Algorithms," Tech. Rep., Centre de Recherche Informatique de Montréal (CRIM), 2006.

[3] Najim Dehak, Patrick Kenny, Reda Dehak, Pierre Dumouchel, and Pierre Ouellet, "Front-End Factor Analysis For Speaker Verification," *IEEE Transactions on Audio, Speech and Language Processing*, 2010.

[4] Fabio Castaldo, Daniele Colibro, Claudio Vair, Sandro Cumani, and Pietro Laface, "Loquendo - Politecnico di Torino's 2010 NIST speaker recognition evaluation system," in *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. May 2011, pp. 5464–5467, IEEE.

[5] Niko Brummer, Luk Burget, Patrick Kenny, Pavel Matjka, Edward Villiers de, Martin Karafit, Marcel Kockmann, Ondej Glembek, Oldich Plchot, Doris Baum, and Mohammed Senoussauoi, "Abc system description for nist sre 2010," in *Proc. NIST 2010 Speaker Recognition Evaluation*. 2010, pp. 1–20, National Institute of Standards and Technology.

[6] Nicolas Scheffer, Yun Lei, Luciana Ferrer, S R I International, and Menlo Park, "Factor Analysis Back Ends for MLLR Transforms in Speaker Recognition," *Interspeech*, , no. August, pp. 257–260, 2011.

[7] Sandro Cumani, Oldrich Plchot, and Martin Karafiat, "Independent component analysis and MLLR transforms for speaker identification," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Mar. 2012, pp. 4365–4368, IEEE.

[8] Pavel Matějka, Ondej Glembek, Fabio Castaldo, Jahangir Alam, Oldich Plchot, Patrick Kenny, Lukáš Burget, and Jan Černocký, "Full-covariance UBM and Heavy-tailed PLDA in I-Vector Speaker Verification," in *Proceedings of the 2011 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2011*. 2011, pp. 4828–4831, IEEE Signal Processing Society.

[9] D Garcia-Romero, X Zhou, D Zotkin, B Srinivasan, Y Luo, S Ganapathy, S Thomas, S Nemala, GSVS Sivaram, M Mirbagheri, SH Mallidi, T Janu, P Rajan, N Mesgarani, M Elhilali, H Hermansky, S Shamma, and R Duraiswami, "The UMD-JHU 2011 speaker recognition system," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Mar. 2012, pp. 4229–4232, IEEE.

[10] Pierre Comon and Christian Jutten, "Handbook of Blind Source Separation: Independent Component Analysis and Applications," Mar. 2010.

[11] David A Harville, *Matrix Algebra From a Statistician's Perspective*, Springer, 2008.

[12] Lukáš Machlica, *High Dimensional Spaces and Modelling in the task of Speaker Recognition*, Ph.D. thesis, submitted for the degree of Doctor of Philosophy at University of West Bohemia, Department of Cybernetics, Czech Republic, 2012.

[13] Daniel Garcia-Romero and Carol Y Espy-Wilson, "Analysis of I-vector Length Normalization in Speaker Recognition Systems," *Interspeech*, pp. 249–252, 2011.

[14] "The NIST Year 2008 Speaker Recognition Evaluation Plan," 2008, `http://www.itl.nist.gov/iad/mig/tests/spk/2008/sre08_evalplan_release4.pdf`.