https://doi.org/10.1016/j.dsp.2021.103055

Investigating and Improving the Utility of Probabilistic Linear Discriminant Analysis for Acoustic Signal Classification

Yuechi Jiang and Frank H. F. Leung

***Abstract*—Probabilistic linear discriminant analysis (PLDA) has achieved good performance in face recognition and speaker recognition. However, the computation of PLDA using the original formulation is inefficient when there are many training data, especially when the dimensionality of the data is high. Faced with this inefficiency issue, we propose scalable formulations for PLDA. The computation of PLDA using the scalable formulations is more efficient than using the original formulation when dealing with many training data. Using the scalable formulations, the PLDA model can significantly outperform other popular classifiers for speaker recognition, such as Support Vector Machine (SVM) and Gaussian Mixture Model (GMM). Besides of directly using PLDA as a classifier, we may also use PLDA as a feature transformation technique. This PLDA-based feature transformation technique can reduce or expand the original feature dimensionality, and at the same time keep the transformed feature vector approximately following the Gaussian distribution. Our experimental results on speaker recognition and acoustic scene classification demonstrate the effectiveness of applying PLDA for feature transformation. It is then promising to combine PLDA with other classification models for improved performance, extending the utility of PLDA to a wider range of areas.**

***Index Terms*—Probabilistic linear discriminant analysis, scalability analysis, acoustic signal classification, feature transformation**

I. Introduction

PROBABILISTIC Linear Discriminant Analysis (PLDA) was first proposed for face recognition [1], and has achieved superior performance in face recognition, face verification, and face clustering [2]. It is quite suitable for some specific applications where there are few samples, such as age-invariant face recognition [3][4], and where there are large variations in the images [5]-[7] or facial expressions in videos [8]. By representing a video using a high-dimensional vector, PLDA achieves the state-of-the-art performance in action recognition [9]. Owing to the capability of PLDA in dealing with high-dimensional feature representations, it is later extended to speaker verification [10], and is currently the state-of-the-art back-end [11] [12]. The similarity score produced by PLDA is also applicable to other applications, such as speaker identification [13], spoofing detection [14], and voice search [15]. PLDA can also be applied to audio-visual cases, where audio information and visual information are fused for improved performance [16][17].

In the perspective of PLDA, a feature vector is assumed to be generated by a between-class latent variable and a within-class latent variable [1]. The between-class latent variable represents the common characteristics shared by the feature vectors within the same class, while the within-class latent variable represents the variation of the characteristics possessed by the feature vectors within the same class. That is to say, the between-class latent variable is supposed to be class-dependent, whereas the within-class latent variable is supposed to be feature-dependent. These latent variables have been shown to be quite useful in handling faces with different poses [2].

Despite of the discriminative nature of PLDA, its application scope is quite narrow, mainly focusing on face verification and speaker verification where there are few samples. It has not been widely used as a classification model, like Support Vector Machine (SVM), which has been applied to a wide range of pattern recognition tasks. The major obstacles are probably the scalability of PLDA, as the original formulation cannot handle many training data. In the original formulation, during the process of estimating the parameters of a PLDA model, the inverse of large matrices needs to be found, whose sizes are proportional to the dimensionality and the quantity of the training data. If the quantity of the training data is large or the dimensionality of the training data is high, inverting those matrices can be quite difficult or even infeasible if exceeding the memory space. This scalability issue also arises during the process of class label prediction.

In the literature, scalable formulations for model parameter estimation have been proposed, such that the parameters of a PLDA model can be efficiently estimated using many training data [18][19]. In this paper, we propose scalable formulations for class label prediction, such that a PLDA model can efficiently make predictions using many training data. Instead of focusing on the simplified formulation of PLDA widely used in speaker verification [20], we shall consider the general formulation.

In the literature, PLDA is usually applied to high-dimensional feature vectors, such as i-vector [12]. In this paper, we find that, when the dimensionality of the feature vector is low, PLDA may not work well. Faced with this issue, we propose a novel

Yuechi Jiang and Frank H. F. Leung are with the Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hung Hom, Hong Kong. (e-mail: yuechi.jiang@connect.polyu.hk, frank-h-f.leung@polyu.edu.hk).

application of PLDA, which uses it for feature transformation. In this way, the between-class latent variable in a PLDA model is used as the transformed feature vector, whose dimensionality can be higher or lower than the original feature vector. This brings more flexibility to the transformed feature vector. As the between-class latent variable carries the class information, the transformed feature vector can be more discriminative, and at the same time keep approximately following a Gaussian distribution. By this means, PLDA can be combined with other classification models, such as Gaussian Mixture Model (GMM).

PLDA can be treated as a probabilistic version of Fisher Linear Discriminant Analysis (LDA). LDA is widely used as a dimensionality reduction technique [21], however, the dimensionality of the LDA-based transformed vector cannot be higher than that of the original feature vector, and the transformed vector will probably not follow Gaussian distributions. In contrast, the dimensionality of the PLDA-based transformed vector is dependent on the dimensionality of the latent variable, which is weakly related to the dimensionality of the original feature vector.

To investigate the utility of PLDA for acoustic signal classification, we consider a speaker recognition task and an acoustic scene classification task. Experimental results on these datasets demonstrate the effectiveness and potential of PLDA. They also demonstrate the feasibility of applying PLDA as a feature transformation technique, which may open a new direction of combining PLDA with other classification models as a feature pre-processing technique. Our major contributions lie in the following aspects:

- We analyze the working mechanism of using PLDA to do classification tasks, including the model parameter estimation stage and the class label prediction stage.
- We propose scalable formulations for PLDA to efficiently make class label predictions using many training data. These formulations, together with the scalable formulation for model parameter estimation, enable PLDA to be used as a general-purpose classifier.
- We propose to use PLDA as a feature transformation technique, which can perform both dimensionality reduction and dimensionality expansion, and at the same time keep the transformed vector approximately following a Gaussian distribution.
- We briefly analyze the relationship between PLDA and some classic techniques, and explain that PLDA can be treated as the generalization of the single Gaussian model and the factor analysis model, and is also closely related to the concept of LDA.
- We conduct experiments on different datasets and different feature vectors, trying to explore the factors that may influence the effectiveness of PLDA. Experimental results show that PLDA behaves differently for speech signals and non-speech signals. Its effectiveness highly depends on the characteristics of the acoustic signals.

The rest of this paper is organized as follows. In Section II, the model assumption of PLDA is introduced, followed by the explanations on the original formulations for model parameter estimation and class label prediction. In Section III, the scalable formulations of PLDA are given, including model parameter estimation and various criteria for class label prediction. Its scalability is also analyzed. In Section IV, the way of using PLDA for feature transformation is described. In Section V, the acoustic features are briefly described. In Section VI, experimental results are presented and discussed. In Section VII, a conclusion is drawn.

II. Original Formulation of PLDA

A PLDA model is intrinsically a latent variable model, which assumes that a feature vector is generated by latent variables. The latent variables are supposed to capture useful information from the feature vector, such as the class information, and thus can be used to represent the feature vector or discriminate between different classes.

Given a set of training and testing feature vectors, in order to use a PLDA model to do the classification, there exist two stages. In the first stage, the parameters of the PLDA model need to be estimated based on the training vectors. In the second stage, prediction criteria are needed to assign the testing vectors into different classes.

This section introduces and explains the model assumptions adopted by PLDA, the original formulation for model parameter estimation, and the original formulation for class label prediction. An illustration of the two stages is shown in Fig. 1, where the color indicates the class of the feature vector. In this example, there are 3 classes and J training vectors for each class. $\boldsymbol{x}_{kj}$ denotes the j-th training vector in the k-th class, and $\boldsymbol{y}$ denotes a testing vector with unknown class. The model parameters $\{\boldsymbol{\mu}, \boldsymbol{F}, \boldsymbol{G}, \boldsymbol{\Sigma}\}$ will be explained later.

A. Model Assumption

In this part, the latent variable model adopted by PLDA is introduced. In a PLDA model, a feature vector is assumed to be generated by a between-class latent variable and a within-class latent variable, and therefore should be able to be expressed as the affine transformation of the two latent variables. Suppose we are given a set of training feature vectors denoted as $\{ \boldsymbol{x}_{11}, \boldsymbol{x}_{12}, \ldots \boldsymbol{x}_{1J}, \boldsymbol{x}_{21}, \boldsymbol{x}_{22}, \ldots \boldsymbol{x}_{2J}, \ldots \boldsymbol{x}_{K1}, \boldsymbol{x}_{K2}, \ldots \boldsymbol{x}_{KJ} \}$, where we assume there are K different classes, and J training vectors for each class. Therefore, $\boldsymbol{x}_{kj}$ represents the j-th training vector

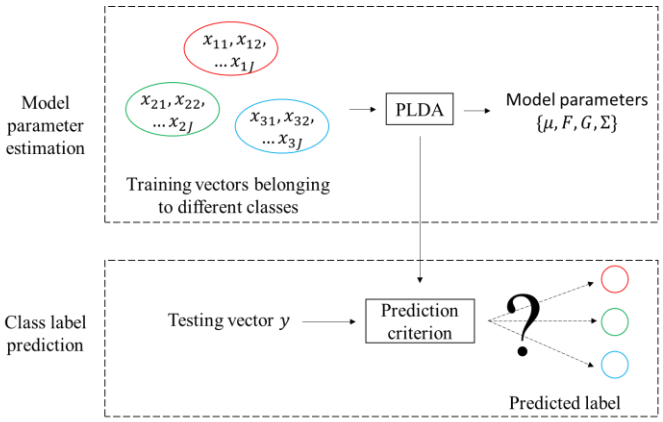


Fig. 1. The working mechanism of PLDA for classification tasks.

in the *k*-th class. It is possible to have different numbers of training vectors in different classes, but we just use the same symbol *J* to represent the number of training vectors in each class for simplicity.

The expression relating a feature vector $\boldsymbol{x}_{kj}$ to its between-class latent variable $\boldsymbol{h}_k$ and its within-class latent variable $\boldsymbol{w}_{kj}$ is given by (1), where $\boldsymbol{\mu}$ is the global mean vector, $\boldsymbol{F}$ and $\boldsymbol{G}$ are factor loading matrices, and $\boldsymbol{\varepsilon}_{kj}$ is the noise variable [1]. The between-class latent variable $\boldsymbol{h}_k$ reflects the common characteristics shared by all the feature vectors in class *k*, whereas the within-class latent variable $\boldsymbol{w}_{kj}$ reflects the variation of the feature vectors in class *k*.

$$\boldsymbol{x}_{kj} = \boldsymbol{\mu} + \boldsymbol{F}\boldsymbol{h}_k + \boldsymbol{G}\boldsymbol{w}_{kj} + \boldsymbol{\varepsilon}_{kj} \tag{1}$$

PLDA is a special form of factor analysis, meaning that it shall abide by the assumptions of factor analysis. Namely, $\boldsymbol{h}_k$ and $\boldsymbol{w}_{kj}$ shall follow Gaussian distributions with zero mean and unit variance, while the noise variable $\boldsymbol{\varepsilon}_{kj}$ shall follow a Gaussian distribution with zero mean and diagonal covariance $\boldsymbol{\Sigma}$. Since $\boldsymbol{h}_k$, $\boldsymbol{w}_{kj}$ and $\boldsymbol{\varepsilon}_{kj}$ are assumed to be independent of each other, $\boldsymbol{x}_{kj}$ shall also follow a Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{F}\boldsymbol{F}^T + \boldsymbol{G}\boldsymbol{G}^T + \boldsymbol{\Sigma}$, as given by (2).

$$p(\boldsymbol{x}_{kj}) = \mathcal{N}(\boldsymbol{x}_{kj} \mid \boldsymbol{\mu}, \boldsymbol{F}\boldsymbol{F}^T + \boldsymbol{G}\boldsymbol{G}^T + \boldsymbol{\Sigma}) \tag{2}$$

The feature vectors in a class are forced to share the same between-class latent variable, thus they can be treated as a whole using the expression given by (3).

$$\begin{bmatrix} \boldsymbol{x}_{k1} \\ \boldsymbol{x}_{k2} \\ \vdots \\ \boldsymbol{x}_{kJ} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\mu} \\ \boldsymbol{\mu} \\ \vdots \\ \boldsymbol{\mu} \end{bmatrix} + \begin{bmatrix} \boldsymbol{F} & \boldsymbol{G} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{F} & \boldsymbol{0} & \boldsymbol{G} & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{F} & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{G} \end{bmatrix} \begin{bmatrix} \boldsymbol{h}_k \\ \boldsymbol{w}_{k1} \\ \boldsymbol{w}_{k2} \\ \vdots \\ \boldsymbol{w}_{kJ} \end{bmatrix} + \begin{bmatrix} \boldsymbol{\varepsilon}_{k1} \\ \boldsymbol{\varepsilon}_{k2} \\ \vdots \\ \boldsymbol{\varepsilon}_{kJ} \end{bmatrix} \tag{3}$$

Eq. (3) can then be simplified to the standard form of a factor analysis model, as given by (4).

$$\boldsymbol{X}_k = \boldsymbol{U} + \boldsymbol{R}\boldsymbol{Y}_k + \boldsymbol{\zeta}_k \tag{4}$$

where

$$\boldsymbol{X}_k = \begin{bmatrix} \boldsymbol{x}_{k1} \\ \boldsymbol{x}_{k2} \\ \vdots \\ \boldsymbol{x}_{kJ} \end{bmatrix},\ \boldsymbol{U} = \begin{bmatrix} \boldsymbol{\mu} \\ \boldsymbol{\mu} \\ \vdots \\ \boldsymbol{\mu} \end{bmatrix},\ \boldsymbol{Y}_k = \begin{bmatrix} \boldsymbol{h}_k \\ \boldsymbol{w}_{k1} \\ \boldsymbol{w}_{k2} \\ \vdots \\ \boldsymbol{w}_{kJ} \end{bmatrix},\ \boldsymbol{\zeta}_k = \begin{bmatrix} \boldsymbol{\varepsilon}_{k1} \\ \boldsymbol{\varepsilon}_{k2} \\ \vdots \\ \boldsymbol{\varepsilon}_{kJ} \end{bmatrix} \tag{5}$$

$$\boldsymbol{R} = \begin{bmatrix} \boldsymbol{F} & \boldsymbol{G} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{F} & \boldsymbol{0} & \boldsymbol{G} & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{F} & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{G} \end{bmatrix}$$

As $\boldsymbol{Y}_k$ and $\boldsymbol{\zeta}_k$ are assumed to be independent of each other, $\boldsymbol{X}_k$ shall follow a Gaussian distribution with mean $\boldsymbol{U}$ and covariance $\boldsymbol{R}\boldsymbol{R}^T + \boldsymbol{\Phi}$, as given by (6), where $\boldsymbol{\Phi}$ is the covariance of $\boldsymbol{\zeta}_k$ [1][2]. Namely, the joint distribution of all the feature vectors in class *k* is a Gaussian distribution.

$$p(\boldsymbol{x}_{k1}, \boldsymbol{x}_{k2} \ldots \boldsymbol{x}_{kJ}) = p(\boldsymbol{X}_k) = \mathcal{N}(\boldsymbol{X}_k \mid \boldsymbol{U}, \boldsymbol{R}\boldsymbol{R}^T + \boldsymbol{\Phi}) \tag{6}$$

where

$$\boldsymbol{\Phi} = \begin{bmatrix} \boldsymbol{\Sigma} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\Sigma} & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{\Sigma} \end{bmatrix} \tag{7}$$

On the one hand, the between-class factor loading matrix $\boldsymbol{F}$ and the within-class factor loading matrix $\boldsymbol{G}$ imitate the roles of the between-class scatter matrix and the within-class scatter matrix used in Linear Discriminant Analysis (LDA) [21]. On the other hand, PLDA can be reformulated in the form of standard Factor Analysis (FA). In this perspective, PLDA can be treated as a fusion of LDA and FA. When the between-class latent variable $\boldsymbol{h}_k$ in (1) vanishes, PLDA becomes FA, where class information disappears. From this perspective, PLDA can be treated as the generalization of FA.

B. Model Parameter Estimation

In this part, the original formulation for estimating the parameters of a PLDA model is introduced. The parameters of a PLDA model can be denoted as $\{\boldsymbol{\mu}, \boldsymbol{F}, \boldsymbol{G}, \boldsymbol{\Sigma}\}$ according to (1). They can also be derived from $\{\boldsymbol{U}, \boldsymbol{R}, \boldsymbol{\Phi}\}$ according to (4), which are the parameters of a standard factor analysis model and can be estimated using the Expectation-Maximization (EM) algorithm [1][22]. The EM algorithm includes an E-step and an M-step. In the E-step, the conditional expected mean $E[\boldsymbol{Y}_k]$ and the conditional expected covariance $E[\boldsymbol{Y}_k \boldsymbol{Y}_k^T]$ are calculated using (8) and (9).

$$E[\boldsymbol{Y}_k] = (\boldsymbol{I} + \boldsymbol{R}^T \boldsymbol{\Phi}^{-1} \boldsymbol{R})^{-1} \boldsymbol{R}^T \boldsymbol{\Phi}^{-1} (\boldsymbol{X}_k - \boldsymbol{U}) \tag{8}$$

$$E[\boldsymbol{Y}_k\boldsymbol{Y}_k^T]=(\boldsymbol{I}+\boldsymbol{R}^T\boldsymbol{\Phi}^{-1}\boldsymbol{R})^{-1}+E[\boldsymbol{Y}_k]E[\boldsymbol{Y}_k]^T \quad (9)$$

In the M-step, $\{\boldsymbol{U},\boldsymbol{R},\boldsymbol{\Phi}\}$ are re-estimated based on $E[\boldsymbol{Y}_k]$ and $E[\boldsymbol{Y}_k\boldsymbol{Y}_k^T]$ obtained from the E-step. This may not be obtained from (4) directly, because the dimensionality of $\boldsymbol{X}_k$ may be different for different classes (because we may have different numbers of training vectors in different classes). Fortunately, (1) can also be formulated in another way as given by (10), where $\boldsymbol{V}$ is the combination of $\boldsymbol{F}$ and $\boldsymbol{G}$, and $\boldsymbol{z}_{kj}$ is the concatenation of $\boldsymbol{h}_k$ and $\boldsymbol{w}_{kj}$.

$$\boldsymbol{x}_{kj}=\boldsymbol{\mu}+\begin{bmatrix}\boldsymbol{F} & \boldsymbol{G}\end{bmatrix}\begin{bmatrix}\boldsymbol{h}_k\\ \boldsymbol{w}_{kj}\end{bmatrix}+\boldsymbol{\varepsilon}_{kj}=\boldsymbol{\mu}+\boldsymbol{V}\boldsymbol{z}_{kj}+\boldsymbol{\varepsilon}_{kj} \quad (10)$$

Based on (10), $\{\boldsymbol{U},\boldsymbol{R},\boldsymbol{\Phi}\}$ can be derived from $\{\boldsymbol{\mu},\boldsymbol{V},\boldsymbol{\Sigma}\}$, which can be calculated based on the conditional expectations $E[\boldsymbol{z}_{kj}]$ and $E[\boldsymbol{z}_{kj}\boldsymbol{z}_{kj}^T]$ using (11) ~ (13) [1]. $E[\boldsymbol{z}_{kj}]$ and $E[\boldsymbol{z}_{kj}\boldsymbol{z}_{kj}^T]$ can be obtained from $E[\boldsymbol{Y}_k]$ and $E[\boldsymbol{Y}_k\boldsymbol{Y}_k^T]$ by examining the relationship between $\boldsymbol{z}_{kj}$ and $\boldsymbol{Y}_k$ given by (10) and (5). Actually $\boldsymbol{z}_{kj}$ is just a part of $\boldsymbol{Y}_k$, and consequently $E[\boldsymbol{z}_{kj}\boldsymbol{z}_{kj}^T]$ is a part of $E[\boldsymbol{Y}_k\boldsymbol{Y}_k^T]$.

$$\boldsymbol{\mu}=\frac{1}{KJ}\sum_{k=1}^{K}\sum_{j=1}^{J}\boldsymbol{x}_{kj} \quad (11)$$

$$\boldsymbol{V}=\left(\sum_{k=1}^{K}\sum_{j=1}^{J}(\boldsymbol{x}_{kj}-\boldsymbol{\mu})E[\boldsymbol{z}_{kj}]^T\right)\left(\sum_{k=1}^{K}\sum_{j=1}^{J}E[\boldsymbol{z}_{kj}\boldsymbol{z}_{kj}^T]\right)^{-1} \quad (12)$$

$$\boldsymbol{\Sigma}=\frac{1}{KJ}\sum_{k=1}^{K}\sum_{j=1}^{J}diag\left((\boldsymbol{x}_{kj}-\boldsymbol{\mu})(\boldsymbol{x}_{kj}-\boldsymbol{\mu})^T-\boldsymbol{V}E[\boldsymbol{z}_{kj}](\boldsymbol{x}_{kj}-\boldsymbol{\mu})^T\right) \quad (13)$$

In brief, the E-step calculates the expectations of the latent variables using (8) and (9), while the M-step re-estimates the model parameters using (11) ~ (13).

C. Class Label Prediction

In this part, the way of using a well-trained PLDA model to predict a testing vector into different classes, is explained. The fundamental assumption is that, vectors belonging to the same class should have the same between-class latent variable, whereas vectors belonging to different classes should have different between-class latent variables. This idea is illustrated in Fig. 2. The training vectors in the k-th class share the same between-class latent variable $\boldsymbol{h}_k$. In order to determine which class a testing vector $\boldsymbol{y}$ should belong to, the idea is to determine which between-class latent variable $\boldsymbol{h}_k$ is the same as the between-class latent variable $\boldsymbol{h}$ of $\boldsymbol{y}$.

In [1] and [2], it is assumed that there is only one training vector for each class, namely the training data are

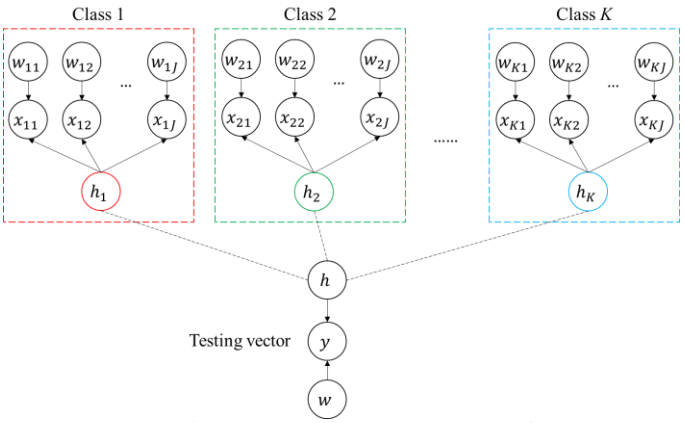


Fig. 2. The method of predicting a testing vector into a specific class.

$\{\boldsymbol{x}_{11},\boldsymbol{x}_{21},\ldots\boldsymbol{x}_{K1}\}$. The probability of predicting a testing vector $\boldsymbol{y}$ to class k, is the probability that the between-class latent variable of $\boldsymbol{y}$ is the same as that of $\boldsymbol{x}_{k1}$ but different from those of the other training vectors [2][23]. This can be reflected from the expression of the joint distribution of $\boldsymbol{y}$ and the training vectors, as given by (14).

$$\begin{aligned}p(\boldsymbol{y},\boldsymbol{x}_{11},\boldsymbol{x}_{21}\ldots\boldsymbol{x}_{K1}\mid k)&=p(\boldsymbol{y},\boldsymbol{x}_{k1})\prod_{i=1,i\neq k}^{K}p(\boldsymbol{x}_{i1})\\&=p(\boldsymbol{y}\mid\boldsymbol{x}_{k1})\prod_{i=1}^{K}p(\boldsymbol{x}_{i1})\end{aligned} \quad (14)$$

Denoting the classified label as $\ell(\boldsymbol{y})$, this prediction criterion can be expressed as (15), which aims at finding the maximum conditional probability. The core idea is that, if $\boldsymbol{y}$ indeed belongs to class k, $\boldsymbol{y}$ and $\boldsymbol{x}_{k1}$ should be jointly distributed, whereas $\boldsymbol{y}$ and the other training vectors should be independently distributed.

$$\ell(\boldsymbol{y})=\arg\max_k p(\boldsymbol{y}\mid\boldsymbol{x}_{k1})\prod_{i=1}^{K}p(\boldsymbol{x}_{i1})=\arg\max_k p(\boldsymbol{y}\mid\boldsymbol{x}_{k1}) \quad (15)$$

If there are J training vectors for each class, denoted as $\{\boldsymbol{x}_{11},\boldsymbol{x}_{12},\ldots\boldsymbol{x}_{1J},\boldsymbol{x}_{21},\boldsymbol{x}_{22},\ldots\boldsymbol{x}_{2J},\ldots\boldsymbol{x}_{K1},\boldsymbol{x}_{K2},\ldots\boldsymbol{x}_{KJ}\}$, the conditional probability prediction criterion given by (15) can be extended to (16) [24][25].

$$\ell(\boldsymbol{y})=\arg\max_k p(\boldsymbol{y}\mid\boldsymbol{x}_{k1},\boldsymbol{x}_{k2}\ldots\boldsymbol{x}_{kJ})=\arg\max_k p(\boldsymbol{y}\mid\boldsymbol{X}_k) \quad (16)$$

As can be seen from (6), the joint distribution of $\{\boldsymbol{x}_{k1},\boldsymbol{x}_{k2},\ldots\boldsymbol{x}_{kJ}\}$ is a Gaussian distribution, because they share the same between-class latent variable $\boldsymbol{h}_k$. If the testing vector $\boldsymbol{y}$ is supposed to belong to class k, $\boldsymbol{y}$ should share the same between-class latent variable with $\{\boldsymbol{x}_{k1},\boldsymbol{x}_{k2},\ldots\boldsymbol{x}_{kJ}\}$. This indicates that, the joint distribution of $\boldsymbol{y}$ and $\{\boldsymbol{x}_{k1},\boldsymbol{x}_{k2},\ldots\boldsymbol{x}_{kJ}\}$ is also a Gaussian distribution, as given by (17), with mean $\boldsymbol{U}'$ and covariance $\boldsymbol{R}'\boldsymbol{R}'^T+\boldsymbol{\Phi}'$ given by (18).

$$p(\boldsymbol{x}_{k1}, \boldsymbol{x}_{k2} ... \boldsymbol{x}_{kJ}, \boldsymbol{y}) = p(\boldsymbol{X}_k, \boldsymbol{y}) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{X}_k \\ \boldsymbol{y} \end{bmatrix} | \boldsymbol{U}', \boldsymbol{R}'\boldsymbol{R}'^T + \boldsymbol{\Phi}' \right) \quad (17)$$

where

$$\boldsymbol{U}' = \begin{bmatrix} \boldsymbol{U} \\ \boldsymbol{\mu} \end{bmatrix}, \ \boldsymbol{R}' = \begin{bmatrix} \begin{bmatrix} & & \\ & \boldsymbol{R} & \\ & & \end{bmatrix} & \begin{matrix} \boldsymbol{0} \\ \boldsymbol{0} \\ \vdots \end{matrix} \\ \begin{matrix} \boldsymbol{F} & \boldsymbol{0} & \boldsymbol{0} & \cdots \end{matrix} & \boldsymbol{G} \end{bmatrix}, \ \boldsymbol{\Phi}' = \begin{bmatrix} \boldsymbol{\Phi} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\Sigma} \end{bmatrix} \quad (18)$$

As both $\boldsymbol{y}$ and $\boldsymbol{X}_k$ follow a Gaussian distribution, the conditional distribution of $\boldsymbol{y}$ given $\boldsymbol{X}_k$ is also a Gaussian distribution, which can be derived from the joint distribution given by (17). As both the mean $\boldsymbol{U}'$ and the covariance $\boldsymbol{R}'\boldsymbol{R}'^T + \boldsymbol{\Phi}'$ are block matrices, they can be partitioned in the form of matrix blocks, as given by (19) [25].

$$\boldsymbol{U}' = \begin{bmatrix} \boldsymbol{U}_y \\ \boldsymbol{U}_X \end{bmatrix}, \quad \boldsymbol{R}'\boldsymbol{R}'^T + \boldsymbol{\Phi}' = \begin{bmatrix} \boldsymbol{\Phi}_{yy} & \boldsymbol{\Phi}_{yX} \\ \boldsymbol{\Phi}_{Xy} & \boldsymbol{\Phi}_{XX} \end{bmatrix} \quad (19)$$

where

$$\boldsymbol{U}_y = \boldsymbol{\mu}, \ \boldsymbol{U}_X = \boldsymbol{U}$$

$$\boldsymbol{\Phi}_{yy} = \boldsymbol{F}\boldsymbol{F}^T + \boldsymbol{G}\boldsymbol{G}^T + \boldsymbol{\Sigma}, \ \boldsymbol{\Phi}_{yX} = \begin{bmatrix} \boldsymbol{F}\boldsymbol{F}^T & \boldsymbol{F}\boldsymbol{F}^T & \cdots & \boldsymbol{F}\boldsymbol{F}^T \end{bmatrix} \quad (20)$$

$$\boldsymbol{\Phi}_{Xy} = \begin{bmatrix} \boldsymbol{F}\boldsymbol{F}^T \\ \boldsymbol{F}\boldsymbol{F}^T \\ \vdots \\ \boldsymbol{F}\boldsymbol{F}^T \end{bmatrix}, \ \boldsymbol{\Phi}_{XX} = \begin{bmatrix} \boldsymbol{\Phi}_{yy} & \boldsymbol{F}\boldsymbol{F}^T & \boldsymbol{F}\boldsymbol{F}^T & \cdots \\ \boldsymbol{F}\boldsymbol{F}^T & \boldsymbol{\Phi}_{yy} & \boldsymbol{F}\boldsymbol{F}^T & \cdots \\ \boldsymbol{F}\boldsymbol{F}^T & \boldsymbol{F}\boldsymbol{F}^T & \boldsymbol{\Phi}_{yy} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

Based on the above partition, the conditional probability of $\boldsymbol{y}$ given $\boldsymbol{X}_k$ can be calculated using (21), where $\boldsymbol{U}_{y|X}$ and $\boldsymbol{\Phi}_{y|X}$ are the conditional mean and the conditional covariance [26].

$$\begin{aligned} p(\boldsymbol{y} \mid \boldsymbol{X}_k) &= \mathcal{N}(\boldsymbol{y} \mid \boldsymbol{U}_{y|X}, \boldsymbol{\Phi}_{y|X}) \\ &= \mathcal{N}(\boldsymbol{y} \mid \boldsymbol{U}_y + \boldsymbol{\Phi}_{yX}\boldsymbol{\Phi}_{XX}^{-1}(\boldsymbol{X}_k - \boldsymbol{U}_X), \boldsymbol{\Phi}_{yy} - \boldsymbol{\Phi}_{yX}\boldsymbol{\Phi}_{XX}^{-1}\boldsymbol{\Phi}_{Xy}) \end{aligned} \quad (21)$$

III. Scalable Formulation of PLDA

Since the original formulations in the model parameter estimation stage and the class label prediction stage require inverting large matrices whose size is proportional to the number of training data in each class, they are not suitable for handling large numbers of training data. It is sometimes even infeasible to do the matrix inversion operation if the matrix size exceeds the memory space. Therefore, this section introduces and explains the scalable formulations for the model parameter estimation stage and the class label prediction stage. The scalable formulations enable a PLDA model to do the classification efficiently with many training data.

A. Model Parameter Estimation

In this part, the scalable formulation for model parameter estimation is introduced, which is based on [19]. The core idea is to transform the large-size matrix inversion operation into several small-size matrix inversion operations, in a mathematically equivalent way.

In the stage of model parameter estimation, the EM algorithm is used. In the original formulation, the E-step requires inverting a large matrix $\boldsymbol{I} + \boldsymbol{R}^T\boldsymbol{\Phi}^{-1}\boldsymbol{R}$, as can be seen from (8) and (9). The size of matrix $\boldsymbol{R}$ is proportional to the dimensionality and the quantity of the training vectors in a class, as can be seen from (5), making finding the inverse of $\boldsymbol{I} + \boldsymbol{R}^T\boldsymbol{\Phi}^{-1}\boldsymbol{R}$ difficult or even infeasible if the quantity of training vectors is large. A scalable formulation for estimating the model parameters has been proposed [19], which partitions $\boldsymbol{I} + \boldsymbol{R}^T\boldsymbol{\Phi}^{-1}\boldsymbol{R}$ into a block matrix and then utilizes the trick of partitioned matrix inversion [26].

In the original formulation, the conditional expectations $E[\boldsymbol{Y}_k]$ and $E[\boldsymbol{Y}_k\boldsymbol{Y}_k^T]$ need to be computed in the E-step; while in the scalable formulation, the conditional expectations $E[\boldsymbol{z}_{kj}]$ and $E[\boldsymbol{z}_{kj}\boldsymbol{z}_{kj}^T]$ need to be computed in the E-step, as given by (22) and (23) [19]. The M-step remains unchanged. As the sizes of the matrices involved in computation are independent of the quantity of the training vectors, as can be seen from (24), this scalable formulation is more efficient in handling many training data.

$$E[\boldsymbol{z}_{kj}] = \begin{bmatrix} E[\boldsymbol{h}_k] \\ E[\boldsymbol{w}_{kj}] \end{bmatrix} = \begin{bmatrix} (\boldsymbol{M}\boldsymbol{F}^T\boldsymbol{\Sigma}^{-1} - \boldsymbol{M}\boldsymbol{\Lambda}^T\boldsymbol{G}^T\boldsymbol{\Sigma}^{-1})\sum_{j=1}^{J}(\boldsymbol{x}_{kj} - \boldsymbol{\mu}) \\ \boldsymbol{L}^{-1}\boldsymbol{G}^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}_{kj} - \boldsymbol{\mu}) - \boldsymbol{\Lambda}E[\boldsymbol{h}_k] \end{bmatrix} \quad (22)$$

$$E[\boldsymbol{z}_{kj}\boldsymbol{z}_{kj}^T] = \begin{bmatrix} \boldsymbol{M} & -\boldsymbol{M}\boldsymbol{\Lambda}^T \\ -\boldsymbol{\Lambda}\boldsymbol{M} & \boldsymbol{L}^{-1} + \boldsymbol{\Lambda}\boldsymbol{M}\boldsymbol{\Lambda}^T \end{bmatrix} + E[\boldsymbol{z}_{kj}]E[\boldsymbol{z}_{kj}]^T \quad (23)$$

where

$$\begin{aligned} \boldsymbol{L} &= \boldsymbol{I} + \boldsymbol{G}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{G} \\ \boldsymbol{\Lambda} &= \boldsymbol{L}^{-1}\boldsymbol{G}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{F} \\ \boldsymbol{M} &= \left(\boldsymbol{I} + J\boldsymbol{F}^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{F} - \boldsymbol{G}\boldsymbol{\Lambda})\right)^{-1} \end{aligned} \quad (24)$$

B. Class Label Prediction

In this part, the scalable formulations for predicting the class label of a testing feature vector are introduced and explained. The core idea is to estimate a class-specific probability with respect to each class, and then the class label is predicted by finding the class having the highest class-specific probability among all.

In the stage of class label prediction, there can be different prediction criteria. For example, for the conditional probability criterion, prediction is made by calculating the conditional probability $p(\boldsymbol{y} \mid \boldsymbol{X}_k)$ with respect to each class. As can be

seen from (21), both the conditional mean $\boldsymbol{U}_{y|X}$ and the conditional covariance $\boldsymbol{\Phi}_{y|X}$ require inverting a large matrix $\boldsymbol{\Phi}_{XX}$. The size of $\boldsymbol{\Phi}_{XX}$ is proportional to the dimensionality and the quantity of the training vectors in a class, as can be seen from (20), making the inverting process difficult or even infeasible if there are many training vectors.

Besides of the conditional probability prediction criterion, we can also employ the joint probability prediction criterion, as given by (25).

$$\ell(\boldsymbol{y}) = \arg\max_k \; p(\boldsymbol{y}, \boldsymbol{x}_{k1}, \boldsymbol{x}_{k2} \ldots \boldsymbol{x}_{kJ}) = \arg\max_k \; p(\boldsymbol{y}, \boldsymbol{X}_k) \quad (25)$$

According to (17), the joint probability can be calculated as given by (26), where D is the dimensionality of a feature vector and J is the number of training vectors in class k. As can be seen, the determinant and the inverse of the covariance $\boldsymbol{R}'\boldsymbol{R}'^T + \boldsymbol{\Phi}'$ are also difficult to calculate directly.

$$\begin{aligned} p(\boldsymbol{y}, \boldsymbol{X}_k) &= p(\boldsymbol{X}_k, \boldsymbol{y}) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{X}_k \\ \boldsymbol{y} \end{bmatrix} | \, \boldsymbol{U}', \boldsymbol{R}'\boldsymbol{R}'^T + \boldsymbol{\Phi}' \right) \\ &= \frac{1}{(2\pi)^{(J+1)D/2} \left| \boldsymbol{R}'\boldsymbol{R}'^T + \boldsymbol{\Phi}' \right|^{1/2}} \\ &\quad \times \exp\left(-\frac{1}{2} \begin{bmatrix} \boldsymbol{X}_k - \boldsymbol{U} \\ \boldsymbol{y} - \boldsymbol{\mu} \end{bmatrix}^T (\boldsymbol{R}'\boldsymbol{R}'^T + \boldsymbol{\Phi}')^{-1} \begin{bmatrix} \boldsymbol{X}_k - \boldsymbol{U} \\ \boldsymbol{y} - \boldsymbol{\mu} \end{bmatrix} \right) \end{aligned} \quad (26)$$

In the following, we provide the scalable formulations for the above-mentioned conditional probability prediction criterion and the joint probability prediction criterion, such that the sizes of the matrices involved in computation are independent of the quantity of the training vectors.

We also provide several pairwise probability-based prediction criteria, such as the pairwise conditional probability criteria and the pairwise joint probability criteria. The pairwise probability is the probability between the testing vector and a training vector, such as the pairwise conditional probability $p(\boldsymbol{y} \mid \boldsymbol{x}_{kj})$ and the pairwise joint probability $p(\boldsymbol{y}, \boldsymbol{x}_{kj})$, instead of the probability between the testing vector and all the training vectors. The pairwise probability-based prediction criteria also make the computation scalable, as the computation of each pairwise probability is efficient. However, if there are many training data, the increased numbers of pairwise probability computation will lead to increased total computation time. Different prediction criteria are also illustrated in Fig. 3.

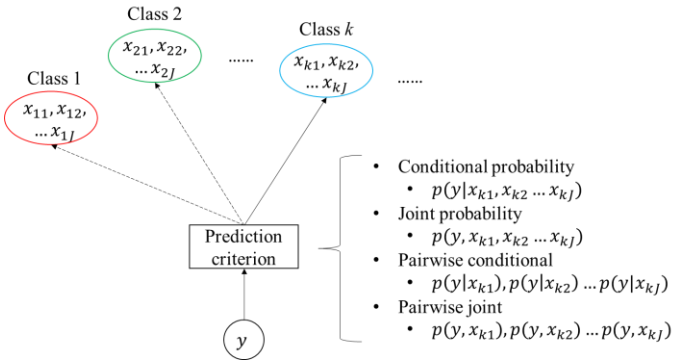


Fig. 3. Different prediction criteria.

1) Conditional Probability Prediction Criterion (OrdinCond)

We name this criterion as Ordinary Conditional (OrdinCond). For this criterion, the major difficulty is the inverse of $\boldsymbol{\Phi}_{XX}$. Fortunately, $\boldsymbol{\Phi}_{XX}$ is a block matrix possessing a very special structure, as can be seen from (20). The inverse should also be a block matrix with the structure as given by (27), where we use a new symbol $\boldsymbol{\Psi}_J$ to represent $\boldsymbol{\Phi}_{XX}$, $\boldsymbol{P}^{(J)}$ and $\boldsymbol{Q}^{(J)}$ are symmetric matrices, and the superscript and subscript J denotes the number of matrix blocks in $\boldsymbol{\Psi}_J$.

$$\begin{aligned} \boldsymbol{\Phi}_{XX}^{-1} &= \begin{bmatrix} \boldsymbol{\Phi}_{yy} & \boldsymbol{F}\boldsymbol{F}^T & \boldsymbol{F}\boldsymbol{F}^T & \cdots \\ \boldsymbol{F}\boldsymbol{F}^T & \boldsymbol{\Phi}_{yy} & \boldsymbol{F}\boldsymbol{F}^T & \cdots \\ \boldsymbol{F}\boldsymbol{F}^T & \boldsymbol{F}\boldsymbol{F}^T & \boldsymbol{\Phi}_{yy} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}^{-1} = \begin{bmatrix} \boldsymbol{P}^{(J)} & \boldsymbol{Q}^{(J)} & \boldsymbol{Q}^{(J)} & \cdots \\ \boldsymbol{Q}^{(J)} & \boldsymbol{P}^{(J)} & \boldsymbol{Q}^{(J)} & \cdots \\ \boldsymbol{Q}^{(J)} & \boldsymbol{Q}^{(J)} & \boldsymbol{P}^{(J)} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \\ &= \boldsymbol{\Psi}_J^{-1} \end{aligned} \quad (27)$$

It can be shown that, $\boldsymbol{P}^{(J)}$ and $\boldsymbol{Q}^{(J)}$ have the expressions as given by (28) [25]. Detailed derivations can be found in the appendix.

$$\begin{aligned} \boldsymbol{P}^{(J)} &= \left(\boldsymbol{F}\boldsymbol{F}^T + \boldsymbol{G}\boldsymbol{G}^T + \boldsymbol{\Sigma} - (J-1)\boldsymbol{F}\boldsymbol{F}^T \boldsymbol{W}^{-1} \boldsymbol{F}\boldsymbol{F}^T \right)^{-1} \\ \boldsymbol{Q}^{(J)} &= -\boldsymbol{W}^{-1} \boldsymbol{F}\boldsymbol{F}^T \boldsymbol{P}^{(J)} \\ \boldsymbol{W} &= \boldsymbol{G}\boldsymbol{G}^T + \boldsymbol{\Sigma} + (J-1)\boldsymbol{F}\boldsymbol{F}^T \end{aligned} \quad (28)$$

Having the inverse of $\boldsymbol{\Phi}_{XX}$, the conditional mean $\boldsymbol{U}_{y|X}$ and the conditional covariance $\boldsymbol{\Phi}_{y|X}$ can be calculated in terms of $\boldsymbol{P}^{(J)}$ and $\boldsymbol{Q}^{(J)}$, as given by (29) and (30).

$$\begin{aligned} \boldsymbol{U}_{y|X} &= \boldsymbol{U}_y + \boldsymbol{\Phi}_{yX} \boldsymbol{\Phi}_{XX}^{-1} (\boldsymbol{X}_k - \boldsymbol{U}_X) \\ &= \boldsymbol{\mu} + \begin{bmatrix} \boldsymbol{F}\boldsymbol{F}^T & \cdots & \boldsymbol{F}\boldsymbol{F}^T \end{bmatrix} \begin{bmatrix} \boldsymbol{P}^{(J)} & \boldsymbol{Q}^{(J)} & \cdots \\ \boldsymbol{Q}^{(J)} & \boldsymbol{P}^{(J)} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_{k1} - \boldsymbol{\mu} \\ \vdots \\ \boldsymbol{x}_{kJ} - \boldsymbol{\mu} \end{bmatrix} \\ &= \boldsymbol{\mu} + \left(\boldsymbol{F}\boldsymbol{F}^T \boldsymbol{P}^{(J)} + (J-1)\boldsymbol{F}\boldsymbol{F}^T \boldsymbol{Q}^{(J)} \right) \sum_{j=1}^{J} (\boldsymbol{x}_{kj} - \boldsymbol{\mu}) \end{aligned} \quad (29)$$

$$\begin{aligned} \boldsymbol{\Phi}_{y|X} &= \boldsymbol{\Phi}_{yy} - \boldsymbol{\Phi}_{yX} \boldsymbol{\Phi}_{XX}^{-1} \boldsymbol{\Phi}_{Xy} \\ &= \boldsymbol{\Phi}_{yy} - \begin{bmatrix} \boldsymbol{F}\boldsymbol{F}^T & \cdots & \boldsymbol{F}\boldsymbol{F}^T \end{bmatrix} \begin{bmatrix} \boldsymbol{P}^{(J)} & \boldsymbol{Q}^{(J)} & \cdots \\ \boldsymbol{Q}^{(J)} & \boldsymbol{P}^{(J)} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} \boldsymbol{F}\boldsymbol{F}^T \\ \vdots \\ \boldsymbol{F}\boldsymbol{F}^T \end{bmatrix} \\ &= \left(\boldsymbol{F}\boldsymbol{F}^T + \boldsymbol{G}\boldsymbol{G}^T + \boldsymbol{\Sigma} \right) - J \left(\boldsymbol{F}\boldsymbol{F}^T \boldsymbol{P}^{(J)} + (J-1)\boldsymbol{F}\boldsymbol{F}^T \boldsymbol{Q}^{(J)} \right) \boldsymbol{F}\boldsymbol{F}^T \end{aligned} \quad (30)$$

Then, the prediction can be made by finding the class with

the highest conditional probability, as given by (31).

$$\ell(\boldsymbol{y}) = \arg\max_k p(\boldsymbol{y} \mid \boldsymbol{X}_k) = \arg\max_k \mathcal{N}(\boldsymbol{y} \mid \boldsymbol{U}_{y|X}, \boldsymbol{\Phi}_{y|X}) \quad (31)$$

2) Joint Probability Prediction Criterion (OrdinJoint)

We name this criterion as Ordinary Joint (OrdinJoint). For this criterion, the major difficulty is the determinant and the inverse of the joint covariance $\boldsymbol{R}'\boldsymbol{R}'^T + \boldsymbol{\Phi}'$, as can be seen from (26). Fortunately, $\boldsymbol{R}'\boldsymbol{R}'^T + \boldsymbol{\Phi}'$ is also a block matrix possessing a similar structure to $\boldsymbol{\Phi}_{XX}$. Its determinant is namely $|\boldsymbol{\Psi}_{J+1}|$, which can be derived using the induction method, as given by (32). Detailed derivations can be found in the appendix.

$$|\boldsymbol{\Psi}_{J+1}| = \left|\boldsymbol{\Phi}_{yy} - J\boldsymbol{F}\boldsymbol{F}^T\left(\boldsymbol{P}^{(J)} + (J-1)\boldsymbol{Q}^{(J)}\right)\boldsymbol{F}\boldsymbol{F}^T\right| \cdot |\boldsymbol{\Psi}_J| \quad (32)$$

The exponential term in (26) can be derived using the fact that $\boldsymbol{R}'\boldsymbol{R}'^T + \boldsymbol{\Phi}' = \boldsymbol{\Psi}_{J+1}$ and the result in (27). The prediction can then be made by finding the class with the highest joint probability, as given by (33). Detailed derivations for $\Delta_k(y)$ can be found in the appendix.

$$\begin{aligned}\ell(\boldsymbol{y}) &= \arg\max_k p(\boldsymbol{y}, \boldsymbol{X}_k) = \arg\max_k \mathcal{N}\left(\begin{bmatrix}\boldsymbol{X}_k \\ \boldsymbol{y}\end{bmatrix} \mid \boldsymbol{U}', \boldsymbol{R}'\boldsymbol{R}'^T + \boldsymbol{\Phi}'\right) \\ &= \arg\max_k \frac{1}{(2\pi)^{(J+1)D/2}|\boldsymbol{\Psi}_{J+1}|^{1/2}} \exp\left(-\frac{\Delta_k(y)}{2}\right)\end{aligned} \quad (33)$$

where

$$\begin{aligned}\Delta_k(y) = &\sum_j (\boldsymbol{x}_{kj} - \boldsymbol{\mu})^T \boldsymbol{P}^{(J+1)} (\boldsymbol{x}_{kj} - \boldsymbol{\mu}) + \sum_i \sum_{j \neq i} (\boldsymbol{x}_{kj} - \boldsymbol{\mu})^T \boldsymbol{Q}^{(J+1)} (\boldsymbol{x}_{ki} - \boldsymbol{\mu}) \\ &+ 2\sum_j (\boldsymbol{x}_{kj} - \boldsymbol{\mu})^T \boldsymbol{Q}^{(J+1)} (\boldsymbol{y} - \boldsymbol{\mu}) + (\boldsymbol{y} - \boldsymbol{\mu})^T \boldsymbol{P}^{(J+1)} (\boldsymbol{y} - \boldsymbol{\mu})\end{aligned} \quad (34)$$

3) Pairwise Probability Prediction Criterion

Besides of calculating the conditional probability $p(\boldsymbol{y} \mid \boldsymbol{X}_k)$ or the joint probability $p(\boldsymbol{y}, \boldsymbol{X}_k)$ with respect to all the training vectors in a class, it is also feasible to calculate the pairwise conditional probability $p(\boldsymbol{y} \mid \boldsymbol{x}_{kj})$ or the pairwise joint probability $p(\boldsymbol{y}, \boldsymbol{x}_{kj})$ with respect to each training vector in a class, and then combine these pairwise probabilities for all the training vectors. $p(\boldsymbol{y} \mid \boldsymbol{x}_{kj})$ and $p(\boldsymbol{y}, \boldsymbol{x}_{kj})$ are given by (35) and (36), similar to (21) and (26).

$$p(\boldsymbol{y} \mid \boldsymbol{x}_{kj}) = \mathcal{N}(\boldsymbol{y} \mid \boldsymbol{U}_y + \boldsymbol{F}\boldsymbol{F}^T\boldsymbol{\Phi}_{yy}^{-1}(\boldsymbol{x}_{kj} - \boldsymbol{\mu}), \boldsymbol{\Phi}_{yy} - \boldsymbol{F}\boldsymbol{F}^T\boldsymbol{\Phi}_{yy}^{-1}\boldsymbol{F}\boldsymbol{F}^T) \quad (35)$$

$$p(\boldsymbol{y}, \boldsymbol{x}_{kj}) = p(\boldsymbol{x}_{kj}, \boldsymbol{y}) = \mathcal{N}\left(\begin{bmatrix}\boldsymbol{x}_{kj} \\ \boldsymbol{y}\end{bmatrix} \mid \begin{bmatrix}\boldsymbol{\mu} \\ \boldsymbol{U}_y\end{bmatrix}, \begin{bmatrix}\boldsymbol{\Phi}_{yy} & \boldsymbol{F}\boldsymbol{F}^T \\ \boldsymbol{F}\boldsymbol{F}^T & \boldsymbol{\Phi}_{yy}\end{bmatrix}\right) \quad (36)$$

Having found the pairwise probabilities for all pairs of $\boldsymbol{y}$ and $\boldsymbol{x}_{kj}$, the prediction can then be made according to the following criteria. In the following expressions, J_k is used to denote the number of training vectors in class *k*, instead of *J*, as there can be different numbers of training vectors in different classes. Criteria *a*) ~ *f*) are named as ArithCond, GeoCond, CmaxCond, ArithJoint, GeoJoint, and CmaxJoint respectively.

a) Arithmetic Mean of Pairwise Conditional (ArithCond)

Prediction is made by finding the class with the highest arithmetic mean of the pairwise conditional probabilities.

$$\ell(\boldsymbol{y}) = \arg\max_k \frac{1}{J_k}\sum_{j=1}^{J_k} p(\boldsymbol{y} \mid \boldsymbol{x}_{kj}) \quad (37)$$

b) Geometric Mean of Pairwise Conditional (GeoCond)

Prediction is made by finding the class with the highest geometric mean of the pairwise conditional probabilities.

$$\ell(\boldsymbol{y}) = \arg\max_k \frac{1}{J_k}\sum_{j=1}^{J_k} \ln p(\boldsymbol{y} \mid \boldsymbol{x}_{kj}) \quad (38)$$

c) Cmax Pairwise Conditional (CmaxCond)

Prediction is made by finding the most frequently occurred class among *C* maximum pairwise conditional probabilities, as given by (39), where $\boldsymbol{x}_i$ represents a training vector, $\{p_i'\}$ is a sequence of $p(\boldsymbol{y} \mid \boldsymbol{x}_i)$ sorted in descending order, and $\mathbf{1}_k(p_i')$ is the indicator function given by (40). This prediction criterion is similar to K Nearest Neighbor (KNN), which counts the occurrence of different classes among K minimum distances.

$$\ell(\boldsymbol{y}) = \arg\max_k \sum_{i=1}^{C} \mathbf{1}_k(p_i') \quad (39)$$

where

$$\begin{aligned}&\{p_i' \mid i = 1, 2 ... \sum_{k=1}^{K} J_k\} = \{p(\boldsymbol{y} \mid \boldsymbol{x}_i) \mid i = 1, 2 ... \sum_{k=1}^{K} J_k\} \quad s.t.\ p_i' \geq p_{i+1}' \\ &\mathbf{1}_k(p_i') = \begin{cases}1, & \boldsymbol{x}_i \in class\ k \\ 0, & otherwise\end{cases}\end{aligned} \quad (40)$$

d) Arithmetic Mean of Pairwise Joint (ArithJoint)

Prediction is made by finding the class with the highest arithmetic mean of the pairwise joint probabilities.

$$\ell(\boldsymbol{y}) = \arg\max_k \frac{1}{J_k}\sum_{j=1}^{J_k} p(\boldsymbol{y}, \boldsymbol{x}_{kj}) \quad (41)$$

e) Geometric Mean of Pairwise Joint (GeoJoint)

Prediction is made by finding the class with the highest geometric mean of the pairwise joint probabilities.

$$\ell(\boldsymbol{y}) = \arg\max_k \frac{1}{J_k}\sum_{j=1}^{J_k} \ln\, p(\boldsymbol{y}, \boldsymbol{x}_{kj}) \tag{42}$$

f) Cmax Pairwise Joint (CmaxJoint)

Prediction is made by finding the most frequently occurred class among C maximum pairwise joint probabilities, as given by (43), where $\boldsymbol{x}_i$ represents a training vector, $\{\ p_i''\ \}$ is a sequence of $p(\boldsymbol{y}, \boldsymbol{x}_i)$ sorted in descending order, and $\mathbf{1}_k(p_i'')$ is the indicator function, as given by (44).

$$\ell(\boldsymbol{y}) = \arg\max_k \sum_{i=1}^{C} \mathbf{1}_k(p_i'') \tag{43}$$

where

$$\begin{aligned} &\{p_i'' \mid i = 1, 2 ... \sum_{k=1}^{K} J_k\} = \{p(\boldsymbol{y}, \boldsymbol{x}_i) \mid i = 1, 2 ... \sum_{k=1}^{K} J_k\} \quad s.t.\ p_i'' \geq p_{i+1}'' \\ &\mathbf{1}_k(p_i'') = \begin{cases} 1, & \boldsymbol{x}_i \in class\ k \\ 0, & otherwise \end{cases} \end{aligned} \tag{44}$$

C. Scalability and Robustness Analysis

In this part, the scalability of the scalable formulations is analyzed, including both the model parameter estimation stage and the class label prediction stage. The properties of different class label prediction criteria are also briefly analyzed.

Suppose there are J training vectors in class k, the dimensionality of a feature vector $\boldsymbol{x}_{kj}$ is D×1, and the dimensionality of the latent variables $\boldsymbol{h}_k$ and $\boldsymbol{w}_{kj}$ is H×1. Then the size of the factor loading matrices $\boldsymbol{F}$ and $\boldsymbol{G}$ is D×H, and the size of $\boldsymbol{\Sigma}$ is D×D.

In the original formulation for model parameter estimation, in the E-step given by (8) and (9), the inverse of $\boldsymbol{I} + \boldsymbol{R}^T\boldsymbol{\Phi}^{-1}\boldsymbol{R}$ needs to be computed. The size of $\boldsymbol{\Phi}$ is JD×JD, and the size of $\boldsymbol{R}$ is JD×(J+1)H, resulting in the size of $\boldsymbol{I} + \boldsymbol{R}^T\boldsymbol{\Phi}^{-1}\boldsymbol{R}$ to be (J+1)H×(J+1)H, which could be too large to be inverted if J is large.

Fortunately, in the scalable formulation, as given by (22) and (23), the computation is based on $\boldsymbol{F}$, $\boldsymbol{G}$ and $\boldsymbol{\Sigma}$, whose sizes are independent of J.

In the original formulation for class label prediction, the conditional probability criterion given by (21) requires computing the inverse of $\boldsymbol{\Phi}_{XX}$, whose size is JD×JD, and the joint probability criterion given by (26) requires computing the determinant and the inverse of $\boldsymbol{R}'\boldsymbol{R}'^T + \boldsymbol{\Phi}'$, whose size is ($J$+1)$D$×($J$+1)$D$. This dependency on J makes the computation inefficient, and sometimes even infeasible.

Fortunately, we have the scalable formulation for the conditional probability criterion as given by (29) and (30), and the scalable formulation for the joint probability criterion as given by (32) and (34), where the computation is based on $\boldsymbol{F}$, $\boldsymbol{G}$, $\boldsymbol{\Sigma}$, $\boldsymbol{P}$ and $\boldsymbol{Q}$, whose sizes are independent of J. Furthermore, we also have several pairwise probability-based prediction criteria as given by (37) ~ (44), where the computation is based on the pairwise conditional probability given by (35), or the pairwise joint probability given by (36), which is also independent of J.

In fact, the pairwise probability-based criteria can be more robust than the conditional probability or the joint probability criterion, in the case where the feature vectors do not follow Gaussian distributions. This situation usually occurs when the dimensionality of the feature vector is too high. The pairwise probability merely assumes that a pair of feature vectors, e.g. $\boldsymbol{y}$ and $\boldsymbol{x}_{kj}$, follow a Gaussian distribution, whereas the conditional probability or the joint probability criterion assumes that a group of feature vectors, e.g. $\boldsymbol{y}$ and $\{\ \boldsymbol{x}_{k1}$, $\boldsymbol{x}_{k2} \ldots \boldsymbol{x}_{kJ}\ \}$, follow a Gaussian distribution, which is quite a strong assumption and may fail when the dimensionality of the feature vector is much higher than the number of training vectors. In particular, CmaxCond and CmaxJoint prediction criteria are supposed to be more robust than ArithCond, GeoCond, ArithJoint and GeoJoint, since only a portion of "important" training vectors are involved in prediction. However, the computation of the conditional probability and the joint probability criteria can be more efficient than the pairwise probability-based criteria, as the expected mean and the expected covariance of the former can be computed for only once, whereas the latter requires the computation of the pairwise probability with respect to each pair of training and testing vectors, whose computation complexity is proportional to the number of training vectors.

IV. PLDA for Feature Transformation

Besides of directly using PLDA for class label prediction, it is feasible to use PLDA to extract feature vectors. As can be seen from (1), each feature vector $\boldsymbol{x}_{kj}$ is associated with a between-class latent variable $\boldsymbol{h}_k$, a within-class latent variable $\boldsymbol{w}_{kj}$, and a noise term $\boldsymbol{\varepsilon}_{kj}$. For prediction purposes, $\boldsymbol{h}_k$ seems to be the most important latent variable, as it carries the class information. If we can determine the $\boldsymbol{h}_k$ of a feature vector $\boldsymbol{y}$, we may then immediately know which class the vector $\boldsymbol{y}$ should belong to. However, it is difficult to obtain the exact value of $\boldsymbol{h}_k$. Fortunately, the posterior mean of $\boldsymbol{h}_k$ can be estimated using (22), meaning that we may approximate $\boldsymbol{h}_k$ using $E[\boldsymbol{h}_k]$. Nevertheless, for a feature vector whose class label is unknown, (22) is still infeasible. In order to estimate the posterior mean $E[\boldsymbol{h}_y]$ for $\boldsymbol{y}$, we may adopt the approximation given by (45), which conducts the estimation using only the given feature vector $\boldsymbol{y}$. Noted that, parameter matrix $\boldsymbol{M}$ is slightly modified.

$$E[\boldsymbol{h}_y] = (\boldsymbol{M}\boldsymbol{F}^T\boldsymbol{\Sigma}^{-1} - \boldsymbol{M}\boldsymbol{\Lambda}^T\boldsymbol{G}^T\boldsymbol{\Sigma}^{-1})(\boldsymbol{y} - \boldsymbol{\mu}) \tag{45}$$

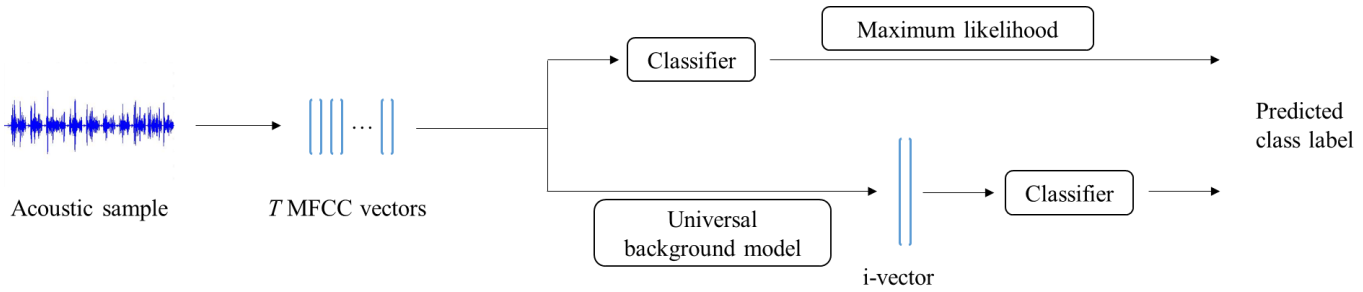


Fig. 4. A general classification system for acoustic signals.

where

$$\boldsymbol{M} = \left(\boldsymbol{I} + \boldsymbol{F}^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{F} - \boldsymbol{G}\boldsymbol{\Lambda})\right)^{-1} \tag{46}$$

By this means, a feature vector $\boldsymbol{y}$ is mapped to another feature space and becomes $E[\boldsymbol{h}_y]$. Ideally, this mapped vector should be more discriminative than $\boldsymbol{y}$, as in the model assumption of PLDA, $E[\boldsymbol{h}_y]$ should carry class information. In addition, $E[\boldsymbol{h}_y]$ can be even more Gaussian than $\boldsymbol{y}$, as in a PLDA model, the between-class latent variable $\boldsymbol{h}$ is assumed to follow a Gaussian distribution. In addition, the dimensionality of $E[\boldsymbol{h}_y]$ can be lower or higher than $\boldsymbol{y}$, which provides more flexibility.

V. Acoustic Signal Feature Extraction

Each acoustic sample is first framed using a Hamming window with 40ms length and 20ms shift. Then the Mel-frequency Cepstral Coefficients (MFCC) [27] are calculated for each frame, yielding a 20-dimension MFCC vector for each frame. Suppose an acoustic sample s is represented by T MFCC vectors, denoted as $\{\boldsymbol{x}_1, \boldsymbol{x}_2 \ldots \boldsymbol{x}_T\}$, where $\boldsymbol{x}_t$ is the t-th MFCC vector. These MFCC vectors are then used to form a single feature vector. The single feature vector representing the whole acoustic sample can be an i-vector [28], which is based on a Universal Background Model (UBM) and trained using the EM algorithm [29]. The UBM is essentially a Gaussian Mixture Model (GMM), which is constructed using the mixture splitting technique [30] and the EM algorithm [31]. If the dimensionality of an MFCC vector is D×1, and the GMM-based UBM has M mixture components, then the dimensionality of i-vector will be DM×1.

Besides of forming a single vector representing an acoustic sample, the MFCC vectors can also be directly used for classification, for example, when using class-specific GMMs. In this way, one GMM is trained using the MFCC vectors of all the acoustic samples in a class, which leads to a total of K GMMs for K classes. The prediction is then made by finding the maximum likelihood among all the GMMs [32].

A general acoustic signal classification system is illustrated in Fig. 4. Since the length of an acoustic sample can be different, it is first divided into equal-length short-time frames to extract the frame-level feature vectors, such as the MFCC vectors. Having obtained the MFCC vectors, we can then use a classifier, e.g., GMM, to make predictions for each MFCC vector. The predicted labels can then be combined based on majority voting or maximum likelihood. These MFCC vectors can also be used to produce a single feature vector representing the whole sample, e.g., i-vector. A classifier, such as PLDA or SVM, can then be used to predict the i-vector into a specific class.

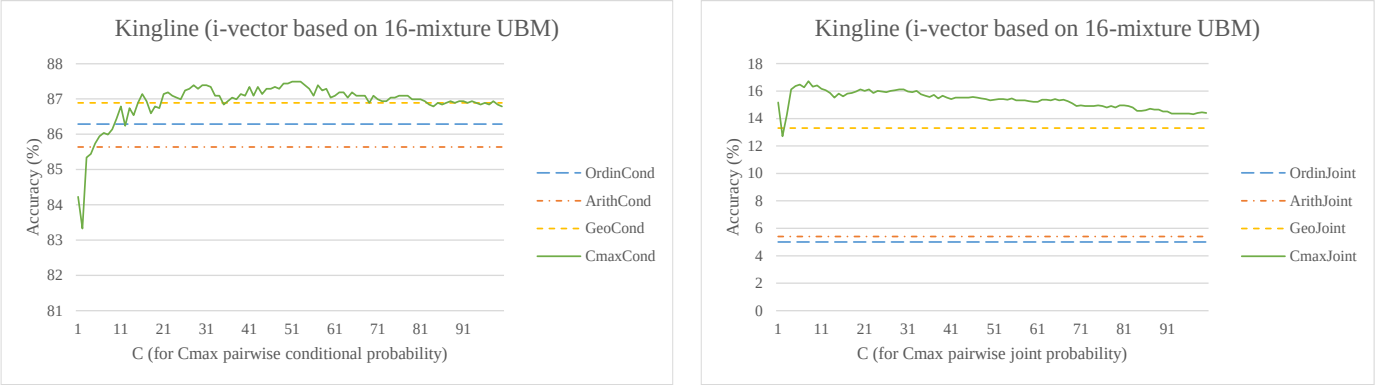
VI. Experimental Results and Discussions

In this section, we conduct experiments on speaker recognition and acoustic scene classification. For speaker recognition, part of Kingline081 [33] American English speech corpus is used, which consists of 20 speakers' speeches. The speeches are recorded in three different sessions, with each session consisting of about 100 speech samples for each speaker. The speeches of the first two sessions are used to form a training set, while the third session is used to form a testing set. This yields a training set consisting of 3997 speech samples and a testing set consisting of 1998 speech samples. Each sample lasts for 2s ~ 5s. For acoustic scene classification, DCASE2013 [34] is used, which consists of 10 acoustic scenes. Each acoustic recording lasts for 30s. The public dataset of DCASE2013 is used for training, and for each acoustic scene, there are 10 acoustic samples. The private dataset of DCASE2013 is used for testing, and for each acoustic scene, there are 10 acoustic samples. This yields a training set of 100 acoustic samples and a testing set of 100 acoustic samples. Details are also summarized in Table I.

For model parameters of PLDA, the factor loading matrices $\boldsymbol{F}$ and $\boldsymbol{G}$, and the noise covariance $\boldsymbol{\Sigma}$, are initialized to have all ones on the principal diagonal and zeros on other positions. Suppose the dimensionality of a feature vector $\boldsymbol{x}_{kj}$ is D×1, and the dimensionality of the latent variables $\boldsymbol{h}_k$ and $\boldsymbol{w}_{kj}$ is H×1, then the size of the factor loading matrices $\boldsymbol{F}$ and $\boldsymbol{G}$ is D×H. In the case where H>D, namely the number of columns is larger than the number of rows in the factor loading matrices, the column vectors starting from the (D+1)-th column are generated as a linear combination of the first D columns, where

TABLE I
Acoustic Signal Dataset

Dataset	Number of classes	Number of samples		Duration of each sample
		Training	Testing	
Kingline081	20	3997	1998	2s ~ 5s
DCASE2013	10	100	100	30s



(a) (b)

Fig. 5. Speaker recognition results using i-vector and PLDA with different prediction criteria. (a) PLDA with conditional probability-based prediction criteria. (b) PLDA with joint probability-based prediction criteria.

the coefficients are generated based on the uniform distribution. The randomly generated column vectors are then normalized to have unit length. For simplicity, the dimensionality of $\boldsymbol{h}_k$ and $\boldsymbol{w}_{kj}$ is kept being the same. The training data is also used to construct the UBM to calculate i-vector.

A. Effects of Different Prediction Criteria for PLDA

In this part, we compare the effects of different prediction criteria, including those based on ordinary and pairwise conditional probabilities and those based on ordinary and pairwise joint probabilities. I-vector is calculated using 5 EM iterations, based on a 16-mixture UBM. The dimensionality of i-vector is 320×1. The model parameters of PLDA is estimated by running the EM algorithm for 3 iterations. The dimensionality of the latent variables is set to be the same as that of i-vector. Speaker recognition results on Kingline081 are shown in Fig. 5, with respect to different C values (varying from 1 to 100) used in CmaxCond and CmaxJoint criteria.

It can be seen from Fig. 5 that the joint probability-based criteria give rather worse performance than the conditional probability-based criteria. The potential reason behind can be explained using (14). The conditional probability-based criteria originate from the idea that, if a testing vector $\boldsymbol{y}$ belongs to class k, $\boldsymbol{y}$ and the training vectors in class k are supposed to be jointly distributed, whereas $\boldsymbol{y}$ and the training vectors in other classes are supposed to be independently distributed. From this perspective, the joint probability-based criteria merely expect that, if $\boldsymbol{y}$ belongs to class k, $\boldsymbol{y}$ and the training vectors in class k should be jointly distributed, but do not expect any relationship between $\boldsymbol{y}$ and the training vectors in other classes. This makes the joint probability-based criteria less discriminative than the conditional probability-based criteria.

It is also observed that, the CmaxCond and the CmaxJoint criteria can be slightly better than the OrdinCond and the OrdinJoint criteria, as the former two select several best matched feature vectors in the training set (i.e., the training vectors giving the first several highest pairwise probabilities) to make predictions, instead of using all the training vectors, in which way only a portion of "important" training vectors are involved in making predictions. Nevertheless, the performance of the CmaxCond and CmaxJoint tend to be the same as OrdinCond and OrdinJoint as C increases (i.e., the number of training vectors involved in predictions increases). However, the pairwise probability-based criteria (ArithCond, GeoCond, CmaxCond, ArithJoint, GeoJoint, CmaxJoint) form a non-parametric model, whose computation time is proportional to the number of training data, whereas the ordinary probability-based criteria (OrdinCond, OrdinJoint) form a parametric model, meaning that the model parameters can be pre-computed and thus more efficient for a large number of training data.

Besides, the GeoCond and the GeoJoint criteria are better than the ArithCond and the ArithJoint. This indicates that the geometric mean seems better than the arithmetic mean. This implies that the pairwise probabilities tend to be independent of each other and follow Gaussian distributions. Mathematically, for a set of vectors, the geometric mean is always smaller than the arithmetic mean, and the equality holds if and only if the vectors are the same. This means the arithmetic mean can be driven to be much larger for a set of feature vectors if some of the pairwise probabilities are highly deviated from the center. From this perspective, the geometric mean should be more robust than the arithmetic mean.

B. Effects of Feature Dimensionality: The Pitfall

In this part, we investigate the performance of PLDA when the dimensionality of i-vector changes, and compare it with SVM. We first consider the speaker recognition task. Experimental results on Kingline081 are shown in Fig. 6. The prediction criterion is OrdinCond. The dimensionality of the latent variables is set to be the same as i-vector. SVM is implemented using LIBSVM [35]. The number of mixture components in the UBM varies from 2 to 128, yielding the i-vectors with the dimensionality varying from 40×1 to 2560×1. It is obvious that PLDA significantly outperforms SVM, demonstrating its superiority and great potential.

The performance of PLDA and SVM may improve as the dimensionality of i-vector increases, however, it tends to saturate and ceases improving if the dimensionality is large enough (e.g., 2560×1). This also shows the importance of the feature dimensionality for PLDA to be effective. According to (1), in a PLDA model, it is assumed that the feature vector can be expressed as the affine transformation of the latent variables.

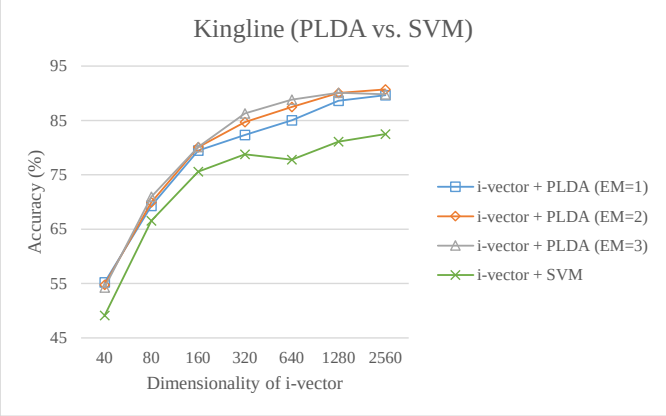


Fig. 6. Effects of the dimensionality of i-vector using Kingline081 dataset.

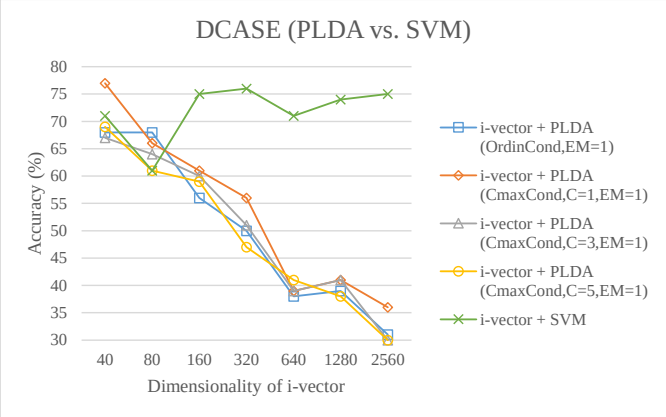


Fig. 7. Effects of the dimensionality of i-vector using DCASE2013 dataset.

This assumption implicitly presumes that the feature vector has a relatively complicated structure and carries a relatively large amount of information, which requires its dimensionality to be high enough. Increasing the number of EM iterations in training the PLDA model tends to increase its performance. Basically, several EM iterations are already good enough.

However, when we consider the acoustic scene classification task, PLDA may not work very well. Experimental results on DCASE2013 are shown in Fig. 7. The prediction criteria include OrdinCond and CmaxCond. As can be seen from Fig. 7, PLDA just slightly outperforms SVM when the dimensionality of i-vector is very low (40×1). The performance of PLDA degrades significantly as the increase of the dimensionality, which is on the contrary of the trend of SVM whose performance improves as the increase of the dimensionality. This is probably because the training data are inadequate. For DCASE2013, there are only 100 training acoustic samples, with 10 for each acoustic scene. It is difficult to estimate a conditional probability using only 10 training vectors whose dimensionality can be as high as 2560×1. Fig. 8 shows how the performance of PLDA is affected by different EM iterations during model parameter estimation. It seems increasing the number of EM iterations tends to degrade the performance, which implies that the model assumption of PLDA is actually violated. This performance degradation implies that, for PLDA to be effectively used, enough training data are necessary, so that the Gaussian assumption involved in using a PLDA model

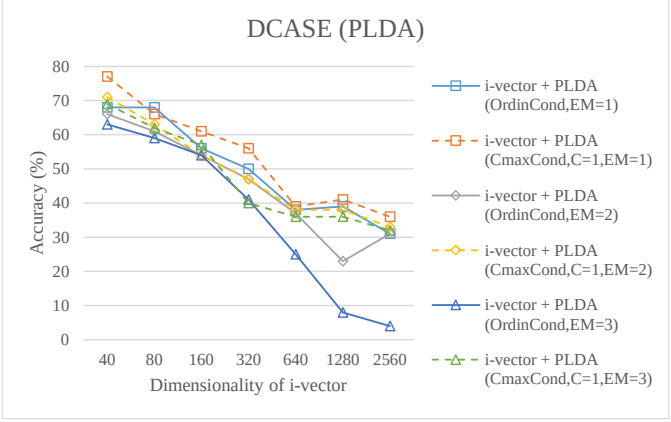


Fig. 8. PLDA with different EM iterations using DCASE2013 dataset.

can be approximately satisfied.

Briefly speaking, in order to effectively employ PLDA for class label prediction, it is necessary to have 1) high-dimensional feature vectors such that the latent variable model assumption is satisfied, and 2) an enough number of training data such that the Gaussian assumption is satisfied. These two requirements seem to be the limitation of applying PLDA, but they also endow PLDA with a great potential when the requirements are fulfilled.

C. PLDA for Feature Transformation

In this part, we investigate an interesting application of PLDA, i.e., using PLDA for feature transformation. The MFCC vectors are used as the raw feature vector, and GMM is used as the classifier. A PLDA model is trained using 1 EM iteration, and then the between-class latent variable is used as the transformed vector according to (45). Suppose the dimensionality of an MFCC vector is D×1, and the dimensionality of the between-class latent variable is H×1, then the transformed vector has a dimensionality of H×1. If H>D, the dimensionality of the original MFCC vector will be expanded, which will then embed more information. At the same time, this dimensionality expansion effect does not violate too much the Gaussian assumption of the MFCC vectors, because the latent variables in a PLDA model also follow Gaussian distributions.

When the feature vector is the MFCC vector whose dimensionality is 20×1, directly using PLDA for class label prediction does not work well, as shown in Fig. 9 and Fig. 10. In this scenario, PLDA uses the same maximum likelihood prediction criterion as GMM, where the probability is computed using the OrdinCond criterion. The reason is that, an MFCC vector does not carry much information, and therefore does not conform to the latent variable model assumption of PLDA given by (1).

Faced with this low-dimensionality issue, we may apply PLDA for feature transformation and dimensionality expansion. Experimental results on Kingline081 and DCASE2013 are shown in Fig. 11 and Fig. 12 respectively. The expansion ratio H/D varies from 1 to 3. It can be seen from Fig. 11 that, increasing the value of H tends to improve the performance of the transformed vector on using the GMM classifier. This

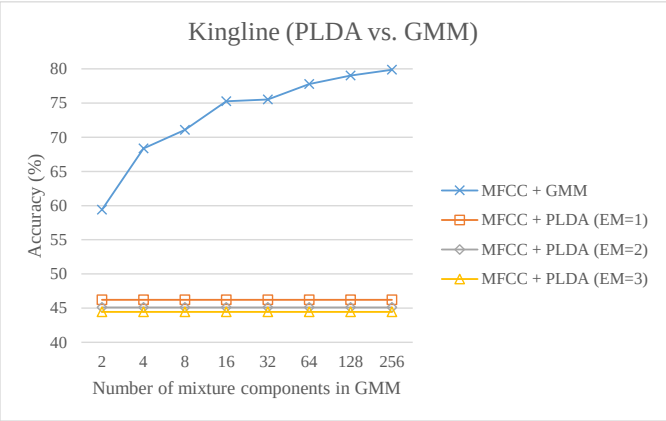


Fig. 9. PLDA with MFCC vector using Kingline081 dataset.

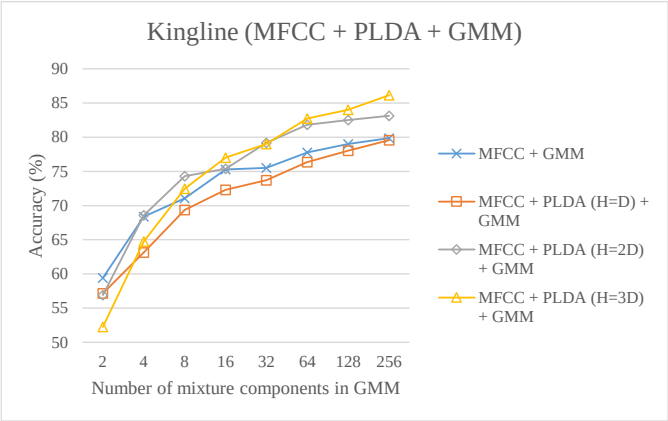


Fig. 11. PLDA for dimensionality expansion using Kingline081 dataset.

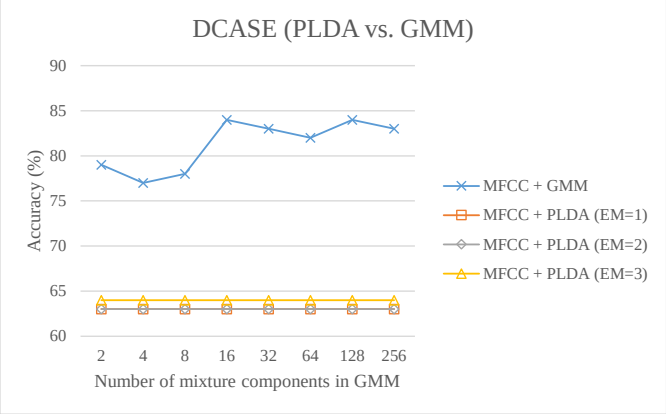


Fig. 10. PLDA with MFCC vector using DCASE2013 dataset.

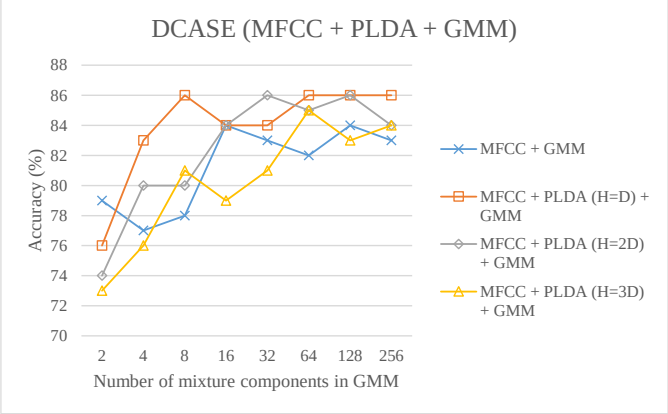


Fig. 12. PLDA for dimensionality expansion using DCASE2013 dataset.

demonstrates the effectiveness of using PLDA for feature dimensionality expansion. It is also noted that the transformed vector still follows a Gaussian distribution, as the performance of GMM improves as the number of mixture components increases. From Fig. 12, we may observe that, H=D gives the best performance. It seems increasing the dimensionality too much does not offer any help. Nevertheless, the transformed vector seems more Gaussian than the original MFCC vector, as the performance of GMM tends to improve with the increase of the number of mixture components. However, in general, the performance of GMM does not always improve with the increase of the number of mixture components, implying the MFCC vectors may not follow Gaussian distributions in this scenario. According to the model assumption of PLDA, the between-class latent variable is supposed to follow a Gaussian distribution theoretically. However, the transformed vector is only the expected value of the latent variable, which may not exactly follow the Gaussian distribution. In addition, the characteristics of the transformed vector also highly depend on the characteristics of the raw feature vector, i.e., the MFCC vector in this scenario.

Besides of directly using the low-dimensional MFCC vector, we may also adopt a neighboring feature concatenation strategy to increase the dimensionality of the raw feature vector. Suppose an acoustic sample is represented by T MFCC vectors denoted as $\{ \boldsymbol{x}_1, \boldsymbol{x}_2 \ldots \boldsymbol{x}_T \}$, the t-th concatenated MFCC vector is then given by $\boldsymbol{x}'_t = \left[\boldsymbol{x}^T_{(t-1)L/2+1} \quad \boldsymbol{x}^T_{(t-1)L/2+2} \quad \cdots \quad \boldsymbol{x}^T_{(t-1)L/2+L} \right]^T$, where L is the concatenation length and $L/2$ is the concatenation shift. If the dimensionality of an MFCC vector is D×1, the dimensionality of a concatenated vector will be DL×1. With the help of the concatenation operation, the raw feature vector has a higher dimensionality and thus may carry more information. However, the higher the dimensionality, the higher the chance of violating the Gaussian assumption. In this case, we may apply PLDA for dimensionality reduction.

Experimental results on DCASE2013 using the concatenated vector are shown in Fig. 13. The classifier is the GMM with 128 mixture components. The concatenation length L varies from 2 to 32 (i.e., the dimensionality D of the concatenated vector varies from 40×1 to 640×1), and the ratio H/D varies from 0.25 to 1. It can be seen from Fig. 13 that, in general the performance of GMM degrades with the increase of the feature dimensionality, as the higher the dimensionality, the higher the chance of violating the Gaussian distribution assumption. Nevertheless, by applying PLDA to the raw concatenated vectors to reduce the dimensionality (i.e., H=0.25D and H=0.5D) or simply keep the same dimensionality (i.e., H=D), the transformed vectors can have improved performance. This demonstrates the capability of PLDA as a dimensionality reduction technique.

When applying PLDA for feature transformation, it inevitably introduces extra time consumption costs. In Fig. 14,

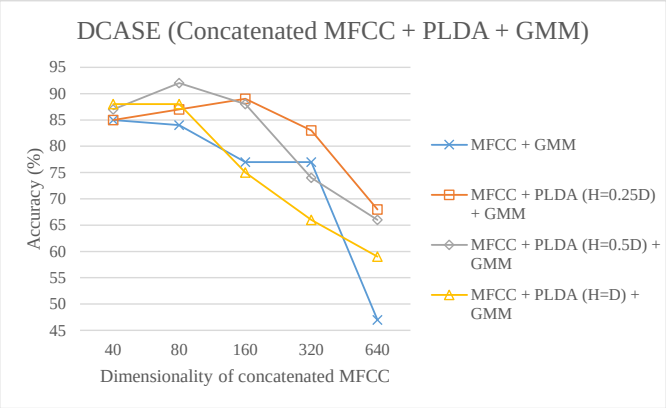


Fig. 13. PLDA for dimensionality reduction using DCASE2013 dataset.

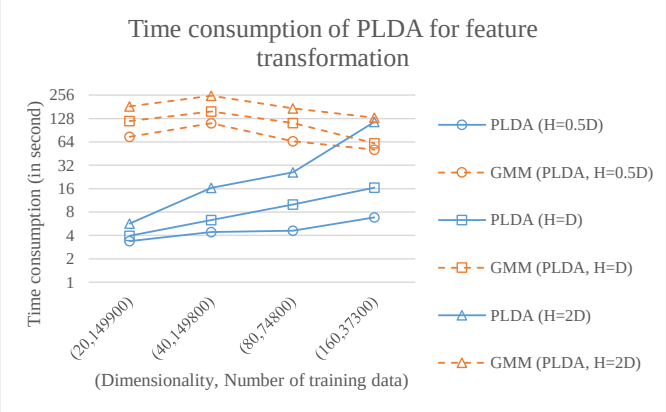


Fig. 14. Time consumption of PLDA-based feature transformation using DCASE2013 dataset.

we record the computation time of applying PLDA to the concatenated MFCC vectors. The dimensionality of the concatenated vector varies from 20×1 to 160×1, and the ratio *H*/*D* varies from 0.5 to 2. Because of the concatenation operation, the number of training vectors (equal to the number of testing vectors) varies from 149900 to 37300. We also record the computation time of constructing the 16-mixture GMM using the PLDA-based transformed vectors, just as a reference. It can be seen that, the computation of PLDA is very efficient when the feature dimensionality is low. However, the computational efficiency degrades with the increase of the feature dimensionality. Based on the formulations of PLDA, the computational complexity is proportional to the number of training data as well as the feature dimensionality. Therefore, the more the training data or the higher the feature dimensionality, the higher the time consumption will be. Nevertheless, when the dimensionality is not very high, the extra computation costs introduced by PLDA-based feature transformation is negligible as compared to the computation costs introduced by constructing GMM.

From these experimental results, it is observed that the PLDA-based feature transformation technique may offer some improvements over the MFCC vector, and the transformed vector can have a higher or lower dimensionality and at the same time keep approximately following the Gaussian distribution. The PLDA-based transformed vector may then be used for different purposes in place of MFCC, for example, to construct i-vector or other types of supervectors such as Gaussian Supervector (GSV) [36]. PLDA may also be used to perform dimensionality reduction in place of LDA or Nuisance Attribute Projection (NAP) [37], working on i-vector or GSV instead of MFCC. This provides a new way of applying PLDA for acoustic signal classification tasks, besides of simply as a classification model.

D. Potentiality of PLDA

In this part, we briefly discuss the potential applications of PLDA and briefly compare PLDA with some other classic methods.

The earlier application of PLDA is face verification and speaker verification, whose objective is to compare whether two feature vectors share the same between-class latent variable (i.e., whether the two feature vectors belong to the same class). This target can be achieved by computing the conditional probability [2] or using the likelihood ratio as a similarity score [19].

A verification task can be treated as a special case of a classification task, which usually involves the comparison between more than two feature vectors. With the scalable formulations proposed in this research, applying PLDA as a general-purpose classifier is feasible. There are also different prediction criteria to choose, such as the conditional probability and the pairwise conditional probability.

On the one hand, PLDA is a probabilistic model that can be used as a probability estimator. On the other hand, PLDA is also a latent variable model, which consists of latent variables. The estimated value of these latent variables can be used as new feature vectors, which enables PLDA to be used as a feature transformation technique.

A closely related feature transformation technique is Fisher Linear Discriminant Analysis (LDA), as the name implies. However, there are several differences between them. First, the model parameters of PLDA are estimated using the EM algorithm, whereas the model parameters of LDA are based on eigen-decomposition of the scatter matrices. Second, PLDA can be directly used for classification purposes as it can estimate probabilities, whereas LDA is merely a feature transformation technique and has to work together with an additional classifier [40][41]. Besides, the dimensionality of the LDA-based transformed vector cannot be higher than the original feature dimensionality, whereas the dimensionality of the PLDA-based transformed vector can be either higher or lower than the original feature dimensionality, depending on the size of the factor-loading matrix. Nevertheless, both aim at capturing the within-class and between-class characteristics, although in different ways. Therefore, to some extent, PLDA can be treated as a probabilistic version of LDA.

PLDA is also related to some other classic techniques. As can be seen from (1), when the dimensionality of the latent variables $\boldsymbol{h}_k$ and $\boldsymbol{w}_{kj}$ becomes zero, the model assumption of PLDA becomes that of the single Gaussian model. This indicates that PLDA is the generalization of the single Gaussian model. If the dimensionality of $\boldsymbol{h}_k$ is zero but the

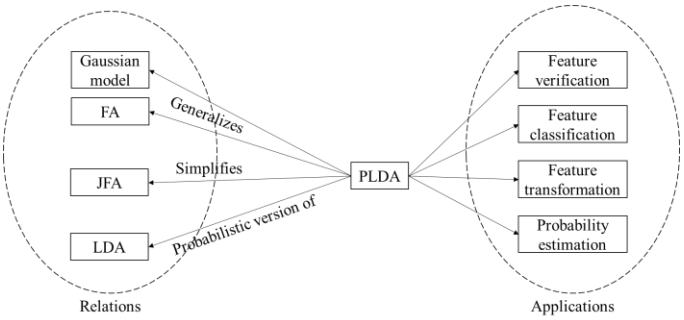


Fig. 15. Applications of PLDA and its connection with other techniques.

dimensionality of $\boldsymbol{w}_{kj}$ is nonzero, PLDA becomes Factor Analysis (FA). This indicates that PLDA is the generalization of FA. Since FA has been widely used as a dimensionality reduction technique [42][43], PLDA will also have such capability. If the expression of PLDA given by (1) includes more latent variables, PLDA can be further extended to Joint Factor Analysis (JFA) [12].

As can be seen from (1), the between-class latent variable $\boldsymbol{h}_k$ carries the class-dependent information, whereas the within-class latent variable $\boldsymbol{w}_{kj}$ carries the sample-dependent information. From this perspective, PLDA is a fusion of both supervised and unsupervised techniques. If the dimensionality of $\boldsymbol{h}_k$ is higher than that of $\boldsymbol{w}_{kj}$, PLDA is more supervised; if the dimensionality of $\boldsymbol{h}_k$ is lower than that of $\boldsymbol{w}_{kj}$, PLDA is more unsupervised. This property endows PLDA with more flexibility. An illustration of the potential applications of PLDA and its connection with other techniques is shown in Fig. 15.

In the future, we plan to extend the scalable formulations to the case of Mixture of PLDA (MPLDA). MPLDA is a collection of multiple PLDA models, which generalizes PLDA and thus can be more powerful [44][45]. In addition, in analogy to the relationship between PLDA, single Gaussian model and FA, MPLDA can be treated as the generalization of GMM and Mixture of Factor Analyzers (MFA) [46].

VII. Conclusion

In this paper, we investigate and try to improve the utility of PLDA for acoustic signal classification. Our major findings are summarized as follows.

First, we comprehensively analyze the formulations of PLDA, and explain the rationale and the core idea behind the model parameter estimation stage and the class label prediction stage. We find that, the original formulation of PLDA is inefficient when the number of training data is large. Therefore, we propose scalable formulations for PLDA, enabling it to make predictions efficiently.

In the scalable formulations, we propose different prediction criteria, which may improve its scalability and robustness. Some prediction criteria can be quite efficient with a large number of training data, but may fail if the number of training data is inadequate. While some prediction criteria can be quite robust even when the number of training data is limited, but may be inefficient with a large number of training data. Under different situations, different prediction criteria should be chosen, in order to maximize the capability of PLDA.

Second, we investigate the effectiveness of PLDA in different acoustic signal classification tasks, including speaker recognition and acoustic scene classification. We observe that, PLDA may not perform well when the dimensionality of the feature vector is low. This ineffectiveness also arises when the number of training data is inadequate, even if the dimensionality of the feature vector is high. These two observations may restrict the utility of PLDA in some scenarios, but they also indicate how to make PLDA effective, i.e., 1) a high dimensionality and 2) an enough number of training data. We believe that, understanding both the advantages and the disadvantages of PLDA may help better apply PLDA for different purposes.

Third, we introduce a novel application of PLDA, which applies it as a feature transformation technique. This technique simply uses the between-class latent variable in the PLDA model as the transformed vector. This transformed vector can have either a lower dimensionality or a higher dimensionality, which makes it more flexible. At the same time, the transformed vector still approximately follows the Gaussian distribution, so that in some cases it can be the substitute of MFCC. It is then promising to combine PLDA with other classification models as a feature pre-processing technique, instead of directly using it for class label prediction.

Finally, we discuss the potential applications of PLDA and its relationship with some classic techniques, such as the single Gaussian model, LDA, FA and JFA. PLDA can be treated as the generalization of the single Gaussian model and FA, and the simplification of JFA. It can also be treated as a probabilistic version of LDA. The potential extension of PLDA, namely, the Mixture of PLDA (MPLDA), further generalizes PLDA, GMM and MFA, and thus may have a wide range of applications and deserves further exploration.

Acknowledgments

The work described in this paper was substantially supported by a grant from The Hong Kong Polytechnic University (Project Account Code: RUG7).

Appendices

A. Supplements to the Conditional Probability

In this part, we derive the expressions of $\boldsymbol{P}^{(J)}$ and $\boldsymbol{Q}^{(J)}$ given by (28). By rearranging (27), we have (47).

$$\begin{bmatrix} \boldsymbol{\Phi}_{yy} & \boldsymbol{F}\boldsymbol{F}^T & \boldsymbol{F}\boldsymbol{F}^T & \cdots \\ \boldsymbol{F}\boldsymbol{F}^T & \boldsymbol{\Phi}_{yy} & \boldsymbol{F}\boldsymbol{F}^T & \cdots \\ \boldsymbol{F}\boldsymbol{F}^T & \boldsymbol{F}\boldsymbol{F}^T & \boldsymbol{\Phi}_{yy} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} \boldsymbol{P}^{(J)} & \boldsymbol{Q}^{(J)} & \boldsymbol{Q}^{(J)} & \cdots \\ \boldsymbol{Q}^{(J)} & \boldsymbol{P}^{(J)} & \boldsymbol{Q}^{(J)} & \cdots \\ \boldsymbol{Q}^{(J)} & \boldsymbol{Q}^{(J)} & \boldsymbol{P}^{(J)} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} & \cdots \\ \boldsymbol{0} & \boldsymbol{I} & \boldsymbol{0} & \cdots \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{47}$$

By equating the block matrices on the left side that produce $\boldsymbol{I}$, we obtain (48). By equating the block matrices on the left

side that produce $\mathbf{0}$, we obtain (49).

$$\boldsymbol{\Phi}_{yy}\boldsymbol{P}^{(J)} + (J-1)\boldsymbol{F}\boldsymbol{F}^T\boldsymbol{Q}^{(J)} = \boldsymbol{I} \tag{48}$$

$$\boldsymbol{\Phi}_{yy}\boldsymbol{Q}^{(J)} + \boldsymbol{F}\boldsymbol{F}^T\boldsymbol{P}^{(J)} + (J-2)\boldsymbol{F}\boldsymbol{F}^T\boldsymbol{Q}^{(J)} = \mathbf{0} \tag{49}$$

By expanding $\boldsymbol{\Phi}_{yy}$, (48) and (49) are equivalent to (50) and (51) respectively. Solving (50) and (51) then gives (28).

$$\left(\boldsymbol{F}\boldsymbol{F}^T + \boldsymbol{G}\boldsymbol{G}^T + \boldsymbol{\Sigma}\right)\boldsymbol{P}^{(J)} + (J-1)\boldsymbol{F}\boldsymbol{F}^T\boldsymbol{Q}^{(J)} = \boldsymbol{I} \tag{50}$$

$$\left(\boldsymbol{F}\boldsymbol{F}^T + \boldsymbol{G}\boldsymbol{G}^T + \boldsymbol{\Sigma}\right)\boldsymbol{Q}^{(J)} + \boldsymbol{F}\boldsymbol{F}^T\boldsymbol{P}^{(J)} + (J-2)\boldsymbol{F}\boldsymbol{F}^T\boldsymbol{Q}^{(J)} = \mathbf{0} \tag{51}$$

B. Supplements to the Joint Probability

In this part, we derive the expression of the joint probability $p(\boldsymbol{X}_k, \boldsymbol{y})$ given by (26). We first consider the determinant of the joint covariance $\boldsymbol{R}'\boldsymbol{R}'^T + \boldsymbol{\Phi}'$. As can be seen from (19), $\boldsymbol{R}'\boldsymbol{R}'^T + \boldsymbol{\Phi}'$ is namely $\boldsymbol{\Psi}_{J+1}$, which possess a very special structure. This special structure makes it possible to calculate the determinant using the induction method, together with the trick of the determinant of block matrices as given by (52) [38][39], where $\boldsymbol{B}_1 \sim \boldsymbol{B}_4$ are matrix blocks.

$$\begin{vmatrix} \boldsymbol{B}_1 & \boldsymbol{B}_2 \\ \boldsymbol{B}_3 & \boldsymbol{B}_4 \end{vmatrix} = \left|\boldsymbol{B}_1 - \boldsymbol{B}_2\boldsymbol{B}_4^{-1}\boldsymbol{B}_3\right| \cdot \left|\boldsymbol{B}_4\right| \tag{52}$$

The steps of the induction method are described below.

1) Step 1:

$$\boldsymbol{\Psi}_1 = \boldsymbol{\Phi}_{yy}, \left|\boldsymbol{\Psi}_1\right| = \left|\boldsymbol{\Phi}_{yy}\right|$$

2) Step 2:

$$\boldsymbol{\Psi}_2 = \begin{bmatrix} \boldsymbol{\Phi}_{yy} & \boldsymbol{F}\boldsymbol{F}^T \\ \boldsymbol{F}\boldsymbol{F}^T & \boldsymbol{\Psi}_1 \end{bmatrix}$$

$$\left|\boldsymbol{\Psi}_2\right| = \left|\boldsymbol{\Phi}_{yy} - \boldsymbol{F}\boldsymbol{F}^T\boldsymbol{\Psi}_1^{-1}\boldsymbol{F}\boldsymbol{F}^T\right| \cdot \left|\boldsymbol{\Psi}_1\right| = \left|\boldsymbol{\Phi}_{yy} - \boldsymbol{F}\boldsymbol{F}^T\boldsymbol{P}^{(1)}\boldsymbol{F}\boldsymbol{F}^T\right| \cdot \left|\boldsymbol{\Psi}_1\right|$$

3) Step 3:

$$\boldsymbol{\Psi}_3 = \begin{bmatrix} \boldsymbol{\Phi}_{yy} & \boldsymbol{F}\boldsymbol{F}^T & \boldsymbol{F}\boldsymbol{F}^T \\ \boldsymbol{F}\boldsymbol{F}^T & \begin{bmatrix} & \\ & \boldsymbol{\Psi}_2 \\ & \end{bmatrix} & \\ \boldsymbol{F}\boldsymbol{F}^T & & \end{bmatrix}$$

$$\begin{aligned} \left|\boldsymbol{\Psi}_3\right| &= \left|\boldsymbol{\Phi}_{yy} - \begin{bmatrix} \boldsymbol{F}\boldsymbol{F}^T & \boldsymbol{F}\boldsymbol{F}^T \end{bmatrix}\boldsymbol{\Psi}_2^{-1}\begin{bmatrix} \boldsymbol{F}\boldsymbol{F}^T \\ \boldsymbol{F}\boldsymbol{F}^T \end{bmatrix}\right| \cdot \left|\boldsymbol{\Psi}_2\right| \\ &= \left|\boldsymbol{\Phi}_{yy} - \begin{bmatrix} \boldsymbol{F}\boldsymbol{F}^T & \boldsymbol{F}\boldsymbol{F}^T \end{bmatrix}\begin{bmatrix} \boldsymbol{P}^{(2)} & \boldsymbol{Q}^{(2)} \\ \boldsymbol{Q}^{(2)} & \boldsymbol{P}^{(2)} \end{bmatrix}\begin{bmatrix} \boldsymbol{F}\boldsymbol{F}^T \\ \boldsymbol{F}\boldsymbol{F}^T \end{bmatrix}\right| \cdot \left|\boldsymbol{\Psi}_2\right| \\ &= \left|\boldsymbol{\Phi}_{yy} - 2\boldsymbol{F}\boldsymbol{F}^T(\boldsymbol{P}^{(2)} + \boldsymbol{Q}^{(2)})\boldsymbol{F}\boldsymbol{F}^T\right| \cdot \left|\boldsymbol{\Psi}_2\right| \end{aligned}$$

4) Step 4:

$$\boldsymbol{\Psi}_4 = \begin{bmatrix} \boldsymbol{\Phi}_{yy} & \boldsymbol{F}\boldsymbol{F}^T & \boldsymbol{F}\boldsymbol{F}^T & \boldsymbol{F}\boldsymbol{F}^T \\ \boldsymbol{F}\boldsymbol{F}^T & \begin{bmatrix} & \\ & \boldsymbol{\Psi}_3 \\ & \end{bmatrix} & & \\ \boldsymbol{F}\boldsymbol{F}^T & & & \\ \boldsymbol{F}\boldsymbol{F}^T & & & \end{bmatrix}$$

$$\begin{aligned} \left|\boldsymbol{\Psi}_4\right| &= \left|\boldsymbol{\Phi}_{yy} - \begin{bmatrix} \boldsymbol{F}\boldsymbol{F}^T & \boldsymbol{F}\boldsymbol{F}^T & \boldsymbol{F}\boldsymbol{F}^T \end{bmatrix}\boldsymbol{\Psi}_3^{-1}\begin{bmatrix} \boldsymbol{F}\boldsymbol{F}^T \\ \boldsymbol{F}\boldsymbol{F}^T \\ \boldsymbol{F}\boldsymbol{F}^T \end{bmatrix}\right| \cdot \left|\boldsymbol{\Psi}_3\right| \\ &= \left|\boldsymbol{\Phi}_{yy} - \begin{bmatrix} \boldsymbol{F}\boldsymbol{F}^T & \boldsymbol{F}\boldsymbol{F}^T & \boldsymbol{F}\boldsymbol{F}^T \end{bmatrix}\begin{bmatrix} \boldsymbol{P}^{(3)} & \boldsymbol{Q}^{(3)} & \boldsymbol{Q}^{(3)} \\ \boldsymbol{Q}^{(3)} & \boldsymbol{P}^{(3)} & \boldsymbol{Q}^{(3)} \\ \boldsymbol{Q}^{(3)} & \boldsymbol{Q}^{(3)} & \boldsymbol{P}^{(3)} \end{bmatrix}\begin{bmatrix} \boldsymbol{F}\boldsymbol{F}^T \\ \boldsymbol{F}\boldsymbol{F}^T \\ \boldsymbol{F}\boldsymbol{F}^T \end{bmatrix}\right| \cdot \left|\boldsymbol{\Psi}_3\right| \\ &= \left|\boldsymbol{\Phi}_{yy} - 3\boldsymbol{F}\boldsymbol{F}^T(\boldsymbol{P}^{(3)} + 2\boldsymbol{Q}^{(3)})\boldsymbol{F}\boldsymbol{F}^T\right| \cdot \left|\boldsymbol{\Psi}_3\right| \end{aligned}$$

5) Step J+1:

$$\boldsymbol{\Psi}_{J+1} = \begin{bmatrix} \boldsymbol{\Phi}_{yy} & \boldsymbol{F}\boldsymbol{F}^T & \cdots & \boldsymbol{F}\boldsymbol{F}^T \\ \boldsymbol{F}\boldsymbol{F}^T & \begin{bmatrix} & \\ & \boldsymbol{\Psi}_J \\ & \end{bmatrix} & & \\ \vdots & & & \\ \boldsymbol{F}\boldsymbol{F}^T & & & \end{bmatrix} \tag{53}$$

$$\left|\boldsymbol{\Psi}_{J+1}\right| = \left|\boldsymbol{\Phi}_{yy} - J\boldsymbol{F}\boldsymbol{F}^T\left(\boldsymbol{P}^{(J)} + (J-1)\boldsymbol{Q}^{(J)}\right)\boldsymbol{F}\boldsymbol{F}^T\right| \cdot \left|\boldsymbol{\Psi}_J\right|$$

Using the above induction method, the determinant of the joint covariance $\boldsymbol{R}'\boldsymbol{R}'^T + \boldsymbol{\Phi}'$ can be calculated step by step, which is namely $\left|\boldsymbol{\Psi}_{J+1}\right|$ in (53). The next step is to calculate the exponential term in (26). The matrix product inside the exponential term can be expanded as follows.

$$
\begin{aligned}
&\begin{bmatrix} \boldsymbol{X}_k - \boldsymbol{U} \\ \boldsymbol{y} - \boldsymbol{\mu} \end{bmatrix}^T (\boldsymbol{R}'\boldsymbol{R}'^T + \boldsymbol{\Phi}')^{-1} \begin{bmatrix} \boldsymbol{X}_k - \boldsymbol{U} \\ \boldsymbol{y} - \boldsymbol{\mu} \end{bmatrix} \\
&= \begin{bmatrix} \boldsymbol{X}_k - \boldsymbol{U} \\ \boldsymbol{y} - \boldsymbol{\mu} \end{bmatrix}^T \boldsymbol{\Psi}_{J+1}^{-1} \begin{bmatrix} \boldsymbol{X}_k - \boldsymbol{U} \\ \boldsymbol{y} - \boldsymbol{\mu} \end{bmatrix} \\
&= \begin{bmatrix} \boldsymbol{x}_{k1} - \boldsymbol{\mu} \\ \vdots \\ \boldsymbol{x}_{kJ} - \boldsymbol{\mu} \\ \boldsymbol{y} - \boldsymbol{\mu} \end{bmatrix}^T \begin{bmatrix} \boldsymbol{P}^{(J+1)} & \boldsymbol{Q}^{(J+1)} & \boldsymbol{Q}^{(J+1)} & \cdots \\ \boldsymbol{Q}^{(J+1)} & \boldsymbol{P}^{(J+1)} & \boldsymbol{Q}^{(J+1)} & \cdots \\ \boldsymbol{Q}^{(J+1)} & \boldsymbol{Q}^{(J+1)} & \boldsymbol{P}^{(J+1)} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_{k1} - \boldsymbol{\mu} \\ \vdots \\ \boldsymbol{x}_{kJ} - \boldsymbol{\mu} \\ \boldsymbol{y} - \boldsymbol{\mu} \end{bmatrix} \\
&= (\boldsymbol{x}_{k1} - \boldsymbol{\mu})^T \boldsymbol{P}^{(J+1)} (\boldsymbol{x}_{k1} - \boldsymbol{\mu}) + \sum_{j \neq 1} (\boldsymbol{x}_{kj} - \boldsymbol{\mu})^T \boldsymbol{Q}^{(J+1)} (\boldsymbol{x}_{k1} - \boldsymbol{\mu}) + \\
&\qquad (\boldsymbol{y} - \boldsymbol{\mu})^T \boldsymbol{Q}^{(J+1)} (\boldsymbol{x}_{k1} - \boldsymbol{\mu}) \\
&+ (\boldsymbol{x}_{k2} - \boldsymbol{\mu})^T \boldsymbol{P}^{(J+1)} (\boldsymbol{x}_{k2} - \boldsymbol{\mu}) + \sum_{j \neq 2} (\boldsymbol{x}_{kj} - \boldsymbol{\mu})^T \boldsymbol{Q}^{(J+1)} (\boldsymbol{x}_{k2} - \boldsymbol{\mu}) + \\
&\qquad (\boldsymbol{y} - \boldsymbol{\mu})^T \boldsymbol{Q}^{(J+1)} (\boldsymbol{x}_{k2} - \boldsymbol{\mu}) \\
&+ \quad \cdots \\
&+ (\boldsymbol{x}_{kJ} - \boldsymbol{\mu})^T \boldsymbol{P}^{(J+1)} (\boldsymbol{x}_{kJ} - \boldsymbol{\mu}) + \sum_{j \neq J} (\boldsymbol{x}_{kj} - \boldsymbol{\mu})^T \boldsymbol{Q}^{(J+1)} (\boldsymbol{x}_{kJ} - \boldsymbol{\mu}) + \\
&\qquad (\boldsymbol{y} - \boldsymbol{\mu})^T \boldsymbol{Q}^{(J+1)} (\boldsymbol{x}_{kJ} - \boldsymbol{\mu}) \\
&+ \sum_j (\boldsymbol{x}_{kj} - \boldsymbol{\mu})^T \boldsymbol{Q}^{(J+1)} (\boldsymbol{y} - \boldsymbol{\mu}) + (\boldsymbol{y} - \boldsymbol{\mu})^T \boldsymbol{P}^{(J+1)} (\boldsymbol{y} - \boldsymbol{\mu}) \\
&= \sum_j (\boldsymbol{x}_{kj} - \boldsymbol{\mu})^T \boldsymbol{P}^{(J+1)} (\boldsymbol{x}_{kj} - \boldsymbol{\mu}) + \sum_i \sum_{j \neq i} (\boldsymbol{x}_{kj} - \boldsymbol{\mu})^T \boldsymbol{Q}^{(J+1)} (\boldsymbol{x}_{ki} - \boldsymbol{\mu}) \\
&\quad + (\boldsymbol{y} - \boldsymbol{\mu})^T \sum_j \boldsymbol{Q}^{(J+1)} (\boldsymbol{x}_{kj} - \boldsymbol{\mu}) + \sum_j (\boldsymbol{x}_{kj} - \boldsymbol{\mu})^T \boldsymbol{Q}^{(J+1)} (\boldsymbol{y} - \boldsymbol{\mu}) \\
&\quad + (\boldsymbol{y} - \boldsymbol{\mu})^T \boldsymbol{P}^{(J+1)} (\boldsymbol{y} - \boldsymbol{\mu}) \\
&= \sum_j (\boldsymbol{x}_{kj} - \boldsymbol{\mu})^T \boldsymbol{P}^{(J+1)} (\boldsymbol{x}_{kj} - \boldsymbol{\mu}) + \sum_i \sum_{j \neq i} (\boldsymbol{x}_{kj} - \boldsymbol{\mu})^T \boldsymbol{Q}^{(J+1)} (\boldsymbol{x}_{ki} - \boldsymbol{\mu}) \\
&\quad + 2 \sum_j (\boldsymbol{x}_{kj} - \boldsymbol{\mu})^T \boldsymbol{Q}^{(J+1)} (\boldsymbol{y} - \boldsymbol{\mu}) + (\boldsymbol{y} - \boldsymbol{\mu})^T \boldsymbol{P}^{(J+1)} (\boldsymbol{y} - \boldsymbol{\mu})
\end{aligned}
\tag{54}
$$

If using a scalar $\Delta_k(y)$ to denote the result in (54), together with the determinant given by (53), the joint probability is then given by (55).

$$
p(\boldsymbol{y}, \boldsymbol{X}_k) = \frac{1}{(2\pi)^{(J+1)D/2} \left| \boldsymbol{\Psi}_{J+1} \right|^{1/2}} \exp\left(-\frac{\Delta_k(y)}{2} \right) \tag{55}
$$

REFERENCES

[1] S. J. D. Prince and J. H. Elder, "Probabilistic linear discriminant analysis for inferences about identity," in *Proc. IEEE Int. Conf. on Computer Vision (ICCV)*, 2007, pp. 1-8.

[2] P. Li, Y. Fu, U. Mohammed, J. H. Elder, and S. J. D. Prince, "Probabilistic models for inference about identity," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 34, no. 1, pp. 144-157, 2012.

[3] L. Liu, C. Xiong, H. Zhang, Z. Niu, M. Wang, and S. Yan, "Deep aging face verification with large gaps," *IEEE Trans. on Multimedia*, vol. 18, no. 1, pp. 64-75, 2016.

[4] H. Zhou and K. M. Lam, "Age-invariant face recognition based on identity inference from appearance age," *Pattern Recognition*, vol. 76, pp. 191-202, 2018.

[5] C. Ding, J. Choi, D. Tao, and L. S. Davis, "Multi-directional multi-level dual-cross patterns for robust face recognition," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 38, no. 3, pp. 518-531, 2016.

[6] K. Anantharajah *et al.*, "Local inter-session variability modeling for object classification," in *Proc. IEEE Conf. on Applications of Computer Vision*, 2014, pp. 309-316.

[7] M. E. Wibowo, D. Tjondronegoro, V. Chandran, R. Pulungan, and J. E. Istiyanto, "Improved face recognition across poses using fusion of probabilistic latent variable models," *Telkomnika*, vol. 15, no. 4, pp. 1976-1986, 2017.

[8] A. Fabris, M. A. Nicolaou, I. Kotsia, and S. Zafeiriou, "Dynamic probabilistic linear discriminant analysis for video classification," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 2781-2785.

[9] D. Roy, K. S. R. Murty, and C. K. Mohan, "Unsupervised universal attribute modeling for action recognition," *IEEE Trans. on Multimedia*, vol. 21, no. 7, pp. 1672-1680, 2019.

[10] D. Garcia-Romero and C. Y. Espy-Wilson, "Analysis of i-vector length normalization in speaker recognition systems," in *Proc. Annual Conf. of the International Speech Communication Association (INTERSPEECH)*, 2011, pp. 249-252.

[11] L. Ferrer and M. McLaren, "Joint plda for simultaneous modeling of two factors," *Journal of Machine Learning Research*, vol. 20, no. 24, pp. 1-29, 2019.

[12] J. H. L. Hansen and T. Hasan, "Speaker recognition by machines and humans: a tutorial review," *IEEE Signal Processing Magazine*, vol. 32, no. 6, pp. 74-99, 2015.

[13] C. Chen, W. Wang, Y. He, and J. Han, "A bilevel framework for joint optimization of session compensation and classification for speaker identification," *Digital Signal Processing*, vol. 89, pp. 104-115, 2019.

[14] C. Hanilci, "Data selection for i-vector based automatic speaker verification anti-spoofing," *Digital Signal Processing*, vol. 72, pp. 171-180, 2018.

[15] V. Vestman, B. Soomro, A. Kanervisto, V. Hautamaki, and T. Kinnunen, "Who do I sound like? Showcasing speaking recognition technology by youtube voice search," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 5781-5785.

[16] G. Sell, K. Duh, D. Snyder, D. Etter, and D. Garcia-Romero, "Audio-visual person recognition in multimedia data from the IARPA Janus program," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 3031-3035.

[17] S. E. Shepstone, Z. H. Tan, and S. H. Jensen, "Using audio-derived affective offset to enhance tv recommendation," *IEEE Trans. on Multimedia*, vol. 16, no. 7, pp. 1999-2010, 2014.

[18] L. E. Shafey, C. McCool, R. Wallace, and S. Marcel, "A scalable formulation of probabilistic linear discriminant analysis: applied to face recognition," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 35, no. 7, pp. 1788-1794, 2013.

[19] Y. Jiang, K. A. Lee, Z. Tang, B. Ma, A. Larcher, and H. Li, "PLDA modeling in i-vector and supervector space for speaker verification," in *Proc. Annual Conference of the International Speech Communication Association (INTERSPEECH)*, 2012, pp. 1680-1683.

[20] P. Rajan, A. Afanasyev, V. Hautamaki, 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.

[21] C. M. Bishop, "Linear models for classification," in *Pattern Recognition and Machine Learning*, Springer, 2006, ch. 4, pp. 179-224.

[22] C. M. Bishop, "Continuous latent variables," in *Pattern Recognition and Machine Learning*, Springer, 2006, ch. 12, pp. 559-603.

[23] S. J. D. Prince, J. H. Elder, J. Warrell, and F. M. Felisberti, "Tied factor analysis for face recognition across large pose differences," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 30, no. 6, pp. 970-984, 2008.

[24] S. Ioffe, "Probabilistic linear discriminant analysis," in *Proc. European Conf. on Computer Vision (ECCV)*, 2006, pp. 531-542.

[25] Y. Jiang and F. H. F. Leung, "The scalable version of probabilistic linear discriminant analysis and its potential as a classifier for audio signal classification," in *Proc. IEEE Int. Joint Conf. on Neural Networks (IJCNN)*, 2018, pp. 1-7.

[26] C. M. Bishop, "Probability distributions," in *Pattern Recognition and Machine Learning*, Springer, 2006, ch. 2, pp. 67-136.

[27] X. Huang, A. Acero, and H. W. Hon, "Speech signal representations," in *Spoken Language Processing: A Guide to Theory, Algorithm and System Development*. Upper Saddle River, NJ: Prentice Hall PTR, 2001, ch. 6, pp. 273-333.

[28] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788-798, 2011.
[29] P. Kenny, G. Boulianne, and P. Dumouchel, "Eigenvoice modeling with sparse training data," *IEEE Trans. on Speech and Audio Processing*, vol. 13, no. 3, pp. 345-354, 2005.
[30] Y. Jiang and F. H. F. Leung, "A class-dependent background model for speech signal feature extraction," in *Proc. IEEE Int. Conf. on Digital Signal Processing (DSP)*, 2018.
[31] D. Reynolds, "Gaussian mixture models," in *Encyclopedia of Biometrics*, 2015, pp. 827-832.
[32] D. A. Reynolds, "Robust text-independent speaker identification using Gaussian mixture speaker models," *IEEE Trans. on Speech and Audio Processing*, vol. 3, no. 1, pp. 72-83, 1995.
[33] KingLine Data Center, American English Speech Recognition Corpus (King-ASR-L-081), Speechocean, 2013.
[34] D. Stowell, D. Giannoulis, E. Benetos, M. Lagrange, and M. D. Plumbley, "Detection and classification of acoustic scenes and events," *IEEE Trans. on Multimedia*, vol. 17, no. 10, pp. 1733-1746, 2015.
[35] C. C. Chang and C. J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. on Intelligent Systems and Technology*, vol. 2, no. 3, pp. 1-27, 2011.
[36] W. M. Campbell, D. E. Sturim, and D. A. Reynolds, "Support vector machines using GMM supervectors for speaker verification," *IEEE Signal Processing Letters*, vol. 13, no. 5, pp. 308-311, 2006.
[37] A. Solomonoff, W. M. Campbell, and I. Boardman, "Advances in channel compensation for SVM speaker recognition," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, 2005, pp. 629-632.
[38] J. R. Silvester, "Determinants of block matrices," *Mathematical Gazette*, vol. 84, no. 501, pp. 460-467, 2000.
[39] P. D. Powell, "Calculating determinants of block matrices," *arXiv preprint*, arXiv:1112.4379, 2011.
[40] A. Glowacz et al., "Detection of deterioration of three-phase induction motor using vibration signals," *Measurement Science Review*, vol. 19, no. 6, pp. 241-249, 2019.
[41] A. Glowacz, "Fault diagnostics of acoustic signals of loaded synchronous motor using SMOFS-25-EXPANDED and selected classifiers," *Tehnički vjesnik*, vol. 23, no. 5, pp. 1365-1372, 2016.
[42] H. Zhao, Z. Li, S. Zhu, and Y. Yu, "Valve internal leakage rate quantification based on factor analysis and wavelet-BP neural network using acoustic emission," *Applied sciences*, vol. 10, no. 16, 2020.
[43] S. B. Zhu, Z. L. Li, S. M. Zhang, L. L. Liang, and H. F. Zhang, "Natural gas pipeline valve leakage rate estimation via factor and cluster analysis of acoustic emissions," *Measurement*, vol. 125, pp. 48-55, 2018.
[44] M. W. Mak, X. M. Pang, and J. T. Chien, "Mixture of PLDA for noise robust i-vector speaker verification," *IEEE/ACM Trans. on Audio Speech and Language Processing*, vol. 24, no. 1, pp. 132-142, 2016.
[45] Y. Liu, J. Zeng, L. Xie, X. Lang, S. Luo, and H. Su, "An improved mixture robust probabilistic linear discriminant analyzer for fault classification," *ISA Transactions*, vol. 98, pp. 227-236, 2020.
[46] Y. Jiang and F. H. F. Leung, "Vector-based feature representations for speech signals: from supervector to latent vector", *IEEE Trans. on Multimedia*, 2020.

Yuechi Jiang received the BEng degree in Electronic Engineering from The Chinese University of Hong Kong in 2015. He is currently a PhD candidate in the department of Electronic and Information Engineering, The Hong Kong Polytechnic University. He has published more than 10 journal papers and conference papers on digital signal processing and pattern recognition. His research interests include acoustic signal processing and pattern recognition. He is a student member of IEEE.

Frank H. F. Leung received the BEng degree and the PhD degree in Electronic Engineering from the Hong Kong Polytechnic in 1988 and 1992, respectively. He is currently an associate professor in the department of Electronic and Information Engineering, The Hong Kong Polytechnic University. He is an active researcher who has published over 210 research papers on Computational Intelligence, Machine Learning, Control, and Power Electronics. He has been serving as editor, guest editor and reviewer for international journals. He is a Chartered Engineer, a corporate member of IET (UK) and the Hong Kong Institution of Engineers, and a senior member of IEEE.