Coupling a generative model with a discriminative learning framework for speaker verification

Xugang Lu[1], Peng Shen[1], Yu Tsao[2], Hisashi Kawai[1]

Abstract— **The task of speaker verification (SV) is to decide whether an utterance is spoken by a target or an imposter speaker. In most studies of SV, a log-likelihood ratio (LLR) score is estimated based on a generative probability model on speaker features, and compared with a threshold for making a decision. However, the generative model usually focuses on individual feature distributions, does not have the discriminative feature selection ability, and is easy to be distracted by nuisance features. The SV, as a hypothesis test, could be formulated as a binary discrimination task where neural network based discriminative learning could be applied. In discriminative learning, the nuisance features could be removed with the help of label supervision. However, discriminative learning pays more attention to classification boundaries, and is prone to overfitting to a training set which may result in bad generalization on a test set. In this paper, we propose a hybrid learning framework, i.e., coupling a joint Bayesian (JB) generative model structure and parameters with a neural discriminative learning framework for SV. In the hybrid framework, a two-branch Siamese neural network is built with dense layers that are coupled with factorized affine transforms as used in the JB model. The LLR score estimation in the JB model is formulated according to the distance metric in the discriminative learning framework. By initializing the two-branch neural network with the generatively learned model parameters of the JB model, we further train the model parameters with the pairwise samples as a binary discrimination task. Moreover, a direct evaluation metric (DEM) in SV based on minimum empirical Bayes risk (EBR) is designed and integrated as an objective function in the discriminative learning. We carried out SV experiments on Speakers in the wild (SITW) and Voxceleb. Experimental results showed that our proposed model improved the performance with a large margin compared with state of the art models for SV.**

I. Introduction

Speaker verification (SV) is a technique to verify whether an acoustic speech is spoken by a target or an imposter speaker. SV is widely used in many speech application systems where speaker information is required from authentication or security perspectives [1], [2], [3]. The basic problem definition of SV is to decide whether two utterances (usually denoted as test and enrollment utterances) are generated from the same or different speakers, i.e., a hypothesis test defined as:

$$\begin{array}{l} H_S : \mathbf{x}_i, \mathbf{x}_j \text{ are spoken by the same speaker} \\ H_D : \mathbf{x}_i, \mathbf{x}_j \text{ are spoken by different speakers} \end{array} \tag{1}$$

where H_S and H_D are the two hypotheses as the same and different speaker spaces, respectively. $(\mathbf{x}_i, \mathbf{x}_j)$ is a tuple with two compared utterances indexed by i and j. For making a decision, it is necessary to estimate the similarity of the two utterances, either calculated as a log likelihood ratio (LLR) or a distance metric measure, and compare it with a threshold. The conventional pipeline in constructing a SV system for doing the hypothesis test defined in Eq. (1) is composed of front-end speaker feature extraction and backend speaker classifier modeling. Front-end feature extraction tries to extract robust and discriminative features to represent speakers, and backend classifier tries to model speakers with the extracted features based on which the similarity or LLR scores could be estimated.

1. National Institute of Information and Communications Technology, Japan. Email: xugang.lu@nict.go.jp

2. Research Center for Information Technology Innovation, Academic Sinica, Taiwan

A. Front-end speaker feature extraction

Historically, in most state of the art frameworks, the front-end speaker feature was based on i-vector representation [3]. In i-vector extraction, speech utterances with variable durations can be converted to fixed dimension vectors with the help of Gaussian mixture models (GMM) on probability distributions of acoustic features. With the resurgence of deep learning techniques, several alternative speaker features have been proposed, e.g., d-vector [4] and X-vector [5]. These features are extracted from a well trained deep neural network with bottleneck layers or statistical pooling. In recent years, X-vector as one of the speaker embedding representations is widely used in most state of the art frameworks [5]. The advantage of X-vector representation is that the model for X-vector extraction could be efficiently trained with a large quantity of speech samples from various speakers. Moreover, in order to explore robust speaker information, data augmentation with various noise types and signal to noise ratios (SNRs) could be easily applied in model training [5]. Since the original front-end feature (e.g., either i-vector or X-vector) encodes various acoustic factors, e.g., speaker factor, channel transmission factor, recording device factor, etc., before classifier modeling, a linear discriminative analysis (LDA), or a local fisher discriminative analysis [6], [7], is usually applied for dimension reduction to eliminate non-speaker specific information.

B. Backend classifier modeling

After speaker features are obtained, how to build a speaker classifier model in backend modeling for SV is important. There are two types of modeling approaches, one is generative modelling, the other is discriminative modeling. In generative modeling, features are regarded as observations from a generation process with certain probability distribution assumptions on the generation variables. Based on the generation model, the hypothesis test defined in Eq. (1) is regarded as a statistical inference from the variable probability distributions. For example, probabilistic linear discriminant analysis (PLDA) modeling was originally proposed for face recognition in [8], and was later improved with many variants for biometric authentication [9]. It has been widely used in SV for building classifier or backend models [10], [11]. PLDA can be applied to model the within-speaker and between-speaker variabilities with linear subspace modeling on speaker and noise spaces in generation. However, it is difficult to determine the dimensions of subspaces, which has a large effect on the final performance. As an alternative, joint Bayesian (JB) modelling [12], [13], which is without subspace model assumptions on speaker and noise spaces, is regarded as a much more efficient model than PLDA. Besides using different modelling assumption from that used in PLDA, JB has a quick convergence speed and accuracy in model parameter estimation with expectation-maximization (EM) iterations [12], [13], [14]. The other approach in backend modeling is discriminative modeling. In the early stage, cosine distance metric as a measure of similarity between two compared speaker embedding features was widely used [3]. With proper speaker feature extractions, the performance based on cosine distance may outperform the PLDA based backend modeling [15]. However, the scores estimated based on cosine distance metric need a lot of careful post processing, for example, score normalization or imposter cohort selection. For unknown or unconstraint environments or conditions, the generative probabilistic models are much more suitable for capturing the latent variations of the acoustic environments. Since the hypothesis test defined in Eq. (1) also can be formulated as a binary classification task, a discriminative modeling approach can be applied with supervised learning algorithms. For example, support vector machines (SVM) were proposed to maximize the between class distance [16], [17], and a neural network based discriminative model was applied to directly maximize classification accuracy with labeled training data sets [18]. The first explicit discriminative training based on pairwise i-vector features was proposed as a binary classification task for SV in [19], and later the idea was further developed to connect the PLDA based scoring to kernel classifier in pairwise i-vector space [20]. In recent years, as a discriminative modeling approach, the supervised end-to-end speaker models which integrate the front-end feature extraction and backend speaker classifier modeling in a unified optimization framework also have been proposed [21], [22]. However, in SV tasks, usually many speakers are not registered in the training data, and test utterances may be recorded from different sessions and environments, so it is difficult for the supervised discriminative modeling to work well if the training and testing conditions are not matched. To deal with the unmatched conditions, several backend algorithms have been proposed [23], [24]. No matter how successful the neural network based discriminative modeling in speech and image, current state of the art pipeline for SV is still the speaker embedding feature extraction (e.g. X-vector) with a generative speaker classifier modeling.

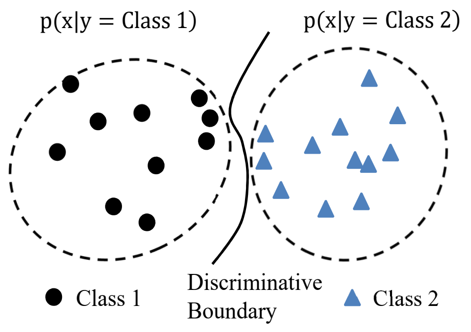

Fig. 1. Generative model learning focuses on class conditional feature distributions (dashed-circles of feature distribution shapes), and discriminative model learning emphasizes the class discriminative boundary (solid curve).

C. Our focus: hybrid generative and discriminative approach for backend modeling

In this study, we focus on backend modeling in SV tasks. We first summarize the different focus of the generative and discriminative modeling approaches as reviewed in section I-B. For a better understanding, a two-class classification task is illustrated in Fig. 1 (circles and triangles for classes 1 and 2 respectively). As shown in this figure, the generative model tries to focus on class conditional feature distributions while the discriminative model tries to pay attention to the classification boundary (solid curve in Fig. 1). As a generative modelling approach, either with PLDA or JB models, prior probability distributions of variables are assumed. If the assumptions are not satisfied, the performance could not be guaranteed. Moreover, it is difficult for the generative model approach to learn data structures in a high dimensional space with complex distributions. And the model does not have the discriminative feature selection ability which may be easily distracted by nuisance features in learning. On the other hand, the discriminative model approach could learn the complex classification boundary with a strong ability to remove nuisance features, but is prone to overfitting to the training data with over estimated label confidence in training. In this study, we try to explicitly integrate both the advantages of generative and discriminative modeling approaches in a unified learning framework for SV. The idea of discriminative training with a generative model scoring is not new [19], [20], [25], [26], the novelty of our work lies in the way of how to exactly coupling the generative model parameters in a discriminative learning framework. Besides, after coupling the generative model parameters in a discriminative learning framework, direct evaluation metrics could be designed as learning objective functions. Our contributions are summarized as follows:

(1) We propose a unified neural network backend framework for SV which couples the JB based generative model parameters in a discriminative learning framework. Although hybrid generative and discriminative modelling has been studied in machine learning for fully utilizing labeled and unlabeled samples, and showed improved performance in classification tasks [27], it is difficult to integrate the generative and discriminative models in SV tasks. The main reason is that in most studies the generative and discriminative models adopted different

modeling structures. In this study, we take the matrix structure of the generative JB model into consideration during the design of a neural network based discriminative modeling framework.

(2) We design a direct evaluation metric based learning objective function which keeps consistency of using the same evaluation metric in both training and testing. In the JB based generative model learning, usually an objective function with negative log-likelihood is minimized, while in a neural network based discriminative model learning, an objective function indicating the classification error rate is minimized. However, the objective for the hypothesis test in SV is different from any of them. In a SV task, the evaluation metric is based on weighting two types of errors (miss and false alarm) [28], [29]. In this study, we formulate this type of objective function in the discriminative learning framework.

(3) We analyze the effects of all components in model structure and parameterizations with detailed SV experiments, and reveal their connections to conventional distance metric learning.

The remainder of the paper is organized as follows. Section II introduces the basic theoretical considerations and the proposed hybrid model framework. Section III describes the implementation details and experiments; in particular, we make deep investigations of the effect of model parameters, and their connections to other related model frameworks. Section IV further checks the effectiveness of the proposed framework on another advanced speaker feature extraction baseline. Conclusions and future work are given in Section V.

II. Proposed hybrid model framework

The generative and discriminative models can be connected with the Bayes theory. Before introducing their connections, we give a brief review of generative and discriminative models.

A. Generative and discriminative models in classification tasks

A generative model tries to capture the data generation process with a fully joint modelling of the relation between feature input and label variables as $p(\mathbf{x}, y)$, while a discriminative model only tries to model the direct relation between input feature and output label as $p(y|\mathbf{x})$, where $\mathbf{x}$ and y are feature and label variables, respectively. Although the generative model is not directly used for classification, a classification model can be deduced from the generative model as model inference based on the Bayes theory as:

$$p(y|\mathbf{x}) = \frac{p(\mathbf{x}, y)}{p(\mathbf{x})} = \frac{p(\mathbf{x}|y)\,p(y)}{p(\mathbf{x})}, \tag{2}$$

where $p(\mathbf{x}|y)$ is the likelihood score of generating feature $\mathbf{x}$ by given a label y. In practical model parameter learning, generative model parameters usually are estimated based on expectation-maximization (EM) like algorithms while discriminative model parameters (neural network) usually are estimated based on gradient descent algorithms. In the following subsections, we show how to integrate them in a hybrid model with careful formulations.

1) Generative model based classification: Given a training data set $\{(\mathbf{x}_i, y_i)\}_{i=1,2,...,N}$, $y_i \in \{1, 2, ..., K\}$ with $\mathbf{x}_i$ and y_i as data feature and label, and K the number of classes, for a classification based on a generative model, based on Eq. (2), the classification model is:

$$p(y=k|\mathbf{x}) = \frac{p(\mathbf{x}|y=k)\,p(y=k)}{\sum\limits_{j=1}^{K} p(\mathbf{x}|y=j)\,p(y=j)}. \tag{3}$$

And Eq. (3) is further cast to:

$$p(y=k|\mathbf{x}) = \frac{1}{1 + \sum\limits_{j=1, j\neq k}^{K} \exp\left(-r_{k,j}(\mathbf{x}, \Theta_G)\right)}, \tag{4}$$

where

$$r_{k,j}(\mathbf{x}, \Theta_G) = \log \frac{p(\mathbf{x}|y=k)\,p(y=k)}{p(\mathbf{x}|y=j)\,p(y=j)}, \tag{5}$$

is a LLR score based on class generative probability model with Θ_G as model parameter set.

2) Discriminative model based classification: Rather than using a generative model, a neural network can be applied to directly approximate the posterior probability function $p(y|\mathbf{x})$. A discriminative learning tries to approximate the mapping between input feature and label with a softmax function defined as:

$$p(y=k|\mathbf{x}) = \frac{\exp(o_k)}{\sum\limits_{j=1}^{K} \exp(o_j)}, \tag{6}$$

where a network mapping function $o_j = \phi_j(\mathbf{x}, \Theta_D)$ is defined as the output corresponding to the j-th class, and Θ_D is the neural network parameters. Eq. (6) is further cast to:

$$p(y=k|\mathbf{x}) = \frac{1}{1 + \sum\limits_{j=1, j\neq k}^{K} \exp\left(-h_{k,j}(\mathbf{x}, \Theta_D)\right)}, \tag{7}$$

where

$$h_{k,j}(\mathbf{x}, \Theta_D) = \phi_k(\mathbf{x}, \Theta_D) - \phi_j(\mathbf{x}, \Theta_D). \tag{8}$$

Comparing Eqs. (7), (8) and (4), (5), we can see that $h_{k,j}(\mathbf{x}, \Theta_D)$ can be connected to the $r_{k,j}(\mathbf{x}, \Theta_G)$ with the LLR in calculation. This connection inspired us to incorporate the LLR of pairwise samples from a generative model to the neural network discriminative training for SV.

B. Connecting generative and discriminative models through Log likelihood ratio for SV

Based on the generative model, given a hypothesis H_S or H_D, the joint probability of generating $(\mathbf{x}_i, \mathbf{x}_j)$ is $p(\mathbf{x}_i, \mathbf{x}_j|H_S)$ or $p(\mathbf{x}_i, \mathbf{x}_j|H_D)$. In making a decision, the LLR is defined as:

$$r_{i,j} \triangleq r(\mathbf{x}_i, \mathbf{x}_j) = \log \frac{p(\mathbf{x}_i, \mathbf{x}_j|H_S)}{p(\mathbf{x}_i, \mathbf{x}_j|H_D)} \tag{9}$$

With a given decision threshold, we can decide whether the two observation vectors are from H_S or H_D (as defined in Eq. (1)). For convenience of formulation, we define a trial as a tuple $\mathbf{z}_{i,j} = (\mathbf{x}_i, \mathbf{x}_j)$, and the two hypothesis spaces are constructed from the two data sets as:

$$\begin{array}{l} S = \{\mathbf{z}_{i,j} = (\mathbf{x}_i, \mathbf{x}_j) \in H_S\} \\ D = \{\mathbf{z}_{i,j} = (\mathbf{x}_i, \mathbf{x}_j) \in H_D\} \end{array} \tag{10}$$

We first derive the LLR score calculation based on the JB based generative model.

1) Joint Bayesian generative model approach: Given an observation X-vector variable $\mathbf{x}$, it is supposed to be generated by a speaker identity variable and a random noise variable (possibly induced by different recording background noise, sessions, or transmission channels, etc.) as:

$$\mathbf{x} = \mathbf{u} + \mathbf{n}, \tag{11}$$

where $\mathbf{u}$ is a speaker identity vector variable, $\mathbf{n}$ represents intra-speaker variation caused by noise. For simplicity, the observation $\mathbf{x}$ is mean subtracted, and the speaker identity and intra-speaker variation variables are supposed to be with Gaussian distributions as:

$$\begin{array}{l} \mathbf{u} \sim N(0, \mathbf{C_u}) \\ \mathbf{n} \sim N(0, \mathbf{C_n}) \end{array}, \tag{12}$$

where $\mathbf{C_u}$ and $\mathbf{C_n}$ are speaker and noise covariance matrices, respectively. In verification, given a trial with $\mathbf{x}_i$ and $\mathbf{x}_j$ generated from Eq. (11), based on the assumption in Eq. (12), the two terms $p(\mathbf{x}_i, \mathbf{x}_j|H_S)$ and $p(\mathbf{x}_i, \mathbf{x}_j|H_D)$ defined in Eq. (9) satisfy zero-mean Gaussian with covariances as:

$$\begin{array}{l} \mathrm{cov}_S = \begin{bmatrix} \mathbf{C_u} + \mathbf{C_n} & \mathbf{C_u} \\ \mathbf{C_u} & \mathbf{C_u} + \mathbf{C_n} \end{bmatrix} \\ \mathrm{cov}_D = \begin{bmatrix} \mathbf{C_u} + \mathbf{C_n} & 0 \\ 0 & \mathbf{C_u} + \mathbf{C_n} \end{bmatrix} \end{array} \tag{13}$$

Based on this formulation, the LLR defined in Eq. (9) could be calculated based on:

$$r(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{A} \mathbf{x}_i + \mathbf{x}_j^T \mathbf{A} \mathbf{x}_j - 2\mathbf{x}_i^T \mathbf{G} \mathbf{x}_j, \tag{14}$$

where

$$\begin{array}{l} \mathbf{A} = (\mathbf{C_u} + \mathbf{C_n})^{-1} - [(\mathbf{C_u} + \mathbf{C_n}) - \mathbf{C_u}(\mathbf{C_u} + \mathbf{C_n})^{-1}\mathbf{C_u}]^{-1} \\ \mathbf{G} = -(2\mathbf{C_u} + \mathbf{C_n})^{-1}\mathbf{C_u}\mathbf{C_n}^{-1} \end{array} \tag{15}$$

As seen from Eq. (15), the generative model parameters Θ_G in estimating LLR are only related to covariance parameters $\mathbf{C_u}$ and $\mathbf{C_n}$ [12], [13]. Given a training data set, the parameters could be estimated using an EM (or EM-like) learning algorithm based on:

$$\Theta_G^* = \arg\min_{\Theta_G} - \sum_i \log p(\mathbf{X}_i|\Theta_G) \tag{16}$$

where $\Theta_G = \{\mathbf{C_u}, \mathbf{C_n}\}$, $\mathbf{X}_i$ is a collection of samples for speaker i.

2) Pairwise discriminative model approach: The binary classification task defined in Eq. (1) can be solved based on discriminative neural network modeling as formulated in Eqs. (6) and (7). In neural network modeling, the parameters are neural weights (affine transform matrices with linear or nonlinear activations). We can connect the model parameters of a generative model with the neural weights and optimize them with an objective function. As a binary classification task, given a trial with two observation X-vector variables $\mathbf{z}_{i,j} = (\mathbf{x}_i,\mathbf{x}_j)$, the classification task is to estimate and compare $p(H_S|\mathbf{z}_{i,j})$ and $p(H_D|\mathbf{z}_{i,j})$. In the discriminative learning, the label is defined as:

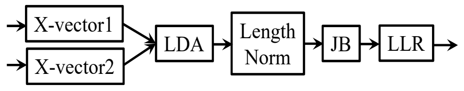

Fig. 2. Pipeline for joint Bayesian based generative modeling on X-vectors for speaker verification, LDA: Linear Discrimination Analysis, JB: Joint Bayesian model, LLR: log likelihood ratio.

$$y_{i,j} = \begin{cases} 1, \mathbf{z}_{i,j} \in H_S \\ 0, \mathbf{z}_{i,j} \in H_D \end{cases} \tag{17}$$

With reference to Eqs. (7) and (8), the posterior probability is estimated based on:

$$p(y_{i,j}|\mathbf{z}_{i,j}) = \begin{cases} \frac{1}{1+\exp(-h_{H_S,H_D}(\mathbf{z}_{i,j},\Theta_D))}; \mathbf{z}_{i,j} \in H_S \\ 1 - \frac{1}{1+\exp(-h_{H_S,H_D}(\mathbf{z}_{i,j},\Theta_D))}; \mathbf{z}_{i,j} \in H_D \end{cases} \tag{18}$$

As we have revealed from Eqs. (4), (5), and (9), we replace the $h_{H_S,H_D}(\mathbf{z}_{i,j}, \Theta_D)$ with a LLR score, and define a mapping as a logistic function with scaled parameters as used in [30], [31]:

$$f(r_{i,j}) \triangleq \frac{1}{1 + \exp(-(\alpha r_{i,j} + \beta))} \tag{19}$$

where $r_{i,j} = r(\mathbf{z}_{i,j}) = r(\mathbf{x}_i, \mathbf{x}_j)$ as defined in Eq. (9), and α and β are gain and bias factors used in the regression model. In Eq. (19), we integrated the LLR score estimated from the JB generative model in a discriminative training framework. The probability estimation in Eq. (18) is cast to:

$$\hat{y}_{i,j} \triangleq p(y_{i,j}|\mathbf{z}_{i,j}) = \begin{cases} f(r_{i,j}); \mathbf{z}_{i,j} \in H_S \\ 1 - f(r_{i,j}); \mathbf{z}_{i,j} \in H_D \end{cases} \tag{20}$$

The training can be based on optimizing the binary cross entropy defined as:

$$L = - \sum_{\mathbf{z}_{i,j} \in \{H_S \cup H_D\}} (y_{i,j} \log f(r_{i,j}) + (1 - y_{i,j}) \log(1 - f(r_{i,j}))) \tag{21}$$

In the following subsection, we investigate the neural network architecture for the hybrid model framework.

C. Coupling generative model parameters with neural network architecture

The conventional state of the art framework for SV based on X-vector and JB model is illustrated in Fig. 2. In this figure, the LDA is applied on the X-vector for discriminative feature extraction and dimension reduction. After the LDA, a vector length normalization is used, then a JB based generative model is applied by which the LLR is estimated. In a pairwise discriminative learning framework, the LLR can be used for a binary classification task, and could be implemented with a discriminative neural network model.

We first explain the LDA which will be approximated by an affine transform as used in the neural network modeling. For input X-vector samples and their corresponding labels

$\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), ..., (\mathbf{x}_M, y_M)\}$, where $\mathbf{x}_i \in \mathrm{R}^l$, and M is the number of samples, the LDA transform is:

$$\mathbf{h}_i = \mathbf{W}^T \mathbf{x}_i, \tag{22}$$

where $\mathbf{W} \in \mathrm{R}^{l\times d}$, l and d are the dimensions of the input X-vectors and the transformed feature vectors, respectively. $\mathbf{W}$ is estimated as follows:

$$\mathbf{W}^* = \arg\max_{\mathbf{W}} tr(\frac{\mathbf{W}^T\mathbf{S}_{\mathrm{b}}\mathbf{W}}{\mathbf{W}^T\mathbf{S}_{\mathrm{w}}\mathbf{W}}), \tag{23}$$

where $tr(.)$ denotes the matrix trace operator, $\mathbf{S}_{\mathrm{w}}$ and $\mathbf{S}_{\mathrm{b}}$ are the intra-class and inter-class covariance matrices. From Eq. (22), we can see that the LDA can be implemented as a linear dense layer in neural network modeling.

We further look at the estimation of the LLR score defined in Eq. (14). In Eq. (14), $\mathbf{A}$ and $\mathbf{G}$ are negative semi-definite symmetric matrices [12], [13], and they can be decomposed as:

$$\begin{aligned} \mathbf{A} &= -\mathbf{P}_A\mathbf{P}_A^T \\ \mathbf{G} &= -\mathbf{P}_G\mathbf{P}_G^T \end{aligned} \tag{24}$$

The LLR score is cast to:

$$r_{i,j} = 2g_i^T g_j - a_i^T a_i - a_j^T a_j \tag{25}$$

with affine linear transforms as:

$$\begin{aligned} a_i = \mathbf{P}_A^T\tilde{\mathbf{h}}_i, a_j = \mathbf{P}_A^T\tilde{\mathbf{h}}_j \\ g_i = \mathbf{P}_G^T\tilde{\mathbf{h}}_i, g_j = \mathbf{P}_G^T\tilde{\mathbf{h}}_j, \end{aligned} \tag{26}$$

where the input to the JB model is the length normalized output from the LDA processing as:

$$\begin{aligned} \tilde{\mathbf{h}}_i &= \frac{\mathbf{h}_i}{||\mathbf{h}_i||} \\ \tilde{\mathbf{h}}_j &= \frac{\mathbf{h}_j}{||\mathbf{h}_j||} \end{aligned} \tag{27}$$

The transforms in Eq. (26) could be implemented in a neural network as linear dense layers. Based on these formulations, a two-branch Siamese neural network is designed as showed in Fig. 3. In this figure, there are two sub-nets, i.e., "LDA_net" and "JB_net". The "LDA_net" is a dense layer net with a transform $\mathbf{W}$ according to Eq. (22). In the "JB_net", the JB model structure is taken into consideration as two-branch $(\mathbf{P}_A, \mathbf{P}_G)$ dense layer network according to Eq. (26). In training the Siamese neural network, the "negative" and "positive" samples are constructed as we did in pairwise discriminative training for language recognition task [32]. In the generative model based backend, a length normalization block is often applied with the purpose of variable Gaussianization for the convenience of generative probability modeling [33]. In our proposed Siamese neural network backend model, the length normalization block is also used. The purpose is twofold: the first one is to exactly fit the generative model structure to the proposed discriminative neural network framework, the second one is to stabilize the neural network training by serving as a nonlinear transform for dynamic range normalization of neural activations.

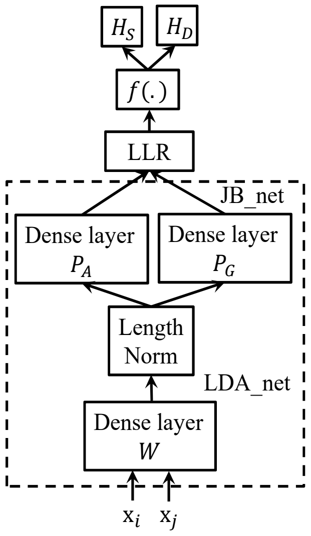

Fig. 3. The proposed two-branch Siamese neural network with coupling of the generative JB model structure for speaker verification (see the text for a detailed explanation). H_D: hypothesis for different speaker, H_S: hypothesis for the same speaker. Dense layers are with linear identity activation functions.

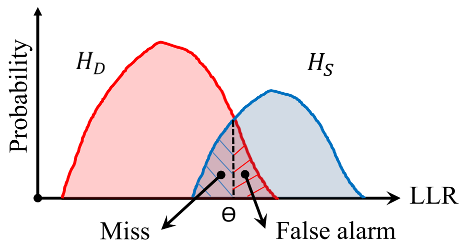

Fig. 4. The LLR distributions in H_S and H_D for the same and different speaker spaces, and two types of errors in the hypothesis test for SV.

D. Direct evaluation metric (DEM): learning objective function based on minimum empirical Bayes risk (EBR)

The cross entropy defined in Eq. (21) can be applied for discriminative training in order to measure the classification error. However, the hypothesis test defined in Eq. (1) is different from a classification goal, and the final evaluation metric for SV usually adopts some different criterions. It is better to optimize model parameters directly based on the evaluation metrics. As a hypothesis test, there are two types of errors, i.e., type I and type II errors [28], [29]. The two types of errors are defined as:

$$\begin{aligned} &\text{Type I error (false alarm): } \mathbf{z}_{i,j} \in H_D, \text{LLR} \geq \theta \\ &\text{Type II error (miss): } \mathbf{z}_{i,j} \in H_S, \text{LLR} < \theta, \end{aligned} \tag{28}$$

where θ is a decision threshold. These two types of errors are further illustrated in Fig. 4 for a SV task. In this figure, the objective for SV is to minimize the target miss P_{miss} (or false reject) and false alarm P_{fa} (or false accept) in the two hypothesis spaces H_S and H_D. By selecting different decision thresholds, a detection error tradeoff (DET) graph could be obtained. In real applications, it is better to generalize the classification errors to a weighing of these two types of errors. With consideration of the prior knowledge in a measure of empirical Bayes risk (EBR), the evaluation metric for SV adopts a detection cost function (DCF) to measure the hardness of the decisions [28]. It is defined as a weighted loss:

$$C_{\text{det}} \overset{\Delta}{=} P_{\text{tar}} C_{\text{miss}} P_{\text{miss}} + (1 - P_{\text{tar}}) C_{\text{fa}} P_{\text{fa}}, \tag{29}$$

where C_{miss} and C_{fa} are user assigned costs for miss and false alarm detections, P_{tar} is a prior of target trials, P_{miss} and P_{fa} are miss and false alarm probabilities defined as:

$$\begin{aligned} P_{\text{fa}} &= \frac{1}{N_{\text{non}}} \sum_{\mathbf{z}_{i,j} \in H_D} u(r_{i,j} \geq \theta) \\ P_{\text{miss}} &= \frac{1}{N_{\text{tar}}} \sum_{\mathbf{z}_{i,j} \in H_S} u(r_{i,j} < \theta) \end{aligned} \tag{30}$$

In Eq. (30), "N_{non}" and "N_{tar}" are the numbers of non-target and target trials, $r_{i,j}$ is the LLR estimated from Eq. (25), θ is a decision threshold, and $u(.)$ is an indictor function for counting the number of trials with scores lower or higher than the decision threshold. In order to change the objective function to be differentiable and thus can be used in gradient based neural network learning, Eq. (30) is approximated by:

$$\begin{aligned} P_{\text{fa}} &\approx \frac{1}{N_{\text{non}}} \sum_{\mathbf{z}_{i,j} \in \{H_S \cup H_D\}} (1 - y_{i,j}) f(r_{i,j}) \\ P_{\text{miss}} &\approx \frac{1}{N_{\text{tar}}} \sum_{\mathbf{z}_{i,j} \in \{H_S \cup H_D\}} y_{i,j} (1 - f(r_{i,j})) \end{aligned} \tag{31}$$

where $f(r_{i,j})$ is a sigmoid logistic function defined the same as in Eq. (19). With regard to the cross-entropy loss defined in Eq. (21), the weighted binary cross entropy loss (WBCE) can be formulated as:

$$L_{CE} = w_S L_{CE\_H_S} + (1 - w_S) L_{CE\_H_D}, \tag{32}$$

where w_S is the weighting coefficient (as prior of target trials P_{tar}), and the target and non-target cross entropy losses are defined as:

$$\begin{aligned} L_{CE\_H_S} &= -\frac{1}{N_{\text{tar}}} \sum_{\mathbf{z}_{i,j} \in H_S} y_{i,j} \log f(r_{i,j}) \\ L_{CE\_H_D} &= \frac{1}{N_{\text{non}}} \sum_{\mathbf{z}_{i,j} \in H_D} (1 - y_{i,j}) \log(1 - f(r_{i,j})) \end{aligned} \tag{33}$$

For a further analysis, we show the loss functions defined in Eqs. (30), (31) and (33) in Fig. 5. From this figure, we can see that the losses defined in Eqs. (31) and (33) have the same monotonic tendency in measuring the loss, and can be regarded as a soft loss of the miss and false alarm as defined in Eq. (30). In addition, from this figure, we can see that the calibrated LLR threshold $f(\theta)$ in loss score estimation is 0.5. Based on this analysis, the definition of losses in Eq. (29) can be regarded as a generalized weighted loss from the definition of Eq. (32).

In mini-batch based gradient back-propagation (BP) learning, the gradient is still estimated based on the chain rule from composition functions. For convenience of analysis, we reformulate the LLR score defined in Eq. (14) as:

$$r_{i,j} = \tilde{\mathbf{h}}_i^T \mathbf{A} \tilde{\mathbf{h}}_i + \tilde{\mathbf{h}}_j^T \mathbf{A} \tilde{\mathbf{h}}_j - 2\tilde{\mathbf{h}}_i^T \mathbf{G} \tilde{\mathbf{h}}_j, \tag{34}$$

where $\tilde{\mathbf{h}}_i$ and $\tilde{\mathbf{h}}_j$ are the length normalized vectors defined in Eq. (27). Then the gradients for JB_net parameters are derived as:

$$\begin{aligned} \Delta \mathbf{P}_A &\propto \frac{\partial C_{\text{det}}}{\partial f(r_{i,j})} \frac{\partial f(r_{i,j})}{\partial r_{i,j}} \frac{\partial r_{i,j}}{\partial \mathbf{A}} \frac{\partial \mathbf{A}}{\partial \mathbf{P}_A} \\ \Delta \mathbf{P}_G &\propto \frac{\partial C_{\text{det}}}{\partial f(r_{i,j})} \frac{\partial f(r_{i,j})}{\partial r_{i,j}} \frac{\partial r_{i,j}}{\partial \mathbf{G}} \frac{\partial \mathbf{G}}{\partial \mathbf{P}_G} \end{aligned} \tag{35}$$

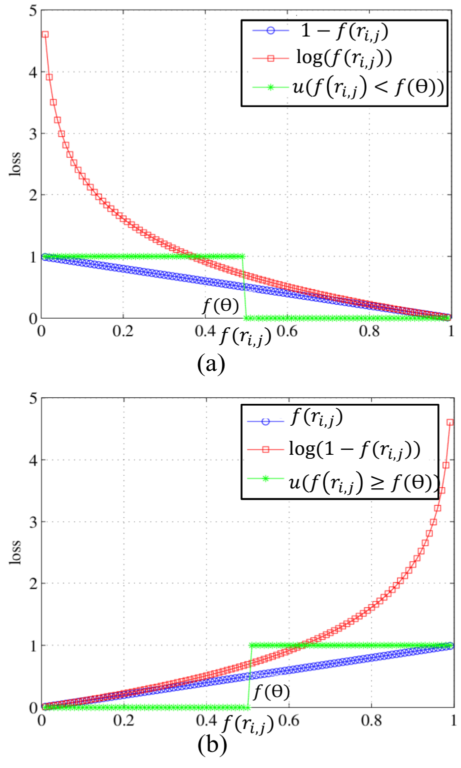

Fig. 5. Loss functions defined in Eqs. (30), (31) and (33): (a) miss loss, and (b) false alarm loss

And the gradients for LDA_net parameters are derived as:

$$\Delta \mathbf{W} \propto \frac{\partial C_{\text{det}}}{\partial f(r_{i,j})} \frac{\partial f(r_{i,j})}{\partial r_{i,j}} \left(\frac{\partial r_{i,j}}{\partial \tilde{\mathbf{h}}_i} \frac{\partial \tilde{\mathbf{h}}_i}{\partial \mathbf{h}_i} \frac{\partial \mathbf{h}_i}{\partial \mathbf{W}} + \frac{\partial r_{i,j}}{\partial \tilde{\mathbf{h}}_j} \frac{\partial \tilde{\mathbf{h}}_j}{\partial \mathbf{h}_j} \frac{\partial \mathbf{h}_j}{\partial \mathbf{W}} \right) \tag{36}$$

Following the definition function in each term, the gradients can be deduced which are used in BP based neural network learning.

III. Experiments and Results

We carried out experiments on SV tasks where the test data sets are from Speakers in the wild (SITW) [34] and Voxceleb [35]. The speaker features and models were trained on Voxceleb (sets 1 and 2) [35]. A state of the art pipeline for constructing the SV system is adopted as shown in Fig. 2. In this figure, the "LDA", "Length Norm", and "JB" blocks are designed independently rather than optimized jointly. The input speaker features in our pipeline are X-vectors. The X-vectors are extracted based on a well trained neural network model which is designed for a speaker classification task [5]. As backend models, both the well-known PLDA and JB based generative models are implemented in our comparisons.

A. Speaker embedding features based on X-vectors

A speaker embedding model is trained for the X-vector extraction. The neural network architecture of the embedding model is composed of deep time delay neural network (TDNN) layers and statistical pooling layers implemented the same as introduced in [5] and used in Kaldi [36]. In training the model, the cross entropy criterion for speaker classification is used as the learning objective function. The training data set includes two data sets from Voxceleb corpus, i.e., the training set of Voxceleb1 corpus from which speakers included in the test set of the SITW are removed, and the training set of Voxceleb2. In total, there are about 7,185 speakers with 1,236,567 utterances used for training. Moreover, data augmentation is applied by adding noise, music, babble with several SNRs, and reverberation with simulated room impulse response is also applied to increase data diversity [36]. Input features for training the speaker embedding model are MFCCs with 30 Mel band bins. The MFCCs are extracted with 25 ms frame length and 10 ms frame shift. Energy based voice activity detection (VAD) is applied to remove silence background regions in speaker feature extraction. More details of feature and model architecture and training procedures were introduced in [5]. The final extracted X-vectors are with 512 dimensions.

B. Backend models

Although X-vectors extracted from the speaker embedding model are supposed to encode speaker discriminative information, they also encode other acoustic factors. In a conventional pipeline as illustrated in Fig. 2, LDA is applied before applying a generative speaker model. In this study, the 512-dimension X-vectors are transformed to 200-dimension vectors by LDA. Correspondingly, in the discriminative neural network model as showed in Fig. 3, a dense layer with 200 neurons is also applied. Moreover, in the discriminative model, two dense layers corresponding to $\mathbf{P}_A$ and $\mathbf{P}_G$ of the JB model are trained with "positive" and "negative" X-vector pairs (pairs from the same and different speakers). Since the discriminative neural network architecture fits well to the pipeline based on a generative model structure, the dense layer parameters could be initialized with the LDA and JB model parameters in training (according to Eqs. (22) and (26)). By this initialization, the discriminative training starts exactly from the model parameters of backend pipeline including the LDA and the generative model, and further refines the discriminability for the SV task. For comparison, the random initialization method with "he_normal" as widely used in deep neural network learning is also applied in experiments [37]. In model training, the Adam algorithm with an initial learning rate of 0.0005 [38] was used. In order to include enough "negative" and "positive" samples, the mini-batch size was set to 4096. The training X-vectors were splitted to training and validation sets with a ratio of $9:1$. The model parameters were selected based on the best performance on the validation set.

C. Results

We first carried out SV experiments on the data sets of SITW. Two test sets are used, i.e., development and evaluation sets, and each is used as an independent test set. The evaluation metrics are equal error rate (EER) and minimum decision cost function (minDCF) (with target priors 0.01 and 0.001) [34]. The EER denotes when the type I and type II errors (as defined in Eq. (28)) are equal, and the minDCF is optimized from the DCF defined in Eq. (29) (with $C_{\text{miss}} = C_{\text{fa}} = 1$). In the optimization of the minDCF, the decision threshold for false alarm and miss error estimations is also jointly optimized. The performance results are showed in Tables I and II. In these two tables, "LDA(200)+PLDA" and "LDA(200)+JB" represent the PLDA and JB generative model based SV systems following the pipeline in Fig. 2 (the dimension for LDA is 200) (replace the block "JB" with "PLDA" for the PLDA based SV system). "Hybrid" denotes the proposed Siamese neural network backend based SV system which takes the JB model structure and parameter coupling in designing the neural model architecture following the pipeline in Fig. 3. And in the "Hybrid" SV system, two model initialization methods are tested in model training as explained in section III-B. From these two tables, we can see that the performance of the JB based generative model is comparable or slightly better than that of the PLDA based model. In the hybrid model, if model parameters ("LDA_net" and "JB_net") are randomly initialized, the performance is worse than the original generative model based results. However, when the neural network parameters are initialized with the "LDA" and "JB" based model parameters, the performance is significantly improved. These results indicate that the discriminative training could further enhance the discriminative power of the generative model when the model parameters are initialized with the generative model based parameters. Otherwise, random initialization in the discriminative learning does not enhance the performance even when the generative model structure is taken into consideration. Following the same process, the experimental results on voxceleb1 test set are showed in Table III. From this table, we could observe the same tendency as in Tables I and II.

TABLE I
PERFORMANCE ON THE DEVELOPMENT SET OF SITW.

Methods	EER (%)	minDCF (0.01)	minDCF (0.001)
LDA(200)+PLDA	3.003	0.3315	0.5198
LDA(200)+JB	3.043	0.3288	0.5019
Hybrid (rand init)	4.159	0.3792	0.5883
Hybrid (JB init)	**2.662**	**0.2972**	**0.4466**

TABLE II
PERFORMANCE ON EVALUATION SET OF SITW.

Methods	EER (%)	minDCF (0.01)	minDCF (0.001)
LDA(200)+PLDA	3.554	0.3526	0.5657
LDA(200)+JB	3.496	0.3422	0.5645
Hybrid (rand init)	4.505	0.3920	0.6003
Hybrid (JB init)	**3.142**	**0.3075**	**0.4619**

TABLE III
PERFORMANCE ON EVALUATION SET OF VOXCELEB1 TEST.

Methods	EER (%)	minDCF (0.01)	minDCF (0.001)
LDA(200)+PLDA	3.128	0.3258	0.5003
LDA(200)+JB	3.105	0.3226	0.4992
Hybrid (rand init)	3.340	0.3778	0.4977
Hybrid (JB init)	**2.837**	**0.3011**	**0.3743**

TABLE IV
PERFORMANCE ON THE DEVELOPMENT SET OF SITW: RANDOM SETTING OF THE CLASSIFIER MODEL ("JB_NET") AND WITH TWO SETTING CONDITIONS FOR THE "LDA_NET". SETTING (A): INDEPENDENT LDA TRANSFORM, SETTING (B): JOINTLY TRAINED LDA TRANSFORM.

LDA_net setting	EER (%)	minDCF (0.01)	minDCF (0.001)
Setting (a)	8.24	0.6546	0.8434
Setting (b)	**8.01**	**0.5968**	**0.7820**

D. Ablation study

Many factors may contribute to the final performance in the proposed framework. In this paper, we consider two aspects which are directly related to our contributions: one is the hybrid generative and discriminative model architecture design, the other is the optimization objective function design. In the model architecture design, there are two important modeling blocks, i.e., "LDA_net" and "JB_net" as illustrated in Fig. 3. The function of the "LDA_net" is extracting low-dimension discriminative speaker representations from X-vectors, whereas the "JB_net" is applied on the extracted feature vectors for speaker modeling. They were jointly learned in a unified framework. In the optimization objective function design, although the direct evaluation metric could be regarded as a generalization from the weighted binary cross entropy function, the degree of penalty for miss and false alarm errors is different. In this subsection, we investigate their effects on SV performance one by one with ablation studies.

1) Effect of the "LDA_net" in learning: X-vectors are extracted from a TDNN based speaker embedding model which is optimized for speaker classification. After the LDA process, the speaker feature has a strong power for speaker discrimination. In the proposed hybrid model, the LDA model is further jointly optimized for the SV task. We verify the discrimination power of speaker representations on SV performance with random setting of the "JB_net" while only setting the parameters of the "LDA_net" with the following conditions (after setting, the model is not further trained any more): (a) setting the "LDA_net" with the LDA parameters (independent LDA transform), (b) setting the "LDA_net" with the jointly trained LDA parameters. The results are showed in table IV. From these results, we can see that after joint training (in setting (b)), the performance is further improved.

2) Effect of $\mathbf{A}$ and $\mathbf{G}$ on SV performance: As showed in Eq. (14), the two terms have different effects on the speaker verification performance. In our discriminative training which integrates the LLR of the JB model, the LLR in Eq. (14) is adapted. With different settings of $\mathbf{A}$ and $\mathbf{G}$ on Eq. (14), we could obtain:

$$r(\mathbf{x}_i, \mathbf{x}_j) = \begin{cases} -2\mathbf{x}_i^T\mathbf{G}\mathbf{x}_j; \text{ for } \mathbf{A} = 0 \\ \mathbf{x}_i^T\mathbf{A}\mathbf{x}_i + \mathbf{x}_j^T\mathbf{A}\mathbf{x}_j; \text{ for } \mathbf{G} = 0 \\ (\mathbf{x}_i - \mathbf{x}_j)^T\mathbf{G}(\mathbf{x}_i - \mathbf{x}_j); \text{ for } \mathbf{A} = \mathbf{G} \\ (\mathbf{x}_i - \mathbf{x}_j)^T\mathbf{A}(\mathbf{x}_i - \mathbf{x}_j); \text{ for } \mathbf{G} = \mathbf{A} \end{cases} \quad (37)$$

Based on this formulation, we could check the different effects of $\mathbf{A}$ and $\mathbf{G}$ on the SV performance. The two matrices $\mathbf{A}$ and $\mathbf{G}$ are connected to the two dense layer branches of the hybrid model with weights $\mathbf{P}_A$ and $\mathbf{P}_G$ (refer to Fig. 3). In our model, the dense layers were first initialized with the parameters from the learned JB based generative model, then the model was further trained with pairwise "negative" and "positive" samples. Only in testing stage, we use different parameter settings for experiments according to Eq. (37), and the results are showed in Tables V and VI for the dev set of SITW before and after discriminative training, respectively. In these two tables, by comparing conditions with $\mathbf{A} = 0$ or $\mathbf{G} = 0$, we can see that the cross term contributes more to the SV performance, i.e., the dense layer branch with neural weight $\mathbf{P}_G$ contributes the most discriminative information in the SV task. Moreover, when keeping the cross term either by setting $\mathbf{A} = \mathbf{G}$ or $\mathbf{G} = \mathbf{A}$, the performance is better than setting any one of them to be zero. In summary, the contribution of discriminative information from feature self-norm associated with matrix $\mathbf{A}$ is less while feature cross-term associated with $\mathbf{G}$ contributes most in the SV task.

TABLE V
PERFORMANCE ON THE DEVELOPMENT SET OF SITW **BEFORE JOINT TRAINING**: SETTING THE LDA_NET AND JB_NET WITH THE INDEPENDENTLY LEARNED LDA AND JB MODEL PARAMETERS, WITH DIFFERENT EXPERIMENTAL SETTINGS OF CLASSIFIER MODEL ("JB_NET").).

Methods	EER (%)	minDCF (0.01)	minDCF (0.001)
A (G=0)	47.71	1.000	1.000
G (A=0)	6.353	0.8261	0.9806
A, G (set G to A)	**3.119**	**0.3604**	**0.5844**
A, G (set A to G)	3.504	0.3978	0.6316

TABLE VI
PERFORMANCE ON THE DEVELOPMENT SET OF SITW **AFTER JOINT TRAINING**: DIFFERENT EXPERIMENTAL SETTINGS OF CLASSIFIER MODEL ("JB_NET").

Methods	EER (%)	minDCF (0.01)	minDCF (0.001)
A (G=0)	50.29	0.9996	0.9996
G (A=0)	4.775	0.4206	0.6340
A, G (set G to A)	**2.811**	**0.2975**	0.4561
A, G (set A to G)	3.080	0.3134	**0.4505**

3) Relation to distance metric learning: Distance metric learning is widely used in discriminative learning with pairwise training samples as input [39], [40]. The Mahalanobis distance metric between two vectors is defined as:

$$d_{i,j} \stackrel{\Delta}{=} d(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)^T\mathbf{M}(\mathbf{x}_i - \mathbf{x}_j), \quad (38)$$

where $\mathbf{M} = \mathbf{P}\mathbf{P}^T$ is a positive definite matrix. Based on this distance metric, the binary classification task for SV can be formulated as:

$$p(y_{i,j}|\mathbf{z}_{i,j}) = \sigma(\lambda(d_0 - d_{i,j})), \quad (39)$$

where $\sigma(x) = (1 + \exp(-x))^{-1}$ is the sigmoid logistic function, d_0 is a distance decision threshold, and λ is a scale parameter for probability calibration. From Eq. (39), we can see that when the Mahalanobis distance $d(\mathbf{x}_i, \mathbf{x}_j) < d_0$, the probability of $\mathbf{x}_i$ and $\mathbf{x}_j$ belonging to the same speaker is high, and vice versa. With pairwise "positive" and "negative" samples, the parameters ($\mathbf{M}$, d_0, and λ) can be learned based on a given training data set as a binary discriminative learning task. Comparing Eqs. (38) and (37), we can see that if we set $\mathbf{A} = \mathbf{G}$ or $\mathbf{G} = \mathbf{A}$, the LLR and Mahalanobis distance have

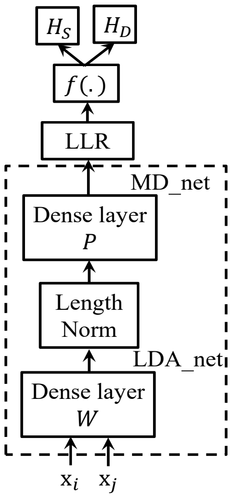

Fig. 6. Siamese neural network with Mahalanobis distance net (MD_net) on X-vector features for speaker verification. Dense layers are with linear identity activation functions.

TABLE VII
PERFORMANCE ON THE DEVELOPMENT SET OF SITW BASED ON THE SIAMESE NEURAL NETWORK WITH "MD_NET" AS CLASSIFIER MODEL.

Methods	EER (%)	minDCF (0.01)	minDCF (0.001)
Random init $\mathbf{P}$	3.966	0.3743	0.5543
Init $\mathbf{P}$ with $\mathbf{P}_A$	**3.621**	**0.3686**	**0.5472**
Init $\mathbf{P}$ with $\mathbf{P}_G$	4.005	0.4060	0.6003

the same formulation form (except the difference in matrix as negative or positive definite), i.e., $d(\mathbf{x}_i, \mathbf{x}_j) \propto -r(\mathbf{x}_i, \mathbf{x}_j)$. In this sense, the distance metric based discriminative learning framework can be regarded as a special case of the hybrid discriminative framework, and the LLR defined in Eq. (9) is cast to:

$$r(\mathbf{x}_i, \mathbf{x}_j) = \log \frac{p(\Delta_{i,j}|H_S)}{p(\Delta_{i,j}|H_D)}, \tag{40}$$

where $\Delta_{i,j} = \mathbf{x}_i - \mathbf{x}_j$. From this definition, we can see that the distance metric based discriminative learning only considers the distribution of the pairwise sample distance space [41]. In our implementation, by merging the two dense layers of the classifier model ("JB_net" with parameters $\mathbf{P}_A$ and $\mathbf{P}_G$), the proposed hybrid framework is changed to be one branch framework as showed in Fig. 6. In this figure, the "MD_net" is the network dense layer for Mahalanobis distance metric with an affine transform matrix $\mathbf{P}$, and it can be initialized with the parameters of the JB based generative model (either $\mathbf{P} = \mathbf{P}_A$ or $\mathbf{P} = \mathbf{P}_G$), or with random values (he_normal). We test this one branch model on the dev set of SITW with different settings of the "MD_net" (the "LDA_net" is initialized with the LDA transform based parameters), and show the results in Table VII. From this table, we can see that when the LDA_net and MD_net of the one branch model are initialized with the LDA and $\mathbf{P}_A$ parameters, the performance is the best. However, no matter in what conditions, comparing results in Tables I and VII, we can see that the hybrid model framework showed the best performance which confirmed that the model structure inspired by the JB based generative model is helpful in the SV task.

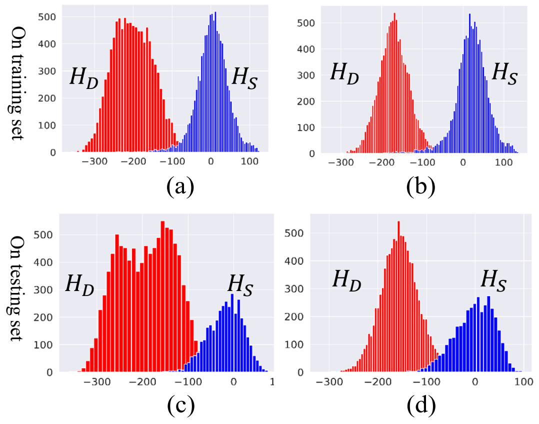

Fig. 7. LLR distributions in H_S and H_D spaces: the first row (a, b) for the training set, the second row (c, d) for the testing set; the left column (a and c) for model setting with generative model parameters learned based on the EM algorithm, and the right column (b and d) for model setting with discriminatively trained parameters after initializing with generative model parameters learned based on the EM algorithm.

TABLE VIII
PERFORMANCE WITH DIFFERENT OPTIMIZATION OBJECTIVE FUNCTIONS (ON THE DEVELOPMENT SET OF SITW).

Objective functions	EER (%)	minDCF (0.01)	minDCF (0.001)
WBCE (Eqs. 32, 33)	2.695	0.3157	0.5670
DEM (Eqs. 29, 31)	**2.662**	**0.2972**	**0.4466**

4) LLR distributions for intra- and inter-speaker spaces: As defined in Eq. (9), the performance is measured based on the LLR distributions in two spaces, i.e., the intra-speaker space H_S and inter-speaker space H_D. The separability can be visualized as the histogram distributions of pairwise distances in the two spaces. We check the histograms of the LLR on the training and test sets based on the hybrid model (refer to network pipeline in Fig. 3) with different parameter settings, and show them in Fig. 7. From this figure, we can see that with the discriminative training, the separation is further enhanced. In particular, the LLR distribution of "negative" sample pairs becomes much more compact for both training and testing data sets.

5) Effect of objective function design: Different objective function may affect the optimization process and hence may result in different performance. Although the direct evaluation metric (DEM) defined in Eqs. (29), (31) can be regarded as a generalization of the weighted binary WBCE (defined in Eqs. (32), (33)), theoretically, the performance based on optimizing this DEM should be better than based on optimizing the W-BCE due to the measurement consistency in both training and testing. We carried out experiments to test model performance when the model was optimized with DEM and WBCE based objective functions, and show the results in Tables VIII and IX for the development and evaluation sets of SITW, respectively. In these two tables, the model parameters are initialized from the JB based generative model, and re-trained based on the two objective functions (setting prior of target trials to 0.01). From these two tables, we can confirm that the direct evaluation metric is much more suitable in discriminative training for the SV tasks especially in terms of minDCF.

2329-9290 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: Indian Institute of TechnologyDharwad. Downloaded on November 23,2021 at 14:36:55 UTC from IEEE Xplore. Restrictions apply.

TABLE IX
PERFORMANCE WITH DIFFERENT OPTIMIZATION OBJECTIVE FUNCTIONS (ON THE EVALUATION SET OF SITW).

Objective functions	EER (%)	minDCF (0.01)	minDCF (0.001)
WBCE (Eqs. 32, 33)	**3.089**	0.3567	0.6163
DEM (Eqs. 29, 31)	3.142	**0.3075**	**0.4619**

TABLE X
PERFORMANCE ON THE EVALUATION SET OF VOXCELEB1 TEST (X-VECTORS EXTRACTED FROM ECAPA-TDNN [50]).

Methods	EER (%)	minDCF (0.01)	minDCF (0.001)
LDA (150)+JB	1.280	0.2013	0.3287
LDA (180)+JB	1.186	0.1844	0.3055
LDA (192)+JB	1.170	0.1858	0.2846
Hybrid (JB init)	**0.803**	**0.0919**	**0.1249**

IV. DISCUSSION

As a detection task, the performance of SV could be benefitted from many aspects. For example, effective neural network architectures for X-vector extraction [42], [43], [44], [45], advanced data augmentation for speaker classification training in robust X-vector extraction [5], [46], borrowing the idea of better margin based objective functions from [47] for training speaker embedding models [48], [49]. Particularly, integrating attention models with the most advanced techniques for X-vector extraction shows significant improvement in SV tasks [50]. The improvement is largely due to the effective exploration of the speaker discriminative information in modeling and learning. Since our proposed discriminative learning framework in this paper is also for enhancing the discriminative power for SV, it is natural to wander: whether the proposed hybrid backend is still effective or not when strong X-vector features are used as inputs. We carried out additional experiments by using X-vectors extracted from ECAPA-TDNN [50] as inputs to our proposed framework. The ECAPA-TDNN was trained using Voxceleb (training sets 1 and 2). Different from the X-vectors extracted in [5] (with dimension 512), the dimension of X-vectors extracted from ECAPA-TDNN is 192. Before designing the hybrid neural backend framework, we first examined the effective dimensions as used in LDA. The results are showed in Table X (as "LDA (dimension)+JB"). From this table, we can see that using full dimensions achieved the best performance. Therefore, in designing the Siamese neural network for backend modeling according to Fig. 3, the dimensions for LDA_net and JB_net were set to 192 neural nodes. All other settings were kept the same as used in experiments in Section III. The results are showed as "Hybrid (JB init)" in Table X. From these results, we can see that there is a large improvement by using the ECAPA-TDNN based X-vector extraction system, and the proposed neural network based backend still provided significant improvement on this strong baseline system. Our results were better or comparable to the best settings in [50] (EER (%)= 0.87, minDCF (0.01)=0.1066) (please note that the settings in training the X-vector model, especially the backend pipelines were different).

V. CONCLUSION AND FUTURE WORK

The current state of the art pipeline for SV is composed of two building models, i.e., a front-end model for speaker feature extraction, and a generative model based backend model for speaker verification. In this study, the X-vector as a speaker embedding feature is extracted in the front-end model which encodes strong speaker discriminative information. Based on this speaker feature, a JB based generative backend model is applied. The JB model tries to model the probability distributions of speaker features, and could predict the conditional probabilities for utterances even from unknown speakers. But as a generative model, the parameter estimation can be easily distracted with nuisance features in a high dimensional space. As an alternative, the SV task can be also regarded as a binary classification task. Correspondingly, a discriminative learning framework can be applied with "positive" and "negative" sample pairs (as from the same speaker and different speakers). Under a discriminative learning framework, discriminative features can be automatically transformed and modeled in a unified optimization framework. In this study, as our main contribution, we proposed to couple the generative model structure and parameters with the dense layers of a neural network learning framework as a hybrid model. The key point is that we reformulated the LLR estimation in the JB model to a distance metric as used in the discriminative learning framework. In particular, the linear matrices in the JB model are factorized to be the linear affine transforms as implemented in dense layers of the neural network model. And the network parameters are connected to the JB model parameters so that they could be initialized by the generatively learned parameters. Moreover, as our second contribution to the discriminative learning framework, rather than simply learning the hybrid model with a conventional binary discrimination objective function, the direct evaluation metric for hypothesis test with consideration of false alarm and miss errors was applied as an objective function in parameter optimization learning.

In this study, the JB based generative model is based on simple Gaussian probability distribution assumptions of speaker features and noise. In real applications, the probability distributions are much more complex. Although it is difficult for a generative model to fit complex shapes of probability distributions in a high dimensional space, it is relatively easy for a discriminative learning framework to approximate the complex distribution shapes. In the future, we will extend the current study for a hybrid model framework to learn more complex probability distributions in SV tasks.

VI. ACKNOWLEDGMENT

Authors would like to thank the anonymous reviewers for helping us in paper revision. The work is partially supported by JSPS KAKENHI No. 19K12035, No. 21K17776.

REFERENCES

[1] J. Hansen, T. Hasan, "Speaker recognition by machines and humans: A tutorial review," *IEEE Signal processing magazine*, vol. 32, no. 6, pp. 74-99, 2015.
[2] H. Beigi, Fundamentals of Speaker Recognition, Springer-Verlag, Berlin, 2011, ISBN 978-0-387-77591-3.

[3] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, “Front-end factor analysis for speaker verification,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788-798, 2011.

[4] E. Variani, X. Lei, E. McDermott, I. L. Moreno and J. Gonzalez-Dominguez, “Deep neural networks for small footprint text-dependent speaker verification,” *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4052-4056, 2014.

[5] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, “X-vectors: Robust DNN embeddings for speaker recognition,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5329-5333, 2018.

[6] M. Sugiyama, ”Local Fisher discriminant analysis for supervised dimensionality reduction,” in *Proc. of the 23rd international conference on Machine learning (ICML)*, pp. 905-912, 2006.

[7] P. Shen, X. Lu, L. Liu, H. Kawai, “Local fisher discriminant analysis for spoken language identification,” in *Proc. of ICASSP*, pp. 5825-5829, 2016.

[8] S. Prince and J. Elder, “Probabilistic linear discriminant analysis for inferences about identity,” in *IEEE International Conference on Computer Vision (ICCV)*, pp. 1-8, 2007.

[9] A. Sizov, K. Lee, and T. Kinnunen, “Unifying probabilistic linear discriminant analysis variants in biometric authentication,” in Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR). Springer, pp. 464-475, 2014.

[10] P. Kenny, “Bayesian speaker verification with heavy tailed priors,” in Odyssey Speaker and Language Recognition Workshop: 14, 2010.

[11] P. Kenny, T. Stafylakis, P. Ouellet, M. J. Alam and P. Dumouchel, “PLDA for speaker verification with utterances of arbitrary duration,” in *Proc. of ICASSP*, pp. 7649-7653, 2013.

[12] D. Chen, X. Cao, L. Wang, F. Wen, and J. Sun, “Bayesian face revisited: A joint formulation,” in *European Conference on Computer Vision*, pp. 566-579, 2012.

[13] D. Chen, X. Cao, D. Wipf, F. Wen, and J. Sun, “An efficient joint formulation for Bayesian face verification,” *IEEE Transactions on pattern analysis and machine intelligence*, vol. 39, pp. 32-46, 2016.

[14] Yiyan Wang, Haotian Xu, Zhijian Ou, ”Joint Bayesian Gaussian Discriminant Analysis for speaker verification,” in *Proceeding of ICASSP*, pp. 5390-5394, 2017.

[15] A. Nagrani, J. Chung, J. Huh, A. Brown, E. Coto, W. Xie, M. McLaren, D. Reynolds and A. Zisserman, ”VoxSRC 2020: The Second VoxCeleb Speaker Recognition Challenge,” https://www.robots.ox.ac.uk/ vgg/data/voxceleb/competition2020.html.

[16] V. Wan, W. Campbell, “Support vector machines for speaker verification and identification,” Neural Networks for Signal Processing X, in *Proceedings of the IEEE Signal Processing Society Workshop*, vol. 2, pp. 775-784, 2000.

[17] W. Campbell, D. Sturim, and D. Reynolds, “Support vector machines using GMM supervectors for speaker verification,” IEEE signal processing letters, vol. 13, no. 5, pp. 308-311, 2006.

[18] J. Villalba, N. Brummer, N. Dehak, “Tied variational autoencoder backends for i-vector speaker recognition,” in *Proc of INTERSPEECH*, pp. 1004-1008, 2017.

[19] L. Burget, O. Plchot, S. Cumani, O. Glembek, P. Matejka and N. Brummer, “Discriminatively trained Probabilistic Linear Discriminant Analysis for speaker verification,” in *Proceeding of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4832-4835, 2011.

[20] S. Cumani, N. Brummer, L. Burget, P. Laface, O. Plchot and V. Vasilakakis, “Pairwise Discriminative Speaker Verification in the I-Vector Space,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 6, pp. 1217-1227, June 2013.

[21] G. Heigold, I. Moreno, S. Bengio, and N. Shazeer, “End-to-end text dependent speaker verification,” in *Proceeding of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 5115-5119, 2016.

[22] L. Wan, Q. Wang, A. Papir and I.Moreno, “Generalized End-to-End Loss for Speaker Verification,” in *Proceeding of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4879-4883, 2018.

[23] L. Ferrer, M. Mclaren, “A Speaker Verification Backend for Improved Calibration Performance across Varying Conditions,”. in *Proc. of Odyssey, the Speaker and Language Recognition Workshop*, pp. 372-379, 2020.

[24] L. Ferrer, M. McLaren, N. Brummer, “A speaker verification backend with robust performance across conditions,” *arXiv:2102.01760*, 2021.

[25] J. Rohdin, A. Silnova, M. Díez, O. Plchot, P. Matejka, L. Burget, O. Glembek, “End-to-end DNN based text-independent speaker recognition for long and short utterances,” Comput. Speech Lang., vol. 59, pp. 22-35, 2020.

[26] S. Ramoji, P. Krishnan, S. Ganapathy, “Neural PLDA Modeling for End-to-End Speaker Verification,” in *INTERSPEECH*, 2020.

[27] A. Lasserre, C. Bishop, T. Minka, “Principled Hybrids of Generative and Discriminative Models,” in *Proceeding of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 87-94, 2006.

[28] N. Brummer, E. Villiers, “The BOSARIS toolkit user guide: Theory, algorithms and code for binary classifier score processing,” Documentation of BOSARIS toolkit, 2011.

[29] E. Lehmann, J Romano, Testing Statistical Hypotheses, Springer-Verlag New York, 2005.

[30] J. Platt, “Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods,” *Advances in large margin classifiers*, pp. 61-74, 1999.

[31] H. Lin, C. Lin, R. Weng, “A note on Platt’s probabilistic outputs for support vector machines,” *Machine Learning*, vol. 68, pp. 267-276, 2007.

[32] X. Lu, P. Shen, Y. Tsao, H. Kawai, “Regularization of neural network model with distance metric learning for i-vector based spoken language identification,” *Computer Speech and Language*, vol.44, pp. 48-60, 2017.

[33] D. Garcia-Romero and C. Y. Espy-Wilson, “Analysis of i-vector length normalization in speaker recognition systems,” in *Proc. of INTERSPEECH*, pp. 249-252, 2011.

[34] M. McLaren, L. Ferrer, D. Castan, and A. Lawson, “The speakers in the wild (SITW) speaker recognition database,” in *Proc. of INTERSPEECH*, pp. 818-822, 2016.

[35] A. Nagrani, J. Chung, W. Xie, A. Zisserman, “Voxceleb: Large-scale speaker verification in the wild,” *Computer Science and Language*, vol. 60, 2020.

[36] https://github.com/kaldi-asr/kaldi/tree/master/egs/voxceleb/v2

[37] K. He, X. Zhang, S. Ren, J. Sun, “Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification,” in *Proceeding of IEEE International Conference on Computer Vision (ICCV)*, pp. 1026-1034, 2015.

[38] D. P. Kingma, J. Ba, “Adam: A Method for Stochastic Optimization,” *the 3rd International Conference on Learning Representations (ICLR)*, 2014.

[39] E. Xing, A. Ng, M. Jordan, and R. Russell, “Distance Metric Learning, with application to Clustering with side-information,” *in Proceeding of Advances in Neural Information Processing Systems*, MIT Press, pp. 521-528, 2002.

[40] K. Weinberger, L. Saul, “Distance Metric Learning for Large Margin Classification,” *Journal of Machine Learning Research*, vol. 10, pp. 207-244, 2009.

[41] B. Moghaddam, T. Jebara, A. Pentland, “Bayesian face recognition,” *Pattern Recognition*, vol. 33, pp. 1771-1782, 2000.

[42] J. S. Chung, A. Nagrani, and A. Zisserman, “Voxceleb2: Deep speaker recognition,” in INTERSPEECH, 2018.

[43] D. Garcia-Romero, A. McCree, D. Snyder, and G. Sell, “JHUHLTCOE system for the VoxSRC speaker recognition challenge,” in *Proc. of ICASSP*, pp. 7559-7563, 2020.

[44] H. Zeinali, S. Wang, A. Silnova, P. Matejka, and O. Plchot, “BUT system description to VoxCeleb speaker recognition challenge 2019,” arXiv:1910.12592

[45] L. Chen, K. Lee, L. He, F. Soong, “On Early-stop Clustering for Speaker Diarization,” in *Proc. of Odyssey, The Speaker and Language Recognition Workshop*, pp. 110-116, 2020.

[46] D. Raj, D. Snyder, D. Povey, S. Khudanpur, “Probing the Information Encoded in X-Vectors,” *IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2019.

[47] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, “ArcFace: Additive angular margin loss for deep face recognition,” in *IEEE/CVPR*, pp. 4685-4694, 2019.

[48] X. Xiang, S. Wang, H. Huang, Y. Qian, K. Yu, “Margin Matters: Towards More Discriminative Deep Neural Network Embeddings for Speaker Recognition,” in *Proc. of APSIPA*, pp. 1652-1656, 2019.

[49] J. Chung, J. Huh, S. Mun, M. Lee, H. Heo, S. Choe, C. Ham, S. Jung, B. Lee, I. Han, “In Defence of Metric Learning for Speaker Recognition,” in *Proc. of INTERSPEECH*, pp. 2977-2981, 2020.

[50] B. Desplanques, J. Thienpondt, and K. Demuynck, “ECAPA-TDNN: Emphasized Channel Attention, Propagation and Aggregation in TDNN Based Speaker Verification,” in *Proc. INTERSPEECH*, pp. 3830-3834, 2020.