Odyssey 2018 The Speaker and Language Recognition Workshop
26-29 June 2018, Les Sables d'Olonne, France

Adversarial Learning and Augmentation for Speaker Recognition

Jen-Tzung Chien *Kang-Ting Peng*

Department of Electrical and Computer Engineering
National Chiao Tung University, Taiwan
jtchien@nctu.edu.tw ktpeng@chien.cm.nctu.edu.tw

Abstract

This paper develops a new generative adversarial network (GAN) to artificially generate i-vectors to deal with the issue of unbalanced or insufficient data in speaker recognition based on the probabilistic linear discriminant analysis (PLDA). Data augmentation is performed to improve system robustness over the variations of i-vectors under different number of training utterances. Our idea is to incorporate the class label into GAN which involves a minimax optimization problem for adversarial learning. We build a generator and a discriminator where the class conditional i-vectors are produced by the generator such that the discriminator can not distinguish them as the fake samples. In particular, multiple learning objectives are optimized to build a specialized deep model for model regularization in speaker recognition. In addition to the minimax optimization of adversarial loss, the posterior probabilities of class labels given real and fake samples are maximized. The cosine similarity between real and fake i-vectors is also minimized to preserve the quality of the generated i-vector. The loss functions for data reconstruction and Gaussian regularization in PLDA model are minimized. The experiments illustrate the merit of multi-objective learning for deep adversarial augmentation for speaker recognition.

1. Introduction

Speaker recognition using i-vectors [1] as the speaker features and probabilistic linear discriminant analysis (PLDA) [2] as the scoring function has achieved state-of-the-art performance in different tasks. PLDA is a linear model which is trained under the assumption that the utterances of the same speaker share a common low dimensional latent variable space. The corresponding i-vectors are efficiently represented in that space. In general, there are two issues which considerably constrain the performance of i-vector combined with PLDA for speaker recognition. First, i-vector may not sufficiently reflect speaker identify due to a mixing condition with the other factors, e.g. channel, noise, language, duration, etc. This issue may be mitigated by conducting the length normalization [3] in the extracted i-vectors which includes the stages of centering and whitening. Second, the number of training utterances or i-vectors of a target speaker may be insufficient and may be varied from that of the other enrolled speakers. The trained PLDA model is deteriorated with such a sparse and unbalanced data problem. To deal with this weakness, a possible solution is to artificially generate the i-vectors to balance and enrich the distribution of training samples for different speakers. This study presents a deep adversarial learning to build a novel generative model to fulfill data augmentation and improve the learning representation to enhance the robustness of speaker recognition under varying conditions of i-vectors for different speakers.

Considering the expanding and emerging researches on deep learning with generative adversarial network (GAN) [4,5], we are motivated to investigate the powerfulness of GAN in finding a deep generative model which samples the informative i-vectors to enhance the data variations and tackle the sparse and unbalanced data problem in PLDA-based speaker recognition. A two-player game is realized to optimize the generator for the fake i-vectors which are difficult to tell from the real i-vectors through a discriminator. The layer-wise generator and discriminator are jointly estimated by solving a minimax optimization problem based on stochastic gradient descent (SGD) algorithm. A distribution model is trained and used to sample new i-vectors in line with real i-vectors from random noise samples. This study implements the auxiliary classifier GAN [6], which is a variant of conditional GAN [7], and incorporates the class label into maximization of class conditional likelihoods of real i-vectors as well as fake i-vectors. The structural information is embedded in the latent space of GAN. To boost the performance of i-vector augmentation, we additionally minimize the cosine similarity between fake i-vectors and real i-vectors in either the observation space or the latent feature space. Moreover, we tightly merge the PLDA scoring into GAN by further optimizing the PLDA scoring for data reconstruction and meeting the Gaussian regularization in PLDA assumption. Experiments on NIST i-vector Speaker Recognition Challenge show how this multi-objective adversarial learning works for data augmentation in PLDA-based speaker verification.

This paper is organized in the following. Section 2 addresses the basics of PLDA for speaker recognition and GAN for data generation. Section 3 details the adversarial learning and augmentation where the cosine similarity in latent features and the reconstruction error due to PLDA model are incorporated into learning objectives. Section 4 reports a set of experiments on speaker recognition to evaluate the effect of latent features as well as the recognition performance with different conditions. Finally, a summary from this study in given in Section 5.

2. Background Survey

This section introduces the fundamentals of speaker recognition and adversarial learning which are integrated to carry out an advanced and specialized solution to data augmentation based on PLDA model.

2.1. Probabilistic linear discriminant analysis

PLDA [2] is a generative model which characterizes both intra-speaker and inter-speaker variations via a linear model

$$\mathbf{x}_{ij} = \mathbf{m} + \mathbf{V}\mathbf{z}_i + \boldsymbol{\epsilon}_{ij} \quad (1)$$

10.21437/Odyssey.2018-48

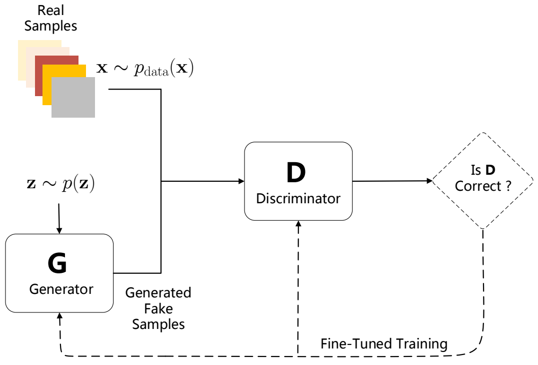


Figure 1: Illustration for generative adversarial network.

where feature vector or i-vector $\mathbf{x}_{ij} \in \mathcal{R}^D$ of speaker i in a session j is represented by factor analysis (FA) with a mean vector $\mathbf{m}$, a factor loading matrix $\mathbf{V} \in \mathcal{R}^{D\times d}$, a common vector $\mathbf{z}_i \in \mathcal{R}^d$ and a residual vector $\boldsymbol{\epsilon}_{ij}$. FA assumes latent vector $\mathbf{z}_i$ and residual vector $\boldsymbol{\epsilon}$ are Gaussian distributed by

$$\mathbf{z}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \qquad \boldsymbol{\epsilon}_{ij} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}) \tag{2}$$

with $d \times d$ identity covariance matrix $\mathbf{I}$ and $D \times D$ covariance matrix $\boldsymbol{\Sigma}$, respectively [8]. PLDA parameters are formed by $\boldsymbol{\theta} = \{\mathbf{m}, \mathbf{V}, \boldsymbol{\Sigma}\}$ which are estimated by maximizing the marginal likelihood function with respect to latent variable $\mathbf{z}_i$ over individual i-vectors [9]

$$\begin{aligned} p(\mathbf{x}_{ij}|\boldsymbol{\theta}) &= \int \mathcal{N}(\mathbf{x}_{ij}|\mathbf{m} + \mathbf{V}\mathbf{z}_i, \boldsymbol{\Sigma})\mathcal{N}(\mathbf{z}_i|\mathbf{0}, \mathbf{I})d\mathbf{z}_i \\ &= \mathcal{N}(\mathbf{x}_{ij}|\mathbf{m}, \mathbf{V}\mathbf{V}^\top + \boldsymbol{\Sigma}) \end{aligned} \tag{3}$$

according to the expectation-maximization (EM) algorithm [10]. E-step is to calculate the posterior probability $p(\mathbf{z}_i|\mathcal{X}, \boldsymbol{\theta}^{\text{old}})$ due to latent vector $\mathbf{z}_i$ by using a training set of i-vectors $\mathcal{X} = \{\mathbf{x}_{ij}\}$ given old parameters $\boldsymbol{\theta}^{\text{old}}$. Using this posterior probability, M-step is to estimate new PLDA parameters $\boldsymbol{\theta}^{\text{new}}$ by maximizing an auxiliary function $Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{\text{old}})$. This function is also viewed as the lower bound $\mathcal{L}(q, \boldsymbol{\theta})$ of the log likelihood function $p(\mathcal{X}|\boldsymbol{\theta})$ where an approximate distribution $q(\mathbf{z}_i) = p(\mathbf{z}_i|\mathcal{X}, \boldsymbol{\theta}^{\text{old}})$ from E-step is merged. Iterative EM steps are performed to find the converged parameters $\boldsymbol{\theta}$. In test phase, PLDA score is calculated for speaker verification according to a likelihood ratio test. We evaluate whether a test speaker's i-vector $\mathbf{x}_s$ and target speaker's i-vector $\mathbf{x}_t$ are from the same speaker or not. The joint Gaussian distribution and the individual Gaussian distributions for $\mathbf{x}_s$ and $\mathbf{x}_t$ are calculated under null hypothesis and alternative hypothesis based on PLDA, respectively. PLDA is recognized as a powerful approach to speaker recognition with different extensions [9, 11, 12]. Nevertheless, the training performance of PLDA is constrained because the number of i-vectors tends to be sparse in various speakers. The generative adversarial network (GAN) provides an attractive solution to construct a high-performance generative model for data augmentation.

2.2. Generative adversarial network

The underlying theory of generative model based on GAN is illustrated in Figure 1. GAN is formulated as a two-player game between a generator with mapping function

$$G(\mathbf{z}) : \mathbf{z} \to \mathbf{x} \tag{4}$$

and a binary discriminator with the mapping

$$D(\mathbf{x}) : \mathbf{x} \to [0, 1] \tag{5}$$

via a minimax optimization over a value function $V(G, D)$ or an adversarial loss $\mathcal{L}_{\text{adv}}$. This optimization problem is expressed by

$$\min_G \max_D V(G, D) \tag{6}$$

where

$$\begin{aligned} V(G, D) &= \mathbb{E}_{\mathbf{x}\sim p_{\text{data}}(\mathbf{x})}[\log D(\mathbf{x})] \\ &\quad + \mathbb{E}_{\mathbf{z}\sim p(\mathbf{z})}[\log(1 - D(G(\mathbf{z})))] \\ &\triangleq \mathcal{L}_{\text{adv}} \end{aligned} \tag{7}$$

In Eq. (7), $p_{\text{data}}(\mathbf{x})$ denotes the data distribution while $p(\mathbf{z})$ is a distribution that draws a noise sample or latent code $\mathbf{z}$. $G(\mathbf{z})$ generates a sample $\mathbf{x}$ from $\mathbf{z}$. The discriminator produces a probability $0 \leq D(\mathbf{x}) \leq 1$ which measures how likely a data point $\mathbf{x}$ is sampled from the data distribution $p_{\text{data}}(\mathbf{x})$ as a real sample or from the generator $G(\mathbf{z})$ (or a distribution of generator $p_{\text{gen}}(\mathbf{x})$) as a fake sample. Generator $G(\mathbf{z})$ and discriminator $D(\mathbf{x})$ are both realized as the fully-connected neural network models. The value function $V(G, D)$ is seen as a negative cross entropy error function for two-player game. This minimax optimization assures the *worst* performance in classifying a fake sample into a real sample. A powerful generator model $G(\mathbf{z})$ is therefore implemented. The minimax problem in Eq. (6) was solved to find an optimal discriminator

$$D^*(\mathbf{x}) = \frac{p_{\text{data}}(\mathbf{x})}{p_{\text{data}}(\mathbf{x}) + p_{\text{gen}}(\mathbf{x})} \tag{8}$$

which was obtained for any given generator G with a distribution $p_{\text{gen}}(\mathbf{x})$ [4]. A global optimum happens in the condition

$$p_{\text{gen}}(\mathbf{x}) = p_{\text{data}}(\mathbf{x}) \tag{9}$$

which results in the worst performance in binary classification, i.e.

$$D^*(\mathbf{x})\big|_{p_{\text{gen}}(\mathbf{x})=p_{\text{data}}(\mathbf{x})} = 0.5. \tag{10}$$

By substituting this discriminator into value function, the estimation of generator G turns out to solve a minimization problem [4]

$$\begin{aligned} &\min_G V(D^*, G) \\ &= \min_G \ \{2\mathcal{D}_{\text{JS}}(p_{\text{data}}(\mathbf{x})\|p_{\text{gen}}(\mathbf{x})) - \log 4\} \end{aligned} \tag{11}$$

where $\mathcal{D}_{\text{JS}}$ denotes the Jensen-Shannon (JS) divergence. Accordingly, the optimal generator G^* is calculated to reflect the generator distribution $p_{\text{gen}}(\mathbf{x})$ which is closest to the real-data distribution $p_{\text{data}}(\mathbf{x})$. GAN encourages G to fit $p_{\text{data}}(\mathbf{x})$ so as to fool D with its generated samples. G and D are trained to update the parameters of both models by *error backpropagation* algorithm.

In practice, the quality of the generated samples $\widehat{\mathbf{x}}$ via GAN can be improved by introducing class label $\mathbf{c}$ as the side information in both generator and discriminator so as to produce the class conditional samples [7]. In [6], the auxiliary classifier GAN (AC-GAN) was proposed to feed side information into the generator and force the discriminator to reconstruct this side information. To do so, the discriminator was modified by merging an auxiliary decoder network which produces the class output for training data. As illustrated in Figure 2, each generated sample $\widehat{\mathbf{x}}$ contains a class label $\mathbf{c}$ as well as a noise sample $\mathbf{z}$. The

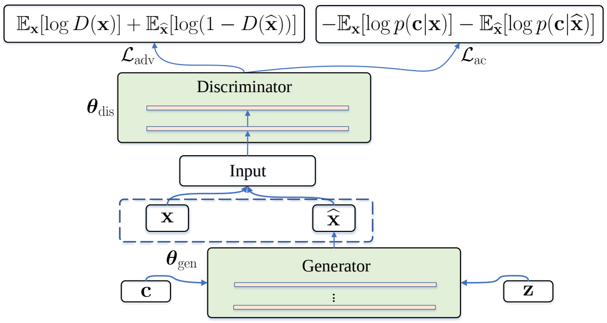

$\mathbb{E}_{\mathbf{x}}[\log D(\mathbf{x})] + \mathbb{E}_{\widehat{\mathbf{x}}}[\log(1 - D(\widehat{\mathbf{x}}))]$
$-\mathbb{E}_{\mathbf{x}}[\log p(\mathbf{c}|\mathbf{x})] - \mathbb{E}_{\widehat{\mathbf{x}}}[\log p(\mathbf{c}|\widehat{\mathbf{x}})]$
$\mathcal{L}_{\text{adv}}$
$\mathcal{L}_{\text{ac}}$
Discriminator
$\boldsymbol{\theta}_{\text{dis}}$
Input
$\mathbf{x}$
$\widehat{\mathbf{x}}$
$\boldsymbol{\theta}_{\text{gen}}$
Generator
$\mathbf{c}$
$\mathbf{z}$

Figure 2: Calculation of loss functions $\mathcal{L}_{\text{adv}}$ and $\mathcal{L}_{\text{ac}}$ for AC-GAN.

generator $G(\mathbf{z}, \mathbf{c})$ produces a sample based on two inputs $\mathbf{z}$ and $\mathbf{c}$. The probability distributions over class label $\mathbf{c}$ given source samples are introduced to carry out an additional loss function

$$\begin{aligned}\mathcal{L}_{\text{ac}} = &-\mathbb{E}_{\mathbf{x}\sim p_{\text{data}}(\mathbf{x})}[\log p(\mathbf{c}|\mathbf{x})] \\ &- \mathbb{E}_{\widehat{\mathbf{x}}\sim p_{\text{gen}}(\mathbf{x})}[\log p(\mathbf{c}|\widehat{\mathbf{x}})].\end{aligned} \tag{12}$$

Discriminator is trained by maximizing $\mathcal{L}_{\text{adv}}$ while generator is trained by minimizing $\mathcal{L}_{\text{adv}}$. Both generator with parameters $\boldsymbol{\theta}_{\text{gen}}$ and discriminator with parameters $\boldsymbol{\theta}_{\text{dis}}$ are trained by minimizing $\mathcal{L}_{\text{ac}}$. In what follows, we develop an adversarial data augmentation method for speaker recognition using i-vectors based on the PLDA speaker model.

3. Adversarial Augmentation and Regularization

This study introduces new GANs to generate i-vectors to tackle the unbalanced and insufficient data problem for speaker recognition. We aim to balance the development data among different speakers. Considering the importance of speaker identity, we carry out AC-GAN and build a speaker-dependent generative model for i-vector augmentation where speaker label $\mathbf{c}$ as well as noise sample $\mathbf{z}$ are incorporated in minimax optimization. In the implementation, the first term and the second term in Eq. (12) express the class conditional likelihoods for real i-vectors

$$\mathbb{E}[\log p(\mathbf{c}|\mathbf{x}_{\text{real}})]$$

and fake i-vectors

$$\mathbb{E}[\log p(\mathbf{c}|\mathbf{x}_{\text{fake}})]$$

respectively. However, direct implementation of AC-GAN for data augmentation does not work well for speaker recognition. Some additional regularization methods should be taken into account. To synthesize the informative i-vectors for PLDA-based speaker recognition, we propose two extensions which tightly integrate different *regularization factors* into optimization procedure.

3.1. Regularization with cosine similarity

Similar to using likelihood ratio test in PLDA model, it is also popular to adopt the cosine similarity as the scoring function to measure the closeness between i-vectors of test speaker and target speaker. Cosine similarity typically achieves desirable performance in speaker recognition. This study is motivated by regularizing the generation of i-vectors via matching with real i-vectors. A loss function for model regularization based on the cosine similarity between real i-vector $\mathbf{x}$ and fake i-vector $\widehat{\mathbf{x}}$ is constructed by

$$\mathcal{L}_{\text{cosx}} = \mathbb{E}_{\mathbf{x}\sim p_{\text{data}},\widehat{\mathbf{x}}\sim p_{\text{gen}}(\mathbf{x})}[\text{cosine}(\mathbf{x}, \widehat{\mathbf{x}})]. \tag{13}$$

This loss is minimized to assure the model regularization for *distribution matching*.

However, the similarity between real and fake samples measured in i-vector space may be distorted because i-vectors contain channel and some other variabilities. The goodness of cosine similarity may be improved by measuring the closeness between real sample and generated sample of i-vectors in latent space. Namely, the loss function of cosine similarity is measured in latent feature space of $\{\mathbf{y}, \widehat{\mathbf{y}}\}$ instead of i-vector space of $\{\mathbf{x}, \widehat{\mathbf{x}}\}$ in a form of

$$\mathcal{L}_{\text{cosy}} = \mathbb{E}_{\mathbf{x}\sim p_{\text{data}},\widehat{\mathbf{x}}\sim p_{\text{gen}}(\mathbf{x})}[\text{cosine}(\mathbf{y}, \widehat{\mathbf{y}})|D(\cdot)]. \tag{14}$$

Since a discriminator $D(\cdot)$ based on deep neural network (DNN) is used to distinguish real i-vector $\mathbf{x}$ and fake i-vector $\widehat{\mathbf{x}}$, we heuristically adopt the corresponding latent vectors $\{\mathbf{y}, \widehat{\mathbf{y}}\}$ in the last layer of discriminator to measure the cosine loss function $\mathcal{L}_{\text{cosy}}$. The cosine similarity is evaluated in latent feature space. Here, the notations Cosx and Cosy denote the regularization of cosine similarity in i-vector space $\mathbf{x}$ and in feature space $\mathbf{y}$. As a result, there are two GANs, Cosx-GAN and Cosy-GAN, which are constructed for adversarial data augmentation. The learning objectives in minimax optimization are constructed by

$$\text{Cosx-GAN:} \qquad \mathcal{L}_{\text{adv}} + \mathcal{L}_{\text{ac}} + \mathcal{L}_{\text{cosx}}$$

and

$$\text{Cosy-GAN:} \qquad \mathcal{L}_{\text{adv}} + \mathcal{L}_{\text{ac}} + \mathcal{L}_{\text{cosy}}$$

Figure 3 shows how four loss functions $\mathcal{L}_{\text{adv}}$, $\mathcal{L}_{\text{ac}}$, $\mathcal{L}_{\text{cosx}}$ and $\mathcal{L}_{\text{cosy}}$ are calculated from generator and discriminator in Cosx-GAN and Cosy-GAN. $\mathcal{L}_{\text{cosy}}$ is calculated by discriminator. Notably, the derivative of cosine similarity with respect to the generated sample $\widehat{\mathbf{x}}$ is required in SGD updating for estimation of model parameters. This derivative is calculated by

$$\frac{\partial}{\partial\widehat{\mathbf{x}}}\text{cosine}(\mathbf{x}, \widehat{\mathbf{x}}) = \frac{\mathbf{x}}{|\mathbf{x}|\,|\widehat{\mathbf{x}}|} - \text{cosine}(\mathbf{x}, \widehat{\mathbf{x}})\frac{\widehat{\mathbf{x}}}{|\widehat{\mathbf{x}}|^2} \tag{15}$$

because

$$\begin{aligned}
&\text{cosine}(\mathbf{x}, \widehat{\mathbf{x}} + d\widehat{\mathbf{x}}) = \frac{\mathbf{x}\cdot\widehat{\mathbf{x}} + \mathbf{x}\cdot d\widehat{\mathbf{x}}}{|\mathbf{x}|\,|\widehat{\mathbf{x}} + d\widehat{\mathbf{x}}|} \\
&\approx \frac{\mathbf{x}\cdot\widehat{\mathbf{x}} + \mathbf{x}\cdot d\widehat{\mathbf{x}}}{|\mathbf{x}|\left(1 + \frac{\widehat{\mathbf{x}}}{|\widehat{\mathbf{x}}|^2}\cdot d\widehat{\mathbf{x}}\right)\,|\widehat{\mathbf{x}}|} \\
&\approx \frac{\mathbf{x}\cdot\widehat{\mathbf{x}} + \mathbf{x}\cdot d\widehat{\mathbf{x}}}{|\mathbf{x}|\,|\widehat{\mathbf{x}}|}\left(1 - \frac{\widehat{\mathbf{x}}}{|\widehat{\mathbf{x}}|^2}\cdot d\widehat{\mathbf{x}}\right) \\
&\approx \frac{\mathbf{x}\cdot\widehat{\mathbf{x}}}{|\mathbf{x}|\,|\widehat{\mathbf{x}}|} + \left(\frac{\mathbf{x}}{|\mathbf{x}|\,|\widehat{\mathbf{x}}|} - \frac{\mathbf{x}\cdot\widehat{\mathbf{x}}}{|\mathbf{x}|\,|\widehat{\mathbf{x}}|}\frac{\widehat{\mathbf{x}}}{|\widehat{\mathbf{x}}|^2}\right)\cdot d\widehat{\mathbf{x}} \\
&\approx \text{cosine}(\mathbf{x}, \widehat{\mathbf{x}}) + \left(\frac{\mathbf{x}}{|\mathbf{x}|\,|\widehat{\mathbf{x}}|} - \text{cosine}(\mathbf{x}, \widehat{\mathbf{x}})\frac{\widehat{\mathbf{x}}}{|\widehat{\mathbf{x}}|^2}\right) d\widehat{\mathbf{x}}.
\end{aligned} \tag{16}$$

Basically, Cosx-GAN and Cosy-GAN are seen as the extended AC-GANs with additional regularization terms of cosine similarity in i-vector space $\mathbf{x}$ and latent feature space $\mathbf{y}$, respectively. Algorithm 1 shows the procedure of implementing Cosx-GAN for finding the DNN parameters for generator $\boldsymbol{\theta}_{\text{gen}}$ and discriminator $\boldsymbol{\theta}_{\text{dis}}$. In the implementation, the discriminator is estimated before the generator. Notably, the loss function of

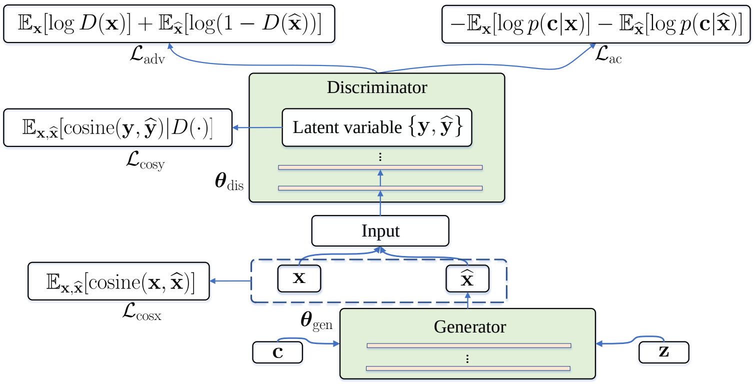

$\mathbb{E}_{\mathbf{x}}[\log D(\mathbf{x})] + \mathbb{E}_{\widehat{\mathbf{x}}}[\log(1 - D(\widehat{\mathbf{x}}))]$
$\mathcal{L}_{\text{adv}}$
$-\mathbb{E}_{\mathbf{x}}[\log p(\mathbf{c}|\mathbf{x})] - \mathbb{E}_{\widehat{\mathbf{x}}}[\log p(\mathbf{c}|\widehat{\mathbf{x}})]$
$\mathcal{L}_{\text{ac}}$
Discriminator
$\mathbb{E}_{\mathbf{x},\widehat{\mathbf{x}}}[\text{cosine}(\mathbf{y}, \widehat{\mathbf{y}})|D(\cdot)]$
Latent variable $\{\mathbf{y}, \widehat{\mathbf{y}}\}$
$\mathcal{L}_{\text{cosy}}$
$\boldsymbol{\theta}_{\text{dis}}$
Input
$\mathbb{E}_{\mathbf{x},\widehat{\mathbf{x}}}[\text{cosine}(\mathbf{x}, \widehat{\mathbf{x}})]$
$\mathbf{x}$
$\widehat{\mathbf{x}}$
$\mathcal{L}_{\text{cosx}}$
$\boldsymbol{\theta}_{\text{gen}}$
Generator
$\mathbf{c}$
$\mathbf{z}$

Figure 3: Calculation of loss functions $\mathcal{L}_{\text{adv}}$, $\mathcal{L}_{\text{ac}}$, $\mathcal{L}_{\text{cosx}}$ and $\mathcal{L}_{\text{cosy}}$ for Cos-GANs.

cosine similarity $\mathcal{L}_{\text{cosx}}$ is only included for estimating the generator rather than discriminator. Minimax optimization is run for discriminator and generator because the sign of adversarial loss function $\mathcal{L}_{\text{adv}}$ is flipped in these two gradients. Algorithm 2 illustrates the implementation procedure for Cosy-GAN. The cosine loss $\mathcal{L}_{\text{cosy}}$ dose not only affect the estimation of discriminator but also that of generator.

Algorithm 1: SGD training for Cosx-GAN. Number of training steps for discriminator k. $k = 3$ is used. SGD with momentum is performed for parameter updating.

```
Initialize the parameters $\{\boldsymbol{\theta}_{\text{dis}}, \boldsymbol{\theta}_{\text{gen}}\}$
for number of training iterations do
    for $k$ steps do
        Sample a minibatch with $m$ i-vector examples
        $\{\mathbf{x}_1, \dots, \mathbf{x}_m\}$ from $p(\mathbf{x})$
        Sample a minibatch with $m$ noise examples
        $\{\mathbf{z}_1, \dots, \mathbf{z}_m\}$ from $p(\mathbf{z})$
        Update the discriminator by ascending its
        stochastic gradient $\frac{1}{m}\sum_{i=1}^{m} \nabla_{\boldsymbol{\theta}_{\text{dis}}}(\mathcal{L}_{\text{ac}} - \mathcal{L}_{\text{adv}})$
    end
    Sample a minibatch of $m$ i-vector examples
    $\{\mathbf{x}_1, \dots, \mathbf{x}_m\}$ from $p(\mathbf{x})$
    Sample a minibatch of $m$ noise examples
    $\{\mathbf{z}_1, \dots, \mathbf{z}_m\}$ from $p(\mathbf{z})$
    Update the generator by ascending its stochastic
    gradient $\frac{1}{m}\sum_{i=1}^{m} \nabla_{\boldsymbol{\theta}_{\text{gen}}}(\mathcal{L}_{\text{ac}} + \mathcal{L}_{\text{adv}} - \mathcal{L}_{\text{cosx}})$
end
```

3.2. Regularization with PLDA reconstruction

The second extension is proposed by seamlessly combining the assumption of PLDA into adversarial learning and augmentation. The combination is driven by the variational autoencoder (VAE) [13,14] where the additional regularization terms are formulated and imposed to generate the meaningful i-vectors for PLDA-based speaker recognition. Different from Cosy-GAN using a standard neural network with the *deterministic* latent features $\mathbf{y}$ to calculate cosine loss function $\mathcal{L}_{\text{cosy}}$ using discriminator $D(\cdot)$, the proposed PLDA-Cos-GAN adopts the PLDA scoring to realize a *stochastic* neural network and carry out the cosine similarity and other regularization terms. We basically represent the latent code of i-vector $\mathbf{x}$ or $\widehat{\mathbf{x}}$ using an encoder

Algorithm 2: SGD training for Cosy-GAN

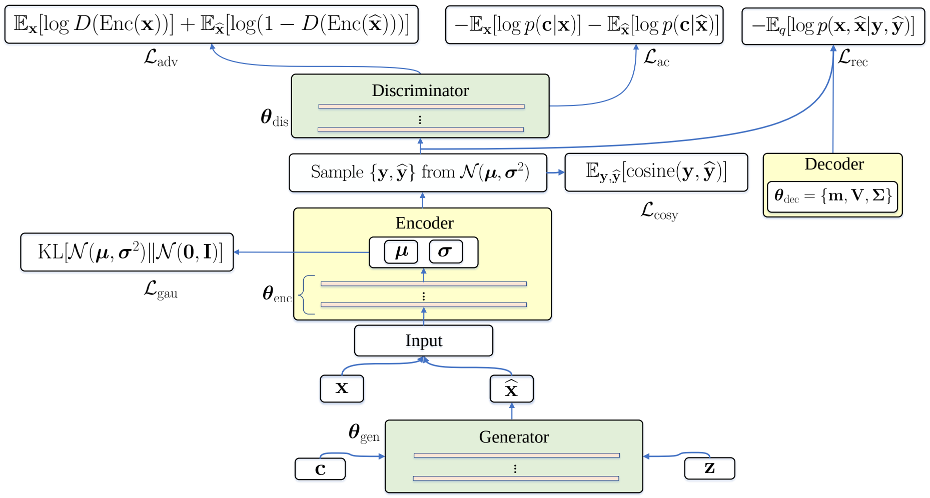
```
Initialize the parameters $\{\boldsymbol{\theta}_{\text{dis}}, \boldsymbol{\theta}_{\text{gen}}\}$
for number of training iterations do
    for $k$ steps do
        Sample a minibatch with $m$ i-vector examples
        $\{\mathbf{x}_1, \dots, \mathbf{x}_m\}$ from $p(\mathbf{x})$
        Sample a minibatch with $m$ noise examples
        $\{\mathbf{z}_1, \dots, \mathbf{z}_m\}$ from $p(\mathbf{z})$
        Update the discriminator by ascending its
        stochastic gradient
        $\frac{1}{m}\sum_{i=1}^{m} \nabla_{\boldsymbol{\theta}_{\text{dis}}}(\mathcal{L}_{\text{ac}} - \mathcal{L}_{\text{adv}} + \mathcal{L}_{\text{cosy}})$
    end
    Sample a minibatch of $m$ i-vector examples
    $\{\mathbf{x}_1, \dots, \mathbf{x}_m\}$ from $p(\mathbf{x})$
    Sample a minibatch of $m$ noise examples
    $\{\mathbf{z}_1, \dots, \mathbf{z}_m\}$ from $p(\mathbf{z})$
    Update the generator by ascending its stochastic
    gradient $\frac{1}{m}\sum_{i=1}^{m} \nabla_{\boldsymbol{\theta}_{\text{gen}}}(\mathcal{L}_{\text{ac}} + \mathcal{L}_{\text{adv}} - \mathcal{L}_{\text{cosy}})$
end
```

with variational distribution $\mathbf{y} \sim q(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\boldsymbol{\mu}(\mathbf{x}), \boldsymbol{\sigma}^2(\mathbf{x}))$ which is Gaussian with i-vector dependent mean $\boldsymbol{\mu}(\mathbf{x})$ and variance $\boldsymbol{\sigma}^2(\mathbf{x})$ via a DNN mapping network with parameters $\boldsymbol{\theta}_{\text{enc}}$. A decoder based on PLDA using parameters $\boldsymbol{\theta}_{\text{dec}} = \{\mathbf{m}, \mathbf{V}, \boldsymbol{\Sigma}\}$ is implemented in accordance with VAE which minimizes the variational bound of negative log likelihood using i-vector $\mathbf{x}$ [14]

$$-\log p(\mathbf{x}) \leq \mathcal{L}_{\text{gau}} + \mathcal{L}_{\text{rec}} \tag{17}$$

where $\mathcal{L}_{\text{gau}} = \text{KL}(q(\mathbf{y}|\mathbf{x}) \| p(\mathbf{y}))$ and $\mathcal{L}_{\text{rec}} = -\mathbb{E}_{q(\mathbf{y}|\mathbf{x})}[\log p(\mathbf{x}|\mathbf{y})]$. In Eq. (17), the variational bound is derived because [15]

$$\begin{aligned}
&\log(p(\mathbf{x})) = \int q(\mathbf{y}|\mathbf{x}) \log(p(\mathbf{x})) d\mathbf{y} \\
&= \int q(\mathbf{y}|\mathbf{x}) \log\left(\frac{p(\mathbf{y},\mathbf{x})}{q(\mathbf{y}|\mathbf{x})}\right) d\mathbf{y} + \int q(\mathbf{y}|\mathbf{x}) \log\left(\frac{q(\mathbf{y}|\mathbf{x})}{p(\mathbf{y}|\mathbf{x})}\right) d\mathbf{y} \\
&\geq \int q(\mathbf{y}|\mathbf{x}) \log\left(\frac{p(\mathbf{y})}{q(\mathbf{y}|\mathbf{x})}\right) d\mathbf{y} + \int q(\mathbf{y}|\mathbf{x}) \log(p(\mathbf{x}|\mathbf{y})) d\mathbf{y} \\
&= -\text{KL}(q(\mathbf{y}|\mathbf{x}) \parallel p(\mathbf{y})) + \mathbb{E}_{q(\mathbf{y}|\mathbf{x})}[\log(p(\mathbf{x}|\mathbf{y}))].
\end{aligned} \tag{18}$$



$\mathbb{E}_{\mathbf{x}}[\log D(\mathrm{Enc}(\mathbf{x}))] + \mathbb{E}_{\widehat{\mathbf{x}}}[\log(1 - D(\mathrm{Enc}(\widehat{\mathbf{x}})))]$
$\mathcal{L}_{\text{adv}}$
$-\mathbb{E}_{\mathbf{x}}[\log p(\mathbf{c}|\mathbf{x})] - \mathbb{E}_{\widehat{\mathbf{x}}}[\log p(\mathbf{c}|\widehat{\mathbf{x}})]$
$\mathcal{L}_{\text{ac}}$
$-\mathbb{E}_q[\log p(\mathbf{x}, \widehat{\mathbf{x}}|\mathbf{y}, \widehat{\mathbf{y}})]$
$\mathcal{L}_{\text{rec}}$
Discriminator
$\boldsymbol{\theta}_{\text{dis}}$
Sample $\{\mathbf{y}, \widehat{\mathbf{y}}\}$ from $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$
$\mathbb{E}_{\mathbf{y},\widehat{\mathbf{y}}}[\mathrm{cosine}(\mathbf{y}, \widehat{\mathbf{y}})]$
$\mathcal{L}_{\text{cosy}}$
Decoder
$\boldsymbol{\theta}_{\text{dec}} = \{\mathbf{m}, \mathbf{V}, \boldsymbol{\Sigma}\}$
Encoder
$\boldsymbol{\mu}$
$\boldsymbol{\sigma}$
$\mathrm{KL}[\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)\|\mathcal{N}(\mathbf{0}, \mathbf{I})]$
$\mathcal{L}_{\text{gau}}$
$\boldsymbol{\theta}_{\text{enc}}$
Input
$\mathbf{x}$
$\widehat{\mathbf{x}}$
$\boldsymbol{\theta}_{\text{gen}}$
Generator
$\mathbf{c}$
$\mathbf{z}$

Figure 4: Calculation of loss functions $\mathcal{L}_{\text{adv}}$, $\mathcal{L}_{\text{ac}}$, $\mathcal{L}_{\text{cosy}}$, $\mathcal{L}_{\text{gau}}$ and $\mathcal{L}_{\text{rec}}$ for PLDA-Cos-GAN.

The additional loss functions are constructed and minimized for data reconstruction $\mathcal{L}_{\text{rec}}$ as well as Gaussian regularization $\mathcal{L}_{\text{gau}}$ under PLDA model where the conditional likelihood

$$p(\mathbf{x}|\mathbf{y}) = \mathcal{N}(\mathbf{m} + \mathbf{V}\mathbf{y}, \boldsymbol{\Sigma}) \tag{19}$$

and the prior assumption

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{20}$$

are considered. The regularization term for Gaussianity is obtained by

$$\begin{aligned} &\mathrm{KL}(\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)\|\mathcal{N}(\mathbf{0}, \mathbf{I})) \\ &= \frac{1}{2}\sum_{d=1}^{D}[\mu_d^2 + \sigma_d^2 + \log(\sigma_d^2) - 1] \end{aligned} \tag{21}$$

where D is the dimension of Gaussians. The regularization for PLDA reconstruction is calculated by using the samples $\{\mathbf{y}_{i,l}\}_{l=1}^{L}$ via

$$\begin{aligned} \mathbb{E}_{q(\mathbf{y}|\mathbf{x})}[\log(p(\mathbf{x}|\mathbf{y}))] = -\frac{1}{2}\sum_i\sum_l\bigg[\log|2\pi\boldsymbol{\Sigma}| \\ +(\mathbf{x}_i - \mathbf{m} - \mathbf{V}\mathbf{y}_{i,l})^\top\boldsymbol{\Sigma}^{-1}(\mathbf{x}_i - \mathbf{m} - \mathbf{V}\mathbf{y}_{i,l})\bigg]. \end{aligned} \tag{22}$$

Interestingly, we implement a variant of adversarial autoencoder [16] for PLDA scoring.

Figure 4 shows how five loss functions are calculated in PLDA-Cos-GAN. The loss functions of $\{\mathcal{L}_{\text{gau}}, \mathcal{L}_{\text{cosy}}\}$, $\{\mathcal{L}_{\text{adv}}, \mathcal{L}_{\text{ac}}\}$ and $\mathcal{L}_{\text{rec}}$ are calculated for *encoder*, *discriminator* and *decoder*, respectively. Notably, all these calculations are based on real i-vector $\mathbf{x}$ and fake i-vector $\widehat{\mathbf{x}}$ obtained from *generator*. The Gaussian samples from variational distribution $q(\mathbf{y}|\mathbf{x})$ are used to calculate four objective functions $\{\mathcal{L}_{\text{adv}}, \mathcal{L}_{\text{ac}}, \mathcal{L}_{\text{cosy}}, \mathcal{L}_{\text{rec}}\}$. Different from Cos-GAN, the cosine loss function $\mathcal{L}_{\text{cosy}}$ in PLDA-Cos-GAN is calculated from encoder rather than discriminator. Minimax optimization is fulfilled to estimate PLDA parameters $\boldsymbol{\theta}_{\text{dec}}$ for decoder and DNN parameters $\{\boldsymbol{\theta}_{\text{gen}}, \boldsymbol{\theta}_{\text{enc}}, \boldsymbol{\theta}_{\text{dis}}\}$ for generator, encoder and discriminator, respectively. Algorithm 3 addresses the implementation procedure for estimating PLDA-Cos-GAN parameters for discriminator, encoder, decoder and generator. Different parameters are estimated by using different loss functions. The loss functions used to estimate a set of parameters depend on whether these loss functions are connected to those parameters.

Algorithm 3: SGD training for PLDA-Cos-GAN. Number of training steps for discriminator k. Hyperparameter for adjusting the losses λ. $k = 3$ and $\lambda = 10$ are used. SGD with momentum is performed for parameter updating.

```
Initialize the parameters $\{\boldsymbol{\theta}_{\text{dis}}, \boldsymbol{\theta}_{\text{gen}}, \boldsymbol{\theta}_{\text{enc}}, \boldsymbol{\theta}_{\text{dec}}\}$
for number of training iterations do
    for $k$ steps do
        Sample a minibatch of $m$ examples
          $\{\mathbf{x}_1, \dots, \mathbf{x}_m\}$ from $p(\mathbf{x})$
        Sample a minibatch of $m$ noise examples
          $\{\mathbf{z}_1, \dots, \mathbf{z}_m\}$ from $p(\mathbf{z})$
        Update the discriminator by ascending its
          stochastic gradient $\frac{1}{m}\sum_{i=1}^{m}\nabla_{\boldsymbol{\theta}_{\text{dis}}}\lambda(\mathcal{L}_{\text{ac}} - \mathcal{L}_{\text{adv}})$
        Update the encoder by ascending its stochastic
          gradient
          $\frac{1}{m}\sum_{i=1}^{m}\nabla_{\boldsymbol{\theta}_{\text{enc}}}(-\mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{gau}} + \lambda(\mathcal{L}_{\text{ac}} - \mathcal{L}_{\text{adv}} + \mathcal{L}_{\text{cosy}}))$
        Update the decoder by ascending its stochastic
          gradient $\frac{1}{m}\sum_{i=1}^{m}\nabla_{\boldsymbol{\theta}_{\text{dec}}}(-\mathcal{L}_{\text{rec}})$
    end
    Sample a minibatch of $m$ examples $\{\mathbf{x}_1, \dots, \mathbf{x}_m\}$
      from $p(\mathbf{x})$
    Sample a minibatch of $m$ noise examples
      $\{\mathbf{z}_1, \dots, \mathbf{z}_m\}$ from $p(\mathbf{z})$
    Update the generator by ascending its stochastic
      gradient $\frac{1}{m}\sum_{i=1}^{m}\nabla_{\boldsymbol{\theta}_{\text{gen}}}(\mathcal{L}_{\text{ac}} + \mathcal{L}_{\text{adv}} - \mathcal{L}_{\text{cosy}})$
end
```

4. Experiments

4.1. Experimental setup

We followed the experimental setup in NIST i-vector Speaker Recognition Challenge [17–19]. The number of i-vectors in development set was 36,572 from 4958 speakers (1930 males and 3028 females), i.e. $\mathbf{c} \in \mathcal{R}^{4958}$. The number of target speaker models was 1,306 with totally 6,530 i-vectors where $\mathbf{x} \in \mathcal{R}^{600}$. The number of test i-vectors was 9,634. There were 12,582,004 trials which included all possible pairs involving a target speaker model and a single i-vector test segment. These trials were divided into a progress subset with 40% of the trials and an evaluation subset with the remaining 60 % of the trials. The equal error rate (EER) (%) and the minimum decision cost function (minDCF (p) for progress subset and minDCF (e) for evaluation set) were examined by using PLDA scoring. PLDA was trained by 450 iterations.

The solutions to i-vector augmentation using Cosx-GAN, Cosy-GAN and PLDA-Cos-GAN were carried out for comparative purposes. The result of AC-GAN [6] was also implemented for comparison. In Cosx-GAN and Cosy-GAN, the same topology of generator was specified as (100 +4958)-1000-1000-1000-600 with three 1000-neuron hidden layers, latent code $\mathbf{z} \in \mathcal{R}^{100}$ and fake i-vector $\widehat{\mathbf{x}} \in \mathcal{R}^{600}$, and the same topology of discriminator 600-1000-1000-1000-(1+4958) was built with three 1000-neuron hidden layers where 1+4958 denotes the number of posterior outputs for 1 real/fake class and 4958 speakers. Adam optimization and Xavier initialization were applied. Mini-batch size was 200 samples. In PLDA-Cos-GAN, we followed the topology of generator used in Cosy-GAN with additional topologies of encoder 600-1000-1000-1000-200 and discriminator 200-(1+4958), i.e. $\mathbf{y}, \widehat{\mathbf{y}} \in \mathcal{R}^{200}$. The decoder was calculated using PLDA parameters. The learning rate was 0.0001 for discriminator and encoder and 0.002 for generator. We performed data augmentation using NIST i-vector dataset. If the number of training samples per speaker was less than n, we used GANs to generate additional samples to reach n for that speaker where n ranges from 2 to 4. Among 4958 speakers, there were 1033 speakers who only uttered one training sample.

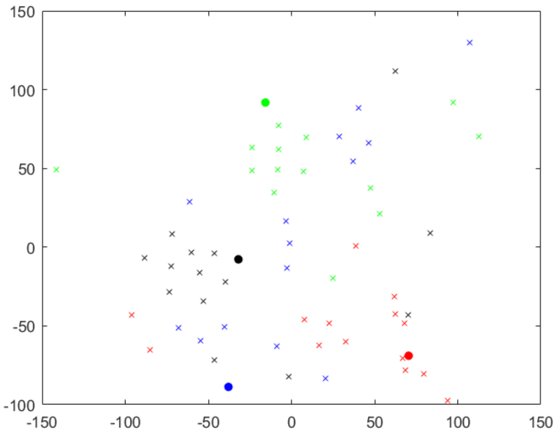


Figure 5: Two-dimensional visualization of latent variables of real i-vector $\mathbf{y}$ (denoted by $\bullet$) and generated i-vector $\widehat{\mathbf{y}}$ (denoted by $\times$) for different speakers (denoted by color) by using PLDA-Cos-GAN.

4.2. Experimental results

Figure 5 displays two-dimensional visualization [20] of latent variables of real $\mathbf{y}$ and generated samples $\widehat{\mathbf{y}}$ where PLDA-Cos-GAN is adopted. We randomly select four speakers with only one training i-vector. Basically, the generated samples are clustered together with real sample although the generation is not only based on cosine similarity $\mathcal{L}_{\text{cosy}}$ but also four other loss functions $\{\mathcal{L}_{\text{adv}}, \mathcal{L}_{\text{ac}}, \mathcal{L}_{\text{gau}}, \mathcal{L}_{\text{rec}}\}$ which are jointly minimized to achieve different objectives for adversarial leaning and augmentation. Table 1 reports the performance of speaker recognition by applying data augmentation using different GANs. Different dimension d of latent code $\mathbf{z}_i$ in PLDA is evaluated. Baseline system is constructed without data augmentation. Increasing d in PLDA does improve system performance. It is obvious that EER and minDCF are consistently decreased by adding one, two and three samples (n=2,3,4,5) across different GANs except AC-GAN. AC-GAN is even worse than baseline. So, it is important to perform regularization in speaker recognition based on cosine similarity in other GANs. Adding two, three and four samples is better than adding one samples. Increasing training samples is helpful. Cosy-GAN does not work better than Cosx-GAN. Nevertheless, the best performance is achieved by using PLDA-Cos-GAN. The cosine similarity calculated from samples of encoder in PLDA-Cos-GAN generates more informative i-vectors for speaker recognition than that from last layer of discriminator in Cosy-GAN. The source codes of Cosx-GAN, Cosy-GAN and PLDA-Cos-GAN are accessible at https://github.com/NCTUMLlab/Kang-Ting-Peng.

5. Conclusions

We have presented the ideas of cosine similarity and probabilistic linear discriminant analysis (PLDA) based reconstruction for regularization in generative adversarial network (GAN) modeling and carried out the generation of i-vectors for PLDA-based speaker recognition. The similarity between real sample and fake sample in i-vector space or in latent feature space was introduced as the objective for adversarial learning. The minimax optimization in the proposed GANs was realized by regularizing the PLDA model by considering cosine similarity, Gaussian assumption and PLDA reconstruction. A specialized GAN model for speaker recognition was developed. From the experimental results, we illustrated the contributions of two extensions of GAN for speaker recognition which minimized the cosine similarity between real and fake i-vectors as well as the reconstruction error of latent codes of real and fake i-vectors based on PLDA scoring.

6. References

[1] N. Dehak, P. J. 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.

[2] S. J. D. Prince and J. H. Elder, “Probabilistic linear discriminant analysis for inferences about identity,” in *Proc. of IEEE International Conference on Computer Vision*, 2007, pp. 1–8.

[3] D. Garcia-Romero and C. Y. Espy-Wilson, “Analysis of i-vector length normalization in speaker recognition systems,” in *Proc. of Annual Conference of International Speech Communication Association*, 2011, pp. 249–252.

[4] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Advances in Neural Information Processing Systems*, 2014, pp. 2672–2680.

Table 1: Comparison of EER (%) and minDCF by using different GANs under different n and d in PLDA scoring.

Model (d, n)	EER (%)	minDCF (p)	minDCF (e)
Baseline (20)	4.60	0.58	0.54
Baseline (50)	2.90	0.37	0.33
Baseline (100)	2.56	0.31	0.28
AC-GAN (20, 2)	4.74	0.59	0.56
AC-GAN (20, 3)	4.62	0.58	0.55
AC-GAN (100, 2)	2.57	0.32	0.28
AC-GAN (100, 3)	2.55	0.31	0.27
Cosx-GAN (20, 2)	4.35	0.56	0.52
Cosx-GAN (20, 3)	4.21	0.55	0.49
Cosx-GAN (20, 4)	4.19	0.54	0.49
Cosx-GAN (20, 5)	4.19	0.54	0.49
Cosx-GAN (100, 2)	2.51	0.31	0.28
Cosx-GAN (100, 3)	2.50	0.31	0.28
Cosx-GAN (100, 4)	2.46	0.31	0.27
Cosx-GAN (100, 5)	2.48	0.31	0.27
Cosy-GAN (20, 3)	4.46	0.58	0.54
Cosy-GAN (20, 4)	4.42	0.56	0.52
Cosy-GAN (50, 3)	2.92	0.37	0.34
Cosy-GAN (50, 4)	2.87	0.37	0.33
Cosy-GAN (100, 3)	2.53	0.31	0.28
Cosy-GAN (100, 4)	2.49	0.32	0.28
PLDA-Cos-GAN (20, 3)	3.73	0.58	0.55
PLDA-Cos-GAN (20, 4)	3.70	0.56	0.53
PLDA-Cos-GAN (50, 3)	2.56	0.36	0.32
PLDA-Cos-GAN (50, 4)	2.52	0.35	0.31
PLDA-Cos-GAN (100, 3)	2.42	0.30	0.27
PLDA-Cos-GAN (100, 4)	2.41	0.30	0.26

[5] N. Li, M.-W. Mak, and J.-T. Chien, “DNN-driven mixture of PLDA for robust speaker verification,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 6, pp. 1371–1383, 2017.

[6] A. Odena, C. Olah, and J. Shlens, “Conditional image synthesis with auxiliary classifier GANs,” *arXiv preprint arXiv:1610.09585*, 2016.

[7] M. Mirza and S. Osindero, “Conditional generative adversarial nets,” *arXiv preprint arXiv:1411.1784*, 2014.

[8] J.-T. Chien and C.-W. Ting, “Factor analyzed subspace modeling and selection,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 1, pp. 239–248, 2008.

[9] M.-W. Mak, X. Pang, and J.-T. Chien, “Mixture of PLDA for noise robust i-vector speaker verification,” *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 24, no. 1, pp. 130–142, 2016.

[10] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the EM algorithm,” *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, no. 1, pp. 1–38, 1977.

[11] N. Li, M.-W. Mak, W.-W. Lin, and J.-T. Chien, “Discriminative subspace modeling of SNR and duration variabilities for robust speaker verification,” *Computer Speech & Language*, vol. 45, pp. 83–103, 2017.

[12] W. Lin, M.-W. Mak, L. Li, and J.-T. Chien, “Reducing domain mismatch by maximum mean discrepancy autoencoders,” in *Proc. of Speaker and Language Recognition Workshop (Odyssey)*, 2018.

[13] D. P. Kingma and M. Welling, “Auto-encoding variational Bayes,” in *Proc. of International Conference on Learning Representations*, 2014.

[14] J.-T. Chien and C.-W. Hsu, “Variational manifold learning for speaker recognition,” in *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2017, pp. 4935–4939.

[15] S. Watanabe and J.-T. Chien, *Bayesian Speech and Language Processing*, Cambridge University Press, 2015.

[16] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, “Adversarial autoencoders,” *arXiv preprint arXiv:1511.05644*, 2015.

[17] C. S. Greenberg, D. Bansé, G. R. Doddington, D. Garcia-Romero, J. J. Godfrey, T. Kinnunen, A. F. Martin, A. McCree, M. Przybocki, and D. A. Reynolds, “The NIST 2014 speaker recognition i-vector machine learning challenge,” in *Proc. of Odyssey: The Speaker and Language Recognition Workshop*, 2014.

[18] J.-T. Chien and C.-H. Chen, “Deep discriminative manifold learning,” in *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2016, pp. 2672–2676.

[19] J.-T. Chien and K.-T. Peng, “Adversarial manifold learning for speaker recognition,” in *Proc. of IEEE Automatic Speech Recognition and Understanding Workshop*, 2017, pp. 599–605.

[20] L. van der Maaten and G. E. Hinton, “Visualizing data using *t*-SNE,” *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.