INTERSPEECH 2017
August 20–24, 2017, Stockholm, Sweden

Tied Variational Autoencoder Backends for i-Vector Speaker Recognition

Jesús Villalba[1], *Niko Brümmer*[2], *Najim Dehak*[1]

[1]Center for Language and Speech Processing, Johns Hopkins University, Baltimore, MD, USA
[2]Nuance Communications, Inc., South Africa

{jvillal7,ndehak3}@jhu.edu, niko.brummer@nuance.com

Abstract

Probabilistic linear discriminant analysis (PLDA) is the de facto standard for backends in i-vector speaker recognition. If we try to extend the PLDA paradigm using non-linear models, e.g., deep neural networks, the posterior distributions of the latent variables and the marginal likelihood become intractable. In this paper, we propose to approach this problem using stochastic gradient variational Bayes. We generalize the PLDA model to let i-vectors depend non-linearly on the latent factors. We approximate the evidence lower bound (ELBO) by Monte Carlo sampling using the reparametrization trick. This enables us to optimize of the ELBO using backpropagation to jointly estimate the parameters that define the model and the approximate posteriors of the latent factors. We also present a reformulation of the likelihood ratio, which we call Q-scoring. Q-scoring makes possible to efficiently score the speaker verification trials for this model. Experimental results on NIST SRE10 suggest that more data might be required to exploit the potential of this method.

Index Terms: speaker recognition, i-vectors, variational autoencoders, stochastic variational inference, PLDA

1. Introduction

The i-vector paradigm provides a method to map a variable length speech utterance into a low dimensional fixed length vector (i-vector) [1]. The i-vector is able to retain the essential information about the utterance, e.g., the speaker identity, language [2], accent [3], age [4], etc. Thus, we can use it as feature for a great variety of classifiers. The state-of-the-art classifier for speaker recognition is probabilistic linear discriminant analysis (PLDA) [5, 6, 7]. PLDA is a linear Gaussian generative model. Currently, we are witnessing the great success of non-linear models like deep neural networks in most fields. For this reason, we want to explore more powerful approaches that can model more complex i-vector distributions.

We propose a probabilistic discriminant model where the parameters of the i-vector conditional likelihood given the latent variables–speaker and channel factors– are given by a neural network. For this type of model, the data marginal likelihood and the latent factor posteriors are intractable. We propose to approximate both by using the variational autoencoder framework [8, 9]. This framework assumes an approximate posterior that belongs to a parametric family of distributions. The evidence lower bound (ELBO) is approximated by Monte Carlo (MC) sampling using the reparametrization trick. The ELBO is differentiable w.r.t. the parameters that define the model and the posteriors, which enables us to optimize these parameters by backpropagation. Variational autoencoders have shown great performance for image modeling [10]. For speech, they have been used to obtain utterance embeddings, which are used for adapting DNN based ASR to the speaker [11]. Also, there is a recent work that uses stochastic variational inference to derive a speaker discriminant manifold [12] from i-vectors.

This work started in the 2016 Stellenbosch Speaker Recognition Workshop (South Africa) organized by Nuance. We would like to thank all the workshop participants for the fruitful discussions.

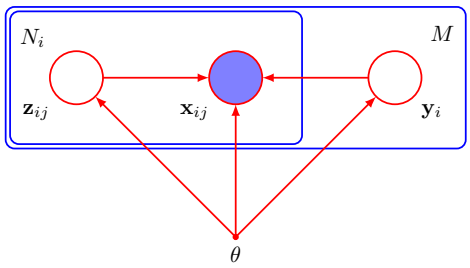

Figure 1: *Probabilistic discriminant analysis graphical model.*

Among the novelties of the paper, we highlight a principled way for computing posteriors of latent factors tied between i-vectors, i.e., the speaker factors. Also, we present an approximation to efficiently evaluate the likelihood ratio of the speaker verification trials, a method that we called Q-scoring.

This paper is organized as follows. Section 2 introduces the probabilistic discriminant analysis framework and describes the PLDA model. Section 3 presents our model, how to compute the latent factor posteriors and ELBO; and Q-scoring. Section 4 describes the experimental setup and presents the results. Finally, Section 5 shows the conclusions.

2. Probabilistic discriminant analysis

Figure 1 displays the graphical model for a general discriminant analysis model. The observed i-vector $\mathbf{x}_{ij}$ is conditioned on two latent variables, $\mathbf{y}_i$ and $\mathbf{z}_{ij}$. The variable $\mathbf{y}_i$ (speaker factor) is shared (tied) across all the N_i i-vectors of speaker i. Thus, $\mathbf{y}_i$ contains the identity information. Meanwhile, $\mathbf{z}_{ij}$ (channel factor) takes a different value for each i-vector and its role is to account for the variability between sessions of the same speaker. The variable θ contains the parameters that define the priors of $\mathbf{y}$ and $\mathbf{z}$ and the likelihood of $\mathbf{x}_{ij}$ given $\mathbf{y}_i$ and $\mathbf{z}_{ij}$.

If we assume a linear mapping from the latent variables to the i-vectors, we obtain probabilistic linear discriminant analysis (PLDA) [5, 6]. In Gaussian PLDA the conditional data likelihood is given by

$$P\left(\mathbf{x}_{ij}|\mathbf{y}_i,\mathbf{z}_{ij},\theta\right)=\mathcal{N}\left(\mathbf{x}_{ij}|\boldsymbol{\mu}+\mathbf{V}\mathbf{y}_i+\mathbf{U}\mathbf{z}_{ij},\mathbf{D}\right) \quad (1)$$

http://dx.doi.org/10.21437/Interspeech.2017-1018

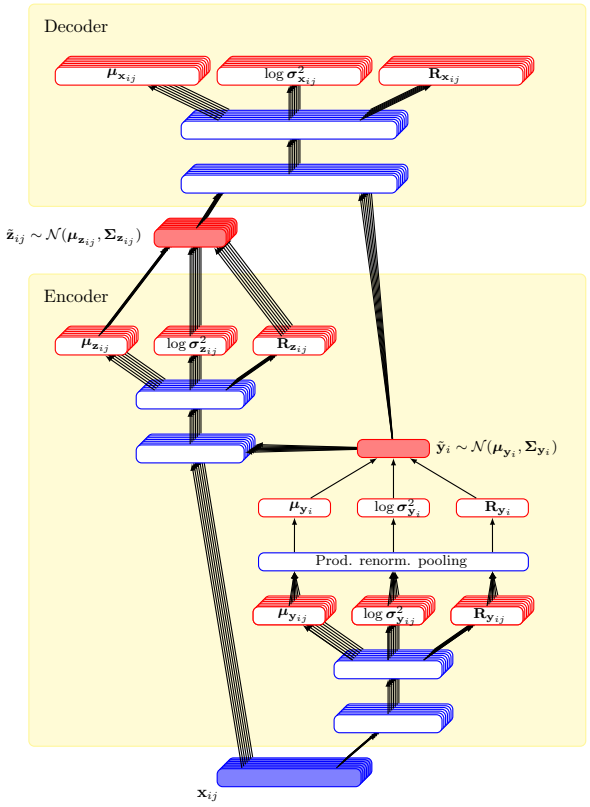

Figure 2: *Computation flow to evaluate the Tied VAE ELBO. The right hand path of the encoder computes* $q_{\phi_\mathbf{y}}(\mathbf{y}_i|\mathbf{X}_i)$ *and draws a sample* $\tilde{\mathbf{y}}_i$*. The left hand path computes* $q_{\phi_\mathbf{z}}(\mathbf{z}_{ij}|\tilde{\mathbf{y}}_i, \mathbf{x}_{ij})$ *and draws* N_i *samples* $\tilde{\mathbf{z}}_{ij}$*.* $\tilde{\mathbf{y}}_i$ *and* $\tilde{\mathbf{z}}_{ij}$ *are plugged into the encoder to compute* $P(\mathbf{x}_{ij}|\tilde{\mathbf{y}}_i, \tilde{\mathbf{z}}_{ij}, \theta)$ *used in Eq.* (13)

where $\boldsymbol{\mu}$ is a speaker independent term, $\mathbf{V}$ is a low-rank eigenvoice matrix; $\mathbf{U}$ is a low-rank eigen-channel matrix; $\mathbf{D}$ is a diagonal covariance matrix; and $\theta = (\boldsymbol{\mu}, \mathbf{V}, \mathbf{U}, \mathbf{D})$. The priors on $\mathbf{y}_i$ and $\mathbf{z}_{ij}$ are assumed standard Normal. When $\mathbf{U}$ is full-rank, this model is equivalent to a simplified PLDA model (SPLDA) without $\mathbf{z}_{ij}$ and full-covariance $\mathbf{D}$ [13].

For PLDA, the latent factor posteriors and the data marginal likelihood have closed form expressions. This makes easy to train the model using the EM algorithm [14]. Unfortunately, if we make the model non-linear most calculations become intractable. To overcome this issue, we propose to use the variational autoencoder framework [8]. PLDA can be seen as an autoencoder where the latent factors are the *code* that compresses the information contained in the i-vector. The path from the latent factors to the i-vector is the decoder. Meanwhile, computing the latent factor posteriors is the encoding process. We can generalize PLDA by making the decoder non-linear as we show in the next section.

3. Tied variational autoencoder as i-vector back-end

3.1. The model (decoder)

Figure 2 depicts the computation flow of the tied variational autoencoder (TVAE) that we propose to perform non-linear discriminant analysis. This is an extension of the original VAE formulation [8], where we added the speaker factor variable $\mathbf{y}_i$ which is tied across samples of the same speaker. The upper part of the figure shows the decoder, which is equivalent to the model in Figure 1. The decoder takes $\mathbf{y}_i$ and $\mathbf{z}_{ij}$ as input and computes the mean and covariance of the conditional likelihood of $\mathbf{x}_{ij}$ as output. Thus, we obtain

$$P(\mathbf{x}_{ij}|\mathbf{y}_i, \mathbf{z}_{ij}, \theta) = \mathcal{N}(\mathbf{x}_{ij}|\boldsymbol{\mu}_\theta(\mathbf{y}_i, \mathbf{z}_{ij}), \boldsymbol{\Sigma}_\theta(\mathbf{y}_i, \mathbf{z}_{ij})) \quad (2)$$

where θ are the parameters of the decoder network. We used the decomposition

$$\boldsymbol{\Sigma}_\theta(\mathbf{y}_i, \mathbf{z}_{ij}) = \boldsymbol{\Sigma}_{\mathbf{x}_{ij}} = \mathbf{R}^{\mathrm{T}}_{\mathbf{x}_{ij}} \mathrm{diag}(\boldsymbol{\sigma}^2_{\mathbf{x}_{ij}}) \mathbf{R}_{\mathbf{x}_{ij}} \quad (3)$$

where $\boldsymbol{\sigma}_{\mathbf{x}_{ij}}$ is a vector with positive values and $\mathbf{R}_{\mathbf{x}_{ij}}$ is an upper triangular matrix with ones in the diagonal [1]. The decoder has a linear output layer that computes $\log \boldsymbol{\sigma}^2_{\mathbf{x}_{ij}}$ and the upper triangular part of $\mathbf{R}_{\mathbf{x}_{ij}}$. In this manner, we assure that $\boldsymbol{\Sigma}_{\mathbf{x}_{ij}}$ is positive definite.

3.2. Posterior of the latent variables (encoder)

The lower part of Figure 2 represents the encoder, i.e., the calculus of the latent factor posteriors. We define $\mathbf{X}_i = \{\mathbf{x}_{ij}\}_{j=1}^{N_i}$ and $\mathbf{Z}_i = \{\mathbf{z}_{ij}\}_{j=1}^{N_i}$, We factorized the approximate posterior for speaker i as

$$q_\phi(\mathbf{y}_i, \mathbf{Z}_i|\mathbf{X}_i) = q_{\phi_\mathbf{y}}(\mathbf{y}_i|\mathbf{X}_i) \prod_{j=1}^{N_i} q_{\phi_\mathbf{z}}(\mathbf{z}_{ij}|\mathbf{y}_i, \mathbf{x}_{ij}) \quad (4)$$

where each factor is a parametric full covariance Gaussian,

$$q_{\phi_\mathbf{y}}(\mathbf{y}_i|\mathbf{X}_i) = \mathcal{N}\left(\mathbf{y}_i|\boldsymbol{\mu}_{\phi_\mathbf{y}}(\mathbf{X}_i), \boldsymbol{\Sigma}_{\phi_\mathbf{y}}(\mathbf{X}_i)\right) \quad (5)$$

$$q_{\phi_\mathbf{z}}(\mathbf{z}_{ij}|\mathbf{x}_{ij}, \mathbf{y}_i) = \mathcal{N}(\mathbf{z}_{ij}|\boldsymbol{\mu}_{\phi_\mathbf{z}}(\mathbf{x}_{ij}, \mathbf{y}_i), \boldsymbol{\Sigma}_{\phi_\mathbf{z}}(\mathbf{x}_{ij}, \mathbf{y}_i)) \; . \quad (6)$$

$\phi_\mathbf{y}$ and $\phi_\mathbf{z}$ are the variational parameters for the $\mathbf{y}$ and $\mathbf{z}$ posteriors; and $\phi = (\phi_\mathbf{y}, \phi_\mathbf{z})$. Note that standard PLDA doesn't need the ϕ parameters because there are closed form equations to calculate the mean and covariances of the posteriors given the model parameters θ. Meanwhile for VAEs, there is no explicit mapping from θ to ϕ so both are jointly trained in the optimization process.

To assure that the variational factors have positive definite covariances, we again used the decomposition (3). By using it, we also simplify the formula for the KL divergences required to compute the ELBO, which we show below. This factorization is also a particular case of the method in [15] to obtain VAEs with flexible posteriors.

As shown in Figure 2, the first step to compute $q_{\phi_\mathbf{y}}(\mathbf{y}_i|\mathbf{X}_i)$ consists in applying a common neural network to each of the speaker's i-vectors. Then, we need to combine the outputs of those networks to produce a unique mean and covariance. This requires some kind of pooling strategy. Most works resort to simple average [16] or max pooling [10]. Other works use LSTM networks [17] but they have the inconvenience that the result would depend on the order in which the i-vectors are processed. We propose a principled way of combining i-vectors that complies with the rules of probability. We could prove that the posterior of $\mathbf{y}$ given N i-vectors can be written as a function

[1] We dropped the dependency of $\boldsymbol{\sigma}$ and $\mathbf{R}$ on θ, $\mathbf{y}_i$ and $\mathbf{z}_{ij}$ to simplify the notation

of the posteriors given each one of the i-vectors $P(\mathbf{y}|\mathbf{x}_j)$ for $j = 1, \ldots, N$,

$$P(\mathbf{y}|\mathbf{x}_1, \ldots, \mathbf{x}_N) \propto \frac{\prod_{j=1}^{N} P(\mathbf{y}|\mathbf{x}_j)}{P(\mathbf{y})^{N-1}} \ . \tag{7}$$

Thus, the encoder computes the approximate individual posteriors

$$q_{\phi_\mathbf{y}}(\mathbf{y}_i|\mathbf{x}_{ij}) = \mathcal{N}\left(\mathbf{y}_i|\boldsymbol{\mu}_{\phi_\mathbf{y}}(\mathbf{x}_{ij}), \boldsymbol{\Sigma}_{\phi_\mathbf{y}}(\mathbf{x}_{ij})\right) \ , \tag{8}$$

and applies (7) to compute the pooled posterior. For the case of Gaussian posteriors and standard Normal priors, the pooled posterior is also Gaussian with covariance and mean,

$$\boldsymbol{\Sigma}_{\phi_\mathbf{y}}(\mathbf{X}_i) = \left(\mathbf{I} + \sum_{j=1}^{N_i} (\boldsymbol{\Sigma}_{\phi_\mathbf{y}}(\mathbf{x}_{ij})^{-1} - \mathbf{I})\right)^{-1} \tag{9}$$

$$\boldsymbol{\mu}_{\phi_\mathbf{y}}(\mathbf{X}_i) = \boldsymbol{\Sigma}_{\phi_\mathbf{y}}(\mathbf{X}_i) \sum_{j=1}^{N_i} \boldsymbol{\Sigma}_{\phi_\mathbf{y}}(\mathbf{x}_{ij})^{-1} \boldsymbol{\mu}_{\phi_\mathbf{y}}(\mathbf{x}_{ij}) \ . \tag{10}$$

In practice, we assume that the layer before the pooling provides the products $\boldsymbol{\Sigma}_{\phi_\mathbf{y}}(\mathbf{x}_{ij})^{-1}\boldsymbol{\mu}_{\phi_\mathbf{y}}(\mathbf{x}_{ij})$ to save computation. Equation (7) basically consists of a product of distributions plus a renormalization to make the distribution to integrate to one. For this reason, we called the pooling layer the *product renormalization* layer.

Once we have $q_{\phi_\mathbf{y}}(\mathbf{y}_i|\mathbf{X}_i)$, we sample $\tilde{\mathbf{y}}_i$ from it and use it as input to the neural networks that compute the $\mathbf{z}_{ij}$ posteriors. If we draw multiple $\tilde{\mathbf{y}}_i$ samples, we obtain multiple posteriors for $\mathbf{z}_{ij}$ and we need to compute expectations to evaluate the ELBO as shown below.

3.3. Stochastic Gradient Variational Bayes

To learn the parameters of the model, we need to maximize the data marginal likelihood $\log P(\mathbf{X}|\theta) = \sum_{i=1}^{M} \log P(\mathbf{X}_i|\theta)$. Unfortunately, $\log P(\mathbf{X}_i|\theta)$ is intractable so we optimized the variational Bayes lower bound (ELBO),

$$\begin{aligned} &\log P(\mathbf{X}_i|\theta) \geq \mathcal{L}(\mathbf{X}_i, \theta, \phi) = \\ &= \mathrm{E}_{q_\phi(\mathbf{y}_i, \mathbf{Z}_i|\mathbf{X}_i)}\left[\log \frac{P(\mathbf{X}_i, \mathbf{y}_i, \mathbf{Z}_i|\theta)}{q_\phi(\mathbf{y}_i, \mathbf{Z}_i|\mathbf{X}_i)}\right] \ . \end{aligned} \tag{11}$$

Applying the factorization in (4), we obtain

$$\begin{aligned} &\mathcal{L}(\mathbf{X}_i, \theta, \phi) = -\mathrm{D}_{\mathrm{KL}}\left(q_{\phi_\mathbf{y}}(\mathbf{y}_i|\mathbf{X}_i)\,||P(\mathbf{y}_i)\right) \\ &+ \sum_{j=1}^{N_i}\Big(\mathrm{E}_{q_{\phi_\mathbf{y}}(\mathbf{y}_i|\mathbf{X}_i) q_{\phi_\mathbf{z}}(\mathbf{z}_{ij}|\mathbf{y}_i, \mathbf{x}_{ij})}\left[\log P(\mathbf{x}_{ij}|\mathbf{y}_i, \mathbf{z}_{ij}, \theta)\right] \\ &- \mathrm{E}_{q_{\phi_\mathbf{y}}(\mathbf{y}_i|\mathbf{X}_i)}\left[\mathrm{D}_{\mathrm{KL}}\left(q_{\phi_\mathbf{z}}(\mathbf{z}_{ij}|\mathbf{y}_i, \mathbf{x}_{ij})\,||P(\mathbf{z}_{ij})\right)\right]\Big) \ . \end{aligned} \tag{12}$$

The KL divergences act as a regularizer for ϕ preventing the posterior to be far from the prior.

The expectations involved in the ELBO cannot be computed in closed form so we approximated it by an empirical expectation,

$$\begin{aligned} &\mathcal{L}(\mathbf{X}_i, \theta, \phi) \approx -\mathrm{D}_{\mathrm{KL}}\left(q_{\phi_\mathbf{y}}(\mathbf{y}_i|\mathbf{X}_i)\,||P(\mathbf{y}_i)\right) \\ &+ \sum_{j=1}^{N_i}\left(\frac{1}{L_1}\sum_{l_1=1}^{L_1}\left(\frac{1}{L_2}\sum_{l_2=1}^{L_2}\log P\left(\mathbf{x}_{ij}|\tilde{\mathbf{y}}_i^{(l_1)}, \tilde{\mathbf{z}}_{ij}^{(l_1,l_2)}, \theta\right)\right.\right. \\ &\left.\left.-\mathrm{D}_{\mathrm{KL}}\left(q_{\phi_\mathbf{z}}\left(\mathbf{z}_{ij}|\tilde{\mathbf{y}}_i^{(l_1)}, \mathbf{x}_{ij}\right)\,||P(\mathbf{z}_{ij})\right)\right)\right) \end{aligned} \tag{13}$$

where we employed the reparametrization trick [8, 9] to draw samples from the approximate posteriors,

$$\tilde{\mathbf{y}}_i^{(l_1)} = \boldsymbol{\mu}_{\phi_\mathbf{y}}(\mathbf{X}_i) + \mathbf{R}_{\phi_\mathbf{y}}(\mathbf{X}_i)^{\mathrm{T}}\left(\boldsymbol{\sigma}_{\phi_\mathbf{y}}(\mathbf{X}_i) \odot \epsilon^{(i,l_1)}\right) \tag{14}$$

$$\begin{aligned} \tilde{\mathbf{z}}_{ij}^{(l_1,l_2)} &= \boldsymbol{\mu}_{\phi_\mathbf{z}}(\tilde{\mathbf{y}}_i^{(l_1)}, \mathbf{x}_{ij}) \\ &+ \mathbf{R}_{\phi_\mathbf{z}}(\tilde{\mathbf{y}}_i^{(l_1)}, \mathbf{x}_{ij})^{\mathrm{T}}\left(\boldsymbol{\sigma}_{\phi_\mathbf{z}}(\tilde{\mathbf{y}}_i^{(l_1)}, \mathbf{x}_{ij}) \odot \epsilon^{(i,j,l_1,l_2)}\right) \end{aligned} \tag{15}$$

$$\epsilon^{(i,l_1)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \qquad \epsilon^{(i,j,l_1,l_2)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \ . \tag{16}$$

L_1 is the number of samples drawn from each $q_{\phi_\mathbf{y}}(\mathbf{y}_i|\mathbf{X}_i)$ and L_2 the number of samples drawn from each $q_{\phi_\mathbf{z}}\left(\mathbf{z}_{ij}|\tilde{\mathbf{y}}_i^{(l_1)}\right)$. In practice, if the mini-batch is large enough we can draw just one sample per data point ($L_1 = L_2 = 1$) [8].

The KL divergence between the Normal and standard Normal distributions can be computed in closed form. Using decomposition (3), it only depends on the vector $\boldsymbol{\sigma}^2$.

$$\mathrm{D}_{\mathrm{KL}}\left(\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})\,||\mathcal{N}(\mathbf{0}, \mathbf{I})\right) = -\frac{1}{2}\sum_{d=1}^{D}(1 + \log\sigma_d^2 - \mu_d^2 - \sigma_d^2) \ . \tag{17}$$

This ELBO is differentiable w.r.t. θ and $\boldsymbol{\phi}$ so we can optimize it by stochastic gradient descent or any of its variants. This procedure is known as stochastic gradient variational Bayes (SGVB).

3.4. Likelihood ratio evaluation: Q-scoring

At evaluation time, given a pair of i-vectors $\mathbf{x}_1$, $\mathbf{x}_2$, we need to evaluate the likelihood ratio

$$R(\mathbf{x}_1, \mathbf{x}_2) = \frac{P(\mathbf{x}_1, \mathbf{x}_2|\mathcal{T}, \theta)}{P(\mathbf{x}_1, \mathbf{x}_2|\mathcal{N}, \theta)} = \frac{P(\mathbf{x}_1, \mathbf{x}_2|\mathcal{T}, \theta)}{P(\mathbf{x}_1|\theta)\,P(\mathbf{x}_2|\theta)} \tag{18}$$

where $P(\mathbf{x}_1, \mathbf{x}_2|\mathcal{T}, \theta)$ is the marginal likelihood assuming that both i-vectors belong to the same speaker and, $P(\mathbf{x}_1, \mathbf{x}_2|\mathcal{N}, \theta)$ assuming that both belong to different speakers. We can approximate these likelihoods with the ELBO in (13). However, we need a good estimation of the ELBO for each hypothesis so drawing one sample per data-point is not enough–we needed at least 100 to obtain a decent result. Evaluating the likelihood ratio in this manner implies a high computational cost.

We propose a more efficient approximation based on manipulating the likelihood ratio to obtain the following expression,

$$R(\mathbf{x}_1, \mathbf{x}_2) = \int \frac{P(\mathbf{y}|\mathbf{x}_1)\,P(\mathbf{y}|\mathbf{x}_2)}{P(\mathbf{y})}\,\mathrm{d}\mathbf{y} \tag{19}$$

where R is written as a function of the $\mathbf{y}$ posteriors given the individual i-vectors $\mathbf{x}_1$ and $\mathbf{x}_2$. If we replace the true posteriors with the approximate posteriors, we obtain what we called *Q-scoring*. For the case of Gaussian posteriors $q_{\phi_\mathbf{y}}(\mathbf{y}|\mathbf{x}_i) = \mathcal{N}(\mathbf{y}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$, Eq. (19) has closed form solution,

$$\begin{aligned} \log R(\mathbf{x}_1, \mathbf{x}_2) \approx &\frac{1}{2}\Big(\log|\boldsymbol{\Sigma}_{1,2}| + \boldsymbol{\mu}_{1,2}^{\mathrm{T}}\boldsymbol{\Sigma}_{1,2}^{-1}\boldsymbol{\mu}_{1,2} \\ &- \sum_{i=1}^{2}\log|\boldsymbol{\Sigma}_i| + \boldsymbol{\mu}_i^{\mathrm{T}}\boldsymbol{\Sigma}_i^{-1}\boldsymbol{\mu}_i\Big) \end{aligned} \tag{20}$$

where $\boldsymbol{\mu}_{1,2}$ and $\boldsymbol{\Sigma}_{1,2}$ are computed using (9) and (10).

This method is very efficient for several reasons. First of all, we eliminate the necessity of sampling. Also, we avoid computing $q_{\phi_\mathbf{y}}(\mathbf{y}|\mathbf{x}_1, \mathbf{x}_2)$ for each pair of i-vectors so, if we want to score N_1 against N_2 i-vectors, we just need to evaluate the encoder $N_1 + N_2$ times instead of $N_1 + N_2 + N_1 N_2$ times.

4. Experiments

4.1. Experimental setup

We evaluated our approach in the NIST SRE10 core extended telephone condition (det5). We used a GMM-UBM of 2048 components and an i-vector extractor of dimension 600. Both were trained on Switchboard II, Switchboard cellular and NIST SRE04-08 data using the Kaldi toolkit[2] [18]. The i-vector preprocessing included centering, whitening and length-normalization [19]. As baseline, we used simplified PLDA given that it provided the same performance as the full PLDA model.

To train the baseline PLDA and the TVAEs, we used either SRE04-08 or pooled switchboard and SRE04-08. We used 90% of the speakers for training and the rest for validation, early stopping and hyperparameter tuning. We implemented the TVAE with the Keras toolkit[3].

We compared TVAEs with diagonal and full covariance encoder and decoder. Furthermore, for the diagonal TVAE, we rotated the i-vectors with LDA (without dimensionality reduction). The justification for this is that, in standard PLDA, a LDA rotation makes the within and between covariances diagonal. As consequence the latent factors posteriors also become diagonal. Thus, LDA becomes a way of alleviating the constraint of using TVAEs with diagonal covariances.

We also compared TVAE with two latent factors ($\mathbf{y}$ and $\mathbf{z}$) versus a simplified model with only speaker factors. In the latter, all the inter-session variability has to be accounted by the covariance of the decoder. This is the TVAE equivalent to the simplified PLDA model.

4.2. Network architectures

In our experiments we used a residual network architecture to calculate the means in the encoder and decoder. Given an input $\mathbf{x}$, the output is given by

$$\mathbf{h}_0 = \mathbf{x} \tag{21}$$

$$\mathbf{h}_k = f(\mathbf{W}_k \mathbf{h}_{k-1} + \mathbf{b}_k) \qquad k = 1, \dots, K \tag{22}$$

$$\mathrm{ResNet}(\mathbf{x}) = \sum_{k=0}^{K} \mathbf{V}_k \mathbf{h}_k \tag{23}$$

where $f(x)$ is the ReLU activation function; $\mathbf{W}$, $\mathbf{V}$ and $\mathbf{b}$ are the network weights and biases; and K is the number of hidden layers. With this architecture, we keep the linear path between the input and output, which is present in standard PLDA. We compared linear encoder-decoder to deep encoder-decoder with ResNets of two hidden layers and 50 units per hidden layer.

Due to the small number of training i-vectors available, we decided to simplify the model to reduce the number of trainable parameters. We assumed that the variances $\boldsymbol{\sigma}^2$ and rotations $\mathbf{R}$ of encoder and decoder are trainable constants instead of depending on the network inputs. This is also the case for standard PLDA.

4.3. Results

Table 1 presents the EER and minimum DCF ($P_{\mathcal{T}} = 0.001$) for different classifiers. The proposed model provided decent performance but was not able to match the PLDA baseline. Adding Switchboard to the training data didn't improve PLDA. However, it consistently improved the TVAE. This indicates that the TVAE needs more training data to perform well. This is probably because the TVAE needs to optimize twice the number of parameters than PLDA. The former needs to train the encoder and decoder parameters while the latter only needs to train the decoder. This is a serious drawback due to the limited availability of datasets with labeled speakers. We didn't obtain any consistent advantage from deep architectures. Again, this could be because the limited data available doesn't allow one to train models with a high number of parameters. Another reason may be that the actual i-vector distribution is well described by a linear model.

Table 1: *EER(%)/MinDCF for different TVAE configurations.*

	SRE Train		SRE+SWB Train	
	EER(%)	MinDCF	EER(%)	MinDCF
PLDA Baseline	**2.09**	**0.42**	**2.08**	**0.41**
Model with $\mathbf{y}$ + $\mathbf{z}$				
Diag TVAE linear	4.37	0.76	3.93	0.62
Diag TVAE deep	4.39	0.78	3.76	0.60
LDA + Diag TVAE linear	**2.42**	**0.51**	2.31	0.46
LDA + Diag TVAE deep	2.50	**0.51**	**2.24**	0.45
FullCov TVAE linear	2.60	0.52	2.39	**0.43**
FullCov TVAE deep	2.91	0.56	2.54	0.46
Model with $\mathbf{y}$				
LDA + Diag TVAE linear	**2.67**	0.52	2.74	0.51
LDA + Diag TVAE deep	2.69	0.51	2.85	0.50
FullCov TVAE linear	2.98	0.55	**2.50**	**0.45**
FullCov TVAE deep	2.80	**0.50**	2.64	0.50

The table compares diagonal TVAE with and without LDA pre-processing; and full covariance TVAE. Using diagonal posteriors with plain i-vectors (without LDA) is an over-simplification that significantly degraded results. The LDA trick solved this issue and enables diagonal TVAE to perform similar to full covariance TVAE.

Finally, we compared TVAE with two latent variables w.r.t. only having speaker factors. Though using $\mathbf{z}$ doesn't benefit standard PLDA, it consistently improved the TVAE.

5. Conclusions

We have presented a framework to perform general non-linear probabilistic discriminant analysis on i-vectors. This framework is based on the variational autoencoder paradigm extended to deal with latent variables that are tied between i-vectors, i.e., speaker factors. We trained this model using stochastic gradient variational Bayes. The paper introduces two important novelties. First, we proposed a principled mechanism to pool the neural networks outputs corresponding to multiple i-vectors and compute the approximate posterior $q_{\phi_{\mathbf{y}}}(\mathbf{y}|\mathbf{x}_1, \dots, \mathbf{x}_N)$. We also proposed an efficient method to evaluate the likelihood ratio (Q-scoring), which only requires evaluation of the posteriors given the individual enrollment and test i-vectors ($q_{\phi_{\mathbf{y}}}(\mathbf{y}|\mathbf{x}_1)$ and $q_{\phi_{\mathbf{y}}}(\mathbf{y}|\mathbf{x}_2)$.

We evaluated this model on NIST SRE10 obtaining performance close to PLDA. Using a deep model did not provide any consistent improvement w.r.t. the linear model. We think that this is mainly because, having more trainable parameters, the VAE requires a significantly larger amount of training data. Is is also possible that the i-vector distribution is not complex enough to benefit from a non-linear model.

[2] https://github.com/kaldi-asr/kaldi/tree/master/egs/sre10/v1

[3] https://github.com/fchollet/keras

6. References

[1] 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, May 2011.

[2] N. Dehak, P. A. Torres-Carrasquillo, D. Reynolds, and R. Dehak, "Language recognition via i-vectors and dimensionality reduction," in *Proceedings of the 12th Annual Conference of the International Speech Communication Association, Interspeech 2011*. Florence, Italy: ISCA, Aug 2011, pp. 857–860.

[3] H. Behravan, V. Hautamaki, S. M. Siniscalchi, T. Kinnunen, and C.-H. Lee, "i-Vector modeling of speech attributes for automatic foreign accent Recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 1, pp. 29–41, Jan 2016. [Online]. Available: http://ieeexplore.ieee.org/document/7295591/

[4] S. O. Sadjadi, S. Ganapathy, and J. W. Pelecanos, "Speaker age estimation on conversational telephone speech using senone posterior based i-vectors," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2016*. Shanghai, China: IEEE, Mar 2016, pp. 5040–5044. [Online]. Available: http://ieeexplore.ieee.org/document/7472637/

[5] S. Ioffe, "Probabilistic linear discriminant analysis," in *Proceedings of the 9th European Conference on Computer Vision, ECCV 2006*, ser. LNCS, vol. 3954. Graz, Austria: Springer-Verlag Berlin, Heidelberg, May 2006, pp. 531–542. [Online]. Available: http://www.springerlink.com/index/654486wq8m531j4j.pdf

[6] S. J. Prince and J. H. Elder, "Probabilistic linear discriminant analysis for inferences about identity," in *Proceedings of the IEEE International Conference on Computer Vision, ICCV 2007*. Rio de Janeiro, Brazil: IEEE, Oct 2007, pp. 1–8. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4409052

[7] P. Kenny, "Bayesian speaker verification with heavy-tailed priors," in *Proceedings of Odyssey 2010 - The Speaker and Language Recognition Workshop*. Brno, Czech Republic: ISCA, Jul 2010.

[8] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proceedings of the International Conference of Learning Representations, ICLR 2014*, Banff, Alberta, Canada, Apr 2014. [Online]. Available: http://arxiv.org/abs/1312.6114

[9] D. J. Rezende, S. Mohamed, and Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," in *Proceedings of the International Conference on Machine Learning, ICML 2014*, Beijing, China, Jun 2014. [Online]. Available: http://arxiv.org/abs/1401.4082

[10] I. Gulrajani, K. Kumar, F. Ahmed, A. A. Taiga, F. Visin, D. Vazquez, and A. Courville, "PixelVAE: a latent variable model for natural images," in *Proceedings of the International Conference of Learning Representations, ICLR 2017*, Nov 2017. [Online]. Available: http://arxiv.org/abs/1611.05013

[11] S. Tan and K. C. Sim, "Learning utterance-level normalisation using Variational Autoencoders for robust automatic speech recognition," in *2016 IEEE Spoken Language Technology Workshop (SLT)*. San Diego, CA, USA: IEEE, Dec 2016, pp. 43–49. [Online]. Available: http://ieeexplore.ieee.org/document/7846243/

[12] J.-T. Chien and C.-W. Hsu, "Variational manifold learning for speaker recognition," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2017*. New Orleans, LA, USA: IEEE, Mar 2017, pp. 4935–4939.

[13] J. Villalba and E. Lleida, "Handling i-vectors from different recording conditions using multi-channel simplified PLDA in speaker recognition," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013*. Vancouver, British Columbia, Canada: IEEE, May 2013, pp. 6763 – 6767.

[14] N. Brummer, "EM for probabilistic LDA," Agnitio Research, Cape Town, South Africa, Tech. Rep. February, Feb 2010. [Online]. Available: https://sites.google.com/site/nikobrummer/EMforPLDA.pdf

[15] D. P. Kingma, T. Salimans, and M. Welling, "Improving variational inference with inverse autoregressive flow," Jun 2016. [Online]. Available: http://arxiv.org/abs/1606.04934

[16] E. Variani, X. Lei, E. McDermott, I. L. Moreno, and J. Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Florence, Italy: IEEE, May 2014, pp. 4052–4056. [Online]. Available: http://ieeexplore.ieee.org/document/6854363/

[17] G. Heigold, I. Moreno, S. Bengio, and N. Shazeer, "End-to-end text-dependent speaker verification," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2015*. Brisbane, Australia: IEEE, Apr 2015. [Online]. Available: http://arxiv.org/abs/1509.08062

[18] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The Kaldi speech recognition toolkit," in *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU2011*. Waikoloa, HI, USA: IEEE, Dec 2011, pp. 1–4.

[19] D. Garcia-Romero and C. Y. Espy-Wilson, "Analysis of i-vector length normalization in speaker recognition systems," in *Proceedings of the 12th Annual Conference of the International Speech Communication Association, Interspeech 2011*. Florence, Italy: ISCA, Aug 2011, pp. 249–252.