INTERSPEECH 2020
October 25–29, 2020, Shanghai, China

Neural Discriminant Analysis for Deep Speaker Embedding

Lantian Li, Dong Wang, Thomas Fang Zheng

Center for Speech and Language Technologies, Tsinghua University

lilt@cslt.org; wangdong99@mails.tsinghua.edu.cn; fzheng@tsinghua.edu.cn

Abstract

Probabilistic Linear Discriminant Analysis (PLDA) is a popular tool in open-set classification/verification tasks. However, the Gaussian assumption underlying PLDA prevents it from being applied to situations where the data is clearly non-Gaussian. In this paper, we present a novel nonlinear version of PLDA named as Neural Discriminant Analysis (NDA). This model employs an invertible deep neural network to transform a complex distribution to a simple Gaussian, so that the linear Gaussian model can be readily established in the transformed space. We tested this NDA model on a speaker recognition task where the deep speaker vectors (x-vectors) are presumably non-Gaussian. Experimental results on two datasets demonstrate that NDA consistently outperforms PLDA, by handling the non-Gaussian distributions of the x-vectors.

Index Terms: speaker recognition, neural discriminant analysis

1. Introduction

Probabilistic Linear Discriminant Analysis (PLDA) [1, 2] has been used in a wide variety of recognition tasks, such as speaker recognition (SRE) [3]. In nearly all the situations, PLDA cooperates with a *speaker embedding* front-end, and plays the role of scoring the similarity between one or a few enrollment utterances and the test utterance, represented in the form of speaker vectors. Traditional speaker embedding approaches are based on statistical models, in particular the i-vector model [4], and recent embedding approaches are based on deep neural networks (DNNs) [5, 6], for which the x-vector model [7, 8, 9] is the most successful.

PLDA is a full-generative model, based on two primary assumptions: (1) all the classes are Gaussians and these Gaussians share the same covariance matrix; (2) the class means are distributed following a Gaussian. The full-generative model offers a principle way to deal with classification tasks where the classes are represented by limited data. Taking SRE as an example, in most cases, a speaker registers itself with only one or a few enrollment utterances, which means that the distribution of the speaker is not fully represented and the identification/verification has to be conducted base on an uncertain model. The PLDA model solves this problem in a Bayesian way: it represents each speaker as a Gaussian with an *uncertain* mean, and computes the likelihood of a test utterance by marginalizing the uncertainty. Due to this elegant uncertainty treatment, PLDA has been widely used in SRE, and has achieved state-of-the-art performance in many benchmarks, in particular when combined with length normalization [10].

Although promising, PLDA suffers from a limited representation capability. Specifically, PLDA is a linear Gaussian model, and the prior, the conditional, and the marginal are all assumed to be Gaussian. This means that if the distribution of speaker vectors do not satisfy this assumption, PLDA cannot represent the data well, leading to inferior performance. This problem is not very serious for speaker vectors that are derived from statistical models where the speaker vectors have been assumed to be Gaussian, e.g., the JFA model [11, 12] and the i-vector model [4]. However, for speaker vectors derived from models that do not possess such a Gaussian assumption, PLDA may be biased. This is particularly the case for x-vectors [7]: they are derived from a deep neural network and there is not any explicit or implicit Gaussian constraint on the prior and the conditional.

In fact, the importance of Gaussianality of the data for PLDA has been noticed for a long time. For example, Kenny et al. [3] found that i-vectors exhibit a heavy-tail property and so are not appropriate to be modeled by Gaussians. They presented a heavy-tail PLDA where the prior and the conditional are set to be Student's t-distributions. Garcia et al [10] found a simple length normalization can improve Gaussianality of i-vectors, and the traditional PLDA model can recover the performance of the heavy-tail PLDA if the test i-vectors are length-normalized.

The non-Gaussianality of x-vectors were recognized by Li et al. [13]. They found that x-vectors exhibit more complex within-class distributions compared to i-vectors, and confirmed the non-Gaussianality of x-vectors by computing the Skewness and Kurtosis of the within-class and between-class distributions. Further analysis was conducted in [14], and a VAE model was used to perform normalization for x-vectors. Recently, a more complex normalization technique based on a normalization flow was conducted by Cai et. al [15]. All these studies try to produce more Gaussian speaker vectors, so that PLDA model can be employed more effectively.

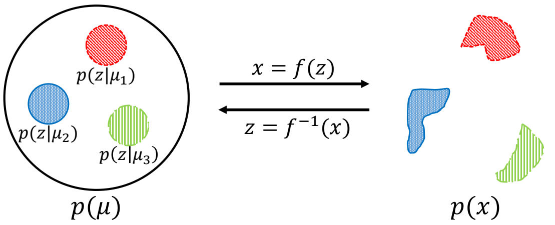

Figure 1: *Neural discriminant analysis. The complex within-class distributions (on the right) are transformed to Gaussians with shared covariance by an invertible transform, and then the linear Bayesian model is established in the transformed space. Each color represents an individual class (speaker).*

All the approaches mentioned above cannot be regarded as perfect. The heavy-tailed PLDA and the length normalization cannot deal with complex distributions, and the normalization techniques do not optimize the normalization model and the scoring model in a joint way. In this paper, we present a Neural Discriminant Analysis (NDA) model that can deal with non-Gaussian data in a principle way. It keeps the strength of PLDA in handling uncertain class representations, while offers the

http://dx.doi.org/10.21437/Interspeech.2020-2542

capability to represent data with complex within-class distributions. The central idea, as shown in Figure 1, comes from the distribution transformation theorem [16], by which any form of non-Gaussian distribution can be transformed to a Gaussian, and so a linear Bayesian model can be established on the transformed variables. We tested the NDA model on SRE tasks with two datasets, VoxCeleb and CNCeleb, and achieved rather promising results.

The organization of this paper is as follows. Section 2 presents the NDA model, and experiments are reported in Section 3. The paper is concluded in Section 4.

2. Method

2.1. Revisit PLDA

PLDA models the generation process of multiple classes. For each class, the generation process is factorized into two steps: firstly sample a class mean $\boldsymbol{\mu}$ from a Gaussian prior, and secondly sample a class member $\boldsymbol{x}$ from a conditional which is another Gaussian centered on $\boldsymbol{\mu}$. Without loss of generality, we shall assume the prior is diagonal and the conditional is isotropic[1]. This leads to the following probabilistic model:

$$p(\boldsymbol{\mu}) = N(\boldsymbol{\mu}; \mathbf{0}, \boldsymbol{\epsilon}\mathbf{I}) \tag{1}$$

$$p(\boldsymbol{x}|\boldsymbol{\mu}) = N(\boldsymbol{x}; \boldsymbol{\mu}, \sigma\mathbf{I}), \tag{2}$$

where $\boldsymbol{\epsilon}\mathbf{I}$ and $\sigma\mathbf{I}$ are the between-class and within-class covariances, respectively.

With this model, the probability that $\boldsymbol{x}_1, ..., \boldsymbol{x}_n$ belong to the same speaker can be computed by:

$$\begin{aligned} p(\boldsymbol{x}_1, ..., \boldsymbol{x}_n) = & \int p(\boldsymbol{x}_1, ..., \boldsymbol{x}_n|\boldsymbol{\mu})p(\boldsymbol{\mu})\mathrm{d}\boldsymbol{\mu} \\ & \propto \prod_j \epsilon_j^{-1/2} \prod_j (\frac{n}{\sigma} + \frac{1}{\epsilon_j})^{-1/2} \\ & \exp\Big\{ - \frac{1}{2\sigma}\{ \sum_i ||\boldsymbol{x}_i||^2 - \sum_j \frac{n^2\epsilon_j}{n\epsilon_j + \sigma}\widehat{x}_j^2 \} \Big\}, \end{aligned} \tag{3}$$

where j indexes the dimension.

For speaker recognition, our essential goal is to estimate the probability $p(\boldsymbol{x}|\boldsymbol{x}_1, ..., \boldsymbol{x}_n)$, where $\boldsymbol{x}$ is the test data and $\boldsymbol{x}_1, ..., \boldsymbol{x}_n$ represent the enrollment data. Focusing on the verification task, the above probability should be normalized by a background probability that $\boldsymbol{x}$ belongs to any possible classes. This leads to a normalized likelihood (NL) for speaker verification:

$$R(\boldsymbol{x}; \boldsymbol{x}_1, ..., \boldsymbol{x}_n) = \frac{p(\boldsymbol{x}|\boldsymbol{x}_1, ..., \boldsymbol{x}_n)}{p(\boldsymbol{x})}.$$

It is easy to verify that the above NL score is equal to the likelihood ratio (LR), the standard form for PLDA scoring in the seminal papers [1, 2]:

$$R(\boldsymbol{x}; \boldsymbol{x}_1, ..., \boldsymbol{x}_n) = \frac{p(\boldsymbol{x}, \boldsymbol{x}_1, ..., \boldsymbol{x}_n)}{p(\boldsymbol{x})p(\boldsymbol{x}_1, ..., \boldsymbol{x}_n)},$$

and all the terms in the numerator and denominator can be computed by Eq. (3).

2.2. Neural discriminant analysis (NDA)

All the derivations for the PLDA model are based on the Gaussian assumption on the prior and conditional distributions, shown by Eq. (1) and Eq. (2). We shall relax this assumption by introducing a new probabilistic discriminant model based on neural net and Bayesian inference.

2.2.1. Distribution transform

Suppose an invertible transform f maps a variable $\boldsymbol{z}$ to $\boldsymbol{x}$, i.e., $\boldsymbol{x} = f(\boldsymbol{z})$. According to the principle of distribution transformation for continuous variables [16], we have:

$$p(\boldsymbol{x}) = p(\boldsymbol{z})|\frac{\partial f^{-1}(\boldsymbol{x})}{\partial \boldsymbol{x}}|, \tag{4}$$

where the second term is the absolute value of the determinant of the Jacobian matrix of f^{-1}, the inverse transform of f. This term reflects the change of the volume of the distribution with the transform, and is often called the *entropy term*. For simplicity, we will denote this term by $J_{\boldsymbol{x}}$:

$$p(\boldsymbol{x}) = p(\boldsymbol{z})J_{\boldsymbol{x}}. \tag{5}$$

Note that $p(\boldsymbol{x})$ and $p(\boldsymbol{z})$ are in different distribution forms. If we assume $p(\boldsymbol{z})$ to be a Gaussian, then this (inverse) transform plays a role of Gaussianization [17]. It was demonstrated that if f is complex enough, any complex $p(\boldsymbol{x})$ can be transformed to a Gaussian or a uniform distribution [18]. Recently, there is numerous research on designing more efficient transform functions by deep neural networks, and most of them adopt a modular architecture by which simple invertible modules are concatenated to attain a complex function. This architecture is often called **Normalizing Flow (NF)** [18, 19, 20, 21, 22] and will be used in this work to implement the transform f.

2.2.2. NDA model

A key idea of NDA is to use an NF to map the distribution of one class in the $\boldsymbol{x}$ space to the $\boldsymbol{z}$ space, where $p(\boldsymbol{z})$ is a Gaussian. After this transform, we can build a linear Gaussian model in the $\boldsymbol{z}$ space to describe the non-Gaussian observations in the $\boldsymbol{x}$ space, as shown in Figure 1.

As in PLDA, we assume that the prior $p(\boldsymbol{\mu})$ for class mean $\boldsymbol{\mu}$ is a diagonal Gaussian, and the conditional distribution $p(\boldsymbol{z}|\boldsymbol{\mu})$ is a standard multivariate Gaussian whose covariance is set to $\mathbf{I}$. This linear Gaussian model will represent the complex non-Gaussian data $\boldsymbol{x}$ via the invertible transform f represented by an NF.

It can be shown that the probability $p(\boldsymbol{x}_1, ..., \boldsymbol{x}_n)$ can be derived as follows:

$$\begin{aligned} p(\boldsymbol{x}_1, ..., \boldsymbol{x}_n) &= \int p(\boldsymbol{x}_1, ..., \boldsymbol{x}_n|\boldsymbol{\mu})p(\boldsymbol{\mu})\mathrm{d}\boldsymbol{\mu} \\ &= \prod_i^n J_{\boldsymbol{x}_i} \int p(\boldsymbol{z}_1, ..., \boldsymbol{z}_n|\boldsymbol{\mu})p(\boldsymbol{\mu})\mathrm{d}\boldsymbol{\mu} \\ &= \prod_i^n J_{\boldsymbol{x}_i} p(\boldsymbol{z}_1, ..., \boldsymbol{z}_n). \end{aligned} \tag{6}$$

Note that the distribution $p(\boldsymbol{\mu})$ and $p(\boldsymbol{z}|\boldsymbol{\mu})$ are both Gaussians, and so $p(\boldsymbol{z}_1, ..., \boldsymbol{z}_n)$ can be easily computed as in PLDA, following Eq. (3).

[1] It can be verified that a linear transform does not change the PLDA score, which makes it possible to transform any form of Gaussians of the prior and the conditional to the form of Eq. (1) and Eq. (2). Besides, PLDA with a low-rank loading matrix can be regarded as a special form of Eq. (1), by setting some of the diagonal elements of $\boldsymbol{\epsilon}$ to be 0.

This is a rather simple form and it indicates that if we can train an NF f, the complex marginal distribution $p(\boldsymbol{x}_1, ..., \boldsymbol{x}_n)$ can be computed by transforming the observation $\boldsymbol{x}_i$ to a latent code $\boldsymbol{z}_i$, and then computing the simple marginal $p(\boldsymbol{z}_1, ..., \boldsymbol{z}_n)$ plus a correction term $\prod_i^n J_{\boldsymbol{x}_i}$. Interestingly, this correction term will be cancelled when computing the likelihood ratio for SRE scoring:

$$\begin{aligned} R(\boldsymbol{x}; \boldsymbol{x}_1, ..., \boldsymbol{x}_n) &= \frac{p(\boldsymbol{x}, \boldsymbol{x}_1, ..., \boldsymbol{x}_n)}{p(\boldsymbol{x})p(\boldsymbol{x}_1, ..., \boldsymbol{x}_n)} \\ &= \frac{p(\boldsymbol{z}, \boldsymbol{z}_1, ..., \boldsymbol{z}_n)}{p(\boldsymbol{z})p(\boldsymbol{z}_1, ..., \boldsymbol{z}_n)}. \end{aligned}$$

We therefore create a discriminant model which keeps the simple linear Gaussian form in the latent space, but can deal with any complex within-class distributions. Finally, we note that if the transform f is linear, NDA falls back to PLDA.

2.2.3. Model training

The NDA model can be trained following the maximum-likelihood (ML) principle. Since all the speakers are independent, the objective function is formulated by:

$$\begin{aligned} \mathcal{L}(f, \boldsymbol{\epsilon}) &= \sum_{k=1}^{K} \log p(\boldsymbol{x}_1, ..., \boldsymbol{x}_{n_k}) \\ &= \sum_{k=1}^{K} \log \big\{ \prod_{i}^{n_k} J_{\boldsymbol{x}_i} p(\boldsymbol{z}_1, ..., \boldsymbol{z}_{n_k}) \big\}, \end{aligned}$$

where K is the number of speakers in the training data. During the training, firstly transform the training samples $\boldsymbol{x}_i$ to $\boldsymbol{z}_i$, and then compute $J_{\boldsymbol{x}_i}$ based on f and $\boldsymbol{x}_i$, secondly compute $p(\boldsymbol{z}_1, ..., \boldsymbol{z}_n)$ following Eq. (3). Note that the covariance of $p(\boldsymbol{z}|\boldsymbol{\mu})$ has been fixed to $\mathbf{I}$ and so σ is not a trainable parameter.

An important issue of the training algorithm is that for each speaker, all the data need to be processed all at once. Therefore, the mini-batch design should be speaker-based. Moreover, we found the training will be unstable if there are too few speakers in one mini-batch. We solve this problem by postponing the model update when adequate speakers have been processed.

3. Experiments

3.1. Data

Three datasets were used in our experiments: VoxCeleb [23, 24], SITW [25] and CNCeleb [26]. More information about these three datasets is presented below.

VoxCeleb: This is a large-scale audiovisual speaker dataset collected by the University of Oxford, UK. The entire database involves VoxCeleb1 and VoxCeleb2. This dataset, after removing the utterances shared by SITW, was used to train the front-end x-vector, PLDA and NDA models. The entire dataset contains $2,000$+ hours of speech signals from $7,000$+ speakers. Data augmentation was applied to improve robustness, with the MUSAN corpus [27] was used to generate noisy utterances, and the room impulse responses (RIRS) corpus [28] was used to generate reverberant utterances.

SITW: This is a standard evaluation dataset excerpted from VoxCeleb1, which consists of 299 speakers. In our experiments, both the *Dev.Core* and *Eval.Core* were used for evaluation.

CNCeleb: This is a large-scale free speaker recognition dataset collected by Tsinghua University. It contains more than 130k utterances from $1,000$ Chinese celebrities. It covers 11 diverse genres, which makes speaker recognition on this dataset much more challenging than on SITW. The entire dataset was split into two parts: *CNCeleb.Train*, which involves $111,257$ utterances from 800 speakers, was used to train the PLDA and the NDA models; *CNCeleb.Eval*, which involves $18,024$ utterances from 200 speakers, was used for evaluation.

3.2. Model Settings

Our SRE system consists of two components: an x-vector frontend that produces speaker vectors, and a scoring model that produces pair-wise scores to make genuine/imposter decisions.

3.2.1. Front-end

x-vector: The x-vector frontend was created using the Kaldi toolkit [29], following the VoxCeleb recipe. The acoustic features are 40-dimensional Fbanks. The main architecture contains three components. The first component is the feature-learning component, which involves 5 time-delay (TD) layers to learn frame-level speaker features. The slicing parameters for these 5 TD layers are: $\{t$-2, t-1, t, t+1, t+2$\}$, $\{t$-2, t, t+2$\}$, $\{t$-3, t, t+3$\}$, $\{t\}$, $\{t\}$. The second component is the statistical pooling component, which computes the mean and standard deviation of the frame-level features from a speech segment. The final one is the speaker-classification component, which discriminates between different speakers. This component has 2 full-connection (FC) layers and the size of its output is $7,185$, corresponding to the number of speakers in the training set. Once trained, the 512-dimensional activations of the penultimate FC layer are read out as an x-vector.

3.2.2. Back-end

PLDA: We implemented the standard PLDA model [1] using the Kaldi toolkit [29].

NDA: We implemented the proposed NDA model in PyTorch. The invertible transform f was implemented using the *RealNVP* architecture [22], a particular NF that does not preserve the volume of the distribution. We used 10 non-volume preserving (NVP) layers, and the Adam optimizer [30] was used to train the model, with the learning rate set to 0.001. For VoxCeleb, each mini-batch covers x-vectors from 600 speakers, and for CNCeleb, each mini-batch covers x-vectors from 200 speakers.

3.3. Basic results

Experimental results on SITW Dev.Core, SITW Eval.Core and CNCeleb.Eval are shown in Table 3. The results are reported in terms of equal error rate (EER) and minimum of the normalized detection cost function (minDCF) with two settings: one with the prior target probability P_{tar} set to 0.01 (DCF(10^{-2})), and the other with P_{tar} set to 0.001 (DCF(10^{-3})).

Firstly, we focus on the full-dimensional PLDA (512) and NDA (512) scoring. It can be observed that NDA scoring consistently outperformed PLDA scoring on the three evaluation datasets, confirming that NDA is effective and more suitable as the x-vector back-end. Besides, the performance of NDA on the CNCeleb.Eval is obviously better than that of PLDA (13.95% vs. 12.51%). Considering the higher complexity of CNCeleb [26], this demonstrates that NDA has better capability in dealing with complicated and challenging test conditions.

Secondly, we discarded some least discriminative dimensions, i.e., dimensions corresponding to the smallest ϵ_i. This approximates the subspace PLDA/NDA. The results are shown in Table 3 as well. It can be found that with this dimensionality reduction, performance improvement was generally observed. Once again, NDA outperforms PLDA on almost all the datasets and with all the settings.

Table 1: *Basic results on three evaluation datasets.*

SITW Dev.Core					
Front-end	Scoring	Dim	DCF(10^{-2})	DCF(10^{-3})	EER(%)
x-vector	PLDA	512	0.485	0.704	4.082
	PLDA	300	0.380	0.581	3.389
	PLDA	**150**	0.307	0.480	**3.196**
	NDA	512	0.480	0.720	4.043
	NDA	300	0.390	0.593	3.466
	NDA	**150**	0.312	0.487	**3.196**

SITW Eval.Core					
Front-end	Scoring	Dim	DCF(10^{-2})	DCF(10^{-3})	EER(%)
x-vector	PLDA	512	0.497	0.764	4.456
	PLDA	300	0.393	0.619	3.745
	PLDA	150	0.333	0.503	3.581
	NDA	512	0.494	0.771	4.155
	NDA	300	0.398	0.637	3.527
	NDA	**150**	0.343	0.516	**3.417**

CNCeleb.Eval					
Front-end	Scoring	Dim	DCF(10^{-2})	DCF(10^{-3})	EER(%)
x-vector	PLDA	512	0.691	0.837	13.95
	PLDA	300	0.674	0.822	13.72
	PLDA	150	0.660	0.816	13.63
	NDA	512	0.623	0.770	12.51
	NDA	**300**	0.613	0.757	**12.45**
	NDA	150	0.612	0.752	12.60

3.4. Analysis for Gaussianality

We have argued that the strength of NDA lies in the fact that the nonlinear transform f can map non-Gaussian observations $\boldsymbol{x}$ to Gaussian latent codes $\boldsymbol{z}$. To test this argument, we compute the Gaussianality of the x-vectors before and after the NDA transform. We compute the Skewness and Kurtosis for the marginal distribution (overall distribution without class labels), conditional distribution (within-class distribution), and prior distribution (distribution of class means). The results are reported in Table 2.

It can be seen that the values of Skewness and Kurtosis of the x-vectors are substantially reduced after NDA transform, especially with the conditional distribution. This is expected as the conditional distribution has been assumed to be Gaussian when NDA is designed and trained. This improved Gaussianality allows a linear Gaussian model in the transformed space, as supposed by NDA.

Table 2: *Gaussianality of x-vectors with/without NDA transform.*

VoxCeleb		Marginal		Conditional		Prior	
Front-end	NDA	Skew	Kurt	Skew	Kurt	Skew	Kurt
x-vector	-	-0.087	-0.361	0.015	1.060	-0.045	-0.524
	+	0.015	0.134	**-0.004**	**0.267**	0.045	0.301

CNCeleb		Marginal		Conditional		Prior	
Front-end	NDA	Skew	Kurt	Skew	Kurt	Skew	Kurt
x-vector	-	-0.139	0.180	-0.034	1.160	-0.160	-0.271
	+	0.002	0.122	**0.001**	**0.163**	-0.022	1.244

3.5. Analysis for LDA pre-processing

It is well known that LDA-based dimension reduction often provides significant performance improvement for x-vector systems [13]. Recently, the authors found that the contribution of LDA for x-vector systems lies in normalization rather than discrimination. More specifically, for x-vectors, the least discriminative dimensions coincide with the most non-Gaussian dimensions. Therefore, LDA may improve the Gaussianality of x-vectors by discarding the least discriminative dimensions [15].

Considering the success of the combination of LDA and PLDA, it is interesting to test if LDA pre-processing contributes to NDA. The results are shown in Table 3, where the dimensionality of the LDA projection space was set to 150 for VoxCeleb dataset and 300 for CNCeleb dataset. These configurations delivered the best performance with both PLDA and NDA.

It can be found that the performance was slightly improved after the LDA pre-processing, with both PLDA and NDA. This is a bit surprising for NDA, as NDA can deal with non-Gaussian data by itself, and so does not require LDA to improve the Gaussianality of the data. One possibility is that the reduced dimensionality allows a better NDA modeling with limited data. However, more investigation is required.

Table 3: *Performance with/without LDA pre-processing.*

SITW Dev.Core				
Front-end	Scoring	DCF(10^{-2})	DCF(10^{-3})	EER(%)
x-vector	PLDA (150)	0.307	0.480	3.196
	NDA (150)	0.312	0.487	3.196
x-vector + LDA (150)	PLDA NDA	0.301 0.295	0.469 0.472	3.157 **3.080**

SITW Eval.Core				
Front-end	Scoring	DCF(10^{-2})	DCF(10^{-3})	EER(%)
x-vector	PLDA (150)	0.333	0.503	3.581
	NDA (150)	0.343	0.516	3.417
x-vector + LDA (150)	PLDA NDA	0.329 0.335	0.496 0.508	3.554 **3.280**

CNCeleb.Eval				
Front-end	Scoring	DCF(10^{-2})	DCF(10^{-3})	EER(%)
x-vector	PLDA (300)	0.674	0.822	13.72
	NDA (300)	0.613	0.757	12.45
x-vector + LDA (300)	PLDA NDA	0.675 0.561	0.821 0.681	13.73 **12.28**

4. Conclusions

We proposed a novel NDA model in this paper. It is a nonlinear extension of PLDA and can deal with data with complex within-class distributions. The key component of NDA is an NF-based invertible transform, which maps a complex distribution to a simple Gaussian so that a linear Gaussian model can be established in the transformed space. We applied NDA to SRE tasks and compared the performance with PLDA. Results on the SITW and the CNCeleb datasets demonstrated that NDA can deliver consistently better performance compared to PLDA. Future work will investigate the joint training of the NDA scoring model and the speaker embedding model, and apply NDA to raw acoustic features directly.

5. Acknowledgements

This work was supported by the National Natural Science Foundation of China No. 61633013. Dong Wang is the corresponding author.

6. References

[1] S. Ioffe, "Probabilistic linear discriminant analysis," in *European Conference on Computer Vision (ECCV)*. Springer, 2006, pp. 531–542.

[2] S. J. Prince and J. H. Elder, "Probabilistic linear discriminant analysis for inferences about identity," in *2007 IEEE 11th International Conference on Computer Vision*. IEEE, 2007, pp. 1–8.

[3] P. Kenny, "Bayesian speaker verification with heavy-tailed priors." in *Odyssey*, 2010, p. 14.

[4] 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.

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

[6] L. Li, Y. Chen, Y. Shi, Z. Tang, and D. Wang, "Deep speaker feature learning for text-independent speaker verification," in *Proceedings of the Annual Conference of International Speech Communication Association (INTERSPEECH)*, 2017, pp. 1542–1546.

[7] 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)*. IEEE, 2018, pp. 5329–5333.

[8] K. Okabe, T. Koshinaka, and K. Shinoda, "Attentive statistics pooling for deep speaker embedding," in *Proceedings of the Annual Conference of International Speech Communication Association (INTERSPEECH)*, 2018, pp. 2252–2256.

[9] W. Cai, J. Chen, and M. Li, "Exploring the encoding layer and loss function in end-to-end speaker and language recognition system," *arXiv preprint arXiv:1804.05160*, 2018.

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

[11] P. Kenny, G. Boulianne, P. Ouellet, and P. Dumouchel, "Joint factor analysis versus eigenchannels in speaker recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 4, pp. 1435–1447, 2007.

[12] P. Kenny, "Joint factor analysis of speaker and session variability: Theory and algorithms," *CRIM, Montreal,(Report) CRIM-06/08-13*, vol. 14, pp. 28–29, 2005.

[13] L. Li, Z. Tang, Y. Shi, and D. Wang, "Gaussian-constrained training for speaker verification," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6036–6040.

[14] Y. Zhang, L. Li, and D. Wang, "VAE-based regularization for deep speaker embedding," *arXiv preprint arXiv:1904.03617*, 2019.

[15] Y. Cai, L. Li, D. Wang, and A. Abel, "Deep normalization for speaker vectors," *arXiv preprint arXiv:2004.04095*, 2020.

[16] W. Rudin, *Real and complex analysis*. Tata McGraw-hill education, 2006.

[17] S. S. Chen and R. A. Gopinath, "Gaussianization," in *NIPS 2001*, 2001.

[18] G. Papamakarios, E. Nalisnick, D. J. Rezende, S. Mohamed, and B. Lakshminarayanan, "Normalizing flows for probabilistic modeling and inference," *arXiv preprint arXiv:1912.02762*, 2019.

[19] E. G. Tabak and C. V. Turner, "A family of nonparametric density estimation algorithms," *Communications on Pure and Applied Mathematics*, vol. 66, no. 2, pp. 145–164, 2013.

[20] O. Rippel and R. P. Adams, "High-dimensional probability estimation with deep density models," *arXiv preprint arXiv:1302.5125*, 2013.

[21] L. Dinh, D. Krueger, and Y. Bengio, "Nice: Non-linear independent components estimation," *arXiv preprint arXiv:1410.8516*, 2014.

[22] L. Dinh, J. Sohl-Dickstein, and S. Bengio, "Density estimation using real NVP," *arXiv preprint arXiv:1605.08803*, 2016.

[23] A. Nagrani, J. S. Chung, and A. Zisserman, "VoxCeleb: a large-scale speaker identification dataset," in *Proceedings of the Annual Conference of International Speech Communication Association (INTERSPEECH)*, 2017.

[24] J. S. Chung, A. Nagrani, and A. Zisserman, "VoxCeleb2: Deep speaker recognition," in *Proceedings of the Annual Conference of International Speech Communication Association (INTERSPEECH)*, 2018, pp. 1086–1090.

[25] M. McLaren, L. Ferrer, D. Castan, and A. Lawson, "The speakers in the wild (SITW) speaker recognition database." in *Proceedings of the Annual Conference of International Speech Communication Association (INTERSPEECH)*, 2016, pp. 818–822.

[26] Y. Fan, J. Kang, L. Li, K. Li, H. Chen, S. Cheng, P. Zhang, Z. Zhou, Y. Cai, and D. Wang, "CN-CELEB: a challenging Chinese speaker recognition dataset," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020.

[27] D. Snyder, G. Chen, and D. Povey, "Musan: A music, speech, and noise corpus," *arXiv preprint arXiv:1510.08484*, 2015.

[28] T. Ko, V. Peddinti, D. Povey, M. L. Seltzer, and S. Khudanpur, "A study on data augmentation of reverberant speech for robust speech recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 5220–5224.

[29] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The Kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*, 2011.

[30] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.