INTERSPEECH 2019
September 15–19, 2019, Graz, Austria

Tied Mixture of Factor Analyzers Layer to Combine Frame Level Representations in Neural Speaker Embeddings

Nanxin Chen, Jesús Villalba, Najim Dehak

Center for Language and Speech Processing
Johns Hopkins University, Baltimore, MD, USA
{bobchennan,jvilla17,ndehak3}@jhu.edu

Abstract

In this paper, a novel neural network layer is proposed to combine frame-level into utterance-level representations for speaker modeling. We followed the assumption that the frame-level outputs of the speaker embedding (a.k.a x-vector) encoder are multi-modal. Therefore, we modeled the frame-level information as a mixture of factor analyzers with latent variable (utterance embedding) tied across frames and mixture components, in as similar way as in the i-vector approach. We denote this layer as Tied Mixture of Factor Analyzers (TMFA) layer. The optimal value of the embedding is obtained by minimizing the reconstruction error of the frame-level representations given the embedding and the TMFA model parameters. However, the TMFA layer parameters (factor loading matrices, means and precisions) were trained with cross-entropy loss as the rest of parameters of the network. We experimented on the Speaker Recognition Evaluation 2016 Cantonese as well as in the Speaker in the Wild datasets. The proposed pooling layer improved w.r.t. mean plus standard deviation pooling–standard in x-vector approach– in most of the conditions evaluated; and obtained competitive performance w.r.t. the recently proposed learnable dictionary encoding pooling method, which also assumes multi-modal frame-level representations.

Index Terms: speaker recognition, least squares, embeddings, x-vectors, pooling

1. Introduction

Speaker recognition systems identify people from their voices. Essentially Given an enrollment recording and a test recording, we decide whether both were uttered by the same speaker or not. Typically, those systems are based on computing a compact vector representation (a.k.a. embedding) for each utterance. These embeddings retain essential information about the speaker like identity, age, emotion, etc. Enrollment and test embeddings are compared to identify the speaker. Speaker recognition has applications in forensics, defense, electronic banking, and human-computer interaction.

Speaker recognition itself has very long history. Starting from the 20th century, researchers proposed using Gaussian Mixture Model-Universal Background Model(GMM-UBM) [1] to model the distribution of acoustic features in a given utterance. Later, the Joint Factor Analysis [2, 3] and i-vector [4] approaches proposed to embed the GMM super-vector means in low-rank subspace. The coordinates of the utterance in that subspace is the embedding that we use in the downstream classifier. The cosine similarities and probabilistic linear discriminant analysis (PLDA) [5] are the common choice for embedding comparison in speaker verification.

Recently with rapid development of deep learning, neural-based speaker recognition systems start to become popular. In the beginning, researchers applied neural network for text-dependent speaker recognition [6, 7, 8, 9] with architectures designed to handle short utterances. Later in [10, 11], authors proposed the combination of Time Delayed Neural Network (TDNN) with statistical pooling layer for text-independent speaker recognition, which is able to handle long utterances. Borrowing the idea from the super-vector mean in Gaussian mixture model, the Learnable Dictionary Encoding (LDE) pooling method treats the frame-level neural network representations as a multi-modal distribution [12, 13, 14]. Both, statistical and LDE pooling, have been proved to be effective in recent NIST Speaker Recognition Evaluation (SRE) 2018.

In this work, a new pooling method is proposed. We adopt similar idea in LDE [12, 13, 14] of assuming multi-modal hidden representations, and combined it with the i-vector approach. In essence, LDE is related to the classical idea of using unconstrained maximum likelihood estimation to obtain the utterance GMM super-vector, and then, project that super-vector into low-dimension embeddings, e.g., using PCA or LDA. Meanwhile, we propose to follow the i-vector approach and model hidden representations as a mixture of factor analyzers with latent variable (embedding) tied across frames and mixture components. In this framework, we do not need to calculate the super-vector mean, which is hidden, and directly optimize the embedding that best explains the hidden representations. In theory, this method has all the potential advantages of i-vectors. For example in LDE, we need to observe samples form the GMM components to get a good estimation of the GMM mean. Meanwhile, with this method, we infer the latent factor given the observed components and we could recover the means of the unobserved ones just multiplying the embedding by the factor loading matrices.

This paper is organized as follows. Section 2 introduces the classical i-vector approach, speaker embeddings and the state-of-the-art pooling methods: statistical pooling and learnable dictionary encoding. The new pooling method we proposed is described with mathematical details in Section 3. Section 4 introduces the experimental setup used for training and evaluation. Results and discussion for the SRE16 Cantonese and Speakers in the Wild datasets are given in Section 5. Section 6 summarizes the conclusions and proposes directions of future research.

2. Previous Work

2.1. i-Vectors

The i-vector framework models the utterance acoustic features as a Gaussian mixture model (GMM). The GMM super-vector mean $\mathbf{m}$ of the utterance is constrained to live in a low-dimensional subspace as

$$\mathbf{m} = \mathbf{M} + \mathbf{T}\mathbf{v} \tag{1}$$

where $\mathbf{M} = (\boldsymbol{\mu}_1^T, \ldots, \boldsymbol{\mu}_C^T)^T$ is the Universal Background Mean trained with a large amount of unlabelled data; and $\mathbf{T}$ is a low-rank matrix–total variability matrix– mapping low-dimensional latent vector $\mathbf{v}$ to the high-dimensional super-vector space. The latent variable $\mathbf{v}$ is standard normal *a priori* while the *maximum a posteriori* value of $\mathbf{v}$ is the embedding–i-vector– that we use for downstream classification tasks. Expectation maximization is used to train, both, UBM and i-vector model parameters.

2.2. x-Vectors

Neural network embeddings (a.k.a. x-vectors) are obtained using a neural network trained to classify the speakers in the training set [10, 11]. x-Vector networks are divided into three parts. First, an encoder network extracts frame level representations from the acoustic features. This is followed by a global temporal pooling layer that produces a single vector per utterance. Finally, a feed forward classification network processes the pooling vector to produce speaker class posteriors. Typically in the evaluation phase, the x-vector is obtained from the first affine transform after the pooling layer, while the last layers of the network are discarded. Different x-vector systems are characterized by different encoder architectures; pooling methods and training objectives. In this paper, we propose a pooling method based on factor analysis. The state-of-the-art pooling methods are described below.

2.3. Global mean and standard deviation pooling

In [11], a statistical pooling layer (SP) is proposed to address problem of variable number of frames. To summarize the whole utterance into a fixed-dimensional network embedding, they calculated mean and standard deviation for each dimension of the representations obtained from a Time Delay Neural Network (TDNN). Experiments show that standard deviation is important to model the speaker improving performance than just average pooling.

An assumption implicitly made by statistical pooling is that outputs from TDNN layers are uni-modal. While this is not true for MFCC and filter-bank features, this may not be a problem for statistical pooling since TDNN may normalize frame representations to make it uni-modal.

2.4. Learnable Dictionary Encoding

While the previous pooling method assumed to be uni-modal, in learnable dictionary encoding [12, 13, 14], the authors assumes that outputs $\mathbf{x}_t$ coming from ResNet-34 are multi-modal. It can be represented by a Gaussian mixture model, similar to classical GMM-UBM [1]. Each mixture has its own parameters, center μ_c and isotropic precision s_c. The responsibility $\gamma_{t,c}$ for each frame $\mathbf{x}_t$ and each cluster c is given by

$$\gamma_{t,c} = \frac{\exp(-\frac{1}{2}s_c \left\|\mathbf{x}_t - \boldsymbol{\mu}_c\right\|^2 + b_c)}{\sum_{c'} \exp(-\frac{1}{2}s_{c'} \left\|\mathbf{x}_t - \boldsymbol{\mu}_{c'}\right\|^2 + b_{c'})} \tag{2}$$

where b_c accounts for the Gaussian log-weight and log-scaling constants. However in our experiments, making b_c equal to zero gave slightly better results. For each cluster, we compute pooling vector by the maximum likelihood deviation from the UBM mean,

$$\mathbf{m}_c^{'} = \frac{\sum_t \gamma_{t,c}\mathbf{x}_t}{\sum_t \gamma_{t,c}} - \boldsymbol{\mu}_c \qquad c = 1, \ldots, C \tag{3}$$

We can estimate a pooling super-vector for each utterance can be estimated by concatenating pooling vectors from each cluster c. Then, $\mathbf{W}$ apply a linear transform to the super-vector as outputs from pooling layer, which is equivalent to

$$\mathbf{v}_{\mathrm{LDE}} = \sum_c \mathbf{W}_c \mathbf{m}_c^{'} \tag{4}$$

where $\mathbf{W}_c$ has similar role as the total variability matrix in i-vector. All model parameters including $\boldsymbol{\mu}_c$, s_c and $\mathbf{W}_c$ are estimated by back-propagation. Objective function again is cross-entropy using one-hot speaker id as true label.

3. Tied Mixture of Factory Analyzers Pooling Layer

3.1. Layer model description

Inspired by the i-vector approach, we proposed to use a similar low-dimensional factorization for the super-vector $\mathbf{m}'$ in LDE as $\mathbf{m}^{'} = \mathbf{T}\mathbf{v}$ where $\mathbf{v}$ is the low-dimensional utterance embedding and $\mathbf{T}$ is the total variability matrix mapping the embedding to high-dimensional super-vector space. Thus, we model observations $\mathbf{x}_t$ using a mixture of factor analyzers model,

$$\begin{aligned} z_{t,c} &\sim \mathrm{Categorical}(\boldsymbol{\pi}) \\ \mathbf{x}_t &\sim \mathcal{N}(\boldsymbol{\mu}_{z_{t,c}} + \mathbf{T}_{z_{t,c}}\mathbf{v}, s_{z_{t,c}}^{-1}\mathbf{I}) \end{aligned} \tag{5}$$

where $\boldsymbol{\pi}$ is the component prior; $\boldsymbol{\mu}_k$, s_k and $\mathbf{T}_k$ are UBM mean, precision and total variability matrix for the k^{th} component; and $\mathbf{v}$ is the latent factor tied across frames and components. The layer output will be the optimum value for $\mathbf{v}$ so we call this layer *tied mixture of factor analyzers* (TMFA) layer. Here $\mathbf{x}_t$ comes from neural network outputs instead of frames, thus we are doing low-dimensional factorization in **hidden space** instead of **feature space**. To get the optimal solution for $\mathbf{v}$, one can treat this problem as a least square problem. The objective is to minimize mean square error:

$$\arg\min_{\mathbf{v}} \sum_{t,c} \gamma_{t,c} s_c \left\|\mathbf{x}_t - \boldsymbol{\mu}_c - \mathbf{T}_c\mathbf{v}\right\|^2 + \alpha \left\|\mathbf{v}\right\|^2 \tag{6}$$

where α is a hyper-parameter used to stabilize the numeric calculation. Another way to consider is to treat it as a normal prior $\mathcal{N}(0, \alpha^{-1}\mathbf{I})$ over $\mathbf{v}$.

Instead of using stochastic gradient descent to find optimal $\mathbf{v}$, because of its linearity property, optimal solution of linear least square can be directly used. Thus, we obtain the familiar i-vector equations, where $\mathbf{v}$ is given by

$$\mathbf{v} = \mathbf{A}^{-1}\mathbf{b} \tag{7}$$

where $\mathbf{A}$ and $\mathbf{b}$ are introduced to simplify equations in following sections,

$$\begin{aligned} \mathbf{A} &= \alpha\mathbf{I} + \sum_{t,c} \gamma_{t,c} s_c \mathbf{T}_c^T \mathbf{T}_c \\ \mathbf{b} &= \sum_{t,c} \gamma_{t,c} s_c \mathbf{T}_c^T (\mathbf{x}_t - \boldsymbol{\mu}_c)\ . \end{aligned} \tag{8}$$

Comparing equation (4) with equation (8), in LDE, all the mixture components have the same importance independently of

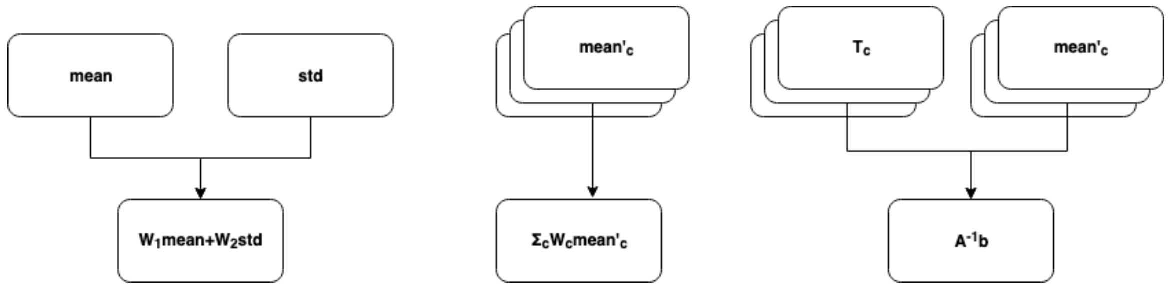

Figure 1: *Left: statistical pooling; middle: Learnable Dictionary Encoding; Right: new pooling method we proposed. Certain linear layer is included for comparison.* $\mathbf{T}$ *and* $\mathbf{W}$ *contains same number of parameters. All pooling methods lead to 400-dimensional output.*

the number of samples observed for each component–since $\mathbf{m}_c^{'}$ is normalized by the counts of component c. Meanwhile, in factor analysis components with more observations have more importance in the calculus of $\mathbf{A}$ and $\mathbf{b}$. Thus, we should expect to obtain the same advantages that the i-vector provides w.r.t. classical MAP, e.g., more robustness to short utterances with unseen Gaussian components.

Note that, the way that the layer parameters are trained are different that in the i-vector approach. In i-vectors, GMM-UBM model, $\mathbf{T}$ and i-vector are optimized to maximize the likelihood of the data, or in auto-encoder jargon, to minimize the reconstruction error. Here, only the embedding $\mathbf{v}$ is calculated to minimize the error between the frame level representation and its corresponding mean. The rest of parameters, e.g., $\boldsymbol{\mu}_c$, s_c, $\mathbf{T}$, are trained discriminatively to minimize the cross-entropy loss.

3.2. Backward Gradient Computation

In back-propagation, we need to calculate derivatives of final loss L with respect to all parameters. Taking $\frac{\partial L}{\partial \mathbf{x}}$ as an example, it can be decomposed into three parts:

$$\frac{\partial L}{\partial \mathbf{x}} = \frac{\partial L}{\partial \mathbf{v}} \frac{\partial \mathbf{v}}{\partial \mathbf{b}} \frac{\partial \mathbf{b}}{\partial \mathbf{x}} \tag{9}$$

The first term is given by back-propagation and the third term is easy to calculate. Here we only show how to calculate the middle term, derivatives of final embedding $\mathbf{v}$ respected to A or b. According to least-square solution, $\mathbf{v}$ is given by

$$\mathbf{v} = \mathbf{A}^{-1}\mathbf{b} \tag{10}$$

$\frac{\partial L}{\partial \mathbf{A}}$ and $\frac{\partial L}{\partial \mathbf{b}}$ are given below:

$$\begin{aligned} \frac{\partial L}{\partial \mathbf{A}} &= -\mathbf{A}^{-1} \frac{\partial L}{\partial \mathbf{v}} \mathbf{v}^T \\ \frac{\partial L}{\partial \mathbf{b}} &= \mathbf{A}^{-1} \frac{\partial L}{\partial \mathbf{v}} \end{aligned} \tag{11}$$

where $\frac{\partial L}{\partial \mathbf{v}}$ is given by back-propagation recursion. More details related to gradient computations can be found in [15]. For $\frac{\partial \mathbf{A}}{\partial \gamma}$, $\frac{\partial \mathbf{A}}{\partial \mathbf{s}}$, $\frac{\partial \mathbf{A}}{\partial \mathbf{T}}$, $\frac{\partial \mathbf{b}}{\partial \gamma}$, $\frac{\partial \mathbf{b}}{\partial \mathbf{s}}$, $\frac{\partial \mathbf{b}}{\partial \mathbf{T}}$, $\frac{\partial \mathbf{b}}{\partial \mathbf{x}}$ and $\frac{\partial \mathbf{b}}{\partial \mu}$, it is relatively easy on mathematics and some of them can be found in original paper of Encoding Layer [12]. Those details are omitted in this paper.

In our case since $\mathbf{A}$ is symmetric, we can utilize Cholesky decomposition to calculate $\mathbf{A}^{-1}\mathbf{b}$ and $\mathbf{A}^{-1}\frac{\partial L}{\partial \mathbf{v}}$. If Cholesky decomposition of $\mathbf{A}$ is $\mathbf{A} = \mathbf{L}\mathbf{L}^T$, then from equation 10 we can get

$$\mathbf{L}(\mathbf{L}^T\mathbf{v}) = \mathbf{b} \tag{12}$$

Since $\mathbf{L}$ is upper triangle matrix, this can be easily solved by using Gaussian elimination algorithm twice. Similarly it can be used to solve equation 11 for backward pass. It is worth to notice that for both forward and backward pass $\mathbf{A}$ needs to be factorized by Cholesky decomposition thus we stored the factorization of $\mathbf{A}$ for both forward and backward, which leads to further speedup.

4. Experiments

4.1. Datasets

We experiment using the setup we used in our participation in NIST SRE16 evaluation [16, 17]. Training set included two parts:

- SWBD: Switchboard 2 Phases 1, 2, and 3 as well as Switchboard Cellular. It includes utterances from 2.6k speakers.
- SRE: 2004-2010 NIST English SRE data from around 4.4k speakers.

For data augmentation, we adopted similar strategy as [11]: babble, music, noise, reverberation are randomly added to original "clean" speech. In total, there were 246, 418 utterances from 6990 speakers.

All frameworks are tested on two different datasets: Cantonese portion of the NIST SRE 2016 evaluation (SRE16) and Speakers in the Wild (SITW) core condition (single speaker). Enrollment utterances contain about 60 seconds of conversational telephone speech while the test utterances varied from 10 - 60 seconds for SRE16. For SITW utterances varies lengths from 6-240 second.

4.2. Network details

All network structures used the same ResNet-34 for feature extraction as [13, 14]. This has been proven to work well for SRE18 [18]. After ResNet-34, a temporal sequence of 128-dimensional vectors are extracted. Pooling layer summarizes those vectors and produces a 400-dimensional embedding, which is fed to final output layer to predict speaker identity. For

Table 1: *Val accuracy for different systems*

System	chunk	Val Acc
ResNet34+SP	2-4s	75.56%
ResNet34+LDE	2-4s	79.88%
ResNet34+TMFA	2-4s	85.96%
ResNet34+SP	5-8s	87.58%
ResNet34+LDE	5-8s	91.17%
ResNet34+TMFA	5-8s	94.12%

Table 2: *Results comparison on SRE16 Cantonese. SP indicates Statistical Pooling*

System	chunk	EER	DCF10^{-3}	DCF10^{-2}
TDNN+SP[11]	2-4s	5.86	0.593	0.410
ResNet34+SP	2-4s	7.05	0.6415	0.4563
ResNet34+LDE	2-4s	5.60	0.5478	0.3637
ResNet34+TMFA	2-4s	6.31	0.5958	0.4273
ResNet34+SP	5-8s	6.44	0.6217	0.4294
ResNet34+LDE	5-8s	4.51	0.5175	0.3442
ResNet34+TMFA	5-8s	6.08	0.6423	0.4382

Table 3: *Results comparison on SITW dev. SP indicates Statistical Pooling. Original x-vector paper[11] didn't report dev scores so it is not included here.*

System	chunk	EER	DCF10^{-3}	DCF10^{-2}
ResNet34+SP	2-4s	6.22	0.8638	0.5716
ResNet34+LDE	2-4s	5.65	0.7542	0.4749
ResNet34+TMFA	2-4s	6.05	0.7968	0.5196
ResNet34+SP	5-8s	6.39	0.8795	0.5338
ResNet34+LDE	5-8s	4.57	0.7341	0.4338
ResNet34+TMFA	5-8s	5.00	0.7280	0.4475

LDE and factor analysis pooling, 32 clusters were used. As shown in Figure 1, we compared these three pooling methods based on similar network architecture. All systems are trained with 100 iterations.

Filter bank features were used for all systems and feature dimension is 23. We applied energy voice activity detection (VAD) for all. During training for each mini-batch, we randomly sampled an utterance length between either 2-4 second or 5-8 second. We analysis influence of train duration for different pooling methods and report all results for all combinations. Validation accuracy for different systems are reported on Table 1. Clearly proposed TMFA layer produced the best validation accuracy.

For evaluation, following previous work [11, 13], outputs before activation function from the first layer after pooling layer were extracted as utterance-level embedding.

4.3. Back-end details

We used the same back-end for all systems following Kaldi [19] scripts. For SRE16 Cantonese, NIST SRE16 development set "major" data were used to center all embeddings for enrollment and test data. For Speaker In the Wild we used development data to center embeddings. Linear discriminant analysis was utilized to reduce embedding dimension from 400 to 150 for all baseline systems. For the embeddings based on FA we used 100 instead. Probabilistic linear discriminant analysis with score normalization is adopted for verification based on length-normalized embeddings. No special adaptation method is used.

5. Results comparison

Results for SRE16 Cantonese, SITW core development and evaluation are reported in Tables 2, 3 and 4, respectively. First of all, we notice that, when ResNet is trained with longer chunks, overall performance is better. For mean+stddev pooling SP the difference between long and short chunks was small. However, for LDE and TMFA, there was a significant improvement when increasing chunk size. We speculate that this is because using longer chunks, the training observes more mixture components per chunk and mini-batch, and this allows to train better the parameters of the LDE and TMFA layers. Secondly, comparing with previous published x-vector work with similar trained setup, TDNN x-vector with SP outperformed w.r.t. ResNet+SP. This can be reasonable since TDNN+SP used larger network (4.2 million parameters in x-vector [11] compared to 2.98 million parameters for ResNet) with much larger embedding dimension. However, both results are not completely comparable since we used single GPU Pytorch implementation with Adam optimizer, while [11] used Kaldi implementation with natural gradient, multi-GPU and model averaging. When we used the same feature extractor network, ResNet34, it is clear that multi-modal based learnable dictionary encoding outperforms uni-modal based statistical pooling. This suggests that outcome from ResNet is multi-modal instead of uni-modal.

Table 4: *Results comparison on SITW eval. SP indicates Statistical Pooling*

System	chunk	EER	DCF10^{-3}	DCF10^{-2}
TDNN+SP[11]	2-4s	6.00	0.677	0.488
ResNet34+SP	2-4s	6.68	0.7673	0.5729
ResNet34+LDE	2-4s	5.74	0.6992	0.5182
ResNet34+TMFA	2-4s	6.34	0.7797	0.5748
ResNet34+SP	5-8s	6.13	0.7333	0.5564
ResNet34+LDE	5-8s	5.29	0.6730	0.4701
ResNet34+TMFA	5-8s	5.74	0.7151	0.5131

In general, LDE layer outperformed the new proposed TMFA layer, even though the latter provided better validation accuracy. In the past, we have observed that validation accuracy is correlated with final EER and cost function. However, it is not the case for TMFA, a matter that needs further investigation. It is possible that stochastic gradient descent is not able to cope with the extra complexity of the TMFA layer, which involve gradients over matrix inversions. Or in certain cases matrix $\mathbf{A}$ may becomes rank deficient so computation of both forward pass and backward pass becomes unstable.

6. Conclusions

In this paper a novel pooling method is proposed. It combines classical *i-vector* idea with recently proposed learnable dictionary encoding pooling. Instead of adapting observed components, low-dimensional factorization is utilized to estimate a low-dimensional embedding. Thus even without observing most components, this embedding has potential to be accurate. We tested on Speaker Recognition Evaluation 16 Cantonese part as well as Speaker in the Wild dataset. Experiments show that pooling method we proposed lead to comparable performance with LDE and in most cases it outperforms statistical pooling.

For future work we are planning to study differences between our pooling method and LDE. Also comparing accuracy with final results, clearly accuracy is not highly related to final performance which indicates system which high accuracy may not generalize well for different data. This suggests the importance of finding new metric instead of validation accuracy for system selection.

7. References

[1] D. A. Reynolds, "Speaker identification and verification using gaussian mixture speaker models," *Speech communication*, vol. 17, no. 1, pp. 91–108, 1995.

[2] 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.

[3] 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.

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

[5] 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.

[6] 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)*. IEEE, 2014, pp. 4052–4056.

[7] Y. Liu, Y. Qian, N. Chen, T. Fu, Y. Zhang, and K. Yu, "Deep feature for text-dependent speaker verification," *Speech Communication*, vol. 73, pp. 1–13, 2015.

[8] S.-X. Zhang, Z. Chen, Y. Zhao, J. Li, and Y. Gong, "End-to-end attention based text-dependent speaker verification," in *2016 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2016, pp. 171–178.

[9] F. Chowdhury, Q. Wang, I. L. Moreno, and L. Wan, "Attention-based models for text-dependent speaker verification," *arXiv preprint arXiv:1710.10470*, 2017.

[10] D. Snyder, D. Garcia-Romero, D. Povey, and S. Khudanpur, "Deep neural network embeddings for text-independent speaker verification," *Proc. Interspeech*, pp. 999–1003, 2017.

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

[12] H. Zhang, J. Xue, and K. Dana, "Deep ten: Texture encoding network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 708–717.

[13] 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.

[14] W. Cai, Z. Cai, X. Zhang, X. Wang, and M. Li, "A novel learnable dictionary encoding layer for end-to-end language identification," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5189–5193.

[15] M. Giles, "An extended collection of matrix derivative results for forward and reverse mode automatic differentiation," 2008.

[16] P. A. Torres-Carrasquillo, F. Richardson, S. Nercessian, D. Sturim, W. Campbell, Y. Gwon, S. Vattam, R. Dehak, H. Mallidi, P. S. Nidadavolu, R. Li, R. R. Pappagari, N. Chen, N. Dehak, and R. Zazo, "The mit lincoln laboratory 2016 speaker recognition system," in *NIST Speaker Recognition Evaluation 2016*, San Diego, California, Dec. 2016.

[17] P. A. Torres-Carrasquillo, F. Richardson, S. Nercessian, D. Sturim, W. Campbell, Y. Gwon, S. Vattam, N. Dehak, H. Mallidi, P. S. Nidadavolu *et al.*, "The mit-ll, jhu and lrde nist 2016 speaker recognition evaluation system," *Proc. Interspeech 2017*, pp. 1333–1337, 2017.

[18] J. Villalba, N. Chen, D. Snyder, D. Garcia-romero, A. Mccree, G. Sell, J. Borgstrom, F. Richardson, S. Shon, G. Francois, L. P. García, P. A. Torres-carrasquillo, and N. Dehak, "The JHU-MIT System Description for NIST SRE18," Johns Hopkins University, Baltimore, MD, Tech. Rep., 2018.

[19] 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," IEEE Signal Processing Society, Tech. Rep., 2011.