Odyssey 2020 The Speaker and Language Recognition Workshop
1-5 November 2020, Tokyo, Japan

COMBINED VECTOR BASED ON FACTORIZED TIME-DELAY NEURAL NETWORK FOR TEXT-INDEPENDENT SPEAKER RECOGNITION

Tianyu Liang, Yi Liu, Can Xu, Xianwei Zhang, Liang He

Department of Electronic Engineering, and Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing 100084, China

ty_liang@foxmail.com, liu-yi15@mails.tsinghua.edu.cn, {ducanxu, zhangxw019}@163.com, heliang@mail.tsinghua.edu.cn

ABSTRACT

Currently, the most effective text-independent speaker recognition method has turned to be extracting speaker embedding from various deep neural networks. Among them, the x-vector extracted from factorized time-delay neural network (F-TDNN) has been demonstrated to be among the best performance on recent NIST SRE evaluations. In our previous works, we have proposed combined vector (c-vector) and proved that the performance can be further improved by introducing phonetic information, which is often ignored in extracting x-vectors. By taking advantages of both F-TDNN and c-vector, we propose an embedding extraction method termed as factorized combined vector (fc-vector). In the NIST SRE18 CTS task, the EER and minDCF18 of fc-vector are 12.1% and 10.5% relatively lower than the x-vector, and 3.4% and 3.9% relatively lower than the c-vector, respectively.

Index Terms: text-independent speaker recognition, x-vector, factorized time-delay neural network, c-vector

1. INTRODUCTION

In the past few years, the state-of-the-art text-independent speaker recognition method has gradually changed from i-vector framework [1] to speaker embedding [2]. The speaker embedding extraction based on deep neural network extracts fixed-length vectors from the speaker's speech as embeddings, and it performs better than the i-vector method in most experiments. In terms of extracting speaker-discriminant vectors, some end-to-end systems based on different network architectures [3, 4, 5, 6, 7] have also demonstrated their effectiveness. Recently, the x-vector architecture [8] and its variants [9, 10] based on the TDNN network [11] have achieved good results in NIST SRE18 speaker recognition evaluation [12]. Among them, the factorized TDNN is one of the best performing speaker recognition systems.

Speech signal consists of content, speaker, gender, emotion, channel and noise information, etc., among which the speech content is one of the main information. Generally, different verification tasks focus on the different target information alone, and ignore the influence of other information. However, the fact is that different speech components share some common information and cannot be completely separated. Based on this idea, some multi-task learning methods have been proposed [13,14,15]. In conventional multi-task learning, hidden layers are shared between different task networks, and different tasks are predicted simultaneously. In [16], the author proposed a multi-task learning network that shared frame-level information. In [17], the level of shared information was changed from frame-level to segment-level based on [16]. Ref. [18], based on [16], proposed a c-vector architecture to additionally integrate phoneme information into the x-vector framework and greatly improved the recognition performance. The disadvantage of c-vector is that it only uses a simple TDNN network.

In this paper, we introduce factorized layers [11] into c-vector and propose an extended network called fc-vector. The experimental results on SRE18 CTS task [19, 20] show that the extended multi-task learning network has better performance.

The organization of this paper is as follows. In the section 2, the speaker embedding extraction method, the method of factorization, and several multi-task learning methods that will be used in this paper are introduced. The section 3 describes our proposed extended multi-task learning network. The experimental setup is given in the section 4. The experimental results are shown in the section 5. Finally, we conclude the paper in the section 6.

2. SPEAKER EMBEDDING AND MULTI-TASK LEARNING

2.1. The factorized TDNN

Speaker embedding [8] is the main speaker recognition method at this stage. Its frame-work is shown in Fig. 1. The input layer is the frame-level acoustic features of the speech. It first passes through several layers of time-delay architecture. Frame-level information is then aggregated in the statistics pooling layer, the mean and standard deviation are calculated, and converted into segment-level information. The statistics pooling layer is followed by two hidden layers and a softmax layer. Finally, the predicted posterior probability is output.

After model training, the output of the hidden layer after the statistics pooling layer will be extracted as speaker embedding. The linear discriminant analysis (LDA) and probabilistic linear discriminant analysis (PLDA) are applied to calculate the score. Factorized TDNN (F-TDNN) [10, 12] is a factored form of TDNNs [11], which is compressed by factorizing the weight matrix between TDNN layers. This method uses less network parameters while maintaining the efficiency and stability of network training, and obtains good results. The weight matrix is factorized into the product of two factor matrices:

$$M = AB \tag{1}$$

Where M is the original weight matrix and matrix B is cons-

10.21437/Odyssey.2020-61

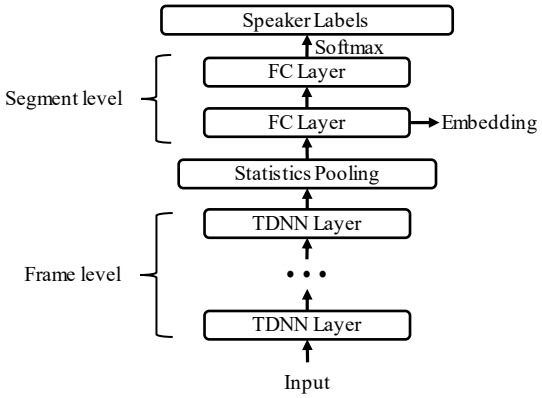

Figure 1. *The x-vector architecture.*

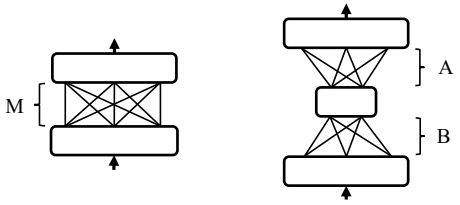

Figure 2. *Differences between standard and factorized layers.*

trained to be semi-orthogonal matrix. The specific differences of this formula in the network are shown in Fig. 2. Ref. [12] proposed an F-TDNN architecture with 14 layers, and its effectiveness has been confirmed in the NIST SRE18 evaluation.

2.2. The c-vector

Although X-vector network performs speaker prediction at the segment level, phonetic information is ignored in this process. Unlike the x-vector network, the output of an ASR network are phonetic labels, and ASR network always keeps its information at the frame level.

Phonetic adaptation method [16] integrates phonetic information into x-vector network. First pre-train an ASR model with a bottleneck layer, and then merge the output of the bottleneck layer as an auxiliary vector into x-vector network.

Different from the above method, hybrid multi-task learning [16] combines x-vector network with ASR network, so that two networks share a part of the frame-level layers. The training process is alternated by the two parts of the combined network. The speaker embedding part of the combined network can learn more about the common information shared by speaker features and phonetic contents, and the recognition effect performs better.

Phonetic adaptation and hybrid multi-task learning focus on two aspects of speech information, the former trying to suppress the negative effect of phonetic content, and the latter trying to learn more effective information from phonetic content. The c-vector network [18] combines these two methods in an attempt to more effectively learn the shared parts of phonetic information and speaker information. The architecture of the c-vector network is shown in Fig. 3 [18].

Similar to the phonetic adaptation method, an ASR network is pre-trained first, features are extracted from its bottleneck layer and merged with the speaker embedding part of hybrid multi-task learning. During hybrid multi-task learning network training, the pre-trained ASR network is no longer updated. After that, the two parts of the hybrid multi-task learning network are alternately trained, and the embedding is extracted from the hidden layer behind the pooling layer of the speaker embedding part.

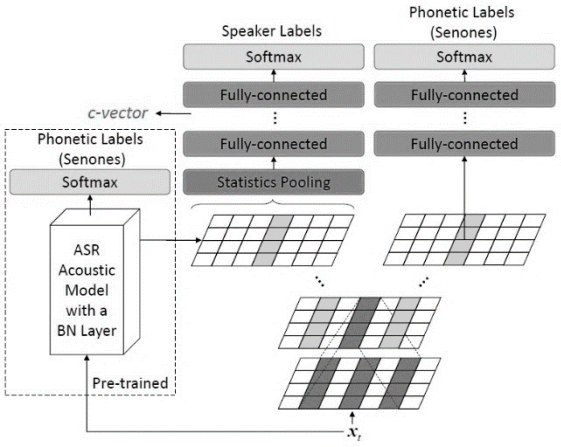

Figure 3. *The c-vector architecture.*

3. EXTENDED C-VECTOR WITH FACTORIZED LAYERS

Many experiments have shown that deeper network architectures improve the whole performance, but this comes with too many parameters and too long training cycles. We tried a new extended-factorized TDNN network architecture (EF-TDNN), which is extended from the F-TDNN system [12], as shown in Table 1. We used this architecture in the NIST SRE19 evaluation. While greatly deepening the network architecture, the network parameter amount is controlled to a certain range, and the performance was significantly improved. In view of the good performance of EF-TDNN network and the effect of phonetic information on speaker recognition system, we introduce EF-TDNN to c-vector, which is called fc-vector.

The way we introduce the factorized layers is a little different from the F-TDNN network in [12]. Skip connection is a network connection method [10], which connects the bottleneck output in the factorization layer to the subsequent layers as part of the input. We use the jump connection method within the layer instead, that is, the input of the factorized layer is multiplied by a coefficient and the output of the factorized layer is added as the new output, similar to the ResNet method [21]. The source code of the Kaldi system has also been updated accordingly.

We use EF-TDNN to replace the part used to extract the embedding in the c-vector network. The replaced part accepts the phonetic bottleneck feature at layer 20 and extracts the embeddings at layer 22. At the same time, we use a simplified EF-TDNN network without pooling layer to replace the ASR part of the multi-task learning in c-vector. The first few layers of the two parts remain the same for sharing. The network architecture of the fc-vector is shown in the Fig. 4.

Table 1. *The EF-TDNN architecture*

Layer	Layer Type	Context factor1	Context factor2	Size	Inner size
1	TDNN-ReLU	t-2: t+2		512	
2	TDNN-ReLU	t		1024	
3	F-TDNN-ReLU	t-2, t	t, t+2	1024	256
4	TDNN-ReLU	t		1024	
5	F-TDNN-ReLU	t	t	1024	256
6	TDNN-ReLU	t		1024	
7	F-TDNN-ReLU	t-3, t	t, t+3	1024	256
8	TDNN-ReLU	t		1024	
9	F-TDNN-ReLU	t	t	1024	256
10	TDNN-ReLU	t		1024	
11	F-TDNN-ReLU	t-3, t	t, t+3	1024	256
12	TDNN-ReLU	t		1024	
13	F-TDNN-ReLU	t-3, t	t, t+3	1024	256
14	TDNN-ReLU	t		1024	
15	F-TDNN-ReLU	t-3, t	t, t+3	1024	256
16	TDNN-ReLU	t		1024	
17	F-TDNN-ReLU	t	t	1024	256
18	TDNN-ReLU	t		2048	
19	TDNN-ReLU	t		2048	
20	TDNN-ReLU	t		2048	
21	Pooling (mean+stddev)	full-seq		2×2048	
22	FC-ReLU			1024	
23	FC-ReLU			1024	
24	Softmax			Num. speakers.	

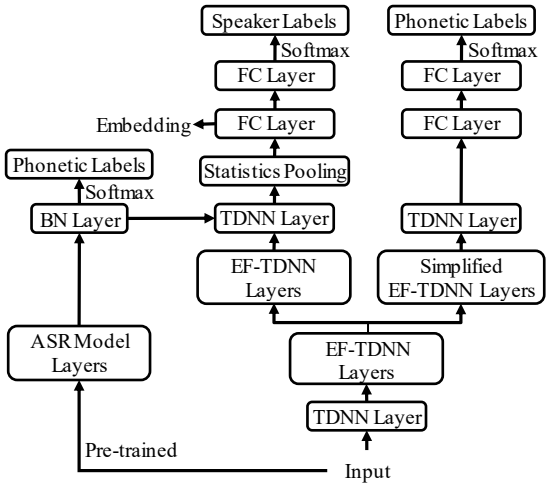

Figure 4. *The fc-vector architecture.*

4. EXPERIMENTAL SETUP

4.1. Dataset

All our experiments are implemented using the Kaldi toolkit [22]. The experiment is performed according to the requirements of the fixed training condition of NIST SRE18 [19, 20]. It should be noted that our training dataset does not include the VoxCeleb and SITW data sets. The Fisher dataset contains phoneme labels, so we use it to train additional ASR parts, and the remaining datasets are used to train neural networks and backends, as shown in Table 2. For each system, MUSAN and RIRS NOISES are used as noise sources to enhance the training data, and the amount of training data is doubled. The test set is the DEV and EVAL datasets of NIST SRE18 CTS task. Details of the test set are shown in Table 3. We use equal error rate (EER) and minimum detection cost function from NIST SRE18 (minDCF18) [19] as the experimental performance evaluation index.

Table 2. *Datasets used for training*

Project	Data set
Phonetic training set	Switchboard-1, Fisher English
Speaker training set	Switchboard-2 Phase 1/2/3, Switchboard Cellular Part 1/2, NIST SRE 2004-2010, 2016

Table 3. *Information of test sets*

Test set		Sample rate	Num. Speakers	Num. Target trials	Num. Impostor trials
SRE 18	CMN2 Dev	8kHz	125	7830	100265
	CMN2 Eval		940	60991	2033832

4.2. Baseline x-vector system and EF-TDNN system

We use the same x-vector network architecture as [8], as shown in Table 4. The input features of network are 23-dim MFCCs. We extract embeddings from layer 7 of the network. We also use the EF-TDNN proposed in Section 3 as a separate network to extract embeddings.

Table 4. *The x -vector network architecture*

Layer	Layer Type	Context	Size
1	TDNN-ReLU	t-2: t+2	512
2	TDNN-ReLU	t-2, t, t+2	512
3	TDNN-ReLU	t-3, t, t+3	512
4	TDNN-ReLU	t	512
5	TDNN-ReLU	t	1500
6	Pooling (mean+stddev)	full-seq	2×1500
7	FC-ReLU		512
8	FC-ReLU		512
9	Softmax		Num. speakers.

4.3. The c-vector system

We use the network architecture described in [18]. The senone transcriptions forced aligned by GMM-HMM are used as phonetic labels. The pre-trained ASR network architecture is shown in Table 5. In hybrid multi-task learning, the architecture of the speaker embedding part is the same as the baseline x-vector system, and the architecture of the ASR part has two differences: the size of the layer 5 is changed to 512 and the layer 6 (pooling layer) is deleted. We only share the first layer of the network.

Since the ASR network is trained using English datasets such as Switchboard and Fisher, it does not fit well with the language of the SRE18 dataset. Based on the experimental results of [5], we set the learning rate scaling factor to 0.2 to reduce this mismatch.

Table 5. *The ASR network for phonetic adaptation*

Layer	Layer Type	Context	Size
1	TDNN-ReLU	t-2: t+2	650
2	TDNN-ReLU	t-1: t+1	650
3	TDNN-ReLU	t-1: t+1	650
4	TDNN-ReLU	t-3, t, t+3	650
5	TDNN-ReLU	t-6, t-3, t	128
6	Softmax		Num. Senones.

4.4. The fc-vector system

The architecture of the pre-trained ASR model is the same as that of the c-vector system. The speaker embedding part of hybrid multi-task learning architecture with factorized layers is shown in Table 1. The first 8 layers of the ASR part of hybrid multi-task learning architecture are the same as the speaker embedding part, followed by two TDNN layers of size 1024, and a softmax layer. We share the first 1, 3, and 5 layers for experiments.

4.5. Back-end processing

All the above experiments are followed by the same back-end processing. After the embeddings are extracted, 200-dimension LDA and PLDA scoring are trained and applied. Due to domain mismatch, a common optimization method is to use sre18-unlabel data for unsupervised PLDA adaptation. But we use another method to get better results in the experiment, which is to perform unsupervised clustering on sre18-unlabel data, and use the clustered data to train PLDA. Then the PLDA is used to score.

5. RESULTS

Table 6 shows the results of the dev and eval sets of different systems in the SRE18 CTS task. The result of EF-TDNN is significantly better than x-vector, and it is also better than c-vector in all indicators (except EER on SRE18 eval set). The experimental results show that fc-vector is more effective.

Next, it is noted that the fc-vector model achieves better results than the c-vector model and EF-TDNN model. The fc-vector system that shared the first layer showed the best performance. Compared with the baseline x-vector system, its EER and minDCF18 on the SRE18EVAL dataset are improved by 12.1% and 10.5%, respectively. Compared with the c-vector system, it also improved by 3.4% and 3.9%, respectively.

It should be noted that in comparison experiments with different numbers of shared layers, although the fc-vector with 5-layers sharing performs best on the EER of the SRE18 DEV dataset, the overall effect of the fc-vector system decreases as the number of shared layers increases. We believe that this is partly due to language mismatch. The training data of the ASR part is spoken in English, while the test data set is spoken in Tunisian Arabic. But through the results we can see that even in this case, the extracted phonetic information can still help improve the effect of speaker recognition.

Table 6. *Results on NIST SRE18 CTS task*

System	SRE18 Dev		SRE18 Eval	
	EER(%)	minDCF18	EER(%)	minDCF18
x-vector	7.43	0.484	7.80	0.550
EF-TDNN	6.35	0.452	7.09	0.500
c-vector (1-layer sharing)	6.86	0.489	7.09	0.512
fc-vector (1-layer sharing)	6.34	**0.418**	**6.85**	**0.492**
fc-vector (3-layer sharing)	6.59	0.433	7.07	0.509
fc-vector (5-layer sharing)	**6.26**	0.467	7.10	0.513

6. CONCLUSION

In this paper, we proposed a fc-vector based on F-TDNN and c-vector. By introducing factorized layers, the number of parameters is relatively reduced and the stability of whole network is guaranteed. By introducing phonetic information, the implicit phonetic alignment boosts the text-independent speaker recognition method. The experimental results on the NIST SRE18 CTS task show that the fc-vector has better results than the x-vector and the c-vector. Meanwhile, our results also show that the method can still obtain good results even in the case of language mismatch.

In our future work, we will train acoustic model by using multilingual methods and data to reduce the impact of language mismatch, and consider adding an auto encoder to optimize the network.

7. 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. D. Snyder, D. Garcia-Romero, D. Povey, and S. Khudanpur, "Deep Neural Network Embeddings for Text-Independent Speaker Verification," in Proceedings of the 18th Annual Conference of the International Speech

Communication Association, INTERSPEECH 2017, Stockholm, Sweden, Aug 2017, pp. 999-1003, ISCA.

3. S.-X. Zhang, Z. Chen, Y. Zhao, J. Li, and Y. Gong, "End-to-end attention based text-dependent speaker verification," in Proc. IEEE Spoken Language Technology Workshop (SLT), 2016, pp. 171–178.
4. D. Snyder, P. Ghahremani, D. Povey, D. Garcia-Romero, Y. Carmiel, and S. Khudanpur, "Deep neural network-based speaker embeddings for end-to-end speaker verification," in Proc. IEEE Spoken Language Technology Workshop (SLT), 2016, pp. 165–170.
5. G. Heigold, I. Moreno, S. Bengio, and N. Shazeer, "End-to-end text-dependent speaker verification," in Proc. IEEE ICASSP, March 2016, pp. 5115–5119.
6. C. Zhang and K. Koishida, "End-to-end text-independent speaker verification with triplet loss on short utterances," in Proc. INTERSPEECH, 2017, pp. 1487–1491.
7. C. Li, X. Ma, B. Jiang, X. Li, X. Zhang, X. Liu, Y. Cao, A. Kannan, and Z. Zhu, "Deep speaker: An end-to-end neural speaker embedding system," arXiv preprint arXiv:1705.02304, 2017.
8. D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-Vectors : Robust DNN Embeddings for Speaker Recognition," in Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018, Alberta, Canada, Apr 2018, pp. 5329–5333, IEEE.
9. D. Snyder, D. Garcia-Romero, G. Sell, A. McCree, D. Povey, and S. Khudanpur, "Speaker Recognition for Multi-Speaker Conversations Using X-Vectors," in Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019. Brighton, UK: IEEE, May 2019.
10. D. Povey, G. Cheng, Y. Wang, K. Li, H. Xu, M. Yarmohamadi, and S. Khudanpur, "Semi-Orthogonal Low-Rank Matrix Factorization for Deep Neural Networks," in Proceedings of the 19th Annual Conference of the International Speech Communication Association, INTERSPEECH 2018, Hyderabad, India, Sep 2018.
11. V. Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in Proc. INTERSPEECH, 2015, pp. 3214-3218.
12. J. Villalba1, N. Chen, D. Snyder, D. Garcia-Romero, A. McCree, G. Sell, J. Borgstrom, F. Richardson, S. Shon, F. Grondin, R. Dehak, L. P. García-Perera, D. Povey, P. A. Torres-Carrasquillo, S. Khudanpur, and N. Dehak, "State-of-the-art Speaker Recognition for Telephone and Video Speech: the JHU-MIT Submission for NIST SRE18," in Proc. INTERSPEECH, 2019, pp. 1488-1492
13. N. Chen, Y. Qian, and K. Yu, "Multi-task learning for text-dependent speaker verification," in Proc. INTERSPEECH, 2015, pp. 185-189.
14. G. Pironkov, S. Dupont, and T. Dutoit, "Speaker-aware long shortterm memory multi-task learning for speech recognition," in Proc. IEEE Signal Processing Conference (EUSIPCO), 2016, pp. 1911–1915.
15. Z. Tang, L. Li, D. Wang, and R. Vipperla, "Collaborative joint training with multitask recurrent model for speech and speaker recognition," IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 25, no. 3, pp. 493-504, 2017.
16. Y. Liu, L. He, J. Liu, and M. T. Johnson, "Speaker embedding extraction with phonetic information," in Proc. INTERSPEECH, 2018, pp. 2247-2251
17. S. Wang, J. Rohdin, L. Burget, O. Plchot, Y. Qian, K. Yu, J. H. Černocký, "On the Usage of Phonetic Information for Text-independent Speaker Embedding Extraction," in Proc. INTERSPEECH, 2019, pp. 1148-1152
18. Y. Liu, L. He, J. Liu, and M. T. Johnson, "Introducing phonetic information to speaker embedding for speaker varication," EURASIP Journal on Audio, Speech, and Music Processing, 2019, 19(2019), doi:10.1186/s13636-019-0166-8
19. NIST 2018 Speaker Recognition Evaluation Plan. https://www.nist.gov/sites/default/files/documents/2018/08/17/sre18_eval_plan_2018-05-31_v6.pdf
20. S. O. Sadjadi, C. Greenberg, E. Singer, D. Reynolds, L. Mason, and J. Hernandez-Cordero, "The 2018 NIST Speaker Recognition Evaluation," in Proc. INTERSPEECH, 2019, pp. 1483-1487
21. K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," dec 2015.
22. D. Povey, A. Ghoshal, G. Boulianne, L. Burget, Ondřej Glembek, N. Goel, M. Hannemann, P. Motlíček, Y. Qian, P. Schwarz, J. Silovský, G. Stemmer, and K. Veselý, "The kaldi speech recognition toolkit," in Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), 2011.