SPEAKER ADAPTATION OF DNN ACOUSTIC MODELS USING X-VECTORS

Qian Liu, Wei-Qiang Zhang, Jia Liu

Beijing National Research Center for Information Science and Technology
Department of Electronic Engineering, Tsinghua University, Beijing 100084, China
q-liu18@mails.tsinghua.edu.cn, {wqzhang,liuj}@tsinghua.edu.cn

ABSTRACT

Speaker adaptation is an important technique for acoustic modeling in automatic speech recognition (ASR). In this paper, we propose to apply x-vectors for speaker adaptation of Deep Neural Network (DNN) acoustic models. Some methods and strategies are applied to improve x-vectors for speaker adaptation task, including data selection and augmentation, speaker split-up and a different extraction setup. The results of our experiments show that x-vectors can achieve a bit better performance than the state-of-the-art baseline with i-vector speaker adaptation. Moreover, the x-vectors and i-vectors are complementary in speaker adaptation and better performance can be achieved when fusion strategy is applied.

Index Terms— x-vector, i-vector, acoustic modeling, speaker adaptation, automatic speech recognition

1. INTRODUCTION

Speaker adaptation is a long-standing technique to improve the performance of acoustic models. The aim of speaker adaptation is to capture important information of the speakers and even the acoustic environments to help improve the performance of ASR systems.

In the early works, the approaches of speaker adaptation are mainly based on transforms, such as Maximum Likelihood Linear Regression (MLLR) [1] and Constrained MLLR (CMLLR) [2]. These approaches utilize an affine transform or feature-space transform to adapt model parameters, and perform well on GMM-HMM acoustic models. Vocal Tract Length Normalization (VTLN) is an approach for speaker normalization in ASR to compensate for speaker variability [3]. With speaker adaptation, the acoustic models are able to generalize better to unseen speakers.

Deep Neural Networks (DNN) have become the dominant method for acoustic modeling in the state-of-the-art research. The traditional speaker adaptation features such as MLLR and CMLLR are usually intended for GMM-HMM acoustic models, and they are not suitable for neural network structure [4]. In [5], researchers use some adaptation data of a given speaker to adapt the parameters of DNN models. However, it leads to overfitting because the number of parameters are too much and the adaptation data is usually insufficient. Another approach is to train DNN acoustic models directly on speaker adaptation features. The features can be obtained based on MLLR transforms [6] or speaker normalization like VTLN [7].

On account of the structure of neural networks, researchers figure out a way to input the speaker information of the utterances into the neural networks. The idea is borrowed from speaker recognition and the speakers are encoded into feature vectors. I-vectors are commonly used speaker embeddings in speaker recognition [8] and have been successfully used in the adaptation of DNN acoustic models [4][9]. I-vectors are fixed length vectors extracted from the feature sequences, capturing speaker information for adaptation of DNN acoustic models. I-vector based adaptation has been verified to be effective in reverberant environments [10].

Some approaches are shared across speaker adaptation in ASR and speaker recognition. On the one hand, transforms for speaker adaptation such as CMLLR are applied in speaker recognition [11]. On the other hand, speaker embeddings such as i-vectors originally used for speaker recognition perform well in speaker adaptation for DNN acoustic models [4][9][12]. Inspired by this, we focus on another kind of speaker embeddings called x-vectors. X-vector is a DNN-based speaker embedding proposed in [13], and has obtained the state-of-the-art performance on text-independent speaker recognition [14]. In [15], x-vectors are applied for speaker adaptive training of DNN acoustic models, and promising results are achieved.

In this paper, we propose a new adaptation method for DNN acoustic models using x-vectors. With some specialized strategies and modifications, we show that adaptation with x-vectors can help improve the performance of DNN acoustic models, achieving better results than i-vector based adaptation on the test data.

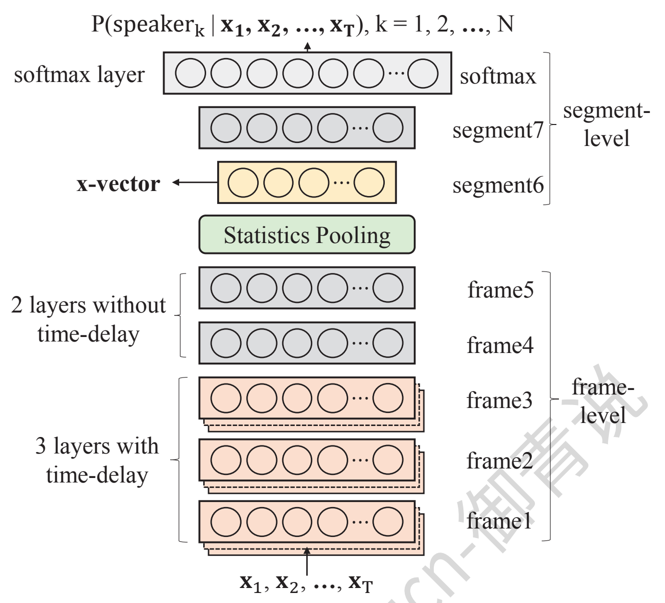
The remainder of the paper is organized as follows. In section 2, we review the theory of DNN based x-vector embeddings, and introduce the DNN architecture for speaker embedding. In section 3, we introduce our data set and the state-of-the-art baseline ASR system with i-vectors for speaker adaptation, and elaborate the experiment setup of using x-vectors. The experiments results are shown in section 4, as well as some discussions on optimizing the x-vectors for speaker adaptation task. The conclusions are presented in section 5.

2. X-VECTOR EMBEDDINGS

X-vectors are segment-level speaker embeddings computed from variable-length speech segments [13]. It proves to be effective to capture speaker information over the entire utterance. The DNN architecture for extracting x-vectors is based a feed-forward DNN shown in Fig. 1. The DNN is trained to predict speakers from variable-length speech segments. The detailed DNN architecture is interpreted as follows.

- *input*: the features of each frame.
- *five frame-level layers*: the first three layers work with a time-delay architecture [16], while the rest two layers do not.
- *a statistics-pooling layer*: it aggregates over the output of the final frame-level layer and computes the mean and standard deviation. The segment-level statistics are then concatenated together and passed to two hidden layers.

This work was supported by the National Natural Science Foundation of China under Grant No. U1836219 and the National Key R&D Program of China.



P(speaker$_k$ | $\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_T}$), k = 1, 2, …, N
softmax layer
softmax
segment7
segment-level
x-vector
segment6
Statistics Pooling
2 layers without time-delay
frame5
frame4
3 layers with time-delay
frame3
frame-level
frame2
frame1
$\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T$

Fig. 1. Structure of the Deep Neural Network for x-vector extraction. All the nonlinearities in the model are rectified linear units (ReLUs).

- *two hidden layers*: the speaker embeddings can be extracted from these two affine layers on top of the statistics.
- *final output layer*: a softmax layer with multi-class cross-entropy objective function. The layer has N output where N is the number of speakers to be classified.

Prior studies have found that x-vectors leverage large-scale training data better than i-vectors, and x-vectors benefit more from data augmentation because of supervised learning [13][14]. Recently, researchers have investigated the information encoded in x-vectors compared to i-vectors through several probing tasks [17]. The results show that in addition to speaker-related information such as speaker and gender, x-vectors can also capture information of utterance length, transcription and channel characteristics. Given the rich information encoded in x-vectors, it is reasonable that we substitute the i-vectors with x-vectors for adaptation of DNN acoustic models.

Layer	Layer Context	Total Context	Input×Output
frame1	[t-2,t+2]	5	200×512
frame2	{t-2,t,t+2}	9	1536×512
frame3	{t-3,t,t+3}	15	1536×512
frame4	{t}	15	512×512
frame5	{t}	15	512×1500
stats pooling	[0,T)	T	1500T×3000
segment6	{0}	T	3000×d
segment7	{0}	T	d×512
softmax	{0}	T	512×N

Table 1. The DNN architecture for x-vectors. In this paper, T, d, N denote the number of utterance frames, the dimension of extracted x-vectors and the number of training speakers respectively.

In this paper, the x-vectors for speaker adaptation are mostly the same with those used for speaker recognition in [14]. The parameter configuration of the DNN architecture is shown in Table 1.

In the first 3 frame-level layers, each layer is built on the temporal context of the earlier layers. The outputs of the statistics pooling layer are 1500 dimensional vectors, and are computed once for each speech segment. Each training example consists of a chunk of speech features and the corresponding label of the speaker. X-vectors are extracted from the first hidden layer on top of statistics pooling layer for speaker recognition in previous work [13]. In our work, we follow the same method and x-vectors are extracted at layer *segment6* before the ReLU nonlinearity.

3. EXPERIMENT SETUP

3.1. Data set

In this paper, we use TED-LIUM release 3 corpus as our data set [18]. It includes 540 hours of audio with transcripts (including noises and silences), which are extracted from 2351 TED talks of 2028 unique speakers. The pronunciation lexicon is also open-sourced in the data set. We use the legacy version of the corpus in order to compare with the results in [18]. Table 2 summarizes the details of the audio and transcripts of the data set.

Data set	Train	Dev	Test
Duration of audio	540.2h	1.7h	3.1h
Number of speakers	2028	8	11
Number of segments	268263	507	1155
Number of words	4.78M	17.8k	27.5k

Table 2. Detail information of TED-LIUM release 3 data set.

3.2. Baseline system

This paper focuses on speaker adaptation of DNN acoustic models, so other components of the ASR system are invariant. We use i-vectors for speaker adaptation as our baseline, achieving the state-of-the-art results [18]. All experiments are conducted using the nnet3 library in the Kaldi Speech Recognition Toolkit [19]. X-vectors are used to replace i-vectors with some refinements at first, and are combined with i-vectors to obtain the best results in our work.

Speech data augmentation has shown great performance for DNN based speaker embeddings [14], because it can increase the amount and diversity of the training data. In this paper, we adopt two low-cost methods for augmentation. The first method is volume perturbation, which help the acoustic models more invariant to test speech volume. The other method is speed perturbation. While transcripts remain invariant, speech audio with different speed can be obtained by time warping [20]. In our work, each audio is perturbed with factors of 0.9, 1.0 and 1.1 in order to get triple training data.

The acoustic features of our ASR system are commonly used Mel-frequency cepstral coefficients (MFCCs). After augmentation of training data, a 40-dimensional MFCC feature is computed at each frame. A GMM-HMM acoustic model is used to generate alignments for training neural networks. We use an 11-layer Factorized Time Delay Neural Network (TDNN-F) as the acoustic model, which achieves the state-of-the-art performance on the data set [18].

The i-vector extraction process for training and decoding is similar to [10]. We do a first-pass decoding and re-segment training data using a GMM-HMM based ASR system. Only the reliable speech segments are kept for further training. About a quarter of the training data are used to train 512 GMMs for a Universal Background Model (UBM) and the estimation of the T matrix [4]. Then the i-vector extractor is trained with the entire training set. I-vectors are extracted in an online fashion used in [10]. Only frames prior to the current frame are used during i-vector extraction. The extracted 100-dimensional i-vectors are appended to the 40-dimensional MFCCs at each frame as the inputs of the neural network.

A small 4-gram language model is used for the first-pass decoding. Additionally, a pruned 4-gram model and an RNN language model (RNNLM) are used for lattice rescoring. These three language models are all publicly available[1], and we directly integrate them into our ASR system.

3.3. X-vector based speaker adaptation

As interpreted in section 2, segment-level speaker embeddings can be directly extracted for the speech segments. In order to make the best of the x-vectors for speaker adaptation, we compare some different strategies for extracting the x-vectors.

3.3.1. Data selection

Before training the embedding DNN for x-vectors, non-speech frames and too short utterances are removed because of their inferior quality. In our experiments, only utterances longer than 500 frames (around 5 seconds) are kept in training the DNN for embedding. Meanwhile, speakers with fewer than 8 utterances are thrown out, in order to make sure each speaker has enough training data.

3.3.2. Method of using x-vectors

The extracted segment-level x-vectors are directly appended to the 40 MFCCs of each frame in the corresponding segment, indicating which speaker this segment belongs. The TDNN-F acoustic model is trained with input MFCC features alongside extracted x-vectors. While decoding, we extract the x-vectors for development or test speech in an offline fashion given the entire speech segment.

3.3.3. Speaker split-up

Inspired by the strategy in [10] for i-vector extraction, we modify the number of utterances per speaker to enrich the speaker variety of training data, expecting better generalization. For speakers with more than two utterances in the training set, we try splitting the utterances into more speakers and each speaker has two utterances.

3.3.4. Extraction strategies

When x-vectors are used in speaker recognition, the speech vectors extracted from the neural network are usually in high dimension like 512. Then the vectors are centered and projected into lower dimension like 150 using Linear Discriminant Analysis (LDA). In our paper, we compare two extraction strategies for x-vectors. The one is based on dimension reduction using LDA, the other is directly extracting x-vectors of the dimension we need.

3.3.5. Combination with i-vectors

In [13], experiment results show that i-vectors and x-vectors are complementary and shows better performance for speaker recognition after combination. In this paper, two combination methods are considered and compared. The first method is feature-level combination, which means that i-vectors and x-vectors are concatenated together for each frame to get a new feature vector. The other method is system-level. The decoding results of two ASR systems with different speaker adaptation are combined with ROVER method [21].

[1]http://www.kaldi-asr.org/models/m5

4. RESULTS AND DISCUSSION

In this section, we present the experiment results. In order to analyze the performance of x-vectors in speaker adaptation, we compare different embedding sizes, numbers of utterances per speaker and extraction strategies. I-vectors and x-vectors are compared across all the experiments. A TDNN-F acoustic model without speaker adaptation is also presented for comparison.

4.1. Primary results

Compared to ASR system without speaker adaptation, x-vectors have good performance on both development and test set as shown in Table 3, achieving more than 5% relative reduction on WER. When compared with i-vector based systems, the x-vectors perform slightly worse on development set, but show competitive results on test set. From the preliminary experiment results, x-vectors prove to be effective for speaker adaptation, though current performance is not as good as the i-vectors. In the following parts, we make some adjustments to obtain better results.

Type	d	WER		WER 4-gram		WER RNNLM	
		Dev	Test	Dev	Test	Dev	Test
i-vector	100	7.85	8.39	7.22	7.76	6.20	6.95
x-vector	100	8.37	8.53	7.82	8.13	6.69	7.07
	200	8.18	8.40	7.73	7.95	6.49	6.94
No	-	8.76	8.82	8.18	8.41	6.83	7.43

Table 3. Primary results of the baseline system and x-vector based system. The results of our baseline is slightly different from [18] because of randomness during training.

Table 3 also compares different sizes of x-vectors. The x-vectors are directly extracted from DNN with the needed dimension in this experiment. While increasing the dimension of x-vectors from 100 to 200, the system achieves lower WER on both development and test set. Therefore, x-vectors with higher dimension can perform better for speaker adaptation.

4.2. Speaker split-up

Before extracting x-vectors for training data, we try modifying the speakers of the utterances. The number of speakers is increased so that more speaker variability is added to the TDNN-F acoustic model.

d	Speaker Modification	WER		WER 4-gram		WER RNNLM	
		Dev	Test	Dev	Test	Dev	Test
100	No	8.37	8.53	7.82	8.13	6.69	7.07
	Yes	8.44	8.28	7.83	7.94	6.76	6.97
200	No	8.18	8.40	7.73	7.95	6.49	6.94
	Yes	8.29	8.36	7.74	7.89	6.48	**6.78**

Table 4. Experiment results of x-vector based speaker adaptation with or without speaker split-up.

Table 4 shows the results of x-vector based speaker adaptation with or without speaker split-up. For different embedding sizes, speaker split-up can generally help reduce WER on test data. Our ASR system achieves 6.78% WER on the test set.

4.3. Extraction strategies

In the above experiments, we directly extract x-vectors of the dimension we need for speaker adaptation. In speaker recognition, the extraction method is usually different. Higher dimensional vectors are usually extracted and projected to lower dimension using LDA. Table 5 shows the experiment results of two different extraction strategies for speaker adaptation. It is worth noting that speaker split-up is applied in the following experiments.

The results indicate that x-vectors directly extracted as needed generally performs better than those from LDA dimension reduction. It is reasonable to assume that when the size of x-vectors is fixed, the x-vectors directly extracted from DNN are of richer speaker information. Some crucial information for speaker adaptation may be lost in LDA, so there is no need to have a backend for dimension reduction.

d	Extraction Strategy	WER		WER 4-gram		WER RNNLM	
		Dev	Test	Dev	Test	Dev	Test
100	LDA	8.52	8.48	7.87	8.05	6.69	7.15
	no LDA	8.44	8.28	7.83	7.94	6.76	6.97
200	LDA	8.31	8.57	7.78	8.01	6.65	6.91
	no LDA	8.29	8.36	7.74	7.89	6.48	6.78

Table 5. Experiment results of different strategies for extracting x-vectors. *LDA*: 512-dimensional x-vectors are extracted a first, and are projected to 100 or 200 dimension using LDA. *no LDA*: the x-vectors are directly extracted in 100 or 200 dimension.

4.4. Comparison of speech duration

Compared with i-vectors, x-vectors have shown better performance on short utterances for speaker recognition [13]. In order to evaluate the performance of x-vector based speaker adaptation across different duration of speech segments, we divide the development and test set into three parts according their duration: short (less than 8 seconds), medium (8∼15 seconds), long (more than 15 seconds).

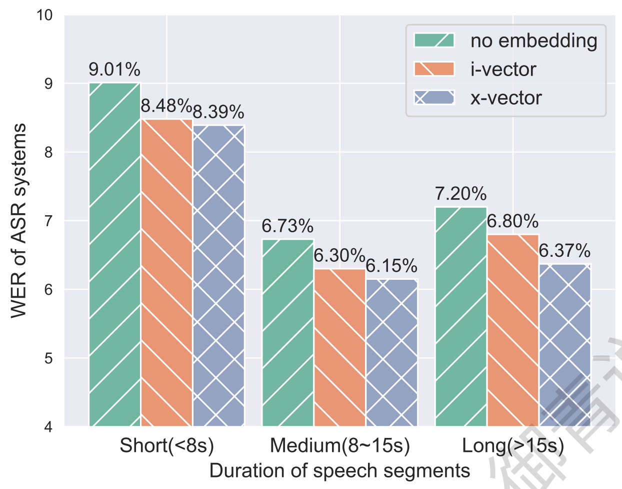


Fig. 2. Performance of ASR systems with different setup of speaker adaptation, evaluated on different duration of speech segments. *no embedding*: only input MFCC features to the TDNN-F acoustic model. *i-vector*: speaker adaptation with i-vectors. *x-vector*: speaker adaptation with x-vectors.

Fig. 2 shows the results of different choices of speaker adaptation. All the results are obtained after lattice rescoring with RNNLM. First of all, it seems our ASR system perform better on medium-length speech segments. WER is obviously higher on short segments. Besides, speaker adaptation with x-vectors achieve the lowest WER across speech segments of different duration, indicating the effectiveness of x-vector based speaker adaptation. However, the improvement compared with i-vectors is limited and there is no evidence that the x-vectors perform better on short segments.

4.5. Combination

In this work, we apply two methods of combing x-vectors and i-vectors as stated in section 3.3.5. Table 6 shows the results of combinations on feature-level and system-level.

system	WER		WER 4-gram		WER RNNLM	
	Dev	Test	Dev	Test	Dev	Test
i-vector	7.85	8.39	7.22	7.76	6.20	6.95
x-vector	8.29	8.36	7.74	7.89	6.48	**6.78**
feature fusion	8.10	8.36	7.46	7.80	6.40	6.90
system fusion	7.70	8.28	7.19	7.71	**6.06**	**6.71**

Table 6. Results of ASR systems with different speaker adaptation methods and fusion strategies. *i-vector*: speaker adaptation with i-vectors. *x-vector*: speaker adaptation with x-vectors. *feature fusion*: concatenating the i-vectors and x-vectors as new feature vectors. *system fusion*: post processing with ROVER method.

With concatenating x-vectors and i-vectors at feature-level, a little improvement is made compared to speaker adaptation with x-vectors only, but still not as good as i-vectors only. It is verified that simply concatenating the feature vectors is not a wise way for fusion. As for system-level fusion, the WER is lower than each individual system after ROVER. This is similar to the fusion strategy in [13], where fusion refers to the equally weighted sum fusion of the Probabilistic LDA (PLDA) scores of i-vectors and x-vectors in post-processing. Therefore, system-level fusion is an ideal method to combine x-vectors and i-vectors. After system-level fusion, we achieve 6.06% WER on development data and 6.71% WER on the test data.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we investigate the methods of applying x-vectors for speaker adaptation of DNN acoustic models in automatic speech recognition. The x-vectors are provided alongside regular MFCC features as inputs to the neural network acoustic model. We present several strategies to improve x-vectors for speaker adaptation, including data selection and augmentation, speaker split-up and a different extraction strategy. We also explore different choices of embedding sizes and methods of combining with i-vectors. Overall, the x-vectors perform slightly better on the test set than an i-vector baseline. X-vectors and i-vectors are complementary when system-level fusion is applied, and 3.45% relative reduction of WER is achieved on the test set compared to the state-of-the-art baseline. In the future, we plan to explore deeper about x-vectors and find out appropriate ways for online decoding with x-vectors.

6. REFERENCES

[1] Christopher J Leggetter and Philip C Woodland, "Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models," *Computer speech & language*, vol. 9, no. 2, pp. 171–185, 1995.

[2] Vassilios V Digalakis, Dimitry Rtischev, and Leonardo G Neumeyer, "Speaker adaptation using constrained estimation of gaussian mixtures," *IEEE Transactions on speech and Audio Processing*, vol. 3, no. 5, pp. 357–366, 1995.

[3] Jordan Cohen, Terri Kamm, and Andreas G Andreou, "Vocal tract normalization in speech recognition: Compensating for systematic speaker variability," *The Journal of the Acoustical Society of America*, vol. 97, no. 5, pp. 3246–3247, 1995.

[4] George Saon, Hagen Soltau, David Nahamoo, and Michael Picheny, "Speaker adaptation of neural network acoustic models using i-vectors," in *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*. IEEE, 2013, pp. 55–59.

[5] Dong Yu, Kaisheng Yao, Hang Su, Gang Li, and Frank Seide, "Kl-divergence regularized deep neural network adaptation for improved large vocabulary speech recognition," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 7893–7897.

[6] Sree Hari Krishnan Parthasarathi, Bjorn Hoffmeister, Spyros Matsoukas, Arindam Mandal, Nikko Strom, and Sri Garimella, "fmllr based feature-space speaker adaptation of dnn acoustic models," in *Sixteenth annual conference of the international speech communication association*, 2015.

[7] Romain Serizel and Diego Giuliani, "Vocal tract length normalisation approaches to dnn-based children's and adults' speech recognition," in *2014 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2014, pp. 135–140.

[8] Najim Dehak, Patrick J Kenny, Réda Dehak, Pierre Dumouchel, and Pierre Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2010.

[9] Mickael Rouvier and Benoit Favre, "Speaker adaptation of dnn-based asr with i-vectors: Does it actually adapt models to speakers?," in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[10] Vijayaditya Peddinti, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur, "Reverberation robust acoustic modeling using i-vectors with time delay neural networks," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[11] Marc Ferras, Cheung Chi Leung, Claude Barras, and Jean-Luc Gauvain, "Constrained mllr for speaker recognition," in *2007 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2007, vol. 4, pp. IV–53.

[12] Yajie Miao, Hao Zhang, and Florian Metze, "Speaker adaptive training of deep neural network acoustic models using i-vectors," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 23, no. 11, pp. 1938–1949, 2015.

[13] David Snyder, Daniel Garcia-Romero, Daniel Povey, and Sanjeev Khudanpur, "Deep neural network embeddings for text-independent speaker verification.," in *Interspeech*, 2017, pp. 999–1003.

[14] David Snyder, Daniel Garcia-Romero, Gregory Sell, Daniel Povey, and Sanjeev 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.

[15] Joanna Rownicka, Peter Bell, and Steve Renals, "Embeddings for dnn speaker adaptive training," *arXiv preprint arXiv:1909.13537*, 2019.

[16] Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[17] Desh Raj, David Snyder, Daniel Povey, and Sanjeev Khudanpur, "Probing the information encoded in x-vectors," *arXiv preprint arXiv:1909.06351*, 2019.

[18] François Hernandez, Vincent Nguyen, Sahar Ghannay, Natalia Tomashenko, and Yannick Estève, "Ted-lium 3: twice as much data and corpus repartition for experiments on speaker adaptation," in *International Conference on Speech and Computer*. Springer, 2018, pp. 198–208.

[19] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al., "The kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*. IEEE Signal Processing Society, 2011.

[20] Tom Ko, Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur, "Audio augmentation for speech recognition," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[21] Jonathan G Fiscus, "A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (rover)," in *1997 IEEE Workshop on Automatic Speech Recognition and Understanding Proceedings*. IEEE, 1997, pp. 347–354.