*INTERSPEECH 2016*
September 8–12, 2016, San Francisco, USA

# Speaker normalization through feature shifting of linearly transformed i-vector

*Jahyun Goo, Younggwan Kim, Hyungjun Lim, Hoirin Kim*

School of Electrical Engineering, KAIST, Daejeon, Republic of Korea
{jahyun.goo, cleanthink, hyungjun.lim, hoirkim}@kaist.ac.kr

## Abstract

In this paper, we propose a simple speaker normalization for deep neural network (DNN) using i-vectors, the state-of-the-art technique for speaker recognition, for automatic speech recognition. There have been already many techniques using i-vectors for speaker adaptation or speaker variability reduction of DNN acoustic models. However, in order to add the speaker information into the acoustic feature, most of those techniques have to train a large number of parameters while dimensionality of the i-vector is quite small. We tried to apply a component-wise shift to the acoustic features by linearly transformed i-vector, and then achieved the better performance than typical approaches. On top of that, we propose to modify this structure to adapt each frame of the features, reducing the number of parameters. Experiments were conducted on the TED-LIUM release-1 corpus, and the proposed method showed some performance gains.

**Index Terms**: speaker normalization, speaker adaptation, i-vector, deep neural networks, speech recognition

## 1. Introduction

In the last few years, deep neural networks (DNN) based acoustic model defeated the conventional acoustic modelling approach, Gaussian mixture model (GMM), and achieved a dominant position in automatic speech recognition (ASR) task. DNN provides more discriminative feature representation than generative one in GMM, whereupon DNN can extract phonetic information effectively with enough computing power. In other words, a DNN-based acoustic model is more robust on speaker variability and also provides better speech recognition performance than an GMM-based one. Unfortunately, because of the robustness, it is hard to propose an effective speaker adaptation technique in spite of their necessity.

Size of DNN-based model is another obstacle to finding adaptation technique. The number of parameters of DNN is naturally high since nonlinear projection to high dimensionality is used to derive high recognition accuracy and speaker invariability in DNN. On the other hand, the number of one speaker's utterances to adapt the model is usually very small. Therefore speaker adaptation of DNN remains as a difficult task and plenty of techniques have been proposed to overcome this difficulty.

According to [1], these techniques can be grouped into 3 categories: Linear transformation, Conservative training, and Speaker-specific subspace method. Linear transformation might be the simplest and most popular way to adapt the DNN. For one certain layer of speaker-independent DNN, linear transformation is applied on top of that layer with the speeches of one speaker [2, 3, 4]. Conservative training is to keep existing parameters unchanged when you want to adapt entire model or part of that, for example the Maximum *A Posteriori* (MAP) adaptation of GMM model. L2-norm regularization [5] and Kullback-Leibler divergence (KLD) regularization [6] are well-known techniques in this category. Recently it was proposed to apply the MAP to the DNN model after the dimensionality reduction using the Singular Value Decomposition (SVD) [7]. In this context, we want to mention Learning Hidden Unit Contribution (LHUC) [8] which is also train the networks' weights speaker-dependently. Lastly, subspace method is the way to extend the input feature space by additional speaker-specific subspace. The most representative way of the subspace method is to concatenate the acoustic feature with pre-generated i-vector [9, 10, 11].

I-vector (*identity* vector) is an approach that have achieved the state-of-the-art performance of speaker recognition [12, 13]. In that task, one can compare an i-vector with several others by appropriate measure such as cosine distance or PLDA scoring, and identify or verify a speaker's identity. It implies that the i-vector extracts and encodes the speaker's discriminative information intensively in relatively small dimension. Therefore, in subspace method of speech adaptation, the robustness on speaker variability is enhanced by concatenating the i-vectors to the acoustic feature vectors. It is, in other words, equal to the speaker normalization, modifying the feature representation based on the speaker information [14, 15]. Many techniques were proposed for speaker normalizations using i-vector aside from the i-vector augmentation.

Garimella *et al.* proposed to use the i-vector generated from HMM-GMM, not UBM-GMM, and proposed a structure in which output vector of i-vector DNN, not i-vector itself, is augmented to the acoustic feature [16]. One more remarkable point is that the authors applied exponentially decaying weights to the utterances used in i-vector extractor, for i-vectors to reflect recent information (they called it causal i-vector.). In Cardinal *et al.*'s paper, i-vectors were concatenated to the bottleneck feature from 2-stage large DNN, demonstrating the performance improvements [17]. Miao *et al.* proposed two speaker adaptive training methods using i-vector, named AdaptNN and iVecNN [18, 19]. AdaptNN has additional layers below the regular DNN to adapt the acoustic feature, and the i-vectors are used as a bias in each adaptation layer. iVecNN is the multi-layer network of the i-vector as input and feature-sized vector as output, and the output vector is summed to the acoustic vector as a speaker-specific linear shift on the feature space. We got inspiration from the iVecNN, and our proposed method will to be explained later.

This paper is organized as follows. A brief explanations of i-vector are in Section 2, and Section 3 gives our simple DNN normalization method and experiment results with our experimental setup. Further structure named one-frame shifting are



http://dx.doi.org/10.21437/Interspeech.2016-819

stated in Section 4, and this paper concludes with future research plan in Section 5.

## 2. I-vector

The i-vector is introduced and widely used in speaker recognition task since each speaker's characteristics are effectively extracted as relatively low-dimensional vector. The reason we use the phrase "relatively low-dimensional" is that many speaker recognition techniques use very high-dimensional vector called *supervector*, which is a concatenated vector of a speaker's GMM mean adapted from the universal background model (UBM). The variability in supervector can be modelled as low-dimensional coordinate vector and corresponding space, and this approach is known as Factor Analysis (FA).

In Joint Factor Analysis (JFA), speaker variability and channel variability are exclusively represented. Total Variability (TV) model, however, extracts the entire variabilities from speakers and channels and then models them onto one space, in which corresponding coordinate vector is considered as a representative of speaker [12, 13]. In the following equation, the speaker variability, presented as a difference between the speaker-dependent supervector $M$ and UBM supervector $m$, is modelled as a subspace of acoustic feature domain $T$ and a corresponding coordinate vector $w$.

$$M = m + Tw \tag{1}$$

The vector $w$ is a standard normal distributed random vector, and the MAP point estimate of $w$ is considered as *identity vector*, or i-vector. The i-vector approach have been successful in speaker recognition task since the i-vector encodes the characteristics of each speaker well in spite of the simple calculation and the small number of parameters [13].

In this paper, we basically used the per-utterance i-vectors, not per-speaker ones. That is because we assume that the channel can change constantly in real environments even if same speaker keeps speaking. Per-utterance i-vector can reflect the channel variability of corresponding utterance since TV matrix involves the concept of the channel-variability matrix as well as speaker variability matrix as mentioned above. Based on this idea, we trained the i-vector extractor and generated the i-vectors per utterance. These i-vectors are used to normalize the DNN acoustic model as described in next section.

## 3. Feature shifting with i-vector

### 3.1. Feature shifting of linearly transformed i-vector

Here we propose the idea that the acoustic feature input can be shifted as a same-size vector generated by linear transformation of the utterance's i-vector, resulting in better recognition performance. For example, you can use 440×100-size matrix to transform the 100-dimensional i-vector if you use the 11 frames of 40-dimensional acoustic feature.

As mentioned before, many authors used the i-vector expecting their model to have invariabilities because the i-vector extracts those speaker and channel information efficiently [10]. They uses the i-vectors as DNN inputs parallel to the acoustic feature, whereas we simply shift the acoustic feature itself.

Miao *et al.* ([18]) already used the i-vector induced feature shifting, but we thought that the linear transformation was enough to utilize the i-vector appropriately for the acoustic features since i-vectors are originally derived from linear mapping of feature information as equation (1). However, we did not use or make the Total Variability matrices ($T$ in eq. (1)) or other matrices but train the network weights with traditional cross-entropy criterion. Overall training procedure resembles the Miao's one: initialize phone classification networks with SI networks, train the i-vector networks only, and then re-train the phone classification networks. The difference is that our proposed approach is linear transform and we used the per-utterance i-vector. On top of that, we are supposed to suggest something further later. Model structure is in figure 1.

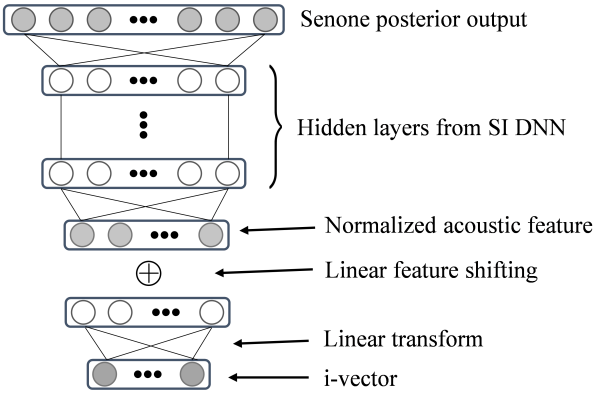


Figure 1: *Diagram of proposed model.*

### 3.2. Experimental setup

In our experiments, we used the TED-LIUM release-1 corpus [20] for entire experiments in this paper. Corpus contains about 118 hours of speeches, which were collected from TED talks of 774 speakers. We used the 95% of the whole corpus for the model training and 5% for validation to prevent the overfitting. Two small sets named dev and test were used for performance measurement (4.2 and 2.5 hours, respectively).

I-vector extractor training, GMM-HMM model training and overall speech recognition experiments were done by open-source Kaldi toolkit [21]. The acoustic features used to train an i-vector extractor was 60-dimensional MFCC (20-dim. MFCC with delta and delta-delta). The extractor was trained on the UBM of 2048 full-covariance Gaussians, and generated the 100-dimensional per-utterance i-vectors.

Before training the DNN model, we made GMM-HMM models to generate alignments and fMLLR-adapted features. We used 39-dim. MFCC (13-dim. MFCC with delta and delta-delta) for training GMM-HMM model and followed the Kaldi's recipe. Finally we had the GMM-HMM models with 3,969 context-dependent phones and 100,133 Gaussians, and fMLLR-adapted 39-dim. MFCC features.

For DNN[1] training and decoding, we basically used the 40-dim. Mel-filterbank (fbank) acoustic features but also fMLLR features are used. We trained two kind of networks, fbank and fMLLR features, for each experiments. For baseline SI model, each network was composed of 6 fully connected logistic sigmoid hidden layers with 2,048 nodes and softmax outputs. 11 frames of the acoustic feature vectors were spliced (current frame and consecutive ± 5 frames) and considered as an input vector of the network. (440-dim. for fbank and 429-dim. for fMLLR feature.) Networks were pre-trained based

[1]We used the word "DNN" to signify the DNN structure, or DNN-HMM hybrid acoustic model.

Table 1: WER (%) of various DNN models for fbank/fMLLR features on TED-LIUM dev/test corpus, respectively.

| Models | fbank | | fMLLR | |
|---|---|---|---|---|
| | dev | test | dev | test |
| Baseline DNN | 18.0 | 16.0 | 16.9 | 15.4 |
| i-vector augmentation | 17.3 | 16.7 | 16.5 | 15.2 |
| SAT-DNN | 16.5 | 14.7 | 16.0 | 14.5 |
| Proposed | **16.2** | **14.3** | **15.8** | **14.3** |
| SAT-DNN + baseline DNN | 17.8 | 15.6 | - | - |
| Proposed + baseline DNN | 17.4 | 15.6 | - | - |

on stacked denoising autoencoder[22] and fine-tuning was executed according to exponentially decaying learning rate.

For comparison, we implemented the i-vector augmentation networks ([9]) and SAT-DNN ([23]). Training procedure of the i-vector augmentation networks were similar to the baseline SI networks except for input vector concatenated by the i-vector. For SAT-DNN network, we used 3-hidden-layer networks as the i-vector-to-feature network and each hidden layer has 512 nodes. SAT-DNN and proposed structure were trained as mentioned before: initialize with SI model, train the i-vector networks only, and then update the phone classification networks only. Random and zero initialization were executed for weights of i-vector part in the i-vector augmentation networks and weights for SAT-DNN and proposed structure, but they did not show meaningful differences. All trainings of the DNN models were done by Kaldi+PDNN framework ([24]) which bridges the gap between the Kaldi and the Theano and be specialized in i-vector speaker adaptation.

### 3.3. Baseline experiments

Table 1 shows the results of various baseline experiments. In general, the results of DNNs with speaker-adapted fMLLR features were better than speaker-independent fbank feature cases. Even though both fbank and fMLLR networks were influenced by the i-vector based speaker adaptation, the speaker-independent fbank networks showed much more improvements of recognition accuracy. That is because, in the fMLLR feature, the speaker variability had already been normalized or modelled by the fMLLR transformation.

The i-vector augmentation showed little amount of performance improvements than the baseline DNN (in some case, performance became worse.). Otherwise, SAT-DNN and our proposed method showed much better performance improvements than i-vector augmentation did. Proposed method defeated the SAT-DNN, and we think this is because the number of parameters is small and the linear transformation resembles the inherent structure of the i-vector extraction as in eq. (1).

The last two rows in Table 1 are the results when we kept the upper networks unchanged while the i-vector network updated. Their performances were slightly better than the results of baseline DNN. It follows from this that feature normalization seemed to be in effect. However, the performances were much worse than regular adaptation techniques. It means that the acoustic models should also adapt to the feature-space changes. Our proposed networks were slightly better than the SAT-DNNs in this case.

Table 2: WER (%) of various DNN models with per-speaker i-vectors.

| Models | fbank | | fMLLR | |
|---|---|---|---|---|
| | dev | test | dev | test |
| Baseline DNN | 18.0 | 16.0 | 16.9 | 15.4 |
| i-vector augmentation | 16.8 | 15.7 | 16.5 | 15.0 |
| SAT-DNN | 16.3 | 14.9 | 16.0 | 14.6 |
| Proposed | **16.0** | **14.6** | **15.9** | **14.5** |

### 3.4. Per-speaker i-vector experiments

We also did the experiments on per-speaker i-vectors, and results are in Table 2. Trend of overall performances is similar to the baseline experiments and proposed method defeated others. One figure that does deserve highlighting in Table 2 is every experiments perform slightly better than corresponding one in Table 1. That is because per-speaker i-vectors are more reliable to hold the speaker information than per-utterance one. In addition, TED talks are held in a stable environment. It means that the channel variability of one speaker in the TED-LIUM database is quite small. Therefore the performances in Table 2 is slightly better than the results in Table 1. Causal i-vector ([16]) can reduce the gap between per-speaker and per-utterance i-vector, but we saved it for future work.

One question remains: Why did the multi-layer networks get worse results than the linear transform, notwithstanding the fact that the multi-layer perceptron is the universal function approximator ([25])? We surmised that the problem is overfitting to the i-vectors of training set. We will apply the L2-norm regularization on both techniques and analyze the results in our future work.

## 4. One-frame feature shifting

In this section, we propose the other kind of the feature shifting approach. The per-utterance i-vectors extract the speaker and channel variabilities between utterances. In other words, within an utterance, the i-vector's effect on each frame is nearly being unchanged. Based on this idea, we suggest a modified approach of the previously proposed method, named one-frame shifting in this section.

In this approach, we train the linear transform from the i-vector to one frame of the acoustic feature, and then use the one-frame sized vector to shift entire input feature vector. For example, in our experimental setup (11 consecutive frames of 40-dim. features), we can make the 440-dim. feature shifting by training just the $40\times100$-sized matrix as in Figure 2. Basically its framework is similar to the one in Section 3, but the size of linear transformation matrix become much smaller in this approach.

Actually, our implementation was different from the Figure 2, constructed in more tricky way. First we made the 40-dim. hidden layer between the i-vector input layer and feature shifting output layer. Then we set the weights between the hidden and the output layer as stacked identity matrix and fixed them.

$$\begin{bmatrix} 1 & 0 & \cdots & 0 & & 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \cdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & & 0 & 0 & \cdots & 1 \end{bmatrix}^T \tag{2}$$

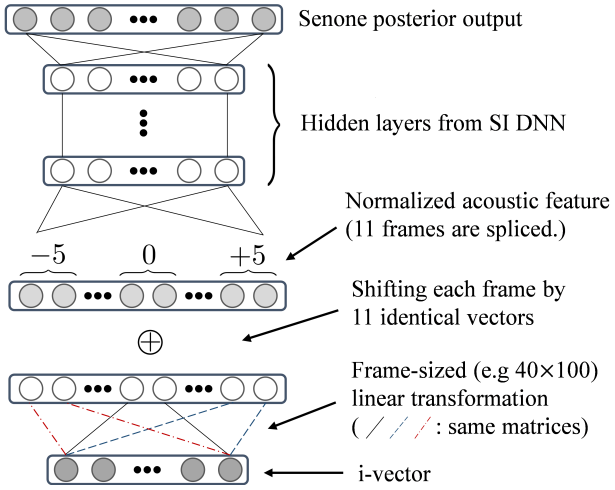


Figure 2: *Diagram of one-frame feature shifting.*

By this fixed weights, values in the hidden nodes copied to the position of each frame. In training, we updated the weights between the input and the hidden layer.

This one-frame shifting has a competitive performance comparing to the other approaches in spite of the parameter reduction. In addition, we expect this approach to be more robust on the variation of the number of input acoustic frames since the number of parameters to tune is unchanged as the number of frames increases. This is a great advantage of one-frame shifting model because nowadays more input frames are being accepted as the DNN models become larger. In large-scale DNN model, one-frame shifting can work better than other i-vector based adaptation approaches.

This model can also cope with the variation of the i-vector/feature vector dimension because of the smaller size of the weight matrix. For example, if the dimensionality of the acoustic feature increases from 40 to 60, then the parameter number increment is $20\times11\times100$ for linear transform model and $20\times100$ for one-frame shifting model respectively. We wanted to know experimentally that our model was robust on variations of these factors: input frame number, feature dimension and the i-vector dimension. However, we could not do these experiments because of the time insufficiency and so leave those as the future work.

### 4.1. Experiments

We used the 1/10 subset of the entire TED-LIUM corpus for the experiments of the one-frame shifting, due to time insufficiency. After choosing 67 speakers randomly, their 5,097 utterances were used in the experiments of this section. As the size of database decreases, we used the smaller DNN to build up baseline model: 6 layers and 1,024 units per each layer. The other factors were same as our experiments in Section 3. 11-frame spliced 40-dim. fbank features were used for speeches to be classified into 3,969 context-dependent phones. The i-vectors used in this experiments were same as in previous experiments.

The results of experiments are in Table 3. Due to the reduction in the database, the baseline DNN performance degenerated. It seemed that overfitting occurred because we decreased the model size by half while the database size was reduced 1/10. The results of the i-vector augmentation got even worse than the baseline DNN, whereupon we became convinced of overfitting. The i-vector augmentation network naturally has a lot more weights than other adaptation methods since additional weights are added between input and hidden layer.

Table 3: WER (%) of various DNN models including one-frame shifting on the fbank features of TED-LIUM 1/10 set.

| models | dev | test |
|---|---|---|
| Baseline DNN | 27.4 | 26.9 |
| i-vector augmentation | 29.3 | 29.6 |
| SAT-DNN | 26.4 | 25.9 |
| Linear transformation (proposed) | 26.0 | 25.7 |
| One-frame shifting (proposed) | **25.5** | **25.0** |
| Linear trans. + baseline DNN | 27.0 | 26.7 |
| One-frame + baseline DNN | 27.1 | 26.5 |

Overall trends were similar to Table 1. The performances of SAT-DNN were good, and our linear transformation results were slightly better than SAT-DNN. On top of that, the one-frame shifting had a lower WER even if its model size was only a eleventh of the our linear transformation.

The last 2 rows of Table 3 are the results of feature shifting only. As in last 2 rows of Table 1, their upper networks kept unchanged while the lower i-vector networks were updated. This means the feature normalization without corresponding model adjustment, as mentioned before. The noticeable point is that the feature normalization version of the one-frame shifting got 26.5%, relatively 6% worse than the model adaptation result (test set number). It was quite small degradation considering that the model was not updated. It means that the one-frame shifting can adjust the acoustic features suitable for SI DNN, as befits the name *speaker normalization*. We will check whether this tendency is still valid on larger database as the future work.

## 5. Conclusions

We have showed a speaker normalization technique using the per-utterance i-vectors in this paper. First we proposed a linear transformation of the i-vectors to adapt the acoustic features, and then presented more simplified version of it, one-frame shifting. Both approaches shift acoustic feature vectors in the speaker-normalized direction by the i-vectors mapped into feature space. Speaker normalized feature itself can show an effect of the performance gain comparing to the baseline DNN, but speaker-independent DNN can be updated further completing the model adaptation. In experiments, both feature shifting showed meaningful performance improvements.

There are many things to do in our plan. First of all, we will apply the L2-norm regularization on the proposed methods. Also, experiments of one-frame shifting will be done with the larger database. For future works, we will check the performance of the causal i-vector [16] and confirm the robustness on the various dimension and splicing scale of the feature vectors.

## 6. Acknowledgements

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. 2014R1A2A2A01007650).

# 7. References


[1] D. Yu and L. Deng, *Automatic Speech Recognition: A Deep Learning Approach*. Springer, 2015.

[2] V. Abrash, H. Franco, A. Sankar, and M. Cohen, "Connectionist speaker normalization and adaptation," in *Proc. Eurospeech*, 1995.

[3] J. Neto, L. Almeida, M. Hochberg, C. Martins, L. Nunes, S. Renals, and T. Robinson, "Speaker-Adaptation for Hybrid HMM-ANN Continuous Speech Recognition System," in *Proc. Eurospeech*, 1995.

[4] R. Gemello, F. Mana, S. Scanzio, P. Laface, and R. De Mori, "Adaptation of Hybrid ANN/HMM Models Using Linear Hidden Transformations and Conservative Training," in *Proc. ICASSP*, vol. 1, 2006.

[5] X. Li and J. Bilmes, "Regularized Adaptation of Discriminative Classifiers," in *Proc. ICASSP*, 2006.

[6] D. Yu, K. Yao, H. Su, G. Li, and F. Seide, "KL-divergence regularized deep neural network adaptation for improved large vocabulary speech recognition," in *Proc. ICASSP*, 2013.

[7] Z. Huang, S. M. Siniscalchi, I.-f. Chen, J. Li, J. Wu, and C.-h. Lee, "Maximum a Posteriori Adaptation of Network Parameters in Deep Models," in *Proc. Interspeech*, 2015.

[8] P. Swietojanski and S. Renals, "Learning hidden unit contributions for unsupervised speaker adaptation of neural network acoustic models," in *Proc. SLT*, 2014.

[9] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny, "Speaker adaptation of neural network acoustic models using i-vectors," in *Proc. ASRU*, 2013.

[10] A. Senior and I. Lopez-Moreno, "Improving DNN speaker independence with I-vector inputs," in *Proc. ICASSP*, 2014.

[11] V. Gupta, P. Kenny, P. Ouellet, and T. Stafylakis, "I-vector-based speaker adaptation of deep neural networks for French broadcast audio transcription," in *Proc. ICASSP*, 2014.

[12] N. Dehak, R. Dehak, P. Kenny, N. Brummer, P. Ouellet, and P. Dumouchel, "Support vector machines versus fast scoring in the low-dimensional total variability space for speaker verification," in *Proc. Interspeech*, 2009.

[13] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE trans. Audio, Speech and Language Processing*, vol. 19, no. 4, pp. 788–798, 2010.

[14] X. Huang, A. Acero, and H.-W. Hon, *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development*. Prentice Hall, 2001.

[15] R. L. Watrous, "Speaker normalization and adaptation using second-order connectionist networks," *IEEE trans. Neural Networks*, vol. 4, no. 1, pp. 21–30, 1993.

[16] S. Garimella, A. Mandal, N. Strom, B. Hoffmeister, S. Matsoukas, S. Hari, and K. Parthasarathi, "Robust i-vector based Adaptation of DNN Acoustic Model for Speech Recognition," in *Proc. Interspeech*, 2015.

[17] P. Cardinal, N. Dehak, Y. Zhang, and J. Glass, "Speaker Adaptation Using the I-Vector Technique for Bottleneck Features," in *Proc. Interspeech*, 2015.

[18] Y. Miao, H. Zhang, and F. Metze, "Towards Speaker Adaptive Training of Deep Neural Network Acoustic Models," in *Proc. Interspeech*, 2014.

[19] ——, "Speaker Adaptive Training of Deep Neural Network Acoustic Models Using I-Vectors," *IEEE/ACM trans. Audio, Speech, and Language Processing*, vol. 23, no. 11, pp. 1938–1949, nov 2015.

[20] A. Rousseau, P. Deléglise, and Y. Estève, "TED-LIUM: an Automatic Speech Recognition dedicated corpus," *Proc. LREC-2012*, 2012.

[21] D. Povey, A. Ghoshal, and G. Boulianne, "The Kaldi speech recognition toolkit," in *Proc. ASRU*, 2011.

[22] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion," *J. Machine Learning Research*, vol. 11, no. 3, pp. 3371–3408, 2010.

[23] Y. Miao, L. Jiang, H. Zhang, and F. Metze, "Improvements to speaker adaptive training of deep neural networks," in *Proc. SLT*, 2014.

[24] Y. Miao, "Kaldi+PDNN: Building DNN-based ASR Systems with Kaldi and PDNN," *arXiv*, 2014. [Online]. Available: http://arxiv.org/abs/1401.6984

[25] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, jan 1989.