INTERSPEECH 2017
August 20–24, 2017, Stockholm, Sweden

Learning Factorized Transforms for Unsupervised Adaptation of LSTM-RNN Acoustic Models

Lahiru Samarakoon[1], *Brian Mak*[1], *Khe Chai Sim*[2]

[1]Hong Kong University of Science and Technology
[2]Google, Inc

lahiruts@cse.ust.hk, mak@cse.ust.hk, khechai@google.com

Abstract

Factorized Hidden Layer (FHL) adaptation has been proposed for speaker adaptation of deep neural network (DNN) based acoustic models. In FHL adaptation, a speaker-dependent (SD) transformation matrix and an SD bias are included in addition to the standard affine transformation. The SD transformation is a linear combination of rank-1 matrices whereas the SD bias is a linear combination of vectors. Recently, the Long Short-Term Memory (LSTM) Recurrent Neural Networks (RNNs) have shown to outperform DNN acoustic models in many Automatic Speech Recognition (ASR) tasks. In this work, we investigate the effectiveness of SD transformations for LSTM-RNN acoustic models. Experimental results show that when combined with scaling of LSTM cell states' outputs, SD transformations achieve 2.3% and 2.1% absolute improvements over the baseline LSTM systems for the AMI IHM and AMI SDM tasks respectively.

Index Terms: Long Short-Term memory (LSTM), Recurrent Neural Networks (RNNs), Speaker Adaptation, Acoustic Modeling

1. Introduction

All machine learning techniques, including deep learning based methods, are susceptible to performance degradation due to the training and testing mismatch. The variabilities caused by the mismatch can be normalized by transforming the model to match testing conditions or by transforming the runtime features to match the model. In automatic speech recognition (ASR), speaker adaptation techniques are developed to minimize the mismatch between the training and testing conditions due to speaker variability.

The adaptation techniques are first developed for conventional Gaussian mixture model (GMM)–hidden Markov model (HMM) systems. The commonly used techniques include maximum a posteriori (MAP) [1] and maximum likelihood linear regression (MLLR) [2, 3]. In addition, speaker adaptive training (SAT) has been applied to GMM-HMM systems [4, 5]. Then, the adaptation techniques were developed for deep neural network (DNN)-HMM hybrid systems. The adaptation of DNNs has found to be effective as these methods improve the performance significantly [6, 7, 8, 9, 10, 11, 12]. However, DNNs are not effective in modeling the temporal dependencies of speech signals. Therefore, the current state of the art in ASR is to use models that are capable of modeling temporal dependencies of speech signals like LSTM-RNNs. Since LSTM-RNNs are more complex structures than DNNs, LSTM-RNN adaptation is more challenging, especially when performed with a small amount of data in an unsupervised fashion.

In this paper, we investigate the FHL-based adaptation for LSTM-RNN acoustic models. The previous work on LSTM-RNN adaptation has employed SD biases to perform adaptation. In [13], speaker code is used for adaptation, while acoustic feature concatenation with speaker representations is used in [14]. In addition, the adaptation of different weight matrices of LSTM-RNN is investigated in [15]. Factorized hidden layer (FHL) adaptation is proposed to adapt DNNs and has shown superior performance over SD bias based methods [16]. In FHL adaptation, a speaker-dependent (SD) transformation matrix and an SD bias are estimated in addition to the standard affine transformation. The SD transformation is a linear combination of rank-1 matrices whereas the SD bias is a linear combination of vectors. In this work, we investigate the SD transformation-based adaptation for LSTM-RNN. Furthermore, since learning hidden unit contributions (LHUC) [17] has been found effective for DNN adaptation, we also investigate the effectiveness of scaling various components of LSTM-RNN for adaptation. We evaluate the proposed method in two benchmark ASR tasks: the Augmented Multi-party Interaction (AMI) [18] individual headset microphone (IHM) and the AMI single distant microphone (SDM) tasks, respectively.

The rest of the paper is organized as follows. Section 2 reviews the LSTM-RNN acoustic model. Section 3 discusses the FHL-based adaptation for LSTM-RNNs while in Section 4, scaling-based LSTM-RNN adaptation is presented. In Section 5, we give the details of the experimental setup. The results are reported in Section 6 and we conclude our work in Section 7.

2. LSTM-RNNs

LSTM is proposed in [19] to avoid the vanishing gradient problem in RNN training using stochastic gradient descent method. LSTM contains memory blocks with self-connections to store the temporal state of the network. Sets of units called gates are used to control the flow of information to each LSTM cell. In general, there are three types of gates called input, output and forget. An input gate controls the flow of input to the memory cell whereas an output gate controls the output flow. Forget gates decide how much information to forget during each time step [20]. In addition to the gates, peephole connections are used to connect cell state information to the gates [21]. Furthermore, it is shown that LSTMP models where a projection layer is used to reduce the network complexity are more effective in ASR [22]. In this paper, the adaptation is investigated for LSTMP acoustic models. The behaviour of an LSTMP may be summarized by the following formulas:

$$\mathbf{i}_t = \sigma(\mathbf{W}_{xi}\mathbf{x}_t + \mathbf{W}_{ri}\mathbf{r}_{t-1} + \mathbf{W}_{ci}\mathbf{c}_{t-1} + \mathbf{b}_i) \quad (1)$$

$$\mathbf{f}_t = \sigma(\mathbf{W}_{xf}\mathbf{x}_t + \mathbf{W}_{rf}\mathbf{r}_{t-1} + \mathbf{W}_{cf}\mathbf{c}_{t-1} + \mathbf{b}_f) \quad (2)$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_{xo}\mathbf{x}_t + \mathbf{W}_{ro}\mathbf{r}_{t-1} + \mathbf{W}_{co}\mathbf{c}_{t-1} + \mathbf{b}_o) \quad (3)$$

http://dx.doi.org/10.21437/Interspeech.2017-1136

$$\mathbf{c}_t = \mathbf{f}_t \circ \mathbf{c}_{t-1} + \mathbf{i}_t \circ \tanh(\mathbf{W}_{xc}\mathbf{x}_t + \mathbf{W}_{rc}\mathbf{r}_{t-1} + \mathbf{b}_c) \quad (4)$$

$$\mathbf{m}_t = \mathbf{o}_t \circ \tanh(\mathbf{c}_t) \quad (5)$$

$$\mathbf{r}_t = \mathbf{W}_{mr}\mathbf{m}_t \quad (6)$$

where t is the timestep, σ is the sigmoid funtion, $\mathbf{i}_t, \mathbf{f}_t, \mathbf{o}_t, \mathbf{c}_t, \mathbf{m}_t, \mathbf{r}_t$ are vectors with input gate, forget gate, output gate, cell state, cell output, and projection values respectively. $\mathbf{W}_{**}$ are weight matrices and $\mathbf{b}_*$ are biases. All peephole weight matrices $\mathbf{W}_{c*}$ are diagonal.

3. FHL-based LSTM-RNN Adaptation

First, we review the FHL adaptation for DNNs. Then, the FHL-based adaptation for LSTM-RNNs is presented.

3.1. FHL Adaptation for DNNs

$$\mathbf{W}^s = \mathbf{W} + \sum_{i=1}^{|\mathbf{d}^s|} \mathbf{d}^s(i)\mathbf{B}(i) \quad (7)$$

where $\{\mathbf{B}(1), \mathbf{B}(2), .., \mathbf{B}(|\mathbf{d}_s|)$ is the set of basis for the SD transformation and $\mathbf{d}^s$ is the SD interpolation vector. Similarly, the SD bias vector, $\mathbf{b}^s$ is given by:

$$\mathbf{b}^s = \mathbf{b} + \sum_{i=1}^{|\mathbf{v}^s|} \mathbf{v}^s(i)\mathbf{u}(k) = \mathbf{b} + \mathbf{U}\mathbf{v}^s \quad (8)$$

where $\mathbf{v}^s$ is the SD interpolation vector.

Furthermore, in [16] $\mathbf{B}(i)$ weight bases are constrained to be rank-1 matrices. This allows us to formulate the SD transformation as:

$$\begin{aligned}\mathbf{W}^s &= \mathbf{W} + \sum_{i=1}^{|\mathbf{d}^s|} \mathbf{d}^s(i)\boldsymbol{\gamma}(i)\boldsymbol{\psi}^\top(i) \\ &= \mathbf{W} + \boldsymbol{\Gamma}\mathbf{D}^s\boldsymbol{\Psi}^\top \end{aligned} \quad (9)$$

where $\mathbf{B}(i) = \boldsymbol{\gamma}(i)\boldsymbol{\psi}^\top(i)$ and $\mathbf{D}^s$ is a diagonal matrix ($\mathbf{D}^s = \text{diag}(\mathbf{d}^s)$) and $\boldsymbol{\gamma}(i)$, $\boldsymbol{\psi}(i)$ are i-th column vectors for $\boldsymbol{\Gamma}$, $\boldsymbol{\Psi}$ respectively.

3.2. FHL Adaptation for LSTM-RNNs

FHL adaptation for LSTM-RNNs can be applied by modelling SD transformations and SD biases for various $\mathbf{W}_{**}$ and $\mathbf{b}_*$ in the LSTM-RNNs (Equations (1) - (6)). For instance, we can estimate the SD transformations on the input feature ($\mathbf{x}_t$) as given below:

$$\mathbf{W}^s_{x*} = \mathbf{W}_{x*} + \boldsymbol{\Gamma}_{x*}\mathbf{D}^s_{x*}\boldsymbol{\Psi}^{l\top}_{x*} \quad (10)$$

where $\mathbf{D}^s_{x*} \in \mathbb{R}^{|\mathbf{d}^s| \times |\mathbf{d}^s|}$ is a diagonal matrix ($\mathbf{D}^s_{x*} = \text{diag}(\mathbf{d}^s)$).

Similarly, an SD transformation is estimated for the recurrence connections as given below:

$$\mathbf{W}^s_{r*} = \mathbf{W}_{r*} + \boldsymbol{\Gamma}_{r*}\mathbf{D}^s_{r*}\boldsymbol{\Psi}^{l\top}_{r*} \quad (11)$$

In our experiments, we investigate the effectiveness of SD transformations on input features as well as recurrence connections. SD transformations are not estimated for peephole weight matrices ($\mathbf{W}_{c*}$), since those connections are diagonal.

Similar to the FHL adaptation for DNNs, the SD bias vector, $\mathbf{b}^s_*$ can be estimated for LSTM-RNNs (Equation 8).

3.3. Increasing the number of Bases

The adaptation power of the SD transformations can be increased by increasing the number of bases for SD transformations. We increase the number of bases by estimating a non-linear projection during training as given below:

$$\hat{\mathbf{d}}^s = \sigma(\boldsymbol{\Lambda}\mathbf{d}^s) \quad (12)$$

where $|\hat{\mathbf{d}}^s| > |\mathbf{d}^s|$. Then, this new SD representation can be used to estimate the bases for SD transformation as given below:

$$\mathbf{W}^s_{**} = \mathbf{W}_{**} + \boldsymbol{\Gamma}_{**}\hat{\mathbf{D}}^s_{**}\boldsymbol{\Psi}^{l\top}_{**} \quad (13)$$

where $\hat{\mathbf{D}}^s_{**} = \text{diag}(\hat{\mathbf{d}}^s)$.

4. Speaker-Dependent Scaling

Inspired by LHUC [17], we investigate scaling of the LSTM components. We describe the scaling of the input gate $\mathbf{i}_t$ below. The scaling of the forget gate $\mathbf{f}_t$ and the output gate $\mathbf{o}_t$ is similar.

$$\mathbf{i}^s_t = \mathbf{A}^s\mathbf{i}_t \quad (14)$$

where $\mathbf{A}^s$ is the SD diagonal matrix as $\mathbf{A}^s = diag(\mathbf{a}^s)$.

In LHUC adaptation, an additional constraint is applied to the diagonal elements which restrict them in the range of $[0, 2]$ as given in equation 15.

$$\mathbf{a}^s = 2 \times \sigma(\mathbf{z}^s) \quad (15)$$

where $\mathbf{z}^s$ is the SD parameter vector for speaker s and these $\mathbf{z}^s$ parameter values are estimated for the test speaker using the adaptation data.

It is also possible to estimate these $\mathbf{z}^s$ parameters from a subspace as done in the subspace LHUC method [23]:

$$\mathbf{z}^s = \mathbf{U}\mathbf{v}^s \quad (16)$$

where $\mathbf{v}^s$ is a low-dimensional vector for speaker s, and $\mathbf{U}$ is the connecting weight matrix and is learned using the training data. Furthermore, this method reduces the per-speaker footprint considerably as $|\mathbf{v}^s| \ll |\mathbf{z}^s|$.

In all the experiments, we use the i-vector as the low-dimensional representation $\mathbf{v}^s$, however, it is possible to use other representations like bottleneck vectors [24, 25].

5. Experiment Setup

In this paper, we use the AMI corpus which contains about 100 hours of meetings conducted in English. The speech is recorded by multiple microphones, including one IHM and a uniform microphone circular array. In the experiments, we use the IHM data and the speech from the first microphone in the array which is known as the SDM. We use the ASR split [26] of the corpus where 78 hours of the data are used for training while about 9 hours each are used for evaluation and development. We use 90% of the training set for training, and the rest is used as the validation set. The results are reported on the evaluation set.

For both IHM and SDM datasets, we extract the Mel-frequency cepstral coefficients (MFCCs) from the speech using a 25 ms window and a 10 ms frame shift. Then the linear discriminant analysis (LDA) features are obtained by first splicing 7 frames of 13-dimensional MFCCs and then projecting downwards to 40 dimensions using LDA. A single semi-tied covariance (STC) transformation [27] is applied on top of the LDA

Table 1: *Word error rates (WER %) for various baseline models.*

Model	IHM	SDM
DNN Baseline	29.0	56.1
LSTM Baseline	28.1	52.3

Table 2: *IHM : WER % for various models with SD bias connected to different parts of the first LSTMP layer.*

SD bias	First Pass	Second Pass	# Speaker Params
Gates ($\mathbf{b}_i^s, \mathbf{b}_f^s, \mathbf{b}_o^s$)	27.4	26.5	300
Cell Input ($\mathbf{b}_c^s$)	27.2	26.2	100
Projection ($\mathbf{b}_p^s$)	27.3	26.5	100

features. The GMM-HMM system for generating the alignments for DNNs and LSTM-RNNs is trained on these 40 dimensional LDA+STC features. We train the DNN-HMM baselines on the LDA+STC features that span a context of 11 neighboring frames. Before being presented to the DNN, cepstral mean variance normalization (CMVN) is performed on the features globally. DNNs have 6 sigmoid hidden layers with 2048 units per layer, and around 4000 senones as the outputs. We trained LSTM-RNNs with 3 LSTM layers of 1024 memory cells and a 512 dimensional projection as in [28]. The input feature is a single frame with a 5 frames shift. For the training, we use truncated back propagation through time (BPTT) with sequences of 20 frames. We process 40 sequences in parallel.

All the models are trained to optimize the cross-entropy criterion using CNTK [29]. Kaldi [30] is used to build GMM-HMM systems and for i-vector extraction. The UBM consists of 128 full Gaussians. For decodings, we use the trigram language model as used in Kaldi, which is an interpolation of trigram language models trained on AMI and Fisher English transcripts.

6. Results

Table 1 shows the results for the baseline DNNs and LSTM-RNNs models trained on the IHM and SDM tasks. For both tasks, LSTM-RNN outperforms the corresponding DNN. The relative gain (3.2%) of the LSTM-RNN trained on IHM is slightly worse than the relative gain (6.8%) of the LSTM-RNN trained on SDM. This can be because LSTM-RNN is more beneficial for the noisy distant microphone speech in the SDM task.

Next, we investigate where to connect the SD bias of the LSTMP layer (Table 2). In these experiments, we only connect the SD bias to the first LSTMP layer. As shown in [16], it is sufficient to connect the SD bias only to the first hidden layer. During the first pass, speaker representations of the test speakers are initialized to the i-vector while during the second pass we update the i-vector for test speakers in unsupervised adaptation fashion as it was shown to be beneficial in [7]. We connected the SD bias to the gates (input, forget, output), cell input activation as well as the projection. Note that, in our models no global bias for the projection is used ($\mathbf{b}_p = 0$). The second pass always improves the performance significantly. The best result is reported when the SD bias is connected to the cell input activation. This is consistent with the findings in the speaker code adaptation of BLSTM models [13]. Therefore, we investigate the effectiveness of SD transformations by connecting them to cell input activation.

In Table 3, we present the results when SD transformations are estimated on the first LSTMP layer. We investigate the effect of SD transformations estimated on the input feature as well as the recurrence. "N" is used to denote that the SD transformation is not estimated while "Y" denotes the presence of the respective SD transformation. For all the models in Table 3, an SD bias is connected to the cell input activation of the first layer. Therefore, the first row of the Table 3 denotes the SD bias only model. As can be seen, the improvement of the SD transformation is small. Furthermore, estimating an SD transformation on the recurrence seems not beneficial. Therefore, in rest of the experiments, SD transformations are only estimated on the input features ($\mathbf{W}_{xc}^s$).

Table 3: *IHM : WER % for various models with SD transformations of input features and / or of recurrence, which is connected to the cell input of the first LSTMP layer.*

Input ($\mathbf{W}_{xc}^s$)	Recurrence ($\mathbf{W}_{rc}^s$)	First Pass	Second Pass
N	N	27.2	26.2
Y	N	27.1	26.1
N	Y	27.1	26.2
Y	Y	27.1	26.1

Table 4: *IHM : WER % for various models with SD transformation connected to different layers of the LSTM-RNN. The number of bases in the SD transformation is given in brackets.*

Layer (#bases)	First Pass	Second Pass	#Speaker Params
1 (100)	27.1	26.1	200
2 (100)	27.0	26.1	200
3 (100)	27.1	26.2	200
2 (1000)	27.0	26.0	1100
2 (5000)	27.0	25.8	5100

In Table 4, we present the results when the SD transformation is connected to different layers of the LSTM-RNN. All models report similar performances. Since the first pass decoding is slightly better when the SD transformation is connected to the second layer, we investigate the effect of increasing the number of bases for that SD transformation. As can be seen in the last two rows of the table, increasing the number of bases for the SD transformation improve the performance slightly. Therefore, for the IHM task, increasing the adaptation power by increasing the number of adaptation parameters is useful.

Table 5 shows the second pass results for scaling and various combinations of different adaptation techniques. As can be seen, input gate scaling (26.6%) performs slightly better than the output gate scaling (26.9%). The scaling of the forget gate is not successful. The performance improves when the input gate scaling is combined with the SD bias and the SD transformation based adaptation. The best performance is obtained when the adaptation combines input gate scaling with the SD bias.

Next, we report the results for AMI SDM task in Table 6. When the SD bias is connected to the cell input activation, the performance is improved by absolute 1.4% over the LSTM-RNN baseline after the second pass adaptation. When the SD transformation is connected to the cell input activation of the second LSTMP layer, the performance is improved further by absolute 0.6% to 50.3%. However, increasing the number of bases of the SD transformation to 1000 degrades the performance for SDM. Similarly, when the cell input scaling is performed in combination with the SD bias and SD transformation, the performance slightly degrades to 50.5%. This is because estimating more adaptation parameters using unsupervised adap-

Table 5: *IHM : Second pass WER % for various combinations of different adaptation techniques.*

Model	WER	# Speaker Params
Baseline	28.1	-
+ Input gate Scale	26.6	3072
+ Output gate Scale	26.9	3072
+ Forget gate Scale	Diverged	3072
+ SD bias	26.2	100
+ Input gate Scale	25.7	3172
+SD trans	26.1	200
+ Input gate Scale	25.8	3272

Table 6: *SDM : WER % for various adaptation combinations.*

Model	First Pass	Second Pass
LSTM Baseline	52.3	-
+SD bias	51.3	50.9
+ SD trans (100)	51.0	50.3
+ Scaling	51.0	50.5
+ Subspace Scaling	50.9	50.2
+ SD trans (1000)	51.1	50.7

tation is more sensitive to the poor quality of the hypotheses. Therefore, for SDM, we perform subspace scaling as mentioned in Equation 16 to scale the input gate. The best performance of 50.2% is achieved when the adaptation is performed in combination with SD transformation, SD bias, and the input gate subspace scaling which is a 2.1% absolute improvement over the baseline.

Finally, in Table 7, we compare the effectiveness of unsupervised adaptation methods on DNN vs LSTM-RNN acoustic models. The results are reported on both IHM and SDM tasks. The baseline results are given in the row where the method is "None". For both tasks, LSTM-RNN model outperforms the corresponding DNN baseline. As can be clearly seen, the SD bias after the second pass of the adaptation, improves the performance consistently for DNNs as well as LSTM-RNNs on both IHM and SDM tasks. In addition, for both IHM and SDM, the gains we observe from the CMLLR features is considerably reduced for LSTM-RNNs in comparison when CMLLR is used with DNNs. For ease of comparison, our best performances of the adaptation are listed for both DNNs and LSTM-RNNs. The best performance for DNN adaptation is observed when FHL is used with 600 and 800 bases on IHM and SDM respectively. For IHM task, we get the best performance of LSTM-RNN adaptation (25.7%) when SD bias is combined with the input gate scaling which is worse than the performance of the best DNN result (25.1%). For the SDM task, the best performance of the LSTM-RNN (50.2%) is significantly better than the that of DNN (51.9%).

As can be clearly seen from Table 7, the relative gains of the LSTM-RNN adaptation is considerably smaller to that of DNN adaptation for both IHM and SDM tasks. Therefore, we can claim that the adaptation of LSTM-RNN is more difficult than adaptation of DNN. One of the reasons for this is that LSTM-RNNs are more complex models than the feedforward DNNs. This increased complexity of LSTM-RNNs makes adaptation more difficult. In addition, LSTM-RNNs may already capture and normalize the speaker characteristics. Moreover, the SD transformations may negatively affect the modeling of the temporal dependencies. Therefore, for IHM, the SD bias and the simple scaling of the input gate outperforms the FHL-based SD transformations.

Table 7: *The comparison of DNN vs LSTM-RNN adaptation results for both IHM and SDM. Relative improvement are given in the brackets.*

Method	Dataset	DNN	LSTM-RNN
None	IHM	29.0 (-)	28.1 (-)
SD bias	IHM	27.0 (6.9)	26.2 (6.8)
CMLLR	IHM	26.3 (9.3)	26.3 (6.4)
Best	IHM	25.1 (13.5)	25.7 (8.5)
None	SDM	56.1 (-)	52.3 (-)
SD bias	SDM	53.7 (4.3)	50.9 (2.7)
CMLLR	SDM	53.2 (5.2)	50.6 (3.3)
Best	SDM	51.9 (7.5)	50.2 (4.0)

As future work, we want to investigate the adaptation of residual memory networks (RMNs) which are more similar to feedforward DNNs and temporal dependencies can be modeled efficiently as good as LSTM-RNNs [31]. Furthermore, it is interesting to employ a student-teacher mechanism to learn the bases of the SD transformation using an FHL adapted DNN model. We believe that it may help to estimate the bases of the SD transformation more effectively. Furthermore, we want to investigate the adaptation of bidirectional LSTM-RNNs and bidirectional RMNs.

7. Conclusions

In this paper, we investigated the effect of FHL-based adaptation on LSTM-RNNs. Experimental results showed that when combined with scaling of LSTM cell states' outputs, SD transformations achieved 2.3% and 2.1% absolute improvements over the baseline LSTM systems for the AMI IHM and AMI SDM tasks respectively. Furthermore, we observed that adaptation of LSTM-RNN is more difficult than DNN adaptation. We also discussed possible reasons for that observation and the potential future directions for LSTM-RNN adaptation.

8. References

[1] J. Gauvain and C. Lee, "Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains," *IEEE Transactions on Speech and Audio Processing*, vol. 2, no. 2, pp. 291–298, 1994.

[2] C. Leggetter and P. 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.

[3] M. Gales, "Maximum likelihood linear transformations for HMM-based speech recognition," *Computer speech & language*, vol. 12, no. 2, pp. 75–98, 1998.

[4] T. Anastasakos, J. McDonough, R. Schwartz, and J. Makhoul, "A compact model for speaker-adaptive training," in *ICSLP*, vol. 2. ISCA, 1996, pp. 1137–1140.

[5] M. Gales, "Cluster adaptive training of hidden Markov models," *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 4, pp. 417–428, 2000.

[6] L. Samarakoon and K. Sim, "Learning factorized transforms for speaker normalization," in *ASRU*. IEEE, 2015.

[7] ——, "On combining i-vectors and discriminative adaptation methods for unsupervised speaker normalization in DNN acoustic models," in *ICASSP*. IEEE, 2016.

[8] 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 *ICASSP*. IEEE, 2013, pp. 7893–7897.

[9] V. Gupta, P. Kenny, P. Ouellet, and T. Stafylakis, "I-vector-based speaker adaptation of deep neural networks for french broadcast audio transcription," in *ICASSP*. IEEE, 2014, pp. 6334–6338.

[10] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny, "Speaker adaptation of neural network acoustic models using i-vectors," in *ASRU*. IEEE, 2013, pp. 55–59.

[11] O. Abdel-Hamid and H. Jiang, "Fast speaker adaptation of hybrid NN/HMM model for speech recognition based on discriminative learning of speaker code," in *ICASSP*. IEEE, 2013, pp. 7942–7946.

[12] D. Yu, , and L. Deng, *Automatic Speech Recognition - A Deep Learning Approach*. New York: Springer London, 2015.

[13] Z. Huang, J. Tang, S. Xue, and L. Dai, "Speaker adaptation of RNN-BLSTM for speech recognition based on speaker code," in *ICASSP*. IEEE, 2016, pp. 5305–5309.

[14] T. Tan, Y. Qian, D. Yu, S. Kundu, L. Lu, K. C. Sim, X. Xiao, and Y. Zhang, "Speaker-aware training of lstm-rnns for acoustic modelling," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 5280–5284.

[15] C. Liu, Y. Wang, K. Kumar, and Y. Gong, "Investigations on speaker adaptation of lstm rnn models for speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 5020–5024.

[16] L. Samarakoon and K. Sim, "Factorized hidden layer adaptation for deep neural network based acoustic modeling," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2016.

[17] P. Swietojanski, J. Li, and S. Renals, "Learning hidden unit contributions for unsupervised acoustic model adaptation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 8, pp. 1450–1463, 2016.

[18] I. McCowan, J. Carletta, W. Kraaij, S. Ashby, S. Bourban, M. Flynn, M. Guillemot, T. Hain, J. Kadlec, V. Karaiskos *et al.*, "The AMI meeting corpus," in *Proceedings of the 5th International Conference on Methods and Techniques in Behavioral Research*, vol. 88, 2005.

[19] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[20] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with lstm," *Neural computation*, vol. 12, no. 10, pp. 2451–2471, 2000.

[21] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber, "Learning precise timing with lstm recurrent networks," *Journal of machine learning research*, vol. 3, no. Aug, pp. 115–143, 2002.

[22] H. Sak, A. W. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling." in *Interspeech*, 2014, pp. 338–342.

[23] L. Samarakoon and K. Sim, "Subspace LHUC for fast adaptation of deep neural network acoustic models," in *Interpeech*. ISCA, 2016.

[24] H. Huang and K. Sim, "An investigation of augmenting speaker representations to improve speaker normalization for DNN-based speech recognition," in *ICASSP*. IEEE, 2015, pp. 4610–4613.

[25] S. Kundu, G. Mantena, Y. Qian, T. Tan, M. Delcroix, and K. Sim, "Joint acoustic factor learning for robust deep neural network based automatic speech recognition," in *ICASSP*. IEEE, 2016.

[26] P. Swietojanski, A. Ghoshal, and S. Renals, "Hybrid acoustic models for distant and multichannel large vocabulary speech recognition," in *ASRU*. IEEE, 2013, pp. 285–290.

[27] M. Gales, "Semi-tied covariance matrices for hidden Markov models," *IEEE Transactions on Speech and Audio Processing*, vol. 7, no. 3, pp. 272–281, 1999.

[28] Y. Zhang, G. Chen, D. Yu, K. Yaco, S. Khudanpur, and J. Glass, "Highway long short-term memory rnns for distant speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 5755–5759.

[29] D. Yu, A. Eversole, M. Seltzer, K. Yao, Z. Huang, B. Guenter, O. Kuchaiev, Y. Zhang, F. Seide, H. Wang *et al.*, "An introduction to computational networks and the computational network toolkit," Tech. Rep. MSR, Microsoft Research, 2014, http://codebox/cntk, Tech. Rep., 2014.

[30] D. Povey, A. Ghoshal, G. Boulianne, N. Goel, M. Hannemann, Y. Qian, P. Schwarz, and G. Stemmer, "The kaldi speech recognition toolkit," in *ASRU*. IEEE, 2011.

[31] M. Baskar, M. Karafit, L. Burget, K. Vesel, F. Grzl, and J. H. ernock, "Residual memory networks: Feed-forward approach to learn long-term temporal dependencies," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017.