SPEAKER ADAPTATION OF RNN-BLSTM FOR SPEECH RECOGNITION BASED ON SPEAKER CODE

Zhiying Huang, Jian Tang, Shaofei Xue, Lirong Dai

National Engineering Laboratory of Speech and Language Information Processing,
University of Science and Technology of China, Hefei, P. R. China
Email: {zyhuang,enjtang,xuesf}@mail.ustc.edu.cn, lrdai@ustc.edu.cn

ABSTRACT

Recently, recurrent neural network with bidirectional Long Short-Term Memory (RNN-BLSTM) acoustic model has been shown to give great performance on the TIMIT [1] and other speech recognition tasks. Meanwhile, the speaker code based adaptation method has been demonstrated as a valid adaptation method for Deep Neural Network (DNN) acoustic model [2]. However, whether the speaker code based adaptation method is also valid for RNN-BLSTM has not been reported to the best our knowledge. In this paper, we study how to conduct effective speaker code based speaker adaptation on RNN-BLSTM and demonstrate that the speaker code based adaptation method is also a valid adaptation method for RNN-BLSTM. Experimental results on TIMIT have shown that the adaptation of RNN-LSTM can achieve over 10% relative reduction in phone error rate (PER) compared to without adaptation. Then, a set of comparative experiments are implemented to analyze the different contribution of the adaptation on *cell input* and each *gate* activation function of the BLSTM. It's found that the adaptation on *cell input* activation function is more effective than the adaptation on each *gate* activation function.

Index Terms— RNN-BLSTM, Speaker Adaptation, Speaker Code, Activation Function

1. INTRODUCTION

Speaker adaptation techniques aim to optimize the performance of speech recognition system for the target speaker or a group of speakers. It can be realized by either transforming a pre-trained speaker-independent (SI) model to match the target speaker or modifying the target speaker features to match the pre-trained SI system with the adaptation data of the target speaker. There are many effective techniques in speaker adaptation for conventional HMM/GMM acoustic model, such as MAP [3][4], MLLR [5][6], and CMLLR [7]. Recently, there are many speaker adaptation techniques proposed that have shown effective in hybrid NN/HMM. For example, Linear Input Network (LIN [8]), Linear Hidden Network (LHN [9]) and Linear Output Network (LON [10]) all attempt to add additional transforming layers to the initial SI neural networks. These methods alleviate over-fitting to some extent. In [11][12], the neural networks adapt the hidden activation functions towards the target speaker. Another way in [13], the technique that uses Kullback-Leibler (KL) divergence as regularization in the training criterion adapts the model conservatively by forcing the senone distribution estimated from the adapted model to be close to that from the unadapted model, by which it can avoid over-fitting during adaptation. In [14][15][16], they utilize speaker i-vector features to facilitate feature-space or model-space speaker adaptation. In [17], singular value decomposition (SVD) on the weight matrices in pre-trained SI DNN was applied, and then the singular value for the target speaker was tuned with adaptation data to realize adaptation. Similarly, SVD method is also used in [18], and adaptation is then performed by updating a square matrix inserted between two low-rank matrices. To enhance the robustness of adaptation, [19][20] estimate the adaptation parameters via maximum a posteriori (MAP) linear regression, which naturally incorporates prior knowledge into the adaptation process. Recently, in [2][21][22][23][24], several fast speaker adaptation methods on DNN and CNN based on the so-called speaker code have been proposed, which have been shown a promising adaptation method in speaker adaptation.

However, whether the speaker code based adaptation method is also valid for RNN-BLSTM has not been reported to the best our knowledge. In this paper, we've studied how to conduct speaker code based adaptation on RNN-BLSTM in small-scale TIMIT speech database, and proved its validity in adaptation for RNN-BLSTM. We first conduct speaker code based adaptation on *cell input* activation function of the BLSTM and compare the performance effect of different speaker code sizes. Then, we've found the code size that yields the best performance was much larger than DNN [21]. Furthermore, there are two adaptation weights which respectively connect speaker code to two different directions, *forward* and *backward*, in each layer of RNN-BLSTM. As a consequence, larger speaker code size and more adaptation weights introduce more parameters than DNN. To address this issue, we make a share of the *forward* and *backward* adaptation weights in each layer of RNN-BLSTM to decrease about half of the introduced parameters for speaker adaptation without a significant loss in final recognition accuracy. Secondly, we investigate different contribution of the adaptation on *cell input* and each *gate* activation function of the BLSTM. It's found that the adaptation on *cell input* activation function is more effective than the adaptation on each *gate* activation function.

2. REVIEW OF RNN-BLSTM

Normally, for a length T input vector $x = (x^1, x^2, ..., x^T)$, a conventional recurrent neural network (RNN) computes its hidden active vector $h = (h^1, h^2, ..., h^T)$ and output vector $y = (y^1, y^2, ..., y^T)$ given t from 1 to T are as follows:

$$h^t = \sigma(W_{xh}x^t + W_{hh}h^{t-1} + b_h) \quad (1)$$

$$y^t = W_{hy}h^t + b_y \quad (2)$$

Where W_{xh},W_{hh} and W_{hy} denote weight matrices, b_h and b_y denote bias vectors, and σ denotes sigmoid activation function in the hidden nodes.

However, in practice, RNN can't model the long term information due to the vanishing gradient and exploding gradient problems as described in [25]. One effective method to address these problems is by using Long Short-Term Memory (LSTM) architecture [26], which uses *memory cells* to store information and gets something useful. Fig. 1 illustrates a LSTM memory cell, which contains one self-connected cell and three controlling gates (the red lines indicate time-delayed connections). In the memory cell, *input gate* and *output gate* manage information flow into and out of the memory cell. Meanwhile, the *forget gate* [27] is used to provide a way for the cell to reset themselves. Furthermore, there are peephole weights connecting the gates to the cell, which is used for obtaining more accurate Constant Error Carousel (CEC) information [28]. The equations of one memory cell are as follows:

$$i^t = \sigma(W_{xi}x^t + W_{hi}h^{t-1} + W_{ci}c^{t-1} + b_i) \quad (3)$$
$$f^t = \sigma(W_{xf}x^t + W_{hf}h^{t-1} + W_{cf}c^{t-1} + b_f) \quad (4)$$
$$a^t = \tanh(W_{xc}x^t + W_{hc}h^{t-1} + b_c) \quad (5)$$
$$c^t = f^t \odot c^{t-1} + i^t \odot a^t \quad (6)$$
$$o^t = \sigma(W_{xo}x^t + W_{ho}h^{t-1} + W_{co}c^t + b_o) \quad (7)$$
$$h^t = o^t \odot \tanh(c^t) \quad (8)$$

Where i, f, o and a are respectively the input gate, forget gate,

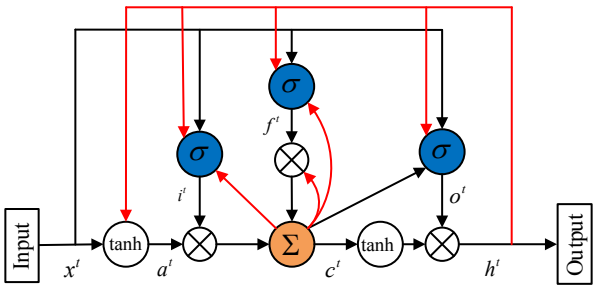

Fig. 1. The architecture of LSTM network with a memory cell [29].

output gate and cell input activation vector, c is a self-connected state vector, all of which are the same size as the hidden vector h. W_{ci}, W_{cf} and W_{co} are peephole connection weights which are diagonal, so element m in each gate vector only receives input from element m of the cell vector.

Nevertheless, RNN with conventional LSTM is unidirectional, can not model the future context. RNN with BLSTM [30] does model the future context by processing the input vector sequence in both *forward* and *backward* directions, with two different hidden weights in the same hidden layer. Generally, RNN-BLSTM may achieve better performance of speech recognition than RNN with conventional LSTM.

3. SPEAKER CODE BASED ADAPTATION OF RNN-BLSTM

As illustrated in Fig. 2, the structure of speaker code based adaptation model proposed in [2] can be regarded as a common model

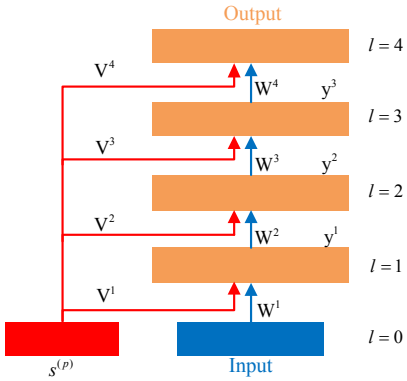

Fig. 2. The structure of speaker code based adaptation model (n = 5 layers).

adaptation structure for different neural network (NN) models, such as DNN as in [2] and RNN-BLSTM to be studied in this paper. $W^{(l)}$ represent all l-th layer weights of NN, $V^{(l)}$ are the connection weights between the speaker codes and l-th layer. $s^{(p)}$ stands for the speaker code that belongs to p-th speaker. For simplicity, taken DNN for example, the propagation equation is as follows:

$$y^{(l)} = \sigma(W^{(l)}y^{(l-1)} + b^{(l)} + V^{(l)}s^{(p)}) \quad (9)$$

Where $y^{(l)}$ denotes the hidden active vector of l-th hidden layer, and $b^{(l)}$ is the bias vector.

As is indicated in eq.(9), speaker code based adaptation can be seen as a kind of speaker adaptation on activation function, in which it biases the inputs of activation function towards the target speakers. For a BLSTM memory cell, it contains one self-connected cell and three controlling gates that control the transmission of the information flow by their corresponding activation functions. As we can see from the equations of one memory cell, there are three kinds of activation functions, *cell input* (eq.(5)), *cell output* (eq.(8)) and *gate* (eqs.(3), (4) and (7)). Especially, the input of the *cell output* activation function is c^t (see eq.(6)), which is computed by f^t, c^{t-1}, i^t and a^t separately from forget gate activation, time-delayed of self-connected state, input gate activation and cell input activation. As a result, the bias of the *cell output* activation function is equivalently included in f^t, c^{t-1}, i^t and a^t. Hence, we don't carry out speaker adaptation on *cell output* activation function.

The joint speaker adaptive training based on speaker codes (SAT-SC) method in [2] is also adopted in this paper. In SAT-SC, the weights $W^{(l)}$ is initiated by a pre-trained one, $s^{(p)}$ and $V^{(l)}$ are all initiated randomly. During training, $W^{(l)}$, $s^{(p)}$ and $V^{(l)}$ are jointly learned using the back propagation (BP) algorithm, where speaker labels are also available for training the speaker codes of all speakers in the training set. During adaptation, supervised method is utilized, where a small number of labeled adaptation utterances are available for each new test speaker. In this phase, $W^{(l)}$ and $V^{(l)}$ remain unchanged, the new speaker code is updated based on the SGD algorithm until convergence. After learning the new speaker code for each test speaker, a speaker-dependent (SD) neural network for each test speaker is reconstructed while testing.

3.1. Speaker Adaptation in Cell Input Activation Function (SA-CIAF)

From the above analysis, we realize that the *cell input* activation function in eq.(5) of BLSTM may play a similar role to DNN hidden activation function in speaker code based adaptation. As a first trial, we choose to do Speaker Adaptation in Cell Input Activation Function (SA-CIAF). In SA-CIAF, eq.(5) is modified into eq.(10):

$$a^t = \tanh(W_{xc}x^t + W_{hc}h^{t-1} + b_c + V_c s^{(p)}) \quad (10)$$

Where V_c denotes the adaptation weights for *cell input* which connect speaker code of p-th speaker, $s^{(p)}$, to current layer.

Frame-level cross entropy (CE) criterion is adopted as the objection function E for SAT-SC of RNN-BLSTM. The gradients of $V_{c_{kj}}$ (the weight that connects between the k-th element in the speaker code and the j-th node in current hidden layer) can be computed as:

$$\frac{\partial E}{\partial V_{c_{kj}}} = \frac{\partial E}{\partial a_j^t}(1 - a_j^t a_j^t)s_k^{(p)} \quad (11)$$

Where a_j^t is j-th element of a^t.

Likewise, the gradient of E with respect to each element of all speaker codes can be computed based on the chain rule. As the propagation errors from all layers contribute to the derivative of $s^{(p)}$, the gradient accumulates across n-1 layers (except input layer) and J nodes as follows:

$$\frac{\partial E}{\partial s_k^{(p)}} = \frac{1}{n-1}\sum_{l=1}^{n-1}\sum_{j=1}^{J}\frac{\partial E}{\partial a_j^t}(1 - a_j^t a_j^t)V_{c_{kj}} \quad (12)$$

3.2. Speaker Adaptation in Gate Activation Function (SA-GAF)

In this section, we discuss how to implement Speaker Adaptation in Gate Activation Function (SA-GAF). The propagation functions of the three gates in eqs.(3), (4) and (7) for SA-GAF can be modified respectively as:

$$i^t = \sigma(W_{xi}x^t + W_{hi}h^{t-1} + W_{ci}c^{t-1} + b_i + V_i s^{(p)}) \quad (13)$$

$$f^t = \sigma(W_{xf}x^t + W_{hf}h^{t-1} + W_{cf}c^{t-1} + b_f + V_f s^{(p)}) \quad (14)$$

$$o^t = \sigma(W_{xo}x^t + W_{ho}h^{t-1} + W_{co}c^t + b_o + V_o s^{(p)}) \quad (15)$$

Where V_i, V_f and V_o represent three kinds of adaptation weights separately for the adaptation of the *input gate*, *forget gate* and *output gate*.

If we define g represents i or f or o, the gradient of E with respect to each element of all speaker codes, $s^{(p)}$, and three kinds of adaptation weights, V_i, V_f and V_o, may be computed as follows:

$$\frac{\partial E}{\partial V_{g_{kj}}} = \frac{\partial E}{\partial g_j^t}(1 - g_j^t)g_j^t s_k^{(p)} \quad (16)$$

$$\frac{\partial E}{\partial s_k^{(p)}} = \frac{1}{n-1}\sum_{l=1}^{n-1}\sum_{j=1}^{J}\frac{\partial E}{\partial g_j^t}(1 - g_j^t)g_j^t V_{g_{kj}} \quad (17)$$

Where $V_{g_{kj}}$ is the connection weight between the k-th element in the speaker code and j-th node in current layer, and g_j^t is the j-th element of g^t.

Generally, RNN with BLSTM [30] does model the future context by processing the input vector sequence in both *forward* and *backward* directions, with two different hidden weights in the same hidden layer. So that the adaptation weights of the *forward* and *backward* directions in the same hidden layer are different, but their derivatives are computed in the same way, which are described in eq.(11) for SA-CIAF or in eq.(16) for SA-GAF.

4. EXPERIMENTS

In this section, we evaluate the adaptation scheme for RNN-BLSTM discussed in section 3 on TIMIT phone recognition task.

We use the standard 462-speaker training set and remove all SA records (i.e., identical sentences for all speakers in the database) since they may bias the results. A separate development set of 50 speakers is used for tuning all the meta parameters. Results are reported using the 24-speaker core test set, which has no overlap with the development set. Each speaker in the test set has eight utterances. The 39 dimensional PLP features (static, first and second derivatives) are extracted and 183 target class labels (3 states for each one of the 61 phones) are labeled for the utterances in training set and development set utterances by GMM-HMM. After decoding, the 61 phone classes were mapped to a set of 39 classes for scoring purpose [31]. In our experiments, a bi-gram language model in phone level, estimated from the training set, is used in decoding.

Before SAT-SC training of RNN-BLSTM, we first pre-train a SI RNN-BLSTM baseline as initialization of the W weights (see Fig. 2), and initiate speaker codes s and adaptation weights V drawn uniformly from [-0.1, 0.1]. During SAT-SC training, the learning rate for s and V fixed at 0.2 and for W fixed at 0.0001 for 3 iterations, and is halved for another 2 iterations, the momentum is kept as 0.9. During adaptation, we use a fixed learning rate of 0.2 to learn speaker code of each target speaker. Since each test speaker has eight utterances in total. Adaptation and testing is conducted for each speaker based on cross validation (CV). In each CV run, for each speaker, eight utterances are divided into 7 utterances for adaptation and the remaining 1 utterance for testing. This is repeated eight times for each speaker. The overall recognition performance is the average of all eight runs.

1) Performance of SA-CIAF with Different Speaker Code Sizes: In this experiment, we've trained a SI RNN-BLSTM model with the size of 3*250 (3 hidden layers, 250 BLSTM memory cells for each layer) as baseline, it achieved PER of 20.9% as illustrated in Table 1. The performance of SA-CIAF with different speaker code sizes (varying from 300 to 2000) is also shown in Table 1. The results in Table 1 show that adaptation performance is not very sensitive to speaker code size when given it from 500 to 2000 (PER varying from 18.8% to 19.3%). However, the performance of speaker code size of 300 is 19.9% in PER (with the relative phone error reduction of 4.78%). This is probably because the speaker code size of 300 is too small to model the information of the target speaker. Besides, SA-CIAF yields the best performance (with the relative phone error reduction of 10.05%) with the speaker code size of 1500.

Table 1. *PERs (in %) of SA-CIAF on RNN-BLSTM (3*250) with different speaker code sizes.*

SC size	baseline	SA-CIAF
300	20.9	19.9
500		19.3
1000		19.2
1500		**18.8**
2000		19.0

It is doubt that whether the improved performance is due to the increased model complexity in SA-CIAF. To clear this doubt out, we've built another bigger SI RNN-BLSTM baseline with the size of 3*326 (3 hidden layers, 326 BLSTM memory cells for each layer), which has roughly the same number of model parameters (6.2 millions) as the one in SA-CIAF. The performance of this larger

baseline is shown in Table 2, which is 20.5% in PER. This indicates that the performance gain of SA-CIAF mainly comes from speaker adaptation not from more model parameters.

Table 2. *Comparison of PERs (in %) of two SI RNN-BLSTM baselines with SA-CIAF.*

RNN-BLSTM	baseline	SA-CIAF
3*250	20.9	18.8
3*326	20.5	-

On the other hand, we try to decrease the number of parameters due to large speaker code size by sharing the *forward* and *backward* adaptation weights. As we can see from Table 3, the performance loss in PER is not significant when the sharing reduces about half of the adaptation parameters.

Table 3. *Comparison of PERs (in %) of without sharing or with sharing the forward and backward adaptation weights of SA-CIAF (speaker code size = 1500).*

without / with sharing	baseline	SA-CIAF
without sharing	20.9	18.8
with sharing		19.2

2) Comparison Between SA-CIAF and SA-GAF: In this part, we first compare the adaptation performances of the joint adaptation of the *cell input* and the three *gate* activation functions at speaker code size of 500, 1000, 1500. The results are shown in Table 4, the speaker code size of 1500 still yields the best performance (PER in 19.1%) in joint train of four kinds of adaptation weights, which is worse than SA-CIFA (PER in 18.8%) and probably is due to the limited data of each speaker for the learning of four times the number of parameters in SA-CIAF.

Table 4. *PERs (in %) of joint adaptation of the cell input and three gate activation functions with different speaker code sizes.*

SC size	baseline	joint adaptation
500	20.9	19.6
1000		19.4
1500		19.1

Secondly, experiments are conducted to compare the adaptation contribution of the *cell input* and each *gate* activation function, while keeping speaker code size at 1500. Results in Table 5 show that the adaptation of *cell input* activation function is more effective in performance than the adaptation of each *gate* activation function. This is mainly due to the different functions between the *cell input* activation function and the *gate* activation functions for a BLSTM memory cell. The *cell input* activation function collects the information of input and recurrent to decide the activation of current layer, but the *input gate*, *forget gate* and *output gate* activation functions are used for providing continuous analogues of read, reset and write operations for the cells [28]. In other words, the *cell input* activation function is a master controller, but the three *gate* activation functions are secondary controllers, and these four activation functions control the information flow synergistically. Hence, it's reasonable that the adaptation of *cell input* activation function can model more information of the target speaker than the adaptation of each *gate* activation function.

Table 5. *PERs (in %) of speaker adaptation on cell input and three gate activation functions respectively.*

model	activation function	baseline	PER
SA-CIAF	cell input	20.9	18.8
SA-GAF	input gate		20.9
	forget gate		20.5
	output gate		20.5

5. CONCLUSION

In this paper, we've studied how to conduct effective speaker adaptation of RNN-BLSTM on TIMIT phone recognition task. Experimental results have shown that the adaptation of RNN-BLSTM can achieve over 10% relative reduction in PER compared to without adaptation, which is better than the adaptation of DNN in [21] (about 7.4% relative error reduction). Then, it's found that *cell input* activation function adaptation is more effective than the adaptation on each *gate* activation function.

In the future, we will attempt to use a smaller speaker code size to reduce the computation complexity. On the other hand, the log Mel-filterbank features show better performance than the PLP features in speech recognition to the best our knowledge. We will implement the unsupervised speaker adaptation of RNN-BLSTM on 320-h Switchboard task by using the log Mel-filterbank features and try to apply MMI criterion based sequence training to the speaker adaptation of RNN-BLSTM.

6. ACKNOWLEDGMENT

We acknowledge the support of the following organizations for research funding: National Nature Science Foundation of China (Grant No.61273264), Science and Technology Department of Anhui Province (Grant No.15CZZ02007), Chinese Academy of Sciences (Grant No.XDB02070006).

7. REFERENCES

[1] Alan Graves, Navdeep Jaitly, and Abdel-rahman Mohamed, "Hybrid speech recognition with deep bidirectional lstm," in *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*. IEEE, 2013, pp. 273–278.

[2] Shaofei Xue, Ossama Abdel-Hamid, Hui Jiang, Lirong Dai, and Qingfeng Liu, "Fast adaptation of deep neural network based on discriminant codes for speech recognition," *Audio, Speech, and Language Processing, IEEE/ACM Transactions on*, vol. 22, no. 12, pp. 1713–1725, 2014.

[3] Jean-Luc Gauvain and Chin-Hui Lee, "Maximum a posteriori estimation for multivariate gaussian mixture observations of markov chains," *Speech and audio processing, ieee transactions on*, vol. 2, no. 2, pp. 291–298, 1994.

[4] SM Ahadi and Philip C Woodland, "Combined bayesian and predictive techniques for rapid speaker adaptation of continuous density hidden markov models," *Computer speech & language*, vol. 11, no. 3, pp. 187–206, 1997.

[5] 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.

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

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

[8] Joao Neto, Luís Almeida, Mike Hochberg, Ciro Martins, Luis Nunes, Steve Renals, and Tony Robinson, "Speaker-adaptation for hybrid hmm-ann continuous speech recognition system," 1995.

[9] Roberto Gemello, Franco Mana, Stefano Scanzio, Pietro Laface, and Renato De Mori, "Linear hidden transformations for adaptation of hybrid ann/hmm models," *Speech Communication*, vol. 49, no. 10, pp. 827–835, 2007.

[10] Bo Li and Khe Chai Sim, "Comparison of discriminative input and output transformations for speaker adaptation in the hybrid nn/hmm systems," 2010.

[11] Sabato Marco Siniscalchi, Jinyu Li, and Chin-Hui Lee, "Hermitian polynomial for speaker adaptation of connectionist speech recognition systems," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 21, no. 10, pp. 2152–2161, 2013.

[12] Yong Zhao, Jinyu Li, Jian Xue, and Yifan Gong, "Investigating online low-footprint speaker adaptation using generalized linear regression and click-through data," .

[13] 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 *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 7893–7897.

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

[15] Vishwa Gupta, Patrick Kenny, Pierre Ouellet, and Themos Stafylakis, "I-vector-based speaker adaptation of deep neural networks for french broadcast audio transcription," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 6334–6338.

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

[17] Shaofei Xue, Hui Jiang, and Lirong Dai, "Speaker adaptation of hybrid nn/hmm model for speech recognition based on singular value decomposition," in *Chinese Spoken Language Processing (ISCSLP), 2014 9th International Symposium on*. IEEE, 2014, pp. 1–5.

[18] Jian Xue, Jinyu Li, Dong Yu, Mike Seltzer, and Yifan Gong, "Singular value decomposition based low-footprint speaker adaptation and personalization for deep neural network," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 6359–6363.

[19] Zhen Huang, Jinyu Li, Sabato Marco Siniscalchi, I-Fan Chen, Chao Weng, and Chin-Hui Lee, "Feature space maximum a posteriori linear regression for adaptation of deep neural networks," in *Proc. INTERSPEECH*, 2014, pp. 2992–2996.

[20] Zhen Huang, Sabato Marco Siniscalchi, I-Fan Chen, Jiadong Wu, and Chin-Hui Lee, "Maximum a posteriori adaptation of network parameters in deep models," *arXiv preprint arXiv:1503.02108*, 2015.

[21] Shaofei Xue, Ossama Abdel-Hamid, Hui Jiang, and Lirong Dai, "Direct adaptation of hybrid dnn/hmm model for fast speaker adaptation in lvcsr based on speaker code," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 6339–6343.

[22] Shaofei Xue, Hui Jiang, Lirong Dai, and Qingfeng Liu, "Unsupervised speaker adaptation of deep neural network based on the combination of speaker codes and singular value decomposition for speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 4555–4559.

[23] Ossama Abdel-Hamid and Hui Jiang, "Fast speaker adaptation of hybrid nn/hmm model for speech recognition based on discriminative learning of speaker code," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 7942–7946.

[24] Ossama Abdel-Hamid and Hui Jiang, "Rapid and effective speaker adaptation of convolutional neural network based models for speech recognition.," in *INTERSPEECH*, 2013, pp. 1248–1252.

[25] Yoshua Bengio, Patrice Simard, and Paolo Frasconi, "Learning long-term dependencies with gradient descent is difficult," *Neural Networks, IEEE Transactions on*, vol. 5, no. 2, pp. 157–166, 1994.

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

[27] Felix Gers, "Long short-term memory in recurrent neural networks," *Unpublished PhD dissertation, École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland*, 2001.

[28] Alex Graves et al., *Supervised sequence labelling with recurrent neural networks*, vol. 385, Springer, 2012.

[29] Xiangang Li and Xihong Wu, "Improving long short-term memory networks using maxout units for large vocabulary speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 4600–4604.

[30] Alex Graves and Jürgen Schmidhuber, "Framewise phoneme classification with bidirectional lstm and other neural network architectures," *Neural Networks*, vol. 18, no. 5, pp. 602–610, 2005.

[31] Kai-Fu Lee and Hsiao-Wuen Hon, "Speaker-independent phone recognition using hidden markov models," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 37, no. 11, pp. 1641–1648, 1989.