JOINT ACOUSTIC FACTOR LEARNING FOR ROBUST DEEP NEURAL NETWORK BASED AUTOMATIC SPEECH RECOGNITION

Souvik Kundu[†], *Gautam Mantena*[†], *Yanmin Qian*[‡], *Tian Tan*[‡], *Marc Delcroix*[§], *Khe Chai Sim*[†]

[†]School of Computing, National University of Singapore, Republic of Singapore
[‡]Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China
[§]NTT Communication Science Laboratories, NTT Corporation, Kyoto, Japan
[†]{souvik,gautam,simkc}@comp.nus.edu.sg, [‡]{yanminqian,tantian}@sjtu.edu.cn, [§]marc.delcroix@lab.ntt.co.jp

ABSTRACT

Deep neural networks (DNNs) for acoustic modeling have been shown to provide impressive results on many state-of-the-art automatic speech recognition (ASR) applications. However, DNN performance degrades due to mismatches in training and testing conditions and thus adaptation is necessary. In this paper, we explore the use of discriminative auxiliary input features obtained using joint acoustic factor learning for DNN adaptation. These features are derived from a bottleneck (BN) layer of a DNN and are referred to as BN vectors. To derive these BN vectors, we explore the use of two types of joint acoustic factor learning which capture speaker and auxiliary information such as noise, phone and articulatory information of speech. In this paper, we show that these BN vectors can be used for adaptation and thereby improve the performance of an ASR system. We also show that the performance can be further improved on augmenting these BN vectors to conventional i-vectors. In this paper, experiments are performed on Aurora-4, REVERB challenge and AMI databases.

Index Terms— deep neural networks, joint factor learning, adaptation, bottleneck vectors, robust speech recognition.

1. INTRODUCTION

In the recent years, many state-of-the-art automatic speech recognition (ASR) systems employ deep neural networks (DNNs) for acoustic modeling. In [1, 2], it has been shown that the DNN-hidden Markov models (HMMs) outperform the conventional Gaussian mixture models (GMMs) - HMM systems. However, like many machine learning algorithms, DNN performance suffers due to mismatch between the training and the testing conditions. Thus adapting DNN acoustic models to unseen test conditions becomes an important research area to build robust ASR systems. Some of the adaptation techniques for GMM-HMM systems are maximum likelihood linear regression (MLLR) [3, 4], constrained MLLR (CMLLR) or feature domain MLLR (fMLLR) [5]. DNNs are discriminative models and thereby require adaptation techniques that are different from those developed for GMMs.

In [6–8], DNN adaptation is performed by adding a linear transformation layer to an existing DNN. This linear transformation layer can be added at the input, hidden or at the output layers. In this approach, only the parameters related to the linear transformation layer are updated. In [9], feature space discriminative linear regression (fDLR) is used for adaptation. As an alternative to this approach, one can also retrain all the parameters of a DNN. However, such an approach may destroy previously learned weights and biases. An approach to overcome this issue is to use regularization techniques during adaptation [10–12]. Adaptation can also be performed by using task specific information such as speaker or noise representative features as auxiliary inputs to a DNN training [13]. It is shown that DNN models trained in such an approach become independent to training conditions and are able to generalize better to unseen test conditions. Some of the feature representations explored for such an approach are speaker codes [14, 15], i-vectors [16] and speaker bottleneck (BN) vectors [17].

Typically, many existing adaptation approaches focus only on a single factor, e.g. either noise or speaker. However, there is an increased interest towards robust ASR approaches dealing with uncertainties due to multiple factors, which is a more realistic and a practical problem to address. Multi-factor adaptation was explored to handle speaker and environmental noise [18, 19] and for low-footprint speech applications [20]. In [21], separate speaker and phoneme neural networks were built and then joined at the BN layer level to enable discriminative joint factor analysis for speaker normalization.

In this paper, we explore feature representation obtained using joint acoustic factor learning. The two types of joint acoustic factor learning used in this paper are multi-task learning (MTL) and joint task training (JTL). In MTL, the network is trained to perform classification on two or more related tasks using a shared representation [22]. In general, in MTL, the parameters of the output layer are not shared and thus each output layer represents a task. In JTL, all the parameters are shared and we use only a single output layer to represent all the tasks. In this paper, we derive vectors from a BN layer of a DNN which is trained using MTL or JTL. These vectors are referred to as BN vectors. As mentioned earlier, BN vectors were also use in [17]. However, these BN vectors are obtained from a BN layer trained on speaker classes and are then expanded into super vectors by incorporating phoneme posterior probabilities. The main contributions of this paper are:

- We explore the use of two types of joint acoustic factor learning, namely multi-task and joint task learning, to derive BN vectors. In this approach, the DNN built to derive BN vectors are trained to capture speaker and other auxiliary information of speech such as noise, phoneme and articulatory information.
- We provide experimental results to show that the BN vectors obtained using joint acoustic factor learning perform better than that of the BN vectors obtained using only the speaker information.
- We also provide experimental results to show that these BN vectors can be used as a complementary information to the

i-vectors and thereby improve the performance of the ASR systems. Note that all the experiments reported in this paper are performed on Aurora-4, REVERB challenge and AMI databases.

The organization of this paper is as follows: Section 2 describes the two techniques used to derive BN vectors using joint acoustic factor learning. In Section 3, experiments are performed on Aurora-4 database using BN vectors derived from speaker and other auxiliary information of speech and a detailed experimental comparison of these vectors is provided. To validate the experimental results we also provide results on using BN vectors for adaptation on REVERB challenge and AMI databases in Sections 4 and 5 respectively.

2. USE OF JOINT ACOUSTIC FACTOR LEARNING TO DERIVE BOTTLENECK VECTORS

In this Section, we describe two types of joint acoustic factor learning to derive BN vectors for adaptation. The two types of joint acoustic factor learning techniques are multi-task learning (MTL) and joint task learning (JTL). In this paper, we place the bottleneck (BN) layer before the output layer [17]. For convenience, the DNNs used to derive BN vectors are referred to as BN-DNN.

In a conventional BN-DNN, the network is trained as a classifier for a single task and the network parameters are learned through optimizing the cross entropy objective function. We refer to this learning as single task learning (STL). For BN vectors obtained using STL, we use a BN-DNN that is trained to output speaker IDs as a one-hot vector.

To make use of multiple factors of speech to train a BN-DNN we employ MTL. In this work, we consider two sets of tasks and are represented as T_s and T_o. T_s represent the speaker task and T_o represent tasks which capture auxiliary information such as noise, phoneme or articulatory information. Let $|T_s|$ represents the number of speaker class labels and T_o represents the number of auxiliary class labels. For MTL, BN-DNN contains two output layers which are of $|T_s|$ and $|T_o|$ dimensions and share the BN layer. Note that for MTL, the BN-DNN outputs two one-hot vectors for each of the tasks. Let J_s and J_o be the cross entropy objective function for speaker and auxiliary tasks respectively. Then, the net objective function, J, is given as: $J = \alpha J_s + (1-\alpha)J_o$, where $\alpha \in [0, 1]$.

As an alternative to MTL, we also explore the use of JTL to train a BN-DNN. Like MTL, JTL also includes 2 tasks. However the difference is that there is only one output layer and each output class represents a combination of two factors, i.e. the number of output classes is equal to $|T_s| \times |T_o|$.

Note that the BN vectors obtained are at a frame level. In order to represent each utterance or a speaker with a single vector, we average all the BN vectors (per utterance or per speaker) and then perform a length normalization on these vectors. For simplicity, in all the future references in this paper, BN vectors refer to the averaged and length normalized vectors.

3. EXPERIMENTS ON AURORA-4 DATABASE

In this Section, we perform experiments on Aurora-4 [23] and the word error rate (WER) is reported. The acoustic model is trained on a window of 11 frames of 40 dimensional fMLLR features. These fMLLR features are obtained using all the data from a given speaker. The DNN model has a 440 dimensional input layer, 7 hidden layers of 2048 dimensions each and a 2013 dimensional output layer. Sigmoid activation functions are used for all the hidden units. All the acoustic models are trained using Kaldi [24] and CNTK [25].

There are 15.11 hours of speech in the multi-condition training set, 8.94 hours of speech in the development set and 9.37 hours of speech in the evaluation set. Note that the development set is used as cross validation during the network training. There are a total of 83 speakers in the training data, 10 speakers in development set and 8 speakers in the test data which are not matched. There are 14 noise conditions which are matched for the training and testing data.

Table 1. WER obtained using fMLLR, fMLLR + i-vector per speaker and fMLLR + i-vector per utterance on Aurora-4.

fMLLR	**WER (%)**				
	A	**B**	**C**	**D**	**Avg.**
baseline	2.6	5.7	4.8	15.3	9.6
+ i-vectors (per speaker)	2.5	5.7	4.4	14.4	9.2
+ i-vectors (per utt.)	2.6	5.6	4.3	14.3	**9.0**

Table 1 shows the WER of the baseline systems obtained using fMLLR in combination with the speaker and utterance level i-vectors. Aurora-4 consists of recordings from 2 different channels, namely Channel-1 (single microphone) and Channel-2 (sampling of 18 different microphones). The test dataset is divided into four subsets: *A* (clean speech + Channel-1), *B* (noisy speech + Channel-1), *C* (clean speech + Channel-2) and *D* (noisy speech + Channel-2). A GMM with 128 Gaussians is used as an universal background model to obtain 100 dimensional i-vectors. These i-vectors are obtained using fMLLR features. It can be seen that the i-vectors per utterance are performing better than that of the speaker level i-vectors. Thus, for Aurora-4, we use i-vectors obtained per utterance for all the experiments.

3.1. Use of BN Vectors for Adaptation

For Aurora-4, we use a 3 hidden layer BN-DNN network. The BN layer is of size 100 and it contains linear activation units. The other 2 hidden layers are of size 2048 and contain Sigmoid activation units. Note that to be consistent with the i-vector baseline we use BN vectors per utterance. The input to the BN-DNN is 40 dimensional fMLLR features with a context window of 11 frames.

As described earlier in Section 2, the primary task is speaker (SP) and the auxiliary tasks are noise (NS), context dependent (CD) units, context independent (CI) units and articulatory (AR) units. For convenience, we refer to the BN features obtained using MTL or JTL as SP-X, where X can represent NS, CD, CI or AR. In Sections 3.1.1-3.1.4, detailed experimental results for BN vectors obtained using STL, MTL and JTL are provided.

3.1.1. Speaker-Noise BN Vectors

Table 2 shows WER obtained using speaker and noise information to train the BN-DNN. Note that we use 84 classes for the speaker task (83 speaker classes + silence class), 14 classes for noise. For MTL, we use $\alpha = 0.5$ (as shown in Section 2). From Table 2, it can be seen that: (a) SP-NS BN vectors obtained using MTL perform similar to that of the SP BN vectors, (b) SP-NS BN vectors obtained using JTL performs better than that of SP BN and SP-NS (using MTL), (c) It can also be seen that i-vectors per utterance (as described in Table 1) perform better than that of SP-NS BN vectors using JTL.

In Section 3.1.2, we explore auxiliary tasks which use phone and articulatory information of speech. The results indicate that, for Aurora-4, phoneme or articulatory classes are a better choice to derive BN vectors for adaptation.

Table 2. WER obtained using fMLLR in combination with SP, SP-NS BN vectors on Aurora-4. Note that SP BN vector is obtained using STL and SP-NS BN vectors are obtained using MTL and JTL

fMLLR	Type	WER(%)				
		A	B	C	D	Avg.
+ SP BN	STL	2.5	5.6	4.6	14.8	9.3
+ SP-NS BN	MTL	2.5	5.6	4.7	14.8	9.3
+ SP-NS BN	JTL	2.6	5.6	4.6	14.4	**9.2**

3.1.2. Speaker-Phone and Speaker-Articulatory BN Vectors

In this Section, we explore the use of phoneme classes of speech as auxiliary information to derive BN vectors. We use 2013 CD units and 42 CI units. These 42 classes of CI units are further grouped into 24 classes based on the articulatory properties of speech sounds such as place and manner of articulation for consonants, duration and tongue position for vowels, fricatives, nasals and semi-vowels [26].

Table 3. WER obtained using fMLLR in combination with SP-CD, SP-CI and SP-AR BN vectors on Aurora-4. Note that SP-CD BN vectors are obtained using MTL and SP-(CI/AR) BN vector are obtained using MTL and JTL.

fMLLR	Type	WER (%)				
		A	B	C	D	Avg.
+ SP-CD BN	MTL	2.5	5.6	4.5	14.3	9.1
+ SP-CI BN	MTL	2.5	5.5	4.5	14.4	9.1
+ SP-AR BN	MTL	2.5	5.6	4.5	14.4	9.1
+ SP-CI BN	JTL	2.6	5.5	4.4	14.2	9.0
+ SP-AR BN	JTL	2.6	5.5	4.3	13.9	**8.8**

Table 3 shows the WER obtained using SP-(CD/CI/AR) BN vectors. Note that for SP-(CI/AR) BN vectors are obtained using MTL and JTL and SP-CD BN vectors can only be obtained using MTL. This is because JTL limits the number of classes and thereby the type of classes for training. From Table 3, it can be seen that the: (a) BN vectors obtained using JTL perform better than that of the BN vectors from MTL, (b) SP-AR BN vectors from JTL perform better than SP-(CD/CI) BN vectors. We use $\alpha = 0.5$ to obtain BN vectors using MTL. However, this value of α might not be an optimum value and thus in Section 3.1.3 we perform experiments to select the optimum value of α.

3.1.3. Effect of α on MTL Training

Fig. 1 shows WER obtained using BN vectors for various values of α. Note that for $\alpha = 1.0$, the training would be that of the STL with a WER of 9.32%. From Fig. 1, it can be seen that SP-NS BN vectors do not perform as well as the SP-(CD/CI/AR) BN vectors. Thus, for Aurora-4, phoneme classes are a better choice of auxiliary information to derive BN vectors using MTL. It can also be seen that SP-CI BN vectors perform slightly better than that of SP-CD and SP-AR BN vectors for $\alpha = 0.8$. Thus, we choose 0.8 to be the optimum value of α and consider SP-CI BN vectors as the optimum BN vectors using MTL for Aurora-4.

3.1.4. Comparison of BN vectors obtained using MTL and JTL

In this Section, we provide results to compare the BN vectors obtained using MTL and JTL. In order to compare we consider the best BN vectors for each of the learning methods, i.e., for MTL we use SP-CI BN vectors (as described in Section 3.1.3) and for JTL we use SP-AR BN vectors (as described in Section 3.1.2). We use $\alpha = 0.8$ to obtain SP-CI BN vectors.

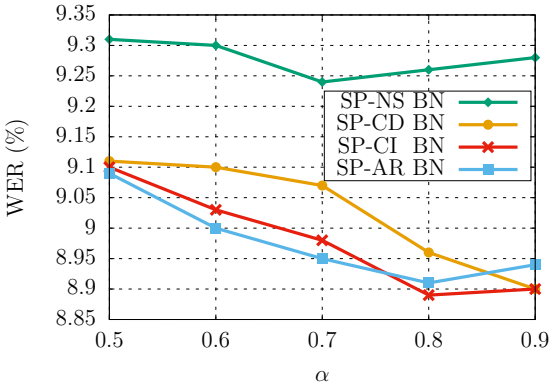

Fig. 1. WER obtained using different values of α to derive SP-(NS/CD/CI/AR) BN vectors. In this figure all the BN vectors are obtained using MTL on Aurora-4.

Table 4. WER obtained using fMLLR in combination with the SP-AR BN , SP-CI BN and i-vectors on Aurora-4 database.

fMLLR	WER (%)				
	A	B	C	D	Avg.
+ SP-AR BN (JTL)	2.6	5.5	4.3	13.9	8.8
+ i-vector + SP-AR BN (JTL)	2.6	5.5	4.0	13.6	**8.7**
+ SP-CI BN (MTL)	2.4	5.4	4.2	14.1	8.9
+ i-vector + SP-CI BN (MTL)	2.6	5.4	4.0	13.6	**8.7**

From Table 1 and Table 4 it can be seen that: (a) SP-AR and SP-CI BN vectors can be used to improve the performance over the i-vector baseline system, (b) SP-CI and SP-AR BN vectors can be used along with the i-vectors to further improve the performance. The performance gain is statistically significant for $p = 0.05$. This improvement is due to the fact that these BN vectors provide complementary information which is not being captured by the i-vectors. From Table 4, it can be seen that the BN vectors obtained using JTL and MTL in combination with the i-vectors have similar performance. However, JTL limits the number of classes and thereby the type of classes for training. Thus JTL poses an issue and can be overcome by using MTL training. Therefore SP-CI BN vectors can be assumed to be a better representation in cases where there are tasks with large number of output targets. In order, to further analyze the performance of SP-CI BN vectors, we perform evaluations on 2 more databases namely REVERB challenge and AMI (as described in Sections 4 and 5).

4. EVALUATIONS ON REVERB CHALLENGE DATABASE

The REVERB challenge database consists of training, development and evaluation datasets [27, 28]. The training data consists of reverberant and noisy speech created by simulation. The utterances are based on the WSJCAM0 5K corpus [29]. The test sets consists of simulated (SimData) and real recordings (RealData). RealData are part of the MC-WSJ-AV corpus [30]. In this paper, we report results only for the RealData set as it is more challenging. The training data

consists of 17.5 h from 92 speakers. The RealData development and evaluation sets consist of 0.3 h from 5 speakers and 0.6 h from 10 speakers, respectively. We use the development data for cross validation and WER is reported for the evaluation data. Refer to [28] for a detailed description of the REVERB challenge data.

In this experiment, we process the 8ch recordings and use weighted prediction error (WPE) algorithm to suppress dereverberation [31]. WPE performs utterance-based dereverberation in the STFT domain. This algorithm achieved high performance on the REVERB challenge [32]. To further suppress noise after dereverberation, we use a weighted delay-sum beamformer [33, 34]. Note that in the experiments below, fMLLR features are computed per speaker. It assumes a-priori knowledge of speaker ID information[1]. Consequently, the results with fMLLR features should be considered carefully when comparing with other results reported on the task that do not exploit such a-priori knowledge.

Table 5. WER obtained using fMLLR and LDA features in combination with SP BN, SP-CI BN and i-vectors on REVERB challenge evaluation dataset. Note that SP and SP-CI BN vectors are obtained using STL and MTL respectively.

System	WER (%)	
	fMLLR	LDA
baseline	20.6	31.4
+ dereverberation & beamforming	11.2	20.1
+ i-vectors	**10.8**	18.1
+ SP BN (STL)	11.3	17.4
+ SP-CI BN (MTL)	11.2	16.9
+ i-vectors + SP-CI BN (MTL)	11.0	**16.1**

For REVERB challenge database, we build two acoustic models using 40 dimensional fMLLR and linear discriminant analysis (LDA) features. The LDA features are obtained by projecting 13 dimensional Mel-frequency cepstral coefficients (with a context window of 7) to a 40 dimensional feature. The DNN and BN-DNN architectures used are similar to that of the models used for Aurora-4. 100 dimensional i-vectors are obtained using a GMM with 128 Gaussians.

Table 5 shows the WER on REVERB challenge evaluation dataset obtained using fMLLR and LDA features. We use 92 speaker classes and 42 CI units to obtain SP and SP-CI BN vectors. We use $\alpha = 0.8$ to obtain SP-CI BN vectors. On applying dereverberation and beamforming, there is a significant improvement for both fMLLR and LDA features. Thus, for future references in this Section, the experiments reported are on the features obtained after dereverberation and beamforming. From Table 5, it can be seen that: (a) For fMLLR features, use of SP and SP-CI BN vectors do not improve the WER and i-vectors perform better than that of the SP and SP-CI BN vectors, (b) For LDA features, use of SP and SP-CI BN vectors improve the WER and SP-CI perform better than that of SP BN vectors. For LDA, it can also be seen that augmenting SP-CI BN to i-vectors improve the performance as compared to that of using only i-vectors or SP-CI BN vectors.

The results reported for the LDA features on REVERB challenge database are consistent with the results seen on Aurora-4. However, this performance gains are not being reflected for fMLLR and a detailed analysis on the features is required to provide further insight into this problem. The analysis on the poor performance of BN vectors using fMLLR on REVERB challenge data will be part of our future investigations.

[1]Note that i-vector based speaker clustering has been successfully used on the REVERB task to remove such a constraint [35].

5. EVALUATIONS ON AMI DATABASE

In this Section, we provide evaluation results using SP-CI BN vectors on AMI database. To train the DNN acoustic model and BN-DNN we use, 9.4 hours (10000 utterances) of data which is randomly chosen from the AMI ihm database. For cross validation and testing, we use 0.85 hours (1108 utterances) and 8.6 hours (12612 utterances) of data respectively. Note that the test set used is the AMI ihm test set. 40 dimensional fMLLR features with a context window of 11 frames is provided as an input to a 6 layer DNN network with 2048 nodes for each of the hidden layer and 3962 output units. To derive the BN vectors, a 3 layer BN-DNN with 128 units for the BN layer and 1024 nodes to the other hidden layers is used. Similar DNN and BN-DNN structures are used in [16, 17]. In this task we use 128 dimensional i-vectors obtained using GMMs consisting of 2048 Gaussians [16]. The number of speakers and CI units are 547 and 187 respectively. We use $\alpha = 0.8$ for MTL training.

Table 6. WER obtained using fMLLR in combination with SP BN, SP-CI BN and i-vectors on AMI ihm 10000-utterance subset.

fMLLR	WER (%)
baseline	34.9
+ i-vectors	34.4
+ SP BN (STL)	34.6
+ SP-CI BN (MTL)	34.3
+ i-vectors + SP-CI BN (MTL)	**34.1**

Table 6 shows the WER obtained using SP BN, SP-CI BN and ivectors for adaptation. From Table 6, it can be seen that: (a) System performance improves on using i-vectors and BN vectors, (b) SP-CI BN vectors perform better than that of SP BN vectors and is comparable to the i-vectors, (c) SP-CI BN vectors can be augmented to the i-vectors to further improve the performance.

6. CONCLUSION

In this paper, we explored two types of joint acoustic factor learning (referred to as MTL and JTL) to derive BN vectors. We have shown that SP-CI BN vectors obtained using MTL and JTL perform better than that of SP BN vectors obtained using STL. For Aurora-4, we have compared the performance of SP-(NS/CD/CI/AR) BN vectors and have shown that SP-CI BN vectors are optimum for adaptation. We have also performed experiments using SP-CI BN vectors on REVERB challenge and AMI databases. For REVERB challenge database, we have shown that the improvements were only obtained using the LDA features. As for AMI database, we have seen improvements on using SP-CI BN vectors over the baseline. We have also provided experimental results to indicate that these BN vectors capture complementary information to that of the i-vectors and thus can be augmented with the i-vectors for adaptation.

ACKNOWLEDGEMENTS

The work reported here was started at JSALT 2015 in UW, Seattle, and was supported by JHU via grants from NSF (IIS), DARPA (LORELEI), Google, Microsoft, Amazon, Mitsubishi Electric, and MERL. Tian Tan is supported by the Program for Professor of Special Appointment (Eastern Scholar) at Shanghai Institutions of Higher Learning, the China NSFC project No. 61222208, JiangSu NSF project No. 201302060012.

7. REFERENCES

[1] G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition," *IEEE Signal Processing Magazine*, vol. 29, pp. 82–97, Nov. 2012.

[2] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 30–42, Jan. 2012.

[3] C. J. Leggetter and P. C. Woodland, "Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models," *Computer Speech and Language*, vol. 9, no. 2, pp. 171 – 185, 1995.

[4] M. J. F. Gales and P. C. Woodland, "Mean and variance adaptation within the MLLR framework," *Computer Speech and Language*, vol. 10, no. 4, pp. 249 – 264, 1996.

[5] M. J. F. Gales, "Maximum likelihood linear transformations for HMM-based speech recognition," *Computer Speech and Language*, vol. 12, no. 2, pp. 75 – 98, 1998.

[6] J. P. Neto, L. B. Almeida, M. Hochberg, C. Martins, L. Nunes, S. Renals, and T. Robinson, "Speaker-adaptation for hybrid HMM-ANN continuous speech recognition system.," in *Proc. of EUROSPEECH*, 1995.

[7] B. Li and K. C. Sim, "Comparison of discriminative input and output transformations for speaker adaptation in the hybrid NN/HMM systems.," in *Proc. of INTERSPEECH*, Sept. 2010, pp. 526–529.

[8] R. Gemello, F. Mana, S. Scanzio, P. Laface, and R. D. Mori, "Linear hidden transformations for adaptation of hybrid ANN/HMM models," *Speech Communication*, vol. 49, no. 10–11, pp. 827 – 835, 2007.

[9] F. Seide, G. Li, X. Chen, and D. Yu, "Feature engineering in context-dependent deep neural networks for conversational speech transcription," in *Proc. of ASRU*, Dec. 2011, pp. 24–29.

[10] D. Albesano, R. Gemello, P. Laface, F. Mana, and S. Scanzio, "Adaptation of artificial neural networks avoiding catastrophic forgetting," in *Proc. of IJCNN*, July 2006, pp. 1554–1561.

[11] X. Li and J. Bilmes, "Regularized adaptation of discriminative classifiers," in *Proc. of ICASSP*, May 2006, vol. 1, pp. I–I.

[12] J. Stadermann and G. Rigoll, "Two-stage speaker adaptation of hybrid tied-posterior acoustic models," in *Proc. of ICASSP*, Mar. 2005, vol. 1, pp. 977–980.

[13] D. Yu and L. Deng, *Automatic Speech Recognition - A Deep Learning Approach*, Springer, October 2014.

[14] 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 *Proc. of ICASSP*, May 2013, pp. 7942–7946.

[15] S. Xue, O. Abdel-Hamid, H. Jiang, and L. Dai, "Direct adaptation of hybrid DNN/HMM model for fast speaker adaptation in LVCSR based on speaker code," in *Proc. of ICASSP*, May 2014, pp. 6339–6343.

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

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

[18] M. L. Seltzer and A. Acero, "Separating speaker and environmental variability using factored transforms.," in *Proc. of INTERSPEECH*, Sept. 2011, pp. 1097–1100.

[19] Y.-Q. Wang and M. J. F. Gales, "Speaker and noise factorisation on the Aurora4 task," in *Proc. of ICASSP*, May 2011, pp. 4584–4587.

[20] J. Li, J.-T. Huang, and Y. Gong, "Factorized adaptation for deep neural network," in *Proc. of ICASSP*, May 2014, pp. 5537–5541.

[21] M. Ferras and H. Bourlard, "MLP-based factor analysis for tandem speech recognition," in *Proc. of ICASSP*, May 2013, pp. 6719–6723.

[22] R. Caruana, "Multitask learning," *Machine Learning*, vol. 28, no. 1, pp. 41–75, 1997.

[23] N. Parihar, J. Picone, D. Pearce, and H.G. Hirsch, "Performance analysis of the Aurora large vocabulary baseline system," in *Proc. of EUSIPCO*, Sept. 2004, pp. 553–556.

[24] D. Povey et al., "The Kaldi speech recognition toolkit," in *Proc. of ASRU*, Dec. 2011.

[25] D. Yu et al., "An introduction to computational networks and the computational network toolkit," Tech. Rep., August 2014.

[26] F. Metze et al., "The spoken web search task at MediaEval 2011," in *Proc. of ICASSP*, Mar. 2012, pp. 5165–5168.

[27] "REVERB Challenge," `http://reverb2014.dereverberation.com`, Sept. 8 2015.

[28] K. Kinoshita et al., "The REVERB challenge: a common evaluation framework for dereverberation and recognition of reverberant speech," in *Proc. of WASPAA*, Oct. 2013, pp. 1–4.

[29] T. Robinson, J. Fransen, D. Pye, J. Foote, and S. Renals, "WSJ-CAM0: A British English speech corpus for large vocabulary continuous speech recognition," in *Proc. of ICASSP*, May 1995, pp. 81–84.

[30] M. Lincoln, "The multi-channel Wall Street Journal audio-visual corpus (MC-WSJ-AV): Specification and initial experiments," in *Proc. of ASRU*, Nov. 2005, pp. 357–362.

[31] T. Yoshioka, T. Nakatani, M. Miyoshi, and H. G. Okuno, "Blind separation and dereverberation of speech mixtures by joint optimization," *IEEE Trans. Audio, Speech, Language Process.*, vol. 19, no. 1, pp. 69–84, 2011.

[32] M. Delcroix, T. Yoshioka, A. Ogawa, Y. Kubo, M. Fujimoto, N. Ito, K. Kinoshita, M. Espi, T. Hori, T. Nakatani, and A. Nakamura, "Linear prediction-based dereverberation with advanced speech enhancement and recognition technologies for the REVERB challenge," in *Proc. of REVERB*, May 2014.

[33] X. Anguera, C. Wooters, and J. Hernando, "Acoustic beamforming for speaker diarization of meetings," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 15, no. 7, pp. 2011–2022, Sept. 2007.

[34] X. Anguera, "BeamformIt," `http://www.xavieranguera.com/beamformit/`, Aug. 18 2014.

[35] X. Feng, K. Kumatani, and J. McDonough, "The CMU-MIT REVERB challenge 2014 system: Description and results," in *Proc. of REVERB*, May 2014.