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

Factorized Linear Input Network for Acoustic Model Adaptation in Noisy Conditions

Dung T. Tran, Marc Delroix, Atsunori Ogawa, Tomohiro Nakatani

NTT Communication Science Laboratories, NTT corporation,
2-4, Hikaridai, Seika-cho (Keihana Science City), Soraku-gun, Kyoto 619-0237 Japan
{dung.tran,marc.delcroix,ogawa.atsunori,nakatani.tomohiro}@lab.ntt.co.jp

Abstract

Deep neural network (DNN) based acoustic models have obtained remarkable performance for many speech recognition tasks. However, recognition performance still remains too low in noisy conditions. To address this issue, a speech enhancement front-end is often used before recognition. Such a front-end can reduce noise but there may remain a mismatch due to the difference in training and testing conditions and the imperfectness of the enhancement front-end. Acoustic model adaptation can be used to mitigate such a mismatch. In this paper, we investigate an extension of the linear input network (LIN) adaptation framework, where the feature transformation is realized as a weighted combination of affine transforms of the enhanced input features. The weights are derived from a vector characterizing the noise conditions. We tested our approach on the real data set of CHiME3 challenge task, confirming the effectiveness of our approach.

Index Terms: speech recognition, deep neural network, adaptation

1. Introduction

Progress in acoustic modeling with deep neural network (DNNs) [1] has significantly improved the performances of automatic speech recognition (ASR). However, DNN-based acoustic models still suffer in adverse environments such as in the presence of noise or reverberation. Using a speech enhancement (SE) front-end prior to ASR has been shown to greatly improve performance in such cases [2, 3, 4, 5]. However, this may not be sufficient as the SE front-end cannot completely remove the mismatch between training and test conditions. This mismatch is mainly due to the imperfectness of speech enhancement that cannot fully suppress noise or that introduce distortions. Moreover, the acoustic models are sometimes trained using noisy speech instead of enhanced speech as it has been shown to be more robust when testing on conditions unseen during training [5, 4]. Consequently adaptation is often used to reduce the mismatch between training and testing conditions and further improve performance.

There are three main approaches for acoustic model adaptation i.e., feature transformation, model compensation and exploiting auxiliary features. Input speech features can be transformed based on generative model such as CMLLR [6] or a discriminative model [7, 8]. For example, linear input network (LIN) [8] simply adds an adaptation layer to the input of a DNN with linear activation. The LIN parameters are obtained to minimize the cross entropy given some adaptation data. In a similar way, other adaptation layers have been used within the network such as linear hidden layer (LHN) [8], linear output layer (LON) [9] and linear hidden unit contributions (LHUC) [10]. A simple alternative consists of retraining all or part of the parameters of a DNN using the adaptation data, which has been shown to work relatively well when dealing with a relatively large amount of adaptation data [4, 5, 11]. Finally, exploiting auxiliary features such as noise estimates or i-vectors is another effective way to generate a DNN adaptive to the acoustic conditions [12, 13, 14, 15, 16]. In particular, noise aware training simply adds an estimate of the noise to the input features of a DNN to make the DNN adaptive to the noise conditions.

In this paper we focus on the situation where adaptation data cover different noise environments. This can have very practical applications if we consider the case where a speaker uses his device in various environments. Adaptation in such scenarios have been studied for legacy GMM-HMM ASR systems [17, 18]. In such a case, globally adapting the acoustic model may not be optimal. Therefore, we propose an extension of LIN, where the feature transformation parameters are made dependent of a noise context feature characterizing the noise environment of an utterance. This is realized by having a set of LIN transforms that process the input features in parallel. Each set of LIN transform is associated with a class of noise environments. The different compensated features are then weighted averaged with weights derived from the noise context features. We call such a structure *factorized LIN* (FLIN). The noise context features are derived from the noisy and enhanced features using a small auxiliary network that is jointly learned with the FLIN transforms. The proposed FLIN approach is related to cluster/context adaptive DNNs [19, 20, 21] with the main difference being that the factorized layer is not learned from training data but from adaptation data. Accordingly, we reduce the number of parameters by using diagonal transformation matrices for the FLIN to cope with the relatively small amount of adaptation data.

We tested our approach on the CHiME3 noisy recognition task. We use an SE front-end to reduce noise and reverberation and a strong deep convolutional neural network (CNN)-based back-end trained on multi-channel noisy speech. We employ adaptation to further reduce mismatch between testing and training conditions. This mismatch originates here from three factors, the environment, speaker and the fact that we use a speech enhancement front-end for testing but not for training. Our experiments reveal that the proposed FLIN can outperform conventional LIN based adaptation when various noise conditions are seen during adaptation. The proposed method can also be combined with network retraining based adaptation, which further improves performance.

In the remainder of the paper, we introduce notations and revise conventional LIN in Section 2. Section 3 discusses the

http://dx.doi.org/10.21437/Interspeech.2016-732

proposed FLIN approach. Some previous related works are discussed in the Section 4. We discuss our experimental settings and results in Section 5. Finally, Section 6 concludes the paper and presents potential future research directions.

2. Linear input network adaptation

LIN is a simple approach to perform adaptation of a neural network. It adds a hidden layer to the bottom of the network with a linear activation. Therefore, LIN performs a linear transformation of the input features as,

$$\hat{\mathbf{x}_t} = \mathbf{L}(\mathbf{x}_t) = \mathbf{W}\mathbf{x}_t + \mathbf{b}, \tag{1}$$

where $\hat{\mathbf{x}}_t$ is the transformed feature vector at time frame t, $\mathbf{x}_t$ is the input feature vector at time frame t, $\mathbf{L}()$ represents the affine transform, which has a weight matrix $\mathbf{W}$ and a bias vector $\mathbf{b}$. The parameters of the affine transform can be computed by error backpropagation given some adaptation data to minimize cross entropy. In this paper, we focus on unsupervised adaptation where the labels are obtained from a first decoding pass using an unadapted acoustic model.

Note that we use here diagonal transformation matrices because using full transformation matrices may not be suitable when using CNN, as it may modify the time-frequency structure of the input features, which CNN relies on. Moreover, using diagonal transformation matrices reduces the number of parameters significantly and may therefore be more suitable when dealing with a relatively limited amount of adaptation data.

LIN has been shown to improve performance in many tasks. However, it assumes that the adaptation data are relatively homogenous since it uses the same affine transform for all features.

3. Factorized LIN

When adaptation data include multiple acoustic conditions, LIN may not be optimal since a single transform may not cover the multiple conditions well. We are interested in scenarios where adaptation data from a speaker cover various noise environments such as in the third CHiME challenge task. In such a case, the LIN transform could potentially perform both speaker and environment adaptation. However, if multiple noise environments are observed for a same speaker, the capability to adapt to a specific environment will be reduced. To deal with such cases, we propose to extend the conventional LIN to FLIN.

3.1. Principles

Figure 1 shows a schematic diagram of the proposed FLIN. The LIN affine transform is factorized in a set of N transforms each associated with a noise context class. The transformed features are obtained by taking the weighted average of the transformed features for each class as,

$$\bar{\mathbf{x}}_t = \sum_{n=1}^{N} \bar{\alpha}_{n,t} \hat{\mathbf{x}}_{n,t}, \tag{2}$$

where $\alpha_{n,t}$ is a scalar value that represents the posterior of the noise context class at time frame t, and $\hat{\mathbf{x}}_{n,t}$ is the output of n^{th} LIN transformation at time frame t. Note that the time frame index t indicates the frame index of the center frame in a block of frames. The output of the n^{th} LIN transform is given by,

$$\hat{\mathbf{x}}_{n,t} = \mathbf{L}_n(\mathbf{x}_t) = \mathbf{W}_n\mathbf{x}_t + \mathbf{b}_n, \tag{3}$$

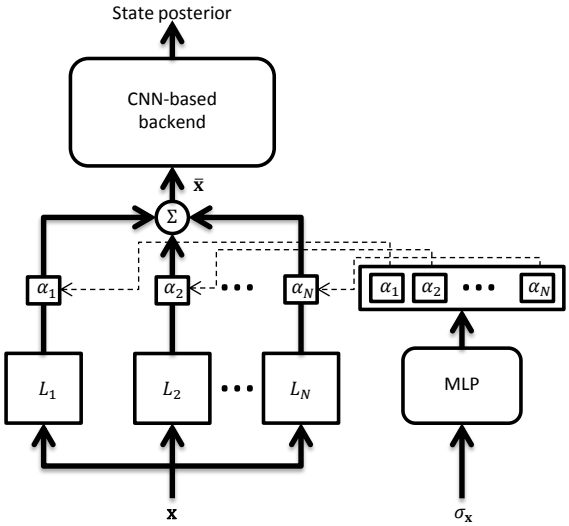

Figure 1: *Schematic diagram of the proposed factorized LIN.*

where $\mathbf{L}_n$ is the affine transform associated with the n^{th} noise context class, $\mathbf{W}_n$ and $\mathbf{b}_n$ are the weight matrix and bias vector associated with the affine transform $\mathbf{L}_n()$. With the proposed FLIN, it becomes possible to select the LIN transforms that correspond to the noise context, which is more flexible than the conventional LIN.

3.2. Noise context posterior computation

A key element of FLIN is the computation of the noise context posteriors. Building on our previous work on speaker adaptation[21], we propose to compute the noise context posteriors using an auxiliary multilayer perceptron (MLP). We derive the posteriors of the noise context $\boldsymbol{\alpha}_t = \begin{bmatrix}\alpha_{1,t} & \alpha_{2,t} & \cdots & \alpha_{N,t}\end{bmatrix}$ by forwarding noise context features $\boldsymbol{\sigma}_{\mathbf{x},t}$ through an MLP as,

$$\boldsymbol{\alpha}_t = \text{MLP}(\boldsymbol{\sigma}_{\mathbf{x},t}). \tag{4}$$

To ensure that the weights $\alpha_{n,t}$ correspond to posteriors, we constrain them to sum up to one. This is realized using a softmax activation function for the output of the auxiliary MLP. We use the squared difference of the noisy and enhanced speech features as noise context features,

$$\boldsymbol{\sigma}_{\mathbf{x},t} = \left(\mathbf{y}_t - \mathbf{x}_t\right)^2, \tag{5}$$

where $\mathbf{y}_t$ is a noisy speech feature at time frame t and $\mathbf{x}_t$ is an enhanced speech feature at time frame t. This choice of auxiliary features is motivated by previous work on uncertainty decoding [22, 23, 24, 25, 26]. We obtain utterance level noise context by simply averaging the frame level noise context features over an utterance. Note that it is possible to connect the auxiliary MLP and the main network so that the factorized LIN parameters and the auxiliary MLP parameters can be jointly trained. This assures that the noise context posteriors are optimal for the cross entropy criterion. We perform adaption by estimating only adaptation parameters, i.e. the auxiliary MLP and the factorized LIN affine transforms, using adaptation data.

4. Relations to prior works

There have been a few related studies investigating the use of auxiliary information for DNN based adaptation in noisy conditions [13, 27, 28, 29]. In [13, 28] an estimate of the noise or the signal to noise ratio (SNR) is added to the input of the DNN to make it adaptive to the noise conditions. In [27], the weight and bias parameters of the hidden layers are expressed as a polynomial function of the SNR, which enables finer adaptation. In these approaches, the network is trained with the auxiliary features, whereas we use auxiliary features for feature compensation and therefore we can re-use an already trained networks.

Li et al [29] proposed to include auxiliary features consisting of noise estimates and noisy speech directly to the input of the softmax layer. This approach is related to vector Taylor series (VTS) based adaptation that was widely used for noise adaptation in legacy GMM-based ASR systems. The main difference with our work is that we perform feature compensation since we expect that the input or lower layers of the network may be more affected by the mismatch between training and testing conditions [27, 30]. Moreover, we exploit the auxiliary information to select the feature transformation related to the noise context instead of inputting it to the softmax layer.

Finally, the proposed FLIN presents a similar factorized structure as context/cluster adaptive DNNs [19, 20, 21, 31]. In particular, an auxiliary network was also introduced in [21] to compute class weights. However, [21] factorized a hidden layer of the network and learned the parameters during training, which enables rapid single pass adaptation. The proposed FLIN performs feature compensation and learns the parameters with adaptation data, which requires two decoding passes but may potentially be better when dealing with acoustic conditions unseen during training. Note that we also use auxiliary features representing the noise context, whereas [21, 31] employed i-vectors.

5. Experiments

5.1. Dataset

We perform experiments using the CHiME-3 corpus [32] that consists of real speech recordings collected in four different environments, i.e. cafe (CAF), street junction (STR), public transport (BUS), and pedestrian area (PED). The corpus includes also simulated training and test data sets. In this paper, we discarded the simulated test data sets from our evaluation. The speech data were recorded using a tablet device with six microphones. The corpus consists of read speech, where the prompts were taken from the WSJ0 corpus. The training set comprises 1600 real and 7138 simulated utterances, which amounts to 18 hours of speech. The development and evaluation sets for the real recordings consist of 1640 and 1320 utterances, respectively, spoken by four different speakers. The test data from a given speaker cover the four different environments. According to the CHiME-3 challenge regulation, it is allowed to exploit speaker labels to perform adaptation. However, it is not allowed to use environment labels.

5.2. Settings

5.2.1. Baseline system

Our baseline system uses of a CNN based acoustic model. It consists of two convolutional layers and two fully connected layers. The first convolutional layer uses (5×11) filters, 3 input and 180 output feature maps. The second convolutional layer uses (1×5) filters, 180 input and 180 output feature maps. After each convolution layer, the resolution of the output feature map is reduced using max-pooling. Three fully connected layers with 2048 output nodes are used. Finally, a softmax layer is used to compute state posteriors. The output consists of 5976 output units corresponding to the Hidden Markov Model (HMM) states. We used sigmoid activation functions for all hidden layers. We used speech features consisting of 40 log mel filterbank coefficients appended with static, Δ and $\Delta\Delta$ coefficients. These features were extracted with a 25-msec sliding window with a 10-msec shift. We employed 11 concatenated speech features as input to the CNN (1320 dimensions in total). These features were arranged in three (40×11) input feature maps, one for static, Δ and $\Delta\Delta$ coefficients. The speech features were processed with utterance level cepstral mean normalization, and further normalized using mean and variance normalization parameters calculated on the training data. The acoustic model was trained using audio from multiple channels, i.e., multi-microphone training. By doing this, the acoustic model is exposed to larger feature variations during training, which makes it more tolerant to environmental variability. Note that during training we used the noisy speech signals without any speech enhancement front-end. We found that this strategy does not only simplifies the experiments with various speech enhancement front-ends, but also improves the robustness of the acoustic model [5]. We trained the acoustic model using mini-batch stochastic gradient descent (SGD) to minimize a cross entropy criterion. We used an initial learning rate of 0.01, a momentum of 0.9 and a batch size of 128. The learning rate was gradually decreased when the frame accuracy did not improve for a cross validation set. The learning was stopped after 20 epochs. We used dropout regularization for all full connected layers. For testing, we used a speech enhancement front-end to reduce noise and reverberation. Our speech enhanced front-end is described in [4, 33]. It consists of two steps: WPE-based dereverberation and MVDR beamforming. The acoustic beam of the MVDR is controlled using steering vectors estimated based on spectral masks. We used a trigram language model for decoding.

5.2.2. Adaptation

We compare three approaches for unsupervised speaker adaptation, i.e. model parameter retraining, LIN and the proposed FLIN. In all three cases, we used the same labels estimated from a first decoding pass and performed adaptation using all data for each speaker separately. Adaptation with model parameter retraining simply retrains all acoustic model parameters. This method was used in the system we submitted to the CHIME-3 challenge [4]. LIN adaptation is described in Section 2. We used diagonal weight matrices for the affine transforms. The initial value of the affine transforms were set to an identity matrix and a zero bias vector so that the adaptation can starts from the conventional CNN configuration performance. For the proposed FLIN, we used 4 noise context classes. The auxiliary MLP for context class weights consists of one hidden layer with 4096 neurons. We also used sigmoid activation functions for the auxiliary MLP. The parameters of the MLP were randomly initialized. However, the initial performance could be preserved because the noise context class weights sum up to 1, and because as for LIN, we used diagonal weight matrices and initialized the weight matrices to a identity matrices and the bias vectors to zero. For all adaptation experiments, we used an initial learning rate value of 5e-4, a momentum of 0.999 and 40

Table 1: *WER for the CHiME3 development set. The results are shown for the baseline system and for unsupervised adaptation with retraining, LIN, FLIN and FLIN with retraining. The best results are highlighted with bold fonts.*

	BUS	CAF	PED	STR	Ave
Baseline	7.82	5.43	5.62	6.05	6.23
Retrain	7.30	4.72	4.70	5.16	5.47
LIN	7.24	4.98	5.19	5.69	5.77
FLIN (utt)	7.15	5.00	5.29	5.66	5.77
FLIN + retrain(utt)	6.95	4.47	4.43	5.09	5.29
FLIN + retrain(frame)	**6.67**	**4.47**	**4.42**	**5.05**	**5.15**

Table 2: *WER for the CHiME3 evaluation set. The results are shown for the baseline system and for unsupervised adaptation with retraining, LIN, FLIN and FLIN with retraining. The best results are highlighted with bold fonts.*

	BUS	CAF	PED	STR	Ave
Baseline	11.89	8.12	8.95	8.90	9.32
Retrain	10.12	6.57	7.54	7.53	7.94
LIN adaptation	10.57	7.36	8.18	7.90	8.50
FLIN (utt)	10.80	7.09	7.90	7.76	8.38
FLIN + retrain(utt)	9.71	6.69	7.73	7.25	7.82
FLIN + retrain(frame)	**9.37**	**6.40**	**7.33**	**7.25**	**7.58**

epochs.

5.3. Result and discussion

Tables 1 and 2 show the results in terms of word error rate (WER) for the development and evaluation sets, respectively. The results are shown for the baseline system and for unsupervised adaptation using retraining, LIN, FLIN and FLIN with retraining. Our baseline system achieves WER of 6.23% and 9.32% WER for the development and evaluation set respectively. These numbers are competitive for the task [32] but cannot be directly compared with the best results we submitted to the CHiME-3 challenge[4], because to speedup the experiment turnover, we used a simpler ASR system with less CNN layers than in [4] and a trigram language model instead of an RNN. Note that we also used a trigram language model to generate the adaptation labels.

Speaker adaptation is very effective for the CHiME-3 task as shown by the large performance improvement obtained when retraining the acoustic models (up to 14 % relative WER reduction). Note that retraining means here updating all parameters of the baseline system. The large improvement observed is due to the fact that we compensate for the mismatch originating from the speaker and from the use of the speech enhancement front-end during testing using the unsupervised adaptation data. The fact that this simple approach performs well despite the large number of parameters it involves is also a sign that the amount of adaptation data is relatively large.

LIN based adaptation also improves performance compared to the baseline but the performance improvement is less than for retraining. This can be explained because LIN only transforms the features and does not adapt the acoustic model parameters. It is therefore less powerful when the amount of adaptation data is sufficient for retraining.

FLIN achieves comparable performance as LIN for the development set and slightly better for the evaluation set. In this case, we use utterance-level noise context features. We used here 4 classes as it performed slightly better than 2 or more classes. We did not observed large performance degradation when using a larger number of classes.

Table 3: *WER for CHiME3 experiment on the development and evaluation sets. The results are shown for each speaker. Comparison of our baseline, baseline with retrain all parameter, FLIN with retrain all parameters. The best results are highlighted with bold fonts.*

	Baseline	Retrain	FLIN + retrain (frame)
F01	6.78	5.33	**5.08**
F04	6.33	5.64	**5.12**
M03	5.22	4.90	**4.49**
M04	6.54	6.01	**5.93**
F05	10.69	8.81	**8.66**
F06	9.08	7.90	**7.39**
M05	7.97	5.89	**5.76**
M06	9.95	9.17	**8.54**

FLIN can be combined with retraining. In this case, all parameters of the network including the CNN back-end, LIN transforms and the auxiliary MLP are updated. Using utterance-level auxiliary features we could observe a small improvement with "FLIN + retrain" compare to "retrain". However, the improvement becomes larger when we use frame-level features ("FLIN + retrain (frame)"). We observed consistent improvements compared to retraining for both development and evaluation sets. FLIN with retraining outperforms baseline with 17.3% and 18.6% relative WER reduction on the development and evaluation sets, respectively. It also outperforms retraining based adaptation with 6.0% and 4.5% relative WER reduction on the development and evaluation sets, respectively.

We further analyzed the performance for each speaker. Table 3 shows the averaged WER per speaker for the baseline, retraining and FLIN with retraining. The upper part of the results correspond to the development set and the lower part to the evaluation set. The results of Table 3 confirm that the proposed FLIN with retraining outperforms the baseline and retraining for all speakers. The relative WER reduction ranges from 9 to 27 % compared to the baseline.

6. Conclusions

In this paper, we investigated an extension of the LIN adaptation framework, where the feature transformation is realized as a weighted combination of affine transforms of the input features. The weights are derived from a noise context vector characterizing the noise environments. We tested our approach on the real data set of the CHiME3 challenge task showing promising results suggesting that the noise context can be combined with speaker information to further improve the performance of noise robust speech recognition. In future works, we will investigate a pre-training step where the MLP is trained to predict the noise context classes in advance. Moreover, we will also explore extension of FLIN for speaker and noise adaptive training[31].

7. Acknowledgements

The authors acknowledge Dr. Yoshioka Takuya for the fruitful discussions and for providing the baseline system.

8. References

[1] G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of fours research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[2] M. Delcroix, Y. Kubo, T. Nakatani, and A. Nakamura, "Is speech enhancement pre-processing still relevant when using deep neural networks for acoustic modeling?" in *INTERSPEECH*, 2013, pp. 2992–2996.

[3] M. L. Seltzer, D. Yu, and Y. Wang, "An investigation of deep neural networks for noise robust speech recognition," in *ICASSP*, 2013, pp. 7398–7402.

[4] T. Yoshioka, N. Ito, M. Delcroix, A. Ogawa, K. Kinoshita, M. Fujimoto, C. Yu, W. J. Fabian, M. Espi, T. Higuchi, S. Araki, and T. Nakatani, "The ntt chime-3 system: Advances in speech enhancement and recognition for mobile multi-microphone devices," in *ASRU*, 2015, pp. 436–443.

[5] M. Delcroix, T. Yoshioka, A. Ogawa, Y. Kubo, M. Fujimoto, N. Ito, K. Kinoshita, M. Espi, S. Araki, T. Hori, and T. Nakatani, "Strategies for distant speech recognitionin reverberant environments," *EURASIP Journal on Advances in Signal Processing*, vol. 1, pp. 1–15, 2015.

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

[7] 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'11*, 2011, pp. 24–29.

[8] 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. of ICASSP'06*, vol. 1, 2006, pp. 1189–1192.

[9] 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'10*, 2010, pp. 526–529.

[10] P. Swietojanski and S. Renals, "Learning hidden unit contributions for unsupervised speaker adaptation of neural network acoustic models," in *Proc of SLT'14*, 2014.

[11] H. Liao, "Speaker adaptation of context dependent deep neural networks." in *ICASSP*, 2013, pp. 7947–7951.

[12] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny, "Speaker adaptation of neural network acoustic models using i-vectors," in *Proc. ASRU'13*, 2013, pp. 55–59.

[13] M. Seltzer, D. Yu, and Y. Wang, "An investigation of deep neural networks for noise robust speech recognition," in *Proc. of ICASSP'13*, 2013, pp. 7398–7402.

[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'13*. IEEE, 2013, pp. 7942–7946.

[15] J. Li, J.-T. Huang, and Y. Gong, "Factorized adaptation for deep neural network," in *Proc of ICASSP'14*, 2014, pp. 5537–5541.

[16] C. Yu, A. Ogawa, M. Delcroix, T. Yoshioka, T. Nakatani, and J. H. Hansen, "Robust i-vector extraction for neural network adaptation in noisy environment," in *Proc of INTERSPEECH'15*, 2015, pp. 2854–2857.

[17] M. Seltzer and A. Acero, "Separating speaker and environmental variability using factored transforms," in *Interspeech*. International Speech Communication Association, August 2011. [Online]. Available: http://research.microsoft.com/apps/pubs/default.aspx?id=155400

[18] Y. Wang and M. J. F. Gales, "Speaker and noise factorization for robust speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 7, pp. 2149–2158, Sept 2012.

[19] C. Wu and M. Gales, "Multi-basis adaptive neural network for rapid adaptation in speech recognition," in *Proc. of ICASSP'15*, 2015, pp. 4315–4319.

[20] T. Tan, Y. Qian, M. Yin, Y. Zhuang, and K. Yu, "Cluster adaptive training for deep neural network," in *Proc. of ICASSP'15*, 2015, pp. 4325–4329.

[21] M. Delcroix, K. Kinoshita, C. Yu, A. Ogawa, T. Yoshioka, and T. Nakatani, "Context adaptive deep neural networks for fast acoustic model adaptation in noisy conditions," in *ICASSP'16*, 2016.

[22] M. Delcroix, T. Nakatani, and S. Watanabe, "Static and dynamic variance compensation for recognition of reverberant speech with dereverberation preprocessing," *IEEE Transatcion on Audio, Speech, and Language Processing*, vol. 17, no. 2, pp. 324–334, 2009.

[23] D. T. Tran, E. Vincent, and D. Jouvet, "Nonparametric uncertainty estimation and propagation for noise robust asr," *IEEE Transatcion on Audio, Speech, and Language Processing*, vol. 23, no. 11, pp. 1835–1846, 2015.

[24] Y. Tachioka and S. Watanabe, "Uncertainty training and decoding methods of deep neural networks based on stochastic representation of enhanced feature," in *INTERSPEECH16*, 2015, pp. 3541–3545.

[25] D. Kolossa and R. Haeb-Umbach, *Robust speech recognition of uncertain or missing data*. New York: Springer, 2011.

[26] M. Delcroix, S. Watanabe, T. Nakatani, and A. Nakamura, "Cluster-based dynamic variance adaptation for interconnecting speech enhancement pre-processor and speech recognizer," *Computer Speech Language.*, vol. 27, no. 1, pp. 350–368, 2000.

[27] R. Zhao, J. Li, and Y. Gong, "Variable-activation and variable-input deep neural network for robust speech recognition." in *Spoken Language Technology Workshop (SLT)*, 2014, pp. 542–547.

[28] ——, "Variable-component deep neural network for robust speech recognition." in *Interspeech*, 2014.

[29] J. Li, J. T. Huan, and Y. Gong, "Factorized adaptation for deep neural network." in *ICASSP*, 2014, pp. 5537–5541.

[30] A. Mohamed, G. Hinton, and G. Penn, "Understanding how deep belief networks perform acoustic modelling," in *Proc. of ICASSP'12,*, 2012, pp. 4273–4276.

[31] M. Delcroix, K. Kinoshita, T. Hori, and T. Nakatani, "Context adaptive deep neural network for fast acoustic model adaptation." in *ICASSP*, 2015, pp. 4535–4539.

[32] J. Barker, R. Marxer, E. Vincent, and S. Watanabe, "The third CHiME speech separation and recognition challenge: Dataset, task and baselines," in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, Dec 2015, pp. 504–511.

[33] T. Higuchi, N. Ito, T. Yoshioka, and T. Nakatani, "Robust mvdr beamforming using time-frequency masks for online/offline asr in noise," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 5210–5214.