UNSUPERVISED SPEAKER ADAPTATION OF BATCH NORMALIZED ACOUSTIC MODELS FOR ROBUST ASR

Zhong-Qiu Wang[♣] *and DeLiang Wang*[♣, ♥]

[♣]Department of Computer Science and Engineering, The Ohio State University, USA
[♥]Center for Cognitive and Brain Sciences, The Ohio State University, USA
{wangzhon, dwang}@cse.ohio-state.edu

ABSTRACT

Batch normalization is a standard technique for training deep neural networks. In batch normalization, the input of each hidden layer is first mean-variance normalized and then linearly transformed before applying non-linear activation functions. We propose a novel unsupervised speaker adaptation technique for batch normalized acoustic models. The key idea is to adjust the linear transformations previously learned by batch normalization for all the hidden layers according to the first-pass decoding results of the speaker-independent model. With the adjusted linear transformations for each test speaker, the test distribution of the input of each hidden layer better matches the training distribution. Experiments on the CHiME-3 dataset demonstrate the effectiveness of the proposed layer-wise adaptation approach. Our overall system obtains 4.24% WER on the real subset of the test data, which represents the best reported result on this dataset to date and a relative 27.3% error reduction over the previous best result.

***Index Terms*—** robust ASR, deep neural networks, batch normalization, unsupervised speaker adaptation, CHiME-3

1. INTRODUCTION

Although DNN-HMM hybrid approaches have shown to be robust to small input variations [1], they still suffer from the mismatch between training and test conditions, just like many other supervised learning based approaches. For real-world ASR systems, many factors can lead to mismatched training and testing conditions, such as different speakers, noises, room reverberations and microphone characteristics. Performing multi-condition training on the data from, say multiple speakers or multiple environments, can only lead to a "flat" model. To deal with the mismatch problem, many adaptation techniques are proposed to compensate for the differences between training and test conditions. These techniques can be roughly categorized into feature-space, model-space and feature augmentation approaches.

The most popular feature-space approach is probably the feature-space MLLR method originally developed for GMM-HMM systems [2], [3]. In this approach, a well-trained GMM-HMM system is employed to obtain the fMLLR features, on which a DNN-HMM hybrid system is built. The linear input network method is another popular feature-space approach that tries to learn a linear transformation of the input of DNNs for adaptation [4], [5]. The linear transformation is normally constrained to be diagonal and tied across neighboring frames to reduce the number of parameters to be learned. For the feature augmentation approach, i-vectors are the commonly used auxiliary features for encoding speaker characteristics [6], [7]. Other auxiliary features, such as bottleneck vectors obtained from speaker classification tasks [8] or environment classification tasks, are utilized to account for speaker or environment variations. For model-space approaches, parameters in DNNs are modified for adaptation. A straightforward way is to adapt all the parameters in the DNN, but learning a large number of parameters would easily lead to overfitting, especially for unsupervised adaptation or when the amount of adaptation data is limited [9]. In [10], a conservative training approach based on KL divergence is proposed to regularize the adapted model to be close to the un-adapted model, while the footprint size of each speaker is large in this case. To limit the number of parameters to be learned when the amount of adaptation data is limited, linear transformation based methods, such as linear hidden network [11], linear output network [12] and singular value decomposition bottleneck adaptation techniques [13], [14], are proposed. Limiting the number of parameters can be considered as adding regularization for acoustic model adaptation. The performance of adaptation is highly dependent on whether the task is supervised or unsupervised, the amount of adaptation data or computations, storage requirements per speaker, and specific tasks.

In this context, we propose an unsupervised speaker adaptation method for batch normalized acoustic models. The key idea is to adjust scaling factors together with shifting factors in batch normalized acoustic models so that the distribution of the test data at every hidden layer better matches that of the training data. On the CHiME-3 dataset that exhibits a large amount of mismatch between training and testing conditions, we obtain a relative 23.9% error reduction when a tri-gram language model is used to generate first-pass decoding results for adaptation, and 34.3% error reduction when a five-gram language model and an RNN language model are used. The overall system achieves state-of-the-art 4.24% WER on the CHiME-3 dataset.

The rest of the paper is organized as follows. We describe our method in Section 2, and our experiments in Section 3 and 4. We conclude this paper in Section 5.

2. SYSTEM DESCRIPTION

We first introduce the batch normalization method. Then we discuss the linear input network (LIN) based approach, and our proposed adaptation strategy for batch normalized acoustic models. The LIN based approach serves as a motivation example for our proposed algorithm.

2.1. Batch Normalization

Batch normalization is a standard training technique for deep neural networks [15]. It has been widely used in many tasks such as image classification [16] and large scale speech recognition [17]. The key idea is to first normalize the input of each hidden layer using the mean and variance calculated from each mini-batch in the forward pass, and then linearly scale and shift the normalized input before applying non-linear activations. Mathematically,

$$h^{(m)} = \delta\left(\gamma^{(m)} \frac{W^{(m)} h^{(m-1)} - \mu^{(m)}}{\sigma^{(m)}} + \beta^{(m)}\right) \qquad (1)$$

where $h^{(m)}$ is the output of the mth hidden layer, δ represents the non-linear activation function, $\mu^{(m)}$, $\sigma^{(m)}$, $\gamma^{(m)}$ and $\beta^{(m)}$ are the mean, standard deviation, scaling factor and shifting factor at the mth hidden layer, respectively, and $h^{(0)}$ is just the input of the network. The bias term of each hidden layer is not included here as it would be cancelled out by the mean subtraction operation in the forward pass.

It is suggested in [15] that during DNN training, the distribution of the input of hidden layers could change frequently, as the parameters in previously layers change. As a result, the optimization process is slowed down significantly. This problem can be alleviated by performing layer-wise normalization. This way, much larger learning rates can be safely used, and thereby much faster convergence and potentially better results could be achieved.

The linear transformation terms, i.e. $\gamma^{(m)}$ and $\beta^{(m)}$, are critical in batch normalization. If they are not incorporated, the input of the non-linear functions would be concentrated around zero. This means that the activations would be close to limited linear transformations for sigmoidal units [15], and approximately half of the activations would be implicitly forced to be zero and half to be positive for ReLUs. With the linear transformation terms, the network can automatically choose which segments of the activation function to use for a better performance.

After the training is done, we feed-forward all the training data to the network and record the mean and variance at every hidden layer. We use the means and variances calculated this way, denoted as $\mu_{train}^{(m)}$ and $\sigma_{train}^{(m)}$, for layer-wise normalization at the test stage.

2.2. Linear Input Network

The idea of the linear input network approach is to learn a linear transformation of the input features of the acoustic model [4], [5], [18]. In this study, as the amount of adaptation data is limited, we constrain the number of parameters to be learned by forcing the linear transformation to be diagonal, with parameters tied across neighboring frames:

$$\hat{x}_{t,f} = w_f \frac{x_{t,f} - \mu_f}{\sigma_f} + b_f \qquad (2)$$

where x denotes the un-normalized input feature, $\hat{x}$ represents the adapted feature, μ and σ stand for the mean and standard deviation computed from the whole training data, and t and f indexes time and frequency, respectively. w and b are the parameters to be learned for each speaker. We initialize w to be an all-one vector and b to be an all-zero vector before adaptation.

This method is reasonable in the sense that the distribution of test data may be very different from that of training data. Therefore, a diagonal linear transformation is learned to scale and shift the normalized test data to better match the mean and variance of the training data.

2.3. Adaptation of Batch Normalized Models

One problem of the LIN approach is that it only tries to match the distribution of test data with that of training data at the input level. However, after many layers of affine transformations and non-linear operations, the distribution of the hidden activations of test data could become more and more mismatched with that of training data. Only matching the distribution at the input level would not be good because the LIN approach itself may not behave well. To deal with this problem, we learn a linear transformation for the input of every hidden layer so that the linear transformed input can better match the distribution of the training data at every hidden layer.

Then, the problem is how to get the distribution of the training data at every hidden layer so that we can match the distribution of the test data onto. Batch normalized acoustic models naturally provide us the $\mu_{train}^{(m)}$ and $\sigma_{train}^{(m)}$ at the mth hidden layer. Therefore, in this study we adjust the scaling factor and shifting factor at every hidden layer for batch normalized acoustic models

$$\hat{h}^{(m)} = \delta\left(\gamma^{(m)} \frac{W^{(m)} \hat{h}^{(m-1)} - \mu_{train}^{(m)}}{\sigma_{train}^{(m)}} + \beta^{(m)}\right) \qquad (3)$$

where $\hat{h}$ is the adapted hidden activations, and $\gamma^{(m)}$ and $\beta^{(m)}$ are the only parameters to be adjusted for each speaker. The dimensions of $\gamma^{(m)}$ and $\beta^{(m)}$ are the same as the number of hidden units at the mth hidden layer. The number of parameters to be learned is therefore limited. Note that we do not change $W^{(m)}$ as modifying it could destroy the well-learned filters in the weight matrix.

In this paper, we perform unsupervised adaptation for each speaker. We first decode all the utterances of each speaker in the test set using the speaker-independent batch normalized acoustic models to obtain the first-pass decoding results, from which we adjust the parameters to minimize the cross-entropy criterion using the back-propagation algorithms.

It should be mentioned that in [19] and [20], Pawel *et al.* propose an adaptation technique based on learning hidden unit contributions (LHUC), which essentially learns a weight for every hidden unit to re-combine all the activations for a target speaker or environment. In [21], a parameterized hidden activation function approach is proposed to re-weight the importance of each hidden unit for speaker adaptation. Different from these studies, our method is proposed for batch normalized acoustic models, and we adjust the shifting factors together with the scaling factors to recombine the activation before, rather than after, the non-linear function at all the hidden layers. This may better adapt an acoustic model to a target speaker or a target environment. In addition, our method does not make any assumptions on the type of activation functions.

3. EXPERIMENTAL SETUP

We evaluate our methods on the recently proposed CHiME-3 corpus [22]. The CHiME-3 dataset includes real and simulated data in four challenging daily environments, i.e. cafe, street junction, public transport, and pedestrian area, and consists of six-channel microphone array data. The real recordings are uttered by real speakers in the abovementioned environments, and are recorded using a specially designed tablet with five microphones mounted in the front and one in the rear. The simulated data is created by first convolving clean utterances with the estimated impulse responses of an environment, and then digitally adding noises recorded in that environment. It represents a significant step towards realism compared with the previous CHiME challenges. The training set contains 7138 simulated utterances from 83 speakers in the WSJ0-5k corpus and 1600 real utterances from four speakers, the development set consists of 1640 simulated and 1640 real utterances from four unseen speakers, and the test set contains 1320 simulated and 1320 real utterances from another four unseen speakers. The transcriptions of all the utterances are based on those in the WSJ0-5k corpus. The text corpus in WSJ0 is available for language modeling. All the submitted systems are ranked according to their word error rates (WER) on the real subset of the test data [22]. Note that speaker labels are allowed to be used for speaker adaptation in this challenge.

We train our acoustic models using noisy signals, while enhanced single-channel signals are used for decoding. This strategy is found to be more robust than training the acoustic model using enhanced signals [23], [5], [18]. In our study, we use the single channel signals from all the channels for acoustic modeling. This way, much more data can be used for acoustic modeling. In our study, the total number of training utterances is 50,828 (7138*6+1600*5, ~104 hours). We exclude the rear microphone of the real recordings due to its low SNR level. Following the common pipelines in the Kaldi toolkit, a GMM-HMM system is first trained, from which alignments are generated and a DNN-HMM hybrid system is built on the fMLLR features. After that, sMBR training is applied. Finally, a five-gram language model and an RNN language model are utilized to re-score the lattices generated by a tri-gram language model.

For speech enhancement, we follow the generalized eigenvalue (GEV) beamformer proposed in [24], [25] to obtain enhanced single-channel signals. The key idea of the GEV beamformer is to estimate a time-frequency mask, which represents the presence probability of speech at every T-F bin, from multi-channel signals. With the estimated mask, spatial covariance matrices of speech and noise can be derived. Then, the beamforming weights are obtained by performing generalized eigen decomposition on these two covariance matrices. A single-channel post-filter based on blind analytic normalization (BAN) is further applied to reduce speech distortions. The key step here is to estimate the time-frequency mask from a noisy utterance. In our study, a DNN is trained on the complementary feature set [26] to predict the ideal ratio mask (IRM) [27], [28] defined in the power spectrogram domain using all the simulated training data (7138*6 utterances in total). In our experiments, the GEV beamformer built in this way gives us substantially better ASR results over the default and official weighted delay-and-sum beamformer implemented using the BeamformIt toolkit.

We refer the readers to [24], [25] and [29] for more details of the GEV beamformer and the RNN language models used in our recipe, and [30], [28] for more details of supervised speech separation.

4. EVALUATION RESULTS

In this section, we first report the performance of our baseline acoustic model trained without batch normalization, and then show the results of using batch normalization for acoustic model training. Finally, we present the performance of the proposed approach for speaker adaptation.

4.1. Baseline Acoustic Model

We build our baseline acoustic model using a DNN with 7 hidden layers in a speaker-independent way. Each hidden layer has 2048 exponential linear units (ELUs) [31]. In our experiments, the ELUs lead to consistently better results over the commonly used ReLUs. The dropout rate is 0.3 for the input layer and all the hidden layers. The DNN is trained for 50 epochs using AdaGrad [32] with a momentum term to minimize the cross-entropy criterion. The momentum is linearly increased from 0.1 to 0.9 in the first five epochs and kept fixed at 0.9 afterwards. The learning rate is fixed at 0.005 in the first 10 epochs and linearly decays to $\mathbf{10^{-5}}$ in the following epochs. The mini-batch size is set to 256. The input feature is 40-dimensional log Mel-spectrogram feature with its delta and double delta components. Sentence-level mean normalization is applied before global mean-variance normalization. Performing sentence-level mean normalization is shown to lead to consistent improvements in [33]. The globally normalized features are then spliced together using a symmetric 11-frame context window. There are 3161 senone states in total in our system. Using this training method and a tri-gram language model for decoding, we obtain 10.40% average WER on the real subset of the test set as reported in Table 1.

4.2. Effects of Batch Normalization

We then report our results of using batch normalization for acoustic model training. As batch normalization allows much larger learning rates for DNN training, we enlarge the learning rate by 10 times, and use the same configuration to train the batch normalized acoustic model. In our experiments, we observe not only faster convergence but also better ASR results. The average WER is improved by absolute 0.89% (from 10.40% to 9.51%). Note that if we apply the same 10-times learning rate when training the baseline acoustic model, the training process diverges after several epochs. In addition, if we do not enlarge the learning rate when doing batch normalization, the performance improvement is small compared with the baseline acoustic model.

4.3. Speaker Adaptation

There are around 410 utterances for each speaker in the development set and approximately 330 utterances for each speaker in the test set. We use all the utterances of each speaker for adaptation. Before adaptation, we turn off the dropout at both the input layer and hidden layers by setting the dropout rates to zero and scaling down all the weight matrices by 0.7 (1-0.3). The speaker-independent acoustic model is then trained for 10 epochs using AdaGrad to minimize the cross-entropy criterion. The learning rate is linearly decreased from 0.005 to 10^{-5}, the momentum is fixed at 0.9, and the mini-batch size is set to 256. We use the model after the last epoch to generate pseudo-likelihood for decoding. This recipe is tuned according to its recognition performance on the development set. In our experiments, we did not observe overfitting on the decoding results as we strictly limit the number of parameters to be learned for adaptation. In our cases, even if the adapted acoustic model fully overfits the first-pass decoding results, the performance would not drop as it will re-generate the same decoding results in the second

Table 1. *ASR performance (%WER) using first-pass decoding results of a tri-gram language model for adaptation.*

Approaches	LMs for decoding	Dev. set			Test set	
		SIMU	REAL	AVG	SIMU	REAL
Baseline acoustic model	Tri-gram	7.22	6.87	7.05	7.86	10.40
Batch normalized acoustic model	Tri-gram	6.85	6.47	6.66	7.17	9.51
LIN adaptation	Tri-gram	5.60	5.79	5.69	6.37	8.34
Scaling and shifting factors adaptation (proposed)	Tri-gram	4.98	**4.92**	4.95	**5.05**	**7.24**
Scaling and shifting factors adaptation (proposed) + LIN adaptation	Tri-gram	**4.93**	4.96	**4.94**	5.10	7.28
LHUC [19]	Tri-gram	5.18	5.36	5.27	5.58	7.78

Table 2. *ASR performance (%WER) using better first-pass decoding results for adaptation.*

Approaches	LMs for decoding	Dev. set			Test set	
		SIMU	REAL	AVG	SIMU	REAL
Batch normalized acoustic model + sMBR	Tri-gram	6.50	6.10	6.30	7.36	8.64
-	Five-gram	5.31	4.96	5.14	6.11	7.35
-	RNNLM	4.63	4.15	4.39	5.52	6.45
LIN adaptation	Tri-gram	5.01	5.29	5.15	6.07	7.59
-	Five-gram and RNNLM	3.55	3.47	3.51	4.47	5.39
Scaling and shifting factors adaptation (proposed)	Tri-gram	3.85	4.20	4.02	4.18	5.77
-	Five-gram and RNNLM	**3.03**	**3.06**	**3.05**	**3.25**	**4.24**
Yoshioka *et al*. [34]	-	3.63	3.45	3.54	4.46	5.83

pass. We report the results of unsupervised speaker adaptation in Table 1 and 2.

In our first experiment, the labels for adaptation come from the first-pass decoding results using the batch normalized acoustic model together with a tri-gram language model. Using the LIN approach for adaptation, we can improve the performance to 8.34% from 9.51%. Note that for the LIN adaptation, the number of parameters to be learned for each speaker is 240 (40*3+40*3). This large improvement may be due to the particularly challenging speaking styles of the test speakers in the CHiME-3 corpus [22]. Much better performance, i.e. 7.24%, is obtained by adapting the scaling factors and shifting factors for all the hidden layers in the batch normalized acoustic model. Combining the LIN adaptation with our proposed method leads to slightly worse performance (from 7.24% to 7.28%). This may be because adapting the scaling and shifting factors of the first hidden layer already incorporates most of the effects of the LIN adaptation. We also compare our approach with the LHUC algorithm [19], where the activations of each hidden layer are re-weighted using a sigmoid-like function. As shown in Table 1, our approach obtains consistently better results over the LHUC algorithm, possibly because our approach adapts the activation before the non-linear functions.

The first-pass decoding results are very important for unsupervised adaptation. Too many errors in the decoding results would mislead the acoustic model during adaptation. To obtain better first-pass decoding results, we first apply sMBR training to the batch normalized acoustic model. After generating the alignments and lattices, we train the acoustic model for two epochs using AdaGrad. As batch normalization allows large learning rates for training, we fix the learning rate at 10^{-4}, which is 10 times the commonly used learning rate in the sequence training code of the Kaldi toolkit. The WER is reduced from 9.51% to 8.64% after sequence training, as reported in the first entry of Table 2. We then refine the decoding results using a more powerful five-gram language model and an RNN language model. The performance is pushed up to 7.35% WER after five-gram language model re-scoring and 6.45% WER after RNN language model re-scoring. With the refined decoding results, we perform adaptation for each test speaker. After that, we first use the adapted model together with the tri-gram language model to generate the lattices, and then use the five-gram language model and the RNN language model again for re-scoring. The results are presented in the second and third entry of Table 2, respectively. By adapting the scaling and shifting factors, we can improve the performance from 6.45% to 4.24%, which represents a relative 34.3% improvement on the real subset of the test data. In addition, the 4.24% WER is the best result on the CHiME-3 dataset reported so far, which represents a relative 27.3% error reduction over the state-of-art 5.83% WER reported in [34]. It is also among the top results in the 6-channel track of the later held CHiME-4 challenge. We report the per-environment WER of our best system in Table 3.

Table 3. *WER (%) per environment of our best system*

Environment	Dev. set		Test set	
	SIMU	REAL	SIMU	REAL
BUS	2.89	3.94	3.14	5.78
CAF	3.50	2.65	3.31	3.47
PED	2.83	2.79	3.40	3.55
STR	2.92	2.88	3.18	4.15

5. CONCLUDING REMARKS

In this study, we have proposed a novel method for the unsupervised speaker adaptation of batch normalized acoustic models. Large improvements have been observed on the CHiME-3 dataset after conducting speaker adaptation using our proposed approach. Moving forward, we plan to modify this method for speaker adaptive training; currently, we only perform test-time adaptation. In addition, auxiliary features such as i-vectors or fMLLR features can be combined with our proposed approach. Furthermore, in our current work, we feed-forward all the data to record $\mu_{train}^{(m)}$ and $\sigma_{train}^{(m)}$ after training, and use them at the test stage for layer-wise normalization. It would be interesting to see whether it is possible to directly use test data to calculate means and variances so that we can perform faster and better adaptation.

6. ACKNOWLEDGEMENTS

We would like to thank Xueliang Zhang for helpful discussions on acoustic beamforming. This research was supported in part by an AFRL contract (FA8750-15-1-0279), an NSF grant (IIS-1409431), and the Ohio Supercomputer Center.

7. REFERENCES

[1] D. Yu, M. L. Seltzer, J. Li, J.-T. Huang, and F. Seide, "Feature Learning in Deep Neural Networks - Studies on Speech Recognition Tasks," *arXiv preprint arXiv:1301.3605*, 2013.

[2] M. J. F. Gales, "Maximum Likelihood Linear Transformations for HMM-based Speech Recognition," *Computer Speech and Language*, vol. 12, no. 2, pp. 75–98, Apr. 1998.

[3] S. Rath, D. Povey, K. Veselý, and J. Cernocký, "Improved Feature Processing for Deep Neural Networks," in *Proceedings of Interspeech*, 2013, pp. 109–113.

[4] F. Seide, G. Li, X. Chen, and D. Yu, "Feature Engineering in Context-dependent Deep Neural Networks for Conversational Speech Transcription," in *IEEE Workshop on Automatic Speech Recognition and Understanding*, 2011, pp. 24–29.

[5] A. Narayanan and D. L. Wang, "Investigation of Speech Separation as a Front-end for Noise Robust Speech Recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 4, pp. 826–835, Apr. 2014.

[6] G. Saon and H. Soltau, "Speaker Adaptation of Neural Network Acoustic Models using I-Vectors," in *IEEE Workshop on Automatic Speech Recognition and Understanding*, 2013, pp. 55–59.

[7] Y. Miao, H. Zhang, and F. Metze, "Speaker Adaptive Training of Deep Neural Network Acoustic Models Using I-Vectors," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 11, pp. 1938–1949, Nov. 2015.

[8] 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 Modeling," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2016, pp. 5280–5284.

[9] H. Liao, "Speaker Adaptation of Context Dependent Deep Neural Networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013.

[10] 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 *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 7893–7897.

[11] B. Li and K. Sim, "Comparison of Discriminative Input and Output Transformations for Speaker Adaptation in the Hybrid NN/HMM Systems," in *Proceedings of Interspeech*, 2010, pp. 526–529.

[12] K. Yao, D. Yu, F. Seide, and H. Su, "Adaptation of Context-dependent Deep Neural Networks for Automatic Speech Recognition," in *IEEE Spoken Language Technology Workshop*, 2012, pp. 366–369.

[13] J. Xue, J. Li, and D. Yu, "Singular Value Decomposition based Low-footprint Speaker Adaptation and Personalization for Deep Neural Network," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2014, pp. 6359–6363.

[14] K. Kumar, C. Liu, and Y. Gong, "Non-negative Intermediate-layer DNN Adaptation for A 10-kB Speaker Adaptation Profile," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2016, pp. 5285–5289.

[15] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," in *arXiv preprint arXiv:1502.03167*, 2015.

[16] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and Tell: A Neural Image Caption Generator," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3156–3164.

[17] D. Amodei, R. Anubhai, E. Battenberg, and C. Case, "Deep Speech 2: End-to-End Speech Recognition in English and Mandarin," in *arXiv preprint arXiv:1512.02595*, 2015.

[18] Z.-Q. Wang and D. L. Wang, "A Joint Training Framework for Robust Automatic Speech Recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 4, pp. 796–806, Apr. 2016.

[19] P. Swietojanski and S. Renals, "Learning Hidden Unit Contributions for Unsupervised Speaker Adaptation of Neural Network Acoustic Models," in *IEEE Spoken Language Technology Workshop*, 2014, pp. 171–176.

[20] P. Swietojanski, J. Li, and S. Renal, "Learning Hidden Unit Contributions for Unsupervised Acoustic Model Adaptation," in *arXiv preprint arXiv:1601.02828v1*, 2016.

[21] C. Zhang and P. C. Woodland, "DNN Speaker Adaptation Using Parameterised Sigmoid and ReLU Hidden Activation Functions," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2016, pp. 5300–5304.

[22] J. Barker, R. Marxer, E. Vincent, and S. Watanabe, "The Third 'CHiME' Speech Separation and Recognition Challenge: Dataset, Task and Baselines," in *IEEE Workshop on Automatic Speech Recognition and Understanding*, 2015, pp. 504–511.

[23] M. L. Seltzer, D. Yu, and Y. Wang, "An investigation of deep neural networks for noise robust speech recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 7398–7402.

[24] J. Heymann, L. Drude, and A. Chinaev, "BLSTM Supported GEV Beamformer Front-end for the 3rd CHiME Challenge," in *IEEE Workshop on Automatic Speech Recognition and Understanding*, 2015, pp. 444–451.

[25] J. Heymann and L. Drude, "Neural Network Based Spectral Mask Estimation for Acoustic Beamforming," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2016, pp. 196–200.

[26] Y. Wang, K. Han, and D. L. Wang, "Exploring Monaural Features for Classification-based Speech Segregation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 2, pp. 270–279, 2013.

[27] A. Narayanan and D. L. Wang, "Ideal Ratio Mask Estimation using Deep Neural Networks for Robust Speech Recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 7092–7096.

[28] Y. Wang, A. Narayanan, and D. L. Wang, "On Training Targets for Supervised Speech Separation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 12, pp. 1849–1858, 2014.

[29] T. Hori, Z. Chen, H. Erdogan, J. R. Hershey, J. Le Roux, V. Mitra, and S. Watanabe, "The MERL/SRI System for the 3rd CHiME Challenge Using Beamforming, Robust Feature Extraction, and Advanced Speech Recognition," in *IEEE Workshop on Automatic Speech Recognition and Understanding*, 2015, pp. 475–481.

[30] Y. Wang and D.L. Wang, "Towards Scaling Up Classification-based Speech Separation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 7, pp. 1381–1390, 2013.

[31] D. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)," in *arXiv preprint arXiv:1511.07289*, 2015.

[32] J. Duchi, E. Hazan, and Y. Singer, "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization," *The Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 2011.

[33] S. Zhao, X. Xiao, Z. Zhang, T. N. T. Nguyen, X. Zhong, B. Ren, L. Wang, D. L. Jones, E. S. Chng, and H. Li, "Robust Speech Recognition using Beamforming with Adaptive Microphone Gains and Multichannel Noise Reduction," in *IEEE Workshop on Automatic Speech Recognition and Understanding*, 2015, pp. 460–467.

[34] 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 *IEEE Workshop on Automatic Speech Recognition and Understanding*, 2015, pp. 436–443.