Wide Residual BLSTM Network with Discriminative Speaker Adaptation for Robust Speech Recognition

Jahn Heymann, Lukas Drude, Reinhold Haeb-Umbach

Paderborn University
Department of Communications Engineering
Paderborn, Germany
{heymann, drude, haeb}@nt.uni-paderborn.de

Abstract

We present a system for the 4th CHiME challenge which significantly increases the performance for all three tracks with respect to the provided baseline system. The front-end uses a bidirectional Long Short-Term Memory (BLSTM)-based neural network to estimate signal statistics. These then steer a Generalized Eigenvalue beamformer. The back-end consists of a 22 layer deep Wide Residual Network and two extra BLSTM layers. Working on a whole utterance instead of frames allows us to refine Batch-Normalization. We also train our own BLSTM-based language model. Adding a discriminative speaker adaptation leads to further gains. The final system achieves a word error rate on the six channel real test data of $3.48\,\%$. For the two channel track we achieve $5.96\,\%$ and for the one channel track $9.34\,\%$. This is the best reported performance on the challenge achieved by a single system, i.e., a configuration, which does not combine multiple systems. At the same time, our system is independent of the microphone configuration. We can thus use the same components for all three tracks.

1. Introduction

Automatic speech recognition has become part of everyday life, not the least because current systems start to achieve remarkable results even in adversarial environments with severe noise conditions. These advances can mainly be attributed to stronger back-ends relying on Deep Neural Networks (DNNs) and by the processing of multiple input channels which can provide spatial selectivity to extract the signal of interest.

Indeed, today's powerful mobile devices are often equipped with multiple microphones, and thus multi-channel signal processing has become a more and more relevant approach to counterfeit more severe signal impairments due to noise or reverberation. The majority of multi-channel speech enhancement systems now consists of some kind of frequency domain beamforming approach. These beamforming systems traditionally relied on model based masking of time frequency (tf) bins [1, 2, 3, 4, 5, 6, 7] but more recently, more data driven and discriminatively trained masking approaches have been proposed [8].

Still, the dispute whether more emphasis should be put on a stronger front-end or just a deeper back-end remains. The latter has been pushed to an extreme, where the multi-channel waveforms or their short time Fourier representations are directly input to the DNN for acoustic modeling. It is then left to the network training to learn that fusion of the channels from the data, which is most effective for ASR performance [9, 10, 11].

We argue that the front-end should be as unobstrusive to the signal as possible. The danger of adding processing artifacts is worse than limited noise reduction and therefore we recommend to only use (masking based) linear beamforming. This still leverages all information contained in correlations between the individual channels and therefore leave all non-linear feature extraction to a deeper and more advanced back-end. Consequently, we employ an explicit acoustic beamforming component, thus taking advantage of recent progress in this field, e.g., by avoiding an explicit speaker localization component and by giving up the assumption of an anechoic environment. Still, we value the power of DNNs by using them for mask estimation: The beamforming coefficients, are estimated from signal statistics, more precisely, from the power spectral density matrices of the target speech and of the distortions. These statistics are obtained from spectral masks, which indicate for each tf bin, whether it is dominated by speech or by noise. And it is this mask estimation which is carried out by means of a DNN.

Nevertheless, a strong back-end is essential for good ASR performance. We employ a Wide Residual Network (WRN) with 22 layers for acoustic modeling. While this network architecture has been used successfully on image recognition tasks [12], it is adapted and used here for the first time for ASR.

The paper is organized as follows. In Sec. 2.1 we give an overview of our front-end design while the back-end is described in Sec. 2.2. The following section (3) shortly describes the database. Detailed experimental results are presented in Sec. 4. At the end we draw conclusions.

2. System Overview

2.1. Front-end

We use the Generalized Eigenvalue (GEV) beamformer which maximizes the signal-to-noise ratio (SNR) of the beamformer output in each frequency bin separately, leading to the beamformer coefficients [13]:

$$\mathbf{F}_{\mathrm{GEV}}(f) = \underset{\mathbf{F}(f)}{\operatorname{argmax}} \frac{\mathbf{F}(f)^{\mathrm{H}}\mathbf{\Phi}_{\mathbf{XX}}(f)\mathbf{F}(f)}{\mathbf{F}(f)^{\mathrm{H}}\mathbf{\Phi}_{\mathbf{NN}}(f)\mathbf{F}(f)}. \tag{1}$$

Here, $\mathbf{\Phi}_{\mathbf{XX}}(f)$ is the target and $\mathbf{\Phi}_{\mathbf{NN}}(f)$ the noise Cross-Power Spectral Density (PSD) matrix for the f-th frequency band. Please note that this does not require any assumptions (e.g., assuming an anechoic environment) regarding the nature of the Acoustic Transfer Function (ATF) from the speech source to the sensors or regarding the spatial correlation of the noise.

The maximization of the coefficient given in Eq. (1) is achieved by solving a generalized eigenvalue problem:

$$\mathbf{\Phi}_{\mathbf{XX}}\mathbf{F} = \lambda\mathbf{\Phi}_{\mathbf{NN}}\mathbf{F}, \tag{2}$$

where the eigenvector corresponding to the largest eigenvalue is the solution to Eq. (1).

This equation, however, does not impose a constraint on the norm of $\mathbf{F}$, and since each frequency is considered independently, this can introduce arbitrary speech distortions.

We handle these distortions by applying the following single channel post filter to the GEV output signal [13]:

$$g_{\mathrm{BAN}}(f) = \frac{\sqrt{\mathbf{F}_{\mathrm{GEV}}(f)^{\mathrm{H}} \mathbf{\Phi}_{\mathbf{NN}}(f) \mathbf{\Phi}_{\mathbf{NN}}(f) \mathbf{F}_{\mathrm{GEV}}(f)/D}}{\mathbf{F}_{\mathrm{GEV}}(f)^{\mathrm{H}} \mathbf{\Phi}_{\mathbf{NN}}(f) \mathbf{F}_{\mathrm{GEV}}(f)}, \tag{3}$$

where D is the number of microphones. This filter performs a so-called Blind Analytic Normalization (BAN) to obtain a distortionless response in the direction of the speaker: The overall ATF from the target source to the post filter output should have unit gain for every frequency bin. If this were achieved perfectly, speech distortions would be removed and one would eventually arrive at the Minimum Variance Distortionless Response (MVDR) beamformer [14, 15].

Another option is to normalize each beamforming vector to unit length. This leaves some distortions in the target signal but those can be handled by the acoustic model if the same kind of distortions occur both in training and test. Indeed, we found out that this matched training scenario even leads to slightly better results, compared to a training on the beamformer output signal after applying BAN. Here, however, we choose to use BAN because we want to train the acoustic model on all channels, and not only on the single beamformer output signal. Then BAN is necessary to reduce the mismatch between the six channels used for training and the beamformer output, which is used for recognition. The benefit of the six times larger training set size more than compensated for the slight loss due to using BAN.

To solve Eq. 2, signal statistics, namely the PSD matrices, are required. We estimate these using a mask based approach. Given non-overlapping masks, $M_{\mathbf{X}}$ for the target signal and $M_{\mathbf{N}}$ for the distortion, we estimate the PSD matrix by calculating the weighted sum of outer products of the microphone signals [16]:

$$\mathbf{\Phi}_{\nu\nu}(f) = \sum_{t=1}^{T} M_{\nu}(t,f) \mathbf{Y}(t,f) \mathbf{Y}(t,f)^{\mathrm{H}}, \tag{4}$$

where $\nu \in \{\mathbf{X}, \mathbf{N}\}$ and $\mathbf{Y}(t,f)$ is the vector of microphone signals at time frame t and frequency bin f.

To obtain an estimation of these masks given our observed signals, we utilize a neural network. Tbl. 1 details its configuration. The network operates on each channel independently yielding D masks for the target and D for the distortions. For each source the masks are condensed into a single mask by median pooling. We opted for this pooling operation because it makes the mask estimation more robust against channel failures compared to computing the average of the masks.

We do not force the values of the estimated masks to be one or zero. Rather, we restrict them to be in the range between one and zero using a Sigmoid non-linearity activation function for both estimates, i.e. we work with soft-masks.

We employ ADAM [17] for training. A fixed learning-rate of 0.001 and full backpropagation through time [18] is used. Additionally, if the norm of a gradient for this network is greater than one, we divide the gradient by its norm [19].

To achieve a better generalization, we use Dropout [20] for the input-hidden connection of the bi-directional Long Short-Term Memory (BLSTM) units [21] and for the input of the Rectified Linear Unit (ReLU) layers [20]. The dropout rate is fixed at $p = 0.5$ for every layer during the whole training. We do not use dropout for the last layer. Additionally we modified the SNR randomly in a range of $0\,\mathrm{dB}$ to $-7\,\mathrm{dB}$. We use the development data for cross-validation, stopping the training when the loss does not decrease anymore after 5 epochs of patience.

Table 1: Network configuration for mask estimation

	Units	Type	Non-Linearity	p_{dropout}
L1	256	BLSTM	Tanh	0.5
L2	513	FF	ReLU	0.5
L3	513	FF	ReLU	0.5
L4	1026	FF	Sigmoid	0.0

We apply Batch-Normalization (BN) [22] for each layer. Statistics for the BN are summarized along the time frame dimension. In contrast to the method proposed in [22], we do not use the population estimates obtained from the training or development data for the mean and variance at test time. Rather, we use the statistics of each utterance for each channel individually also for the test data.

The ideal binary masks used as training targets are defined as:

$$\mathrm{IBM}_{\mathbf{N}}(t,f) = \begin{cases} 1, & \frac{||\mathbf{X}(t,f)||}{||\mathbf{N}(t,f)||} < 10^{\mathrm{th}_{\mathbf{N}}(f)}, \\ 0, & \text{else}, \end{cases} \tag{5}$$

and

$$\mathrm{IBM}_{\mathbf{X}}(t,f) = \begin{cases} 1, & \frac{||\mathbf{X}(t,f)||}{||\mathbf{N}(t,f)||} > 10^{\mathrm{th}_{\mathbf{X}}(f)}, \\ 0, & \text{else}, \end{cases} \tag{6}$$

respectively.

The two thresholds $\mathrm{th}_{\mathbf{X}}$ and $\mathrm{th}_{\mathbf{N}}$ are not identical. Their values range from -5 to 10 depending on the frequency and are hand-tuned. They are chosen such that a decision in favor of speech/noise is only taken if the instantaneous SNR is high/low enough to ensure a low false acceptance rate. The network is trained on all utterances and all channels using the binary cross-entropy cost.

2.2. Back-end

2.2.1. Network configuration

For the back-end network we combine a slightly modified design of a WRN [12] with BLSTM layers. This configuration is motivated by the fact that each layer type has its own distinct advantages which complement each other in a unified architecture [23] and by recent findings about Convolutional Neural Networks (CNNs) by the image community [12, 24, 25]. An overview of the structure, which we call Wide Residual BLSTM Network (WRBN), is given in Fig. 3 while Fig. 2 and Fig. 1 detail the building blocks.

The first part of the network is composed of a WRN. The WRN consists of three residual building blocks which again consist of smaller building blocks (BlockA and BlockB). The difference between BlockA and BlockB is rather small. BlockA can reduce the frequency resolution by having a stride ≥ 1 and it increases the number of channels. Due to this change of the size of the tensor, a direct residual connection to the output of the block is not possible. BlockA therefore has an

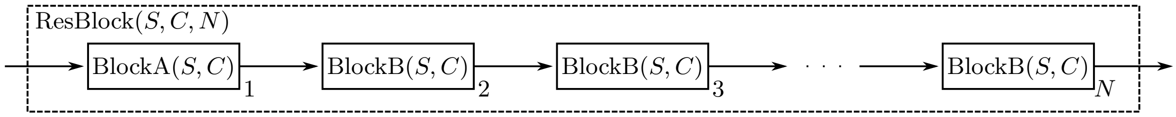

Figure 1: Detailed view of a ResBlock. A ResBlock is parameterized by its striding S, the number of output channels C and the number of inner blocks N. Accordingly, BlockB is repeated $N - 1$ times.

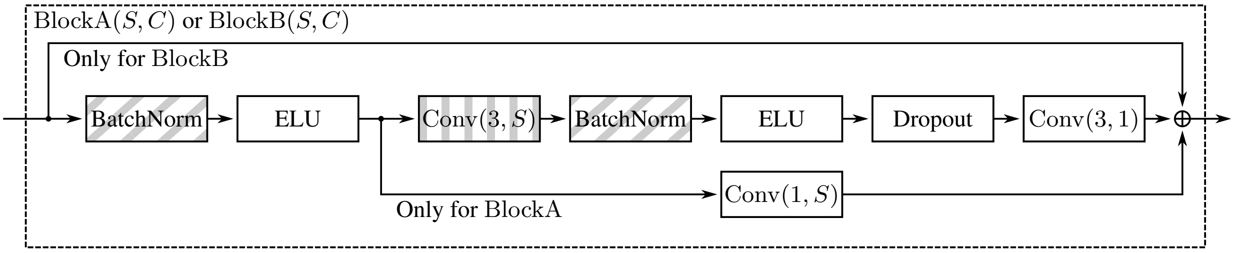

Figure 2: Detailed view of the building blocks BlockA or BlockB. The batch normalization collects statistics along the frequency band axis and along the time frame axis. A convolution block Conv(A, S) is parameterized by the filter size $A \times A$, the zero padding $(A - 1)/2$ in both directions and the consecutive striding S.

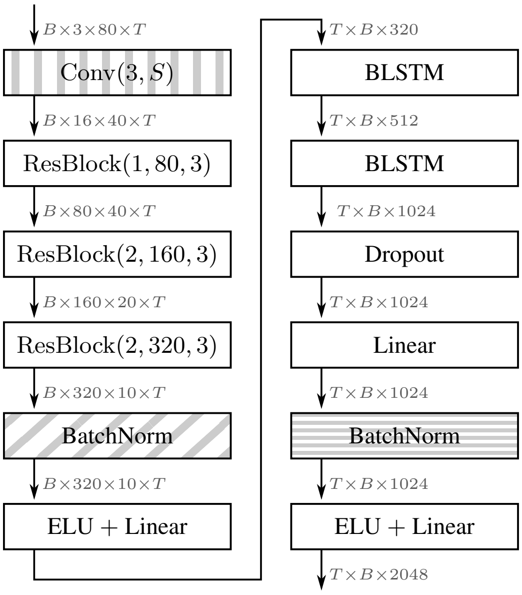

Figure 3: Overview of the back-end structure. The annotations in gray indicate the dimension of the tensors where B is the mini-batch size and T is the number of frames of the largest utterance within the batch. The building blocks are explained in Fig. 1 and Fig. 2. The convolution and the diagonally striped batch normalization is defined as in in Fig. 2. The horizontally striped batch normalization just collects statistics along the time frame axis.

additional convolution operation with filter size 1×1 which acts as the residual connection but also changes the size accordingly. Other than that, the two blocks are identical. A Batch-Normalization [22] normalizes the output of the preceding convolution and an Exponential Linear Unit (ELU) non-linearity [26] is applied afterwards. Before the last convolution of a block we use Dropout [20] with $p = 0.5$.

After the residual blocks, we get 320, each with a dimension of $10 \times T$ where T describes the number of frames and the first dimension can be interpreted as frequency bands. These are then weighted and combined for each channel with learnable weights resulting in a feature dimension of 320 per frame. These frames are used as the input for two consecutive BLSTM layers with 512 units for each direction. The output of the directions is merged by a sum after the first BLSTM layer and by a concatenation after the second BLSTM layer. To prevent overfitting we use Dropout on the input of each layer. Additionally we also use Dropout for the hidden-hidden transitions. Instead of sampling the dropout masks individually per frame, however, we sample the mask once per sequence with $p = 0.5$ [27]. This sampling strategy avoids losing temporal information as a result of Dropout.

The last part of the network consists of two feed-forward layers with Batch-Normalization and an ELU non-linearity. The final output are the posterior probabilities for the 2042 context-dependent states for each frame.

2.2.2. *Training*

We first extract the alignments with the baseline back-end and our front-end using all six channels (*Kaldi+GEV*). We then train our network with a cross-entropy criterion and Adam [17] with $\alpha = 10^{-4}$ on the unprocessed training data from all six microphones. We use 80 dimensional mean-normalized log-mel filterbank features as input. Their delta and delta-delta features are used for two additional input channels. We do not train the network on a window of n frames with truncated backpropagation. Instead, we train it on a whole utterance with full backpropagation through time. We see two main advantages in this strategy. First, the CNN and especially the BLSTM is able to exploit the full temporal context and we can avoid zero-padding within the utterance. Second, we can make efficient use of Batch-Normalization as described in the following.

2.2.3. Batch-Normalization

Batch-Normalization was first proposed in the context of image recognition and has been shown to improve convergence as well as generalization [22]. However, a drawback of this approach is that it relies on statistics accumulated on training and/or development data at test time. Calculating the statistics during test would lead to a dependency on the mini-batch constellation since the statistics are aggregated over the batch dimension. Here, we treat the batch dimension as an independent dimension. We can then calculate the statistics also at test time without losing determinism. This is possible because using the whole utterance we can get a reliable estimate of the statistics without including other utterances. Thus each utterance is normalized separately. For the tensors within the WRN we calculate the statistics over the height (frequency) and width (time) for each channel separately. For the other tensors we calculate the statistics over time and normalize the feature dimension.

2.2.4. Adaptation

For (speaker) adaptation, we train an additional layer consisting of a 80×80 weight matrix for each speaker and each track [28]. That layer with tied weights is applied to all three feature channels equally. Although CNNs can provide some translation invariance, we found that the additional transformation of the input features improves performance. It helps to reduce the mismatch between the unprocessed data at training time and the beamformed data at test time. We opted for the single layer since preliminary results got worse when we adapted the whole network or parts of it. Training is done by first decoding the utterances with our best speaker-independent model to get an alignment for each utterance for each track. We then prepend the layer to the network and train it with backpropagation for 5 epochs and $\alpha = 10^{-5}$.

2.2.5. Language model

The baseline system features three different language models. First, the search graphs are created using a standard 3-gram model provided by the WSJ database [29]. The graph is then rescored with a 5-gram Kneser-Ney [30] language model trained on the provided training data. Finally, the scores are interpolated (rescored) with a recurrent neural network language model [31]. Here, we aim to replace the latter by a stronger one. To this aim, we employ a two layer Long Short-Term Memory (LSTM) language model with 650 hidden units each – similar to the example provided by [32].

Instead of training on an endless word stream (initial state of next batch is end state of current batch), we found that training on complete sentences from the provided language model training data in a random mini-batch improved cross validation scores slightly (6 % relative word error rate (WER) improvement compared to an endless stream).

Again we use Adam [17] with the parameters proposed in the aforementioned paper for optimization for 39 epochs. The main benefit of using Adam besides a slightly improved WER was the fact, that a learning rate did not have to be tuned manually.

We experimented with ZoneOut [33] as a regularization technique for recurrent neural networks but ended up using regular dropout in the vertical connections only.

Global gradient clipping with a maximum value of 5 is used. All weight matrices and bias vectors, including the embedding matrix, are initialized with random weights sampled from a uniform distribution in $[-0.1, 0.1]$.

We experiment with restricted training sets limiting the maximum number of unknown symbols during training. This yielded reduced cross validation perplexities. Nevertheless, we finally selected a model trained on unrestricted training data, since this resulted in the lowest development test WERs.

Although, the training objective for the language model was perplexity, it turned out to beneficial to select the final language model based on the actual WER on the development set.

3. Database

The dataset from the fourth CHiME challenge [34] features three different tracks with real and simulated audio data of prompts taken from the 5k WSJ0-Corpus [29] with 4 different types of real-world background noise. The noise as well as the real utterances were recorded in a pedestrian, in a cafe, on the street and in a bus. The recording device was a tablet with six microphones mounted on its frame. The tracks were differentiated by the number of microphones used at test time. All were used in the six channel track, while in the two and one channel track the microphones were sampled randomly.

4. Experimental evaluation

Tbl. 2 gives an overview of all experiments and their results.

Concentrating on the effect of the front-end first, we can conclude that for the two channel track using our front-end (*Kaldi+GEV*) instead of the baseline front-end (*Baseline*) gives noticeable improvements in terms of WER. For the six channel track, just exchanging the front-end even decreases the WER by about 50 %, clearly showing the effectiveness of our approach. Nevertheless, there is still a big gap between the six channel and the two channel track. While this shows that our front-end is able to leverage additional microphones, it also shows that there is still room for improvements.

The advances in acoustic modelling are best visible for the one channel track. Compared to the Baseline, our proposed acoustic model achieves significantly lower WERs. Especially when comparing the results on the development and the test data, we can see that the gap is much smaller for our model, indicating its ability to generalize to different noise conditions. Looking at the six channel track we can conclude that the gap between the baseline model and our model gets smaller as the quality of the input signal improves (*Basline* vs. *WRBN+BFIT* and *Kaldi+GEV* vs. *WRBN+GEV*). This tendency is also visible for the two channel track.

For all tracks, we are able to further improve the results employing different methods presented in Sec. 2.2. The biggest gain here can be attributed to Batch-Normalization at test time.

Detailed results for the best system for each track are shown in Tbl. 3. Here, the results are splitted according to the four different environments. We can see that the more microphones we use, the less sensitive the result is to a specific environment. Especially for the one channel track the bus environment performs worse with the street environment having nearly half of the WER for the real test set.

Table 2: Average WER (%) for the tested systems. Bold results correspond to the officially submitted results. The individual abbreviations mean: ”Kaldi”: baseline backend, ”WRBN”: our WRBN (Section 2.2.1), ”+BN”: with Batch-Normalization (Sec. 2.2.3), ”+SA”: with additional linear speaker adaptation layer (Sec. 2.2.4) ”+NTLM”: with own language model (Sec. 2.2.5), ”+GEV”: with GEV beamformer (Sec. 2.1), ”+BFIT”: with baseline front-end beamformer

Track	System	Dev		Test	
		real	simu	real	simu
1ch	Baseline	11.57	12.98	23.70	20.84
	WRBN	6.64	9.09	11.8	13.78
	+BN	5.69	7.53	10.4	12.67
	+SA	5.5	7.18	9.88	11.68
	+NTLM	**5.19**	**6.69**	**9.34**	**11.11**
2ch	Baseline	8.23	9.50	16.58	15.33
	Kaldi+GEV	6.93	8.03	13.76	9.9
	WRBN+GEV	4.67	5.38	7.65	6.53
	+BN	4	4.76	6.96	6.22
	+SA	3.8	4.45	6.44	5.38
	+NTLM	**3.54**	**4.05**	**5.96**	**5.16**
6ch	Baseline	5.76	6.77	11.51	10.90
	Kaldi+GEV	3.7	3.72	5.66	4.34
	WRBN+BFIT	4.43	5.27	7.33	7.85
	WRBN+GEV	3.16	3.2	4.52	3.41
	+BN	3.06	2.99	4.07	3.51
	+SA	2.84	2.75	3.85	3.11
	+NTLM	**2.73**	**2.34**	**3.48**	**2.76**

5. Conclusions

Comparing the presented system with the baseline system, two components can be identified which provided significant improvements (on the order of 20 % – 50 %): first the neural network supported GEV beamformer turned out to be more effective than the baseline BeamformIt! [35] beamformer, and, second, the WRBN acoustic model significantly improved over the standard DNN backend. Further, the proposed batch normalization per utterance, the additional linear layer at the WRBN input for speaker adaptation, and the LSTM language model delivered additional improvements (each on the order of 5 % – 10 %). It is further worth mentioning that, up to the speaker adaptation, this is a single-pass recognition system. The described setup can be considered light-weight, as it is a single system and not a combination of multiple systems. While it achieved the best reported single-system results on the CHiME-4 challenge, even better error rates can be achieved by a system combination, as can be seen, e.g., in a companion paper [36].

6. Acknowledgments

This research was supported by the Deutsche Forschungsgemeinschaft (DFG) under Contract No. Ha3455/11-1. Computational resources were provided by the Paderborn Center for Parallel Computing.

Table 3: WER (%) per environment for the best system.

Track	Env	Dev		Test	
		real	simu	real	simu
1ch	BUS	6.82	5.41	13.22	8.07
	CAF	5.28	9.29	9.45	13.17
	PED	3.7	5.21	7.75	10.22
	STR	4.96	6.86	6.93	12.98
2ch	BUS	4.23	3.2	7.85	3.88
	CAF	3.61	5.4	5.79	5.85
	PED	2.86	3.67	4.97	5.21
	STR	3.44	3.92	5.23	5.7
6ch	BUS	2.92	2.14	3.76	2.71
	CAF	2.65	2.63	3.25	2.88
	PED	2.67	2.14	3.33	2.97
	STR	2.67	2.45	3.57	2.48

7. References

[1] H. Sawada, S. Araki, and S. Makino, “Underdetermined Convolutive Blind Source Separation via Frequency Bin-wise Clustering and Permutation Alignment,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 3, pp. 516–527, 2011.

[2] N. Ito, S. Araki, T. Yoshioka, and T. Nakatani, “Relaxed Disjointness Based Clustering for Joint Blind Source Separation and Dereverberation,” in *Acoustic Signal Enhancement (IWAENC), 2014 14th International Workshop on*, Sept 2014, pp. 268–272.

[3] D. H. T. Vu and R. Haeb-Umbach, “Blind Speech Separation Employing Directional Statistics in an Expectation Maximization Framework,” in *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, March 2010, pp. 241–244.

[4] N. Ito, S. Araki, and T. Nakatani, “Permutation-free Convolutive Blind Source Separation via Full-band Clustering based on Frequency-independent Source Presence Priors,” in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, May 2013, pp. 3238–3242.

[5] 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 *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, Dec 2015, pp. 436–443.

[6] S. Araki and T. Nakatani, “Hybrid Approach for Multichannel Source Separation Combining Time-frequency Mask with Multichannel Wiener Filter,” in *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2011, pp. 225–228.

[7] S. Araki, M. Okada, T. Higuchi, A. Ogawa, and T. Nakatani, “Spatial Correlation Model based Observation Vector Clustering and MVDR Beamforming for Meeting Recognition,” in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2016, pp. 385–389.

[8] J. Heymann, L. Drude, A. Chinaev, and R. Haeb-Umbach, “BLSTM supported GEV beamformer front-end for the 3RD CHiME challenge,” in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, Dec 2015, pp. 444–451.

[9] Y. Hoshen, R. J. Weiss, and K. W. Wilson, “Speech Acoustic Modeling from Raw Multichannel Waveforms,” in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, April 2015, pp. 4624–4628.

[10] T. N. Sainath, R. J. Weiss, K. W. Wilson, A. Narayanan, M. Bacchiani, and A. Senior, "Speaker Location and Microphone Spacing Invariant Acoustic Modeling from Raw Multichannel Waveforms," in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, Dec 2015, pp. 30–36.

[11] T. N. Sainath, R. J. Weiss, K. W. Wilson, A. Narayanan, and M. Bacchiani, "Factored Spatial and Spectral Multichannel Raw Waveform CLDNNs," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, 2016.

[12] S. Zagoruyko and N. Komodakis, "Wide Residual Networks," *CoRR*, vol. abs/1605.07146, 2016. [Online]. Available: http://arxiv.org/abs/1605.07146

[13] E. Warsitz and R. Haeb-Umbach, "Blind Acoustic Beamforming Based on Generalized Eigenvalue Decomposition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 5, pp. 1529–1539, July 2007.

[14] B. D. Van Veen and K. M. Buckley, "Beamforming Techniques for Spatial Filtering," *Digital Signal Processing Handbook*, 1997.

[15] U. K. Simmer, J. Bitzer, and C. Marro, "Post-filtering Techniques," in *Microphone Arrays*. Springer, 2001, pp. 39–60.

[16] M. Souden, S. Araki, K. Kinoshita, T. Nakatani, and H. Sawada, "A Multichannel MMSE-Based Framework for Speech Source Separation and Noise Reduction," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 9, pp. 1913–1928, Sept 2013.

[17] D. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[18] P. J. Werbos, "Backpropagation through time: what it does and how to do it," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, Oct 1990.

[19] R. Pascanu, T. Mikolov, and Y. Bengio, "Understanding the Exploding Gradient Problem," *CoRR*, vol. abs/1211.5063, 2012. [Online]. Available: http://arxiv.org/abs/1211.5063

[20] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014. [Online]. Available: http://jmlr.org/papers/v15/srivastava14a.html

[21] W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent Neural Network Regularization," *CoRR*, vol. abs/1409.2329, 2014. [Online]. Available: http://arxiv.org/abs/1409.2329

[22] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," *CoRR*, vol. abs/1502.03167, 2015. [Online]. Available: http://arxiv.org/abs/1502.03167

[23] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, "Convolutional, Long Short-Term Memory, fully connected Deep Neural Networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, April 2015, pp. 4580–4584.

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[25] ——, "Identity Mappings in Deep Residual Networks," *CoRR*, vol. abs/1603.05027, 2016. [Online]. Available: http://arxiv.org/abs/1603.05027

[26] D. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)," *CoRR*, vol. abs/1511.07289, 2015. [Online]. Available: http://arxiv.org/abs/1511.07289

[27] S. Semeniuta, A. Severyn, and E. Barth, "Recurrent Dropout without Memory Loss," *CoRR*, vol. abs/1603.05118, 2016. [Online]. Available: http://arxiv.org/abs/1603.05118

[28] V. Abrash, H. Franco, A. Sankar, and M. Cohen, "Connectionist Speaker Normalization and Adaptation," in *in Eurospeech*. Citeseer, 1995.

[29] J. Garofalo *et al.*, "CSR-I (WSJ0) complete," *Linguistic Data Consortium, Philadelphia*, 2007.

[30] R. Kneser and H. Ney, "Improved Backing-off for M-gram Language Modeling," in *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, vol. 1, May 1995, pp. 181–184 vol.1.

[31] T. Mikolov, S. Kombrink, A. Deoras, L. Burget, and J. Černocký, "RNNLM - Recurrent Neural Network Language Modeling Toolkit," in *Proceedings of ASRU 2011*. IEEE Signal Processing Society, 2011, pp. 1–4. [Online]. Available: http://www.fit.vutbr.cz/research/view_pub.php?id=10087

[32] W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent Neural Network Regularization," *arXiv preprint arXiv:1409.2329*, 2014.

[33] D. Krueger, T. Maharaj, J. Kramár, M. Pezeshki, N. Ballas, N. R. Ke, A. Goyal, Y. Bengio, H. Larochelle, A. C. Courville, and C. Pal, "Zoneout: Regularizing rnns by randomly preserving hidden activations," *CoRR*, vol. abs/1606.01305, 2016. [Online]. Available: http://arxiv.org/abs/1606.01305

[34] E. Vincent, S. Watanabe, A. A. Nugraha, J. Barker, and R. Marxer, "An Analysis of Environment, Microphone and Data Simulation Mismatches in Robust Speech Recognition," *Computer Speech and Language*, 2016, to appear.

[35] X. Anguera, C. Wooters, and J. Hernando, "Acoustic Beamforming for Speaker Diarization of Meetings," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 7, pp. 2011–2022, Sept 2007.

[36] T. Menne, J. Heymann, A. Alexandridis, K. Irie, A. Zeyer, M. Kitza, P. Golik, I. Kulikov, L. Drude, R. Schlüter, H. Ney, R. Haeb-Umbach, and A. Mouchtaris, "The RWTH/UPB/FORTH System Combination for the 4th CHiME Challenge Evaluation," 2016, to appear.