COMPACT NETWORK FOR SPEAKERBEAM TARGET SPEAKER EXTRACTION

Marc Delcroix[1], *Katerina Zmolikova*[*2], *Tsubasa Ochiai*[1],
Keisuke Kinoshita[1], *Shoko Araki*[1], *Tomohiro Nakatani*[1]

[1]NTT Communication Science Laboratories, NTT Corporation, Kyoto, Japan
[2]Brno University of Technology, Speech@FIT and IT4I Center of Excellence, Czech Republic
marc.delcroix@ieee.org

ABSTRACT

Speech separation that separates a mixture of speech signals into each of its sources has been an active research topic for a long time and has seen recent progress with the advent of deep learning. A related problem is target speaker extraction, i.e. extraction of only speech of a target speaker out of a mixture, given characteristics of his/her voice. We have recently proposed SpeakerBeam, which is a neural network-based target speaker extraction method. SpeakerBeam uses a speech extraction network that is adapted to the target speaker using auxiliary features derived from an adaptation utterance of that speaker. Initially, we implemented SpeakerBeam with a factorized adaptation layer, which consists of several parallel linear transformations weighted by weights derived from the auxiliary features. The factorized layer is effective for target speech extraction, but it requires a large number of parameters. In this paper, we propose to simply scale the activations of a hidden layer of the speech extraction network with weights derived from the auxiliary features. This simpler approach greatly reduces the number of model parameters by up to 60%, making it much more practical, while maintaining a similar level of performance. We tested our approach on simulated and real noisy and reverberant mixtures, showing the potential of SpeakerBeam for real-life applications. Moreover, we showed that speech extraction performance of SpeakerBeam compares favorably with that of a state-of-the-art speech separation method with a similar network configuration.

Index Terms— Target speech extraction, Neural network, Adaptation, Auxiliary feature, Speech enhancement

1. INTRODUCTION

In our everyday life, people's speech often overlaps with noise or other people's voice, making it hard to understand especially for automatic speech recognition (ASR) systems. Recently, there has been a great interest in using deep learning approaches to tackle this problem [1–4]. Most of the works focus on blind source separation (BSS) approaches such as deep clustering [2] or permutation invariant training (PIT) [4, 5], to separate a mixture of speech signals into each of its sources. For example, PIT consists of a neural network that predicts masks for extracting speech for each source in the mixture. The training of such a network is made possible thanks to the use of a permutation invariant loss. Deep learning-based BSS approaches have shown to be very successful at separating speech mixtures even when using a single microphone. However, BSS approaches may suffer from a *global source permutation issue*, i.e. the output associated with a given speaker may change depending on the mixtures being processed. Moreover, BSS approaches usually require knowing or estimating the number of sources in the mixture.

In many applications, we may not be interested in separating all speech signals, but rather *extract only speech of a target speaker*. BSS approaches may not be the best candidate to tackle this problem because of the global source permutation issue. We have recently proposed a speaker aware neural network-based approach called SpeakerBeam [6, 7] for target speaker extraction. SpeakerBeam uses an adaptation utterance spoken by a target speaker to extract auxiliary features representing his/her speech characteristics and adapts a speech extraction neural network to extract his/her voice based on these auxiliary features. By focusing only on extracting the target speaker voice, SpeakerBeam can mitigate the global source permutation problem of BSS approaches, and does not require information about the number of sources in the mixture.

The adaptation process used by SpeakerBeam is related to auxiliary feature based speaker adaptation of acoustic models for ASR. A common approach consists of concatenating i-vectors with the input of the neural network [8]. However, this approach was found to be not powerful enough for the target speech extraction task [6, 9]. Instead, we employed a factorized adaptation layer [6, 10], i.e. a hidden layer that consists of several parallel linear transforms weighted by weights derived from the auxiliary feature representing the speech characteristics of the target speaker. The factorized layer can greatly modify the behavior of the network based on the auxiliary features, which appears to be essential for target speech extraction. However, it requires a large number of parameters, and many matrix multiplications, which makes it computationally intensive, memory demanding, and therefore slow to train and use in practice.

In this paper, we investigate an alternative approach for adaptation to reduce the number of model parameters. Instead of the factorized layer, we perform adaptation by scaling the activation of a hidden layer of the speech extraction neural network, with weights derived from the auxiliary features. This approach is similar in concept to subspace learning hidden unit contributions (LHUC) [11] that was recently proposed for acoustic model adaptation. We present experimental results on the publicly available MERL data set, showing that this new architecture for SpeakerBeam greatly reduces the model size by up to 60% compared to previous versions of SpeakerBeam while maintaining a similar level of performance. This new implementation of SpeakerBeam also achieves competitive performance compared to a PIT-based BSS method, with Oracle target speaker selection from its outputs. Finally, we investigate SpeakerBeam for target speech extraction in more challenging conditions consisting of reverberant speech mixtures with background noise and music.

In the remainder of the paper, we describe SpeakerBeam and

[*]Katerina Zmolikova work was partly supported by Technology Agency of the Czech Republic project No. TJ01000208 "NOSICI".

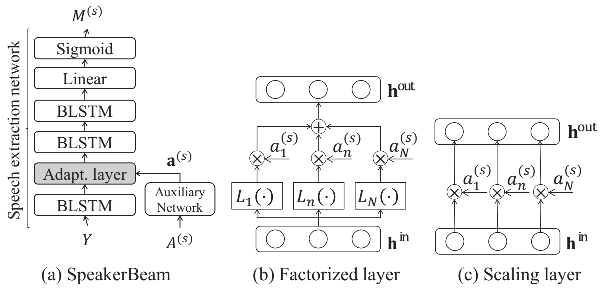

Fig. 1. Network architecture of SpeakerBeam. $A^{(s)} = \{\mathbf{a}_{t'}^{(s)}; t' = 1, \ldots, T'\}$ is the set of amplitude spectrum features of the adaptation utterance

the proposed scaling adaptation layer in Section 2. We discuss the relation of SpeakerBeam with related works in Section 3. Section 4 presents experimental results, and Section 5 concludes the paper.

2. SPEAKERBEAM

Figure 1-(a) is a schematic diagram of the network architecture of SpeakerBeam, which consists of a speech extraction network and an auxiliary network. The speech extraction network predicts a time-frequency mask, $M^{(s)}$, that can extract the target speaker, s, out of the mixture as,

$$\hat{X}^{(s)} = M^{(s)} \odot Y, \tag{1}$$

where $Y = \{\mathbf{y}_t; t = 1, \ldots, T\}$ is the set of amplitude spectrum features of the mixture signal, T is the number of time frames, $\hat{X}^{(s)}$ is the corresponding extracted speech, and $\odot$ is an element-wise product. The time-frequency mask of the target speaker is computed as,

$$M^{(s)} = F(Y, \mathbf{a}^{(s)}), \tag{2}$$

where $F(\cdot)$ is the non-linear transformations of the speech extraction neural network, and $\mathbf{a}^{(s)}$ is an auxiliary feature vector representing the characteristics of the target speaker. The speech extraction network consists of several bidirectional long-short-term memory (BLSTM) layers followed by a linear output layer with sigmoid activation to predict masks in the range $[0, 1]$. This architecture is similar to that of PIT-based separation networks [5], but SpeakerBeam inserts an adaptation layer after the first BLSTM layer to adapt the network to the target speaker. This layer inputs the auxiliary feature vector. In the following subsection, we describe the auxiliary feature used, the factorized adaptation layer, and the proposed scaling adaptation layer.

2.1. Auxiliary features

We can use i-vectors [12] or other features representing speech characteristics of the target speaker as auxiliary features. In our previous work [7], we found that better performance could be obtained when using a sequence summary network, which extracts the speech characteristics directly from the adaptation utterance as [13],

$$\mathbf{a}^{(s)} = \frac{1}{T'} \sum_{t'=1}^{T'} G(\mathbf{a}_{t'}^{(s)}), \tag{3}$$

where $\mathbf{a}_{t'}^{(s)}$ represents the amplitude spectrum of the t'^{th} frame of the adaptation utterance, T' is the number of frames of the adaptation utterance, and $G(\cdot)$ consists of an auxiliary neural network. The averaging operation maps the sequence of features of the adaptation utterance to a single vector.

2.2. Adaptation layer

2.2.1. *Factorized adaptation layer*

A common approach to realize speaker adaptation would consists of simply concatenating the target speaker characteristics with the input features of the speech extraction network. However, this realizes only adaptation of the bias of the input layer, which we found not powerful enough for target speech extraction [7]. Instead, we employed a factorized adaptation layer that realizes adaptation of the weight matrices and bias of the internal layers of the network, thus offers more control over the behavior of the network.

Figure 1-(b) is a schematic diagram of a factorized layer, which consists of a weighted-sum of N parallel linear transforms or factors as,

$$\mathbf{h}^{\text{out}} = \sum_{n=1}^{N} a_n^{(s)} L_n(\mathbf{h}^{\text{in}}), \tag{4}$$

where $\mathbf{h}^{\text{out}}$ and $\mathbf{h}^{\text{in}}$ are the output and input of the adaptation layer, $L_n(\cdot)$ and $a_n^{(s)}$ are linear transformations and factor weights associated with the n^{th} factor. The factor weight $a_n^{(s)}$ corresponds to the n^{th} component of the output of the auxiliary network, $\mathbf{a}^{(s)}$. The factor weights can control the internal behavior of the network so that it can focus on extracting the target speaker. Typically, based on our preliminary experiments, we use $N = 30$ factors. The size of the linear transforms is related to the size of the hidden layers U of the speech extraction network (i.e. size of $\mathbf{h}^{\text{in}}$), e.g. 512×512 in the experiments of Section 4. The factorized adaptation layer adds thus $N \times U \times U$ parameters, which significantly increases the model size. Such large models are not practical and slow to train because of their large memory requirements and computational complexity.

2.2.2. *Proposed scaling adaptation layer*

To reduce the model size, we propose using a scaling adaptation layer. Figure 1-(c) is a schematic diagram of a scaling adaptation layer. The output of the layer is obtained as an element-wise product of the auxiliary feature vector, $\mathbf{a}^{(s)}$, and the input of the layer,

$$\mathbf{h}^{\text{out}} = \mathbf{a}^{(s)} \odot \mathbf{h}^{\text{in}}. \tag{5}$$

Similar to the factorized adaptation layer, the internal behavior of the network can be changed based on the auxiliary features. The scaling activation layer is equivalent to inserting a diagonal matrix, whose diagonal corresponds to the coefficients of the auxiliary feature vector. Therefore, the scaling adaptation layer also adapts the internal weight matrix of the network but does not perform bias adaptation. Note that with the scaling adaptation layer, the size of the auxiliary feature N should match the size of $\mathbf{h}^{\text{in}}$, U, which means that we may have a larger output layer for the auxiliary network (e.g. $N = 512$) than in the case of the factorized adaptation layer (e.g. $N = 30$). However, this increased number of parameters of the auxiliary network is negligible compared to the size of the linear transforms used in the factorized adaptation case.

The scaling adaptation layer is similar to a gate function or to subspace-LHUC [11]. In subspace LHUC, a similar scaling of the activation was proposed for acoustic model adaptation. The weights were derived from i-vectors and constraint to be in a range $[0, 2]$ using a sigmoid function. Here, we control the weights directly from the adaptation utterance through the auxiliary network and do not constrain the weights $\mathbf{a}^{(s)}$ to a specific range.

2.3. SpeakerBeam training

Following the literature on deep-learning based speech separation, we employ the phase sensitive mean squared error (MSE) between masked signals and target speech as training criterion [5],

$$L = \sum_{s=1}^{S} ||M^{(s)} \cdot |Y| - |X^{(s)}| \max(\cos(\theta_Y - \theta_X), 0)||^2, \quad (6)$$

where S is the number of speakers in the mixtures, $M^{(s)} = F(Y, \mathbf{a}^{(s)})$ is the output of SpeakerBeam, $X^{(s)}$ is the sequence of amplitude spectrum features of the target speech, θ_Y and θ_X are the phase of the observed and target speech respectively.

The parameters of the speech extraction and the auxiliary networks are trained jointly from random initialization. To train SpeakerBeam, we require the triplets of speech mixtures, corresponding clean speech signals, and adaptation utterances of the target speakers. By training the network with mixtures from many different target speakers, it can learn to adapt its behavior for the extraction of target speakers unseen during training.

3. RELATED WORKS

Target speech extraction has received increased interest recently [14–16]. In [14], the authors derived a speech extraction method based on the deep attractor framework [17]. Like SpeakerBeam, they extracted a target speaker embedding vector from an adaptation utterance but concatenated it with embedding vectors of the mixture to form an attractor vector for the target speech. This approach appeared effective even for adaptation utterances of less than 1 sec. However, it remains to be tested with noise and reverberation.

In [14], the authors proposed using a d-vectors extractor [18] trained on a large data set instead of the summary network. For speech extraction, they simply concatenated the d-vector of the target speaker to the mixture processed by some convolutional layers, and input them to a BLSTM. In [14], the target speaker seems to be dominant over the interferences, which may explain why the simple concatenation approach worked well in their settings.

Finally, [16] proposed to select the target speaker out of the output of a deep attractor network. This approach requires an additional module for speaker selection that may introduce errors. Moreover, it cannot exploit the auxiliary information about the target speaker during the extraction process as SpeakerBeam does.

4. EXPERIMENTS

We performed experiments on two data sets, the MERL 2 mixture data set and a corpus of Japanese noisy and reverberant speech mixtures. Table 1 summarizes the characteristics of the two corpora.

4.1. Experimental settings

4.1.1. MERL 2 mixture data

We use the MERL 2 mixture data set [2] since it has been widely used for single-channel speech separation. The corpus consists of simulated mixtures of 2 speakers taken from the WSJ0 data set [19]. It contains no noise or reverberation. The signal-to-interference ratio (SIR) of the mixtures ranges from -5 to +5 dB. As adaptation utterance, we randomly selected a clean utterance of the target speaker different from that in the mixture.

Table 1. Characteristics of the corpora used for evaluation.

	MERL			Japanese mixtures		
	train	cv	test	train	cv	test
Duration	30 h	10 h	5 h	157h	2.7 h	2.3 h
No of Speakers	101	101	18	574	10	10
Sampling Freq.		8kHz			16 kHz	
SDR	-	-	0.15	-	-	-2.04
STOI	-	-	0.601	-	-	0.473

4.1.2. Japanese noisy and reverberant mixtures

We created a different corpus to test SpeakerBeam in noisy and reverberant conditions. The corpus consists of simulated mixtures of 2 speakers selected from the corpus of spontaneous Japanese (CSJ) corpus [20], and mixed at SIR randomly chosen between 0 and 20dB. The utterances were convolved with room impulse responses measured in an office room. Babble noise recorded in the same room was also added as well as music played by a single loudspeaker at signal-to-noise ratio (SNR) between 0 and 20 dB. The cross-validation (cv) and test sets consist of 10 different speakers each, which are different from those in the training set. As adaptation utterance, we randomly selected utterances of the target speakers that differ from the utterances in the mixtures. The adaptation utterances were reverberant and contained background babble noise, but no music or interference speakers. This corpus is much more challenging as seen by the signal to distortion ratio (SDR) [21, 22], and short term objective intelligibility (STOI) [23] baseline shown in Table 1. Note that these conditions are more severe than those investigated by most recent works on speech separation/extraction.

4.1.3. Network configuration

The input features of the speech extraction and auxiliary networks consisted of amplitude spectra of the speech mixture and adaptation utterance, respectively. The spectra were computed using STFT with 64 ms window length and 16 ms window shift. Consequently, the input of the network for the MERL data experiments was 257 dimensions and 513 for the Japanese mixtures because of the different sampling frequencies. Except for the input feature size, all other hyper-parameters were the same for all experiments.

We used the network architecture shown in Figure 1 for all investigations. The speech extraction network consisted of 3 BLSTM layers with 512 units for the forward and backward passes. Our BLSTM implementation included 512×1024 projection layers at the output of each BLSTM to map the concatenated outputs of the forward and backward LSTMS back to a 512-dimension vector. The projection layers were followed by a tanh activation function. The adaptation layer was inserted after the first BLSTM layer (after the projection, but before the tanh activation function). For the auxiliary network, we used 2 fully connected layers with 200 units and ReLu activation functions. The output layer of the auxiliary network consisted of a Linear layer followed by a time averaging operation and no additional activation function.

We compared SpeakerBeam with a PIT-based separation network with a similar network configuration, which also consisted of 3 BLSTM layers followed by two parallel linear output layers with sigmoid activation to output masks for each speaker in the mixture. Compared to PIT-based network, SpeakerBeam has an internal adaptation layer that increases its number of parameters. However, since the adaptation layer only performs a linear transformation (there is no additional non-linear activation function), simply adding a linear

Table 2. SDR and STOI results for MERL 2 mixture data set.

	Size	Δ SDR			Δ STOI
		diff	same	avg	
PIT	134 M	10.7	6.4	8.6	0.101
SpeakerBeam FA 10	160 M	11.3	7.4	9.4	0.106
SpeakerBeam FA 20	186 M	11.4	7.6	9.6	0.107
SpeakerBeam FA 30	212 M	11.5	7.9	9.7	0.110
SpeakerBeam SA	134 M	11.4	7.8	9.6	0.110

layer with similar size inside the PIT network should not cause any significant performance difference.

For training all networks, we used the Adam optimizer [24] with an initial learning rate of 0.0001 and ran the training for 200 epochs for the MERL data set, and 25 epochs for the Japanese noisy mixture set. We did not use dropout.

4.2. Results

The results of all experiments were obtained by averaging the extraction performance of both speakers in the mixture.

4.2.1. MERL data set

Table 2 shows the SDR and STOI improvements for PIT-based separation and SpeakerBeam with a factorized adaptation layer (SpeakerBeam-FA) with 10, 20 and 30 factors, and the proposed scaling adaptation layer (SpeakerBeam-SA). We show SDR improvement for mixtures of different (diff) and same (same) genders, as well as the averaged improvement (avg). Note that PIT performs separation and not target speech extraction. For a fair comparison, PIT-based separation should be combined with an additional module to identify the target speaker among the network outputs [16]. In these experiments, we used Oracle target speaker identification for PIT. Consequently, PIT results should be considered as an upper bound for PIT-based target speech extraction.

We observe that all investigated configurations of SpeakerBeam outperformed PIT for this task. Optimal performance was achieved with SpeakerBeam-FA with 30 factors. The performance gradually degraded when using fewer classes. The proposed SpeakerBeam-SA performed similarly to SpeakerBeam-FA with 30 factors although the network size is significantly smaller as indicated in Table 2.

4.2.2. Japanese noisy and reverberant mixtures

We trained a similar network on the corpus of Japanese noisy and reverberant mixtures. Table 3 shows the SDR and STOI improvements on that task. Although the improvement is smaller than for the MERL data, we can confirm that SpeakerBeam also works in presence of noise and reverberation. Moreover, it performs slightly better than PIT with oracle speaker selection in terms of average SDR and STOI.

As seen in Table 2 and 3, speech extraction performance is much better when processing mixtures of different genders. This performance gap is also apparent for PIT, indicating that it is challenging to separate speakers of same-genders. In addition, SpeakerBeam sometimes confuses the target speaker when it is of the same gender as the interference speaker. Such errors appeared more frequent when dealing with challenging noisy and reverberant conditions and may be responsible for the poorer performance for same gender mixtures in Table 3 compared with PIT that uses Oracle target speaker selection. Improving speaker discriminative performance in such conditions remains an important future research direction.

Table 3. SDR and STOI results for simulated Japanese mixtures.

	Δ SDR			Δ STOI
	diff	same	avg	
PIT	8.2	5.4	6.8	0.078
SpeakerBeam SA	9.1	4.9	7.0	0.092

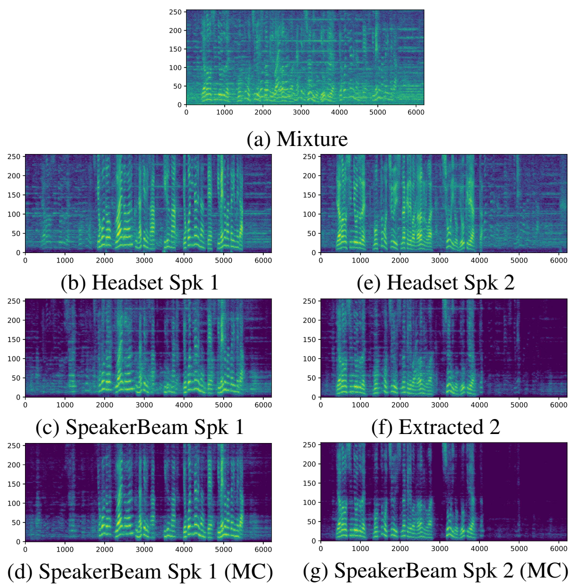

Fig. 2. Spectrograms of mixture, headset and SpeakerBeam outputs for single and multi-channel (MC) processing for real recordings.

4.2.3. Real recordings

We tested the model trained on Japanese noisy and reverberant mixtures on real recordings in a different office room with noise, reverberation, and music playing on the background. Figure 2 shows the spectrograms of (a) a speech mixture of a male and female speaker, (b),(e) headset microphone signals of both speakers, and extracted speech of the two speakers with (c),(f) single channel and (d),(g) multi-channel (MC) processing. For multi-channel processing, we combined SpeakerBeam with a GEV beamformer as described in [6]. The spectrograms clearly show that SpeakerBeam succeeded in extracting the target speakers. In addition, interested readers can refer to our demo videos [25] to evaluate the effect of SpeakerBeam on real recordings.

5. CONCLUSION

In this paper, we investigated a new architecture for SpeakerBeam based target speech extraction. We used a scaling adaptation layer instead of a factorized layer, therefore greatly reducing the network size while maintaining a similar level of performance. We tested the new architecture on MERL simulated mixture corpus, and on more challenging noisy and reverberant recordings. The latter experiments confirmed the potential of SpeakerBeam to tackle real-life conditions. Future work will include investigations on approaches to improve speech extraction performance of mixtures with similar voices.

6. REFERENCES

[1] D. Wang and J. Chen, "Supervised speech separation based on deep learning: An overview," *IEEE/ACM Trans. ASLP*, vol. 26, no. 10, pp. 1702–1726, 2018.

[2] J. R. Hershey, Z. Chen, J. Le Roux, and S. Watanabe, "Deep clustering: Discriminative embeddings for segmentation and separation," in *Proc. of ICASSP'16*, 2016, pp. 31–35.

[3] C. Weng, D. Yu, M. L. Seltzer, and J. Droppo, "Deep neural networks for single-channel multi-talker speech recognition," *IEEE Trans. ASLP*, vol. 23, no. 10, pp. 1670–1679, 2015.

[4] Y. Dong, K. Morten, T. Zheng-Hua, and J. Jesper, "Permutation invariant training of deep models for speaker-independent multi-talker speech separation," in *Proc. of ICASSP'17*, 2017, pp. 241–245.

[5] M. Kolbaek, D. Yu, Z. Tan, and J. Jensen, "Multitalker speech separation with utterance-level permutation invariant training of deep recurrent neural networks," *IEEE/ACM Trans. ASLP*, vol. 25, no. 10, pp. 1901–1913, 2017.

[6] K. Zmolikova, M. Delcroix, K. Kinoshita, T. Higuchi, A. Ogawa, and T. Nakatani, "Speaker-aware neural network based beamformer for speaker extraction in speech mixtures," in *Proc. of Interspeech'17*, Aug 2017, pp. 2655–2659.

[7] K. Zmolikova, M. Delcroix, K. Kinoshita, T. Higuchi, A. Ogawa, and T. Nakatani, "Learning speaker representation for neural network based multichannel speaker extraction," in *Proc of ASRU'17*, 2017, pp. 8–15.

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

[9] M. Delcroix, K. Zmolikova, K. Kinoshita, A. Ogawa, and T. Nakatani, "Single channel target speaker extraction and recognition with SpeakerBeam," in *Proc. of ICASSP'18*, 2018, pp. 5554–5558.

[10] M. Delcroix, K. Kinoshita, A. Ogawa, C. Huemmer, and T. Nakatani, "Context adaptive neural network based acoustic models for rapid adaptation," *IEEE/ACM Trans. ASLP*, vol. 26, no. 5, pp. 895–908, 2018.

[11] L. Samarakoon and K. C. Sim, "Subspace LHUC for fast adaptation of deep neural network acoustic models," in *Proc. of Interspeech'16*, 2016, pp. 1593–1597.

[12] N. Dehak, Patrick J. Kenny, Réda Dehak, Pierre Dumouchel, and Pierre Ouellet, "Front-end factor analysis for speaker verification," *IEEE Trans. ASLP*, vol. 19, no. 4, pp. 788–798, 2011.

[13] K. Vesely, S. Watanabe, K. Zmolikova, M. Karafiat, L. Burget, and J. H. Cernocky, "Sequence summarizing neural network for speaker adaptation," in *Proc. of ICASSP'16*, 2016, pp. 5315–5319.

[14] Q. Wang, H. Muckenhirn, K. Wilson, P. Sridhar, Z. Wu, J. Hershey, R. A. Saurous, R. J. Weiss, Y. Jia, and I. L. Moreno, "Voicefilter: Targeted voice separation by speaker-conditioned spectrogram masking," *arXiv:1810.04826*, 2018.

[15] J. Wang, J. Chen, D. Su, L. Chen, M. Yu, Y. Qian, and D. Yu, "Deep extractor network for target speaker recovery from single channel speech mixtures," *arXiv preprint arXiv:1807.08974*, 2018.

[16] L. Drude, T. von Neumann, and R. Haeb-Umbach, "Deep attractor networks for speaker re-identification and blind source separation," in *Proc. of ICASSP'18*, April 2018, pp. 11–15.

[17] Z. Chen, Y. Luo, and N. Mesgarani, "Deep attractor network for single-microphone speaker separation," in *Proc. of ICASSP'17*, 2017, pp. 246–250.

[18] L. Wan, Q. Wang, A. Papir, and I. L. Moreno, "Generalized end-to-end loss for speaker verification," in *in Proc. ICASSP'18*, 2018, pp. 4879–4883.

[19] J. Garofolo, "CSR-I (WSJ0) Complete LDC93S6A," `https://catalog.ldc.upenn.edu/ldc93s6a`, 1993.

[20] K. Maekawa, H. Koiso, S. Furui, and Isahara H., "Spontaneous speech corpus of Japanese," in *Proc. of LREC'00*, 2000, pp. 947–952.

[21] E. Vincent, R. Gribonval, and C. Févotte, "Performance measurement in blind audio source separation," *IEEE trans. ASLP*, vol. 14, no. 4, pp. 1462–1469, 2006.

[22] C. Raffel, B. McFee, E. J Humphrey, J. Salamon, O. Nieto, D. Liang, D. PW Ellis, and C C. Raffel, "mir_eval: A transparent implementation of common mir metrics," in *Proc. of ISMIR'14*, 2014.

[23] C. H Taal, R. C Hendriks, R. Heusdens, and J. Jensen, "An algorithm for intelligibility prediction of time–frequency weighted noisy speech," *IEEE Trans. ASLP*, vol. 19, no. 7, pp. 2125–2136, 2011.

[24] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arxiv*, vol. abs/1412.6980, 2014.

[25] "SpeakerBeam," *`https://youtu.be/7FSHgKip6vI` (English), `https://youtu.be/BM0DXWgGY5A` (Japanese)*, Cited Oct. 25 2018.