SpeakerBeam: Speaker Aware Neural Network for Target Speaker Extraction in Speech Mixtures

Kateřina Žmolíková, *Student Member, IEEE*, Marc Delcroix, *Senior Member, IEEE*, Keisuke Kinoshita, *Member, IEEE*, Tsubasa Ochiai, *Member, IEEE*, Tomohiro Nakatani, *Senior Member, IEEE*, Lukáš Burget, *Member, IEEE*, and Jan Černocký, *Senior Member, IEEE*

***Abstract*—The processing of speech corrupted by interfering overlapping speakers is one of the challenging problems with regards to today's automatic speech recognition systems. Recently, approaches based on deep learning have made great progress toward solving this problem. Most of these approaches tackle the problem as speech separation, i.e., they blindly recover all the speakers from the mixture. In some scenarios, such as smart personal devices, we may however be interested in recovering one target speaker from a mixture. In this paper, we introduce SpeakerBeam, a method for extracting a target speaker from the mixture based on an adaptation utterance spoken by the target speaker. Formulating the problem as speaker extraction avoids certain issues such as label permutation and the need to determine the number of speakers in the mixture. With SpeakerBeam, we jointly learn to extract a representation from the adaptation utterance characterizing the target speaker and to use this representation to extract the speaker. We explore several ways to do this, mostly inspired by speaker adaptation in acoustic models for automatic speech recognition. We evaluate the performance on the widely used WSJ0-2mix and WSJ0-3mix datasets, and these datasets modified with more noise or more realistic overlapping patterns. We further analyze the learned behavior by exploring the speaker representations and assessing the effect of the length of the adaptation data. The results show the benefit of including speaker information in the processing and the effectiveness of the proposed method.**

***Index Terms*—Speaker extraction, speaker-aware neural network, multi-speaker speech recognition.**

I. Introduction

AUTOMATIC speech recognition systems are now becoming widely deployed in real applications, which increases the need for robustness in adverse conditions. One particularly challenging problem, commonly occurring in spontaneous conversations and human-machine communication, is speech corrupted by interfering speakers. This type of interference has shown to be very difficult to reduce and greatly deteriorates the quality of speech transcriptions. Most of the research dealing with overlapping speech has focused on speech separation [1], [2], where all the source signals are recovered from the observed mixture signal. This problem has been studied in the past using methods, such as Computational auditory scene analysis [3], [4], Non-negative matrix factorization [5], [6] and Factorial Hidden Markov models [7], [8], and was greatly advanced recently thanks to deep learning based approaches [2], [9]–[11]. However, in some practical situations, such as smart personal devices, we may be interested in recovering a single target speaker while reducing noise and the effect of interfering speakers [12]–[15]. We call this problem *target speaker extraction*. In contrast to speech separation, extracting the target speaker avoids problems such as label permutation, dependence on the number of speakers and the speaker-tracing problem (see Section II-A for further discussion).

Most previous studies aiming to extract the target speaker [16]–[18] realized their aim by training a neural network on the target speaker data only, thus creating a model specifically designed to extract this particular speaker. The models are trained either in a speaker-pair-dependent mode, where both the target speaker and the interferer are observed in the training data, or in a target-dependent mode where the model can generalize to unseen interfering speakers. Both of these modes rely on the assumption of having substantial amount of data from the target speaker and do not allow the extraction of a speaker that was unseen during the training.

In this work, we follow the idea of target speaker extraction using a neural network, but rather than using a specialized model for a particular target speaker, we train a speaker independent model and inform it about the target speaker using additional speaker information. The network can use this information to focus on the target speaker, considering all the others as interference. We call this approach SpeakerBeam. The neural network in SpeakerBeam can be trained on a variety of speakers and employed to extract speakers unseen during the training. The additional speaker information determining the target speaker is obtained from an adaptation utterance spoken by the target speaker. In practice, this adaptation utterance could be obtained, for example, from part of a conversation without any overlap or pre-recorded by the target user on his/her personal device.

Manuscript received November 17, 2018; revised March 12, 2019 and May 3, 2019; accepted May 29, 2019. Date of publication June 13, 2019; date of current version July 25, 2019. This paper was supported in part by the Technology Agency of the Czech Republic project No. TJ01000208 "NOSICI," in part by the Czech National Science Foundation (GACR) project "NEUREM3" No. 19-26934X, and in part by the National Programme of Sustainability (NPU II) project "IT4Innovations excellence in science—LQ1602." The guest editor coordinating the review of this paper and approving it for publication was Dr. Michael Seltzer. *(Corresponding author: Kateřina Žmolíková.)*

K. Žmolíková, L. Burget, and J. Černocký are with Speech@FIT, Brno University of Technology, Brno 60190, Czech Republic (e-mail: izmolikova@fit.vutbr.cz; burget@fit.vutbr.cz; cernocky@fit.vutbr.cz).

M. Delcroix, K. Kinoshita, T. Ochiai, and T. Nakatani are with NTT Communications Science Laboratories, NTT Corporation, Kyoto 619-0237, Japan (e-mail: marc.delcroix@ieee.org; kinoshita.k@lab.ntt.co.jp; ochiai.tsubasa@lab.ntt.co.jp; tnak@ieee.org).

Digital Object Identifier 10.1109/JSTSP.2019.2922820

We explore different approaches for utilizing the information from adaptation utterances to cause the neural network to extract the target speaker. Most of these approaches are inspired by speaker adaptation of acoustic models. There are two main problems to be solved: a) *how to use the speaker information to modify the behavior of the neural network*, and b) *how to extract the speaker information from the adaptation utterance*. For the first problem, we look into three different methods: input bias adaptation [19]–[21], factorized layer [22] and scaled activations [23]. To extract the speaker information from the adaptation utterance, we can either use speaker representations that have been widely used for speaker identification tasks, such as i-vectors [24] or jointly learn the speaker representation using sequence summarization [21] or its modification with a simple attention mechanism.

In this paper, we first explain how this work relates to recent speech separation and extraction methods and our previous work (Section II). Then, we describe the proposed SpeakerBeam method and its variants (Section III). Section IV describes the integration of the method with multichannel processing and an automatic speech recognizer. Sections V and VI outline the datasets used and the experimental setup. Finally, the results are reported in Section VII and further analysis is provided in Section VIII.

II. Relationship to Previous Work

A. Related Speech Separation Work

Most recent work on the neural network processing of overlapped speech tackles the problem from a speech separation perspective, i.e. recovering all the sources from the given mixture. Compared with the separation of speech and non-speech signals (e.g. speech-noise or speech-music mixtures), where the individual sources have inherently different characteristics, speech-speech separation gives rise to several problems that require more specialized approaches. To introduce these problems, let us consider a simple approach, where the neural network processes a mixture signal and produces all the source signals as individual outputs. This approach suffers from the following problems:

1) *dependence on the number of speakers* — the architecture of the neural network inherently limits the number of speakers in the mixture, that can be processed.
2) *label-permutation problem* — the correspondence between outputs of the network and the speakers is arbitrary, therefore there are multiple possible correct outputs of the network where the speaker order varies. This makes it difficult to define the targets for the network and to compute the error function during training.
3) *speaker-tracing problem* — when processing a mixture with a network frame-by-frame or block-by-block, the order of the speakers on the output may change arbitrarily and proper alignment across the frames or blocks needs to be ensured.

The two main approaches that address neural network based speech separation are Deep clustering (DC) and Permutation Invariant Training (PIT). In DC [10], [25] and its variants [26], a neural network is used to compute embeddings for all time-frequency bins. These embeddings can be then clustered into group time-frequency bins corresponding to the same speaker. This solves the label-permutation problem as the estimated embeddings are ignorant as regards the order of the sources. The architecture of such a network is also independent of the number of speakers, although this number must be determined during the clustering step.

In PIT [11], [27], the neural network outputs estimations of all source signals. The main idea is to solve the label permutation problem by finding the permutation of the estimated sources on the output of the network that best matches the desired targets. Kolbæk *et al.* [27] have also shown that the same network can be used to process mixtures with different numbers of speakers as long as we can define a maximum, which can be a reasonable assumption in many scenarios. The objective of PIT is more closely related to the actual separation task than in DC and can be more easily combined with the joint training of e.g. an ASR system.

For the speaker tracing problem, both the DC and PIT methods rely on the ability of a recurrent architecture to keep its outputs consistent over time. In DC, the network should keep the embeddings for the same speaker in the same part of the embedding space, and in PIT, it should keep assigning the same speaker to the same output of the network. This has proven to work well in cases where the mixture is short and fully overlapped, but can cause problems for longer recordings or more complicated overlapping patterns, which naturally occur in real conversations.

The proposed SpeakerBeam method does not suffer from problems 1) and 2) as the neural network predicts the speech of the target speaker only. Additionally, it also solves the speaker tracing problem as the explicit speaker information enables the neural network to follow the same speaker over different frames or processing segments.

B. Relationship With Our Previous Work

We gradually built and refined the SpeakerBeam approach over several studies [28]–[31]. In this section, we clarify the relationship between this work and our previous research.

In [28], we first introduced the speaker-aware extraction scheme as part of a multi-channel system and experimented with different speaker-dependent neural network architectures. The work in [28] focused mainly on a closed-speaker-set case and evaluated a factorized layer scheme (see Section III) as the most suitable method. We later extended this method with sequence summarization in [29] to improve the performance in an open-speaker set scenario. Therein, we also evaluated SpeakerBeam as the front-end of an automatic speech recognition system. In [30], the automatic speech recognition performance was further improved by exploring the joint training of the SpeakerBeam front-end with an ASR system. While these studies [28]–[30] focused on a multichannel case, in [31], we investigated the ASR performance in a single-channel setting.

This paper builds upon the previous ones, summarizes the findings, and brings new modifications, evaluation and analysis. In particular, we provide a thorough evaluation of the

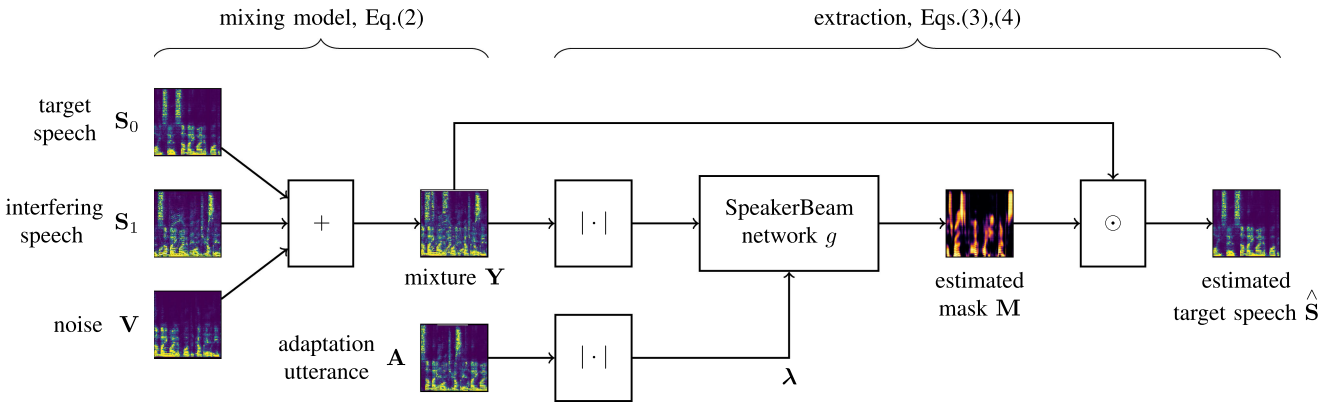


Fig. 1. Overall scheme of single-channel extraction for an example with one interfering speaker and noise.

single-channel scenario and different variants of SpeakerBeam on standard WSJ0-2mix and WSJ0-3mix datasets. We also create new WSJ0-2mix-long and WSJ0-2mix-noisy datasets to explore the effect of more natural overlapping patterns and a higher amount of noise on the results. Furthermore, we experiment with a combination of SpeakerBeam and DC, leading to improved performance. We finally provide an analysis of the learned embeddings and behavior with different lengths of adaptation utterance.

C. Related Speaker Extraction Work

After our proposal of SpeakerBeam in [28], several other studies followed the idea of extracting a target speaker using an adaptation utterance [15], [32], [33]. The authors of [15] built upon deep attractor networks [26] and suggested using the adaptation utterance to map the time-frequency points of the mixture into a canonical embedding space, where the embeddings corresponding to the target speaker are pulled together. The results show effectiveness even for very short adaptation utterances, however, the approach remains to be tested on a publicly available dataset or under more challenging conditions.

The work reported in [32] realized target speech extraction by combining speech separation and speaker identification. The authors proposed making use of embeddings in deep attractor networks to identify the target speaker in the extracted signals. This approach cannot exploit auxiliary information about the target speaker to improve the separation process. Moreover, it requires an additional module for speaker selection that may introduce speaker identification errors.

The method introduced in [33] proposes concatenating a d-vector [34] extracted from the adaptation utterance with one of the layers of the neural network to achieve the target speaker extraction. A similar way of using the speaker representation did not work well in our experiments (see 'input-bias' method in Section VII-A). This may possibly be due to the difference in the experimental settings, i.e. in [33], the target speaker was notably dominant over the interference (10.1 dB signal-to-distortion ratio), while in our experiments, the target and interference are equally strong on average (0.2 dB SDR).

III. Proposed SpeakerBeam Method

In this section, we formally define the problem of speaker extraction, introduce the notation we use and describe the proposed SpeakerBeam method. Figure 1 shows the overall scheme of the mixing model and the extraction.

A. Problem Definition

The problem of speaker extraction is to isolate the speech of a target speaker from an observed mixture of multiple overlapping speakers and optionally an additional noise. We assume a mixing model:

$$y^{(m)}[n] = s_0^{(m)}[n] + \sum_{i=1}^{I-1} s_i^{(m)}[n] + v^{(m)}[n], \qquad (1)$$

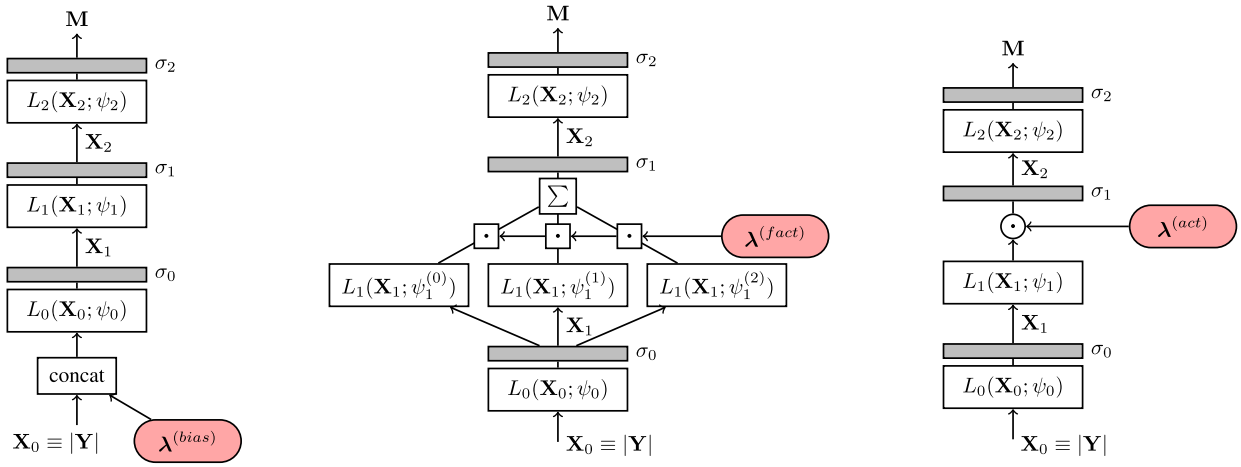
where n is the discrete time index, I is the number of speakers in the mixture, $s_i^{(m)}[n]$ for $i = 0, \ldots, I-1$ is the speech signal of the ith speaker as captured by microphone m with $i = 0$ being the target speaker, $v^{(m)}[n]$ is the additional noise and $y^{(m)}[n]$ is the observed mixture.

In this work, we perform the extraction in the short-time Fourier transform (STFT) domain, where we can model the mixing process as

$$Y^{(m)}[t,f] = S_0^{(m)}[t,f] + \sum_{i=1}^{I-1} S_i^{(m)}[t,f] + V^{(m)}[t,f]. \qquad (2)$$

Here, $[t, f]$ are the indexes corresponding to the time frame and the frequency bin and Y, S_i, V are the STFT-domain counterparts of y, s_i, v, respectively. We will use the notation $\mathbf{Y}$, $\mathbf{S}_i$, $\mathbf{V}$ for the $T \times F$ matrices comprising all time-frequency points $Y[t,f], S_i[t,f], V[t,f]$, respectively, with T being the number of time frames and F the number of frequency bins in the STFT representation of given signal. In the remainder of this section, we will focus on a single channel case (in this case, index $^{(m)}$ can be omitted). A multi-channel extension of SpeakerBeam will be addressed in Section IV.

Our method extracts the target speaker from the mixture, using additional information about the target speaker in the form of an *adaptation utterance*. This utterance will be denoted $a[n]$ in the time domain, $A[t,f]$ in the STFT domain and $\mathbf{A}$ for a $T_a \times F$



(a) Input bias adaptation. (b) Factorized layer. (c) Scaled activations.

Fig. 2. Three different methods of informing the neural network about the target speaker. The red box represents the speaker information $\boldsymbol{\lambda}$ derived from the adaptation utterance. $\boxdot$; denotes vector-scalar multiplication, $\odot$; denotes element-wise vector-vector multiplication.

matrix comprising all time-frequency points, where T_a is the number of frames in the adaptation utterance. The adaptation utterance $a[n]$ contains speech from the same speaker as $s_0[n]$, however it is always a different utterance from $s_0[n]$.

The extraction is performed by means of a neural network that takes the mixture as an input, the adaptation utterance as auxiliary information and provides a mask that can be used to obtain an estimate of the target speech:

$$\mathbf{M} = g(|\mathbf{Y}|, |\mathbf{A}|), \tag{3}$$

$$\hat{\mathbf{S}}_0 = \mathbf{M} \odot \mathbf{Y}, \tag{4}$$

where g is the transformation carried out by the mask estimation neural network, $|\cdot|$ denotes the magnitude of a given STFT signal, $\mathbf{M}$ is the estimated mask, $\odot$ denotes element-wise multiplication and $\hat{\mathbf{S}}_0$ is the estimated target-speaker STFT signal. In general, it would be possible to process directly the complex spectrum of the signals, but in this work, we limit ourselves to using the magnitudes only as in most of the related studies [10], [11], [26].

B. Informing the Network

Modifying the behavior of a neural network using additional speaker information is a task that has been heavily explored for speaker adaptation in acoustic models. The methods applied in our work are thus inspired by previous findings in this field. We explore three different ways of informing the network — input bias adaptation, a factorized layer and scaled activations, as depicted in Figure 2. Please note, that the figure depicts a rough schematic view of the network, and a more precise description of the layers and the architecture will be given in the System configuration sub-section in Section VI. All three methods make use of the speaker information $\boldsymbol{\lambda}$ (red box in Figure 2). In III-C, we will specify how $\boldsymbol{\lambda}$ is obtained from the adaptation utterance.

1) Input Bias Adaptation: The most straightforward technique, commonly used in acoustic modeling, is to append the speaker information to the features on the input of the neural network [19]–[21]. This effectively performs the adaptation of the biases in the first layer of the network [22]. We can express the neural network processing as

$$\mathbf{X}_1 = \sigma_0(L_0([\mathbf{X}_0, \boldsymbol{\lambda}^{(\text{bias})}]; \psi_0)), \tag{5}$$

$$\mathbf{X}_{k+1} = \sigma_k(L_k(\mathbf{X}_k; \psi_k)) \quad \text{for } k \geq 1, \tag{6}$$

where $\mathbf{X}_k$ is the input to the kth layer, $L_k(\mathbf{X}_k, \psi_k)$ is the transformation computed by the kth layer parameterized by ψ_k, and σ_k is an activation function. For example, with fully connected layers, $\psi = \{\mathbf{W}, \mathbf{b}\}$ and $L(\mathbf{X}, \psi) = \mathbf{W}\mathbf{x} + \mathbf{b}$, where $\mathbf{W}$ is a weight matrix and $\mathbf{b}$ is a bias vector.

2) Factorized Layer: Previous literature has shown, that a more powerful adaptation than simply adapting the input bias can be achieved by modifying all the parameters in one of the layers of the network. In a method introduced in [35], one of the layers of the network is factorized into multiple sublayers, which are then combined using weights derived from the speaker information. Following the previous notation and denoting the index of the factorized layer q and the number of sub-layers as J, the network processing is defined as

$$\mathbf{X}_{k+1} = \begin{cases} \sigma_k(L_k(\mathbf{X}_k; \psi_k)) & \text{for } k \neq q, \\ \sigma_k(\sum_{j=0}^{J-1} \lambda_j^{(\text{fact})} L_k(\mathbf{X}_k; \psi_k^{(j)})) & \text{for } k = q. \end{cases} \tag{7}$$

The network thus learns common basis for all the speakers, which then can be combined with different weights $\boldsymbol{\lambda}^{(\text{fact})}$ to make the network extract different speakers. The size of vector $\boldsymbol{\lambda}^{(\text{fact})}$ is determined by the number of factorized sub-layers J, which is chosen as a hyper-parameter.

3) Scaled Activations: An alternative speaker adaptation method is introduced in [23], [36], where the output of each unit in one of the layers of the network is scaled by weights

derived from the speaker information. This method is similar to the factorized layer approach, however it is computationally simpler. In this case, the processing performed by the neural network is:

$$\mathbf{X}_{k+1} = \begin{cases} \sigma_k(L_k(\mathbf{X}_k; \psi_k)) & \text{for } k \neq q, \\ \sigma_k(\boldsymbol{\lambda}^{(\mathrm{act})} \odot L_k(\mathbf{X}_k; \psi_k)) & \text{for } k = q. \end{cases} \tag{8}$$

Here the size of vector $\boldsymbol{\lambda}^{(\mathrm{act})}$ is determined by the size of the adaptive layer, rather than the number of factorized sub-layers as in the previous approach. Note that although the method introduced here follows the same idea as in [23], [36], it differs slightly in how the scaling weights are obtained and where exactly they are applied.

C. Obtaining the Speaker Information

In this section, we describe methods for extracting speaker information $\boldsymbol{\lambda}$ from an adaptation utterance, which is then used to inform the network, as described in the previous section. We explore three different methods — i-vector based extraction, a sequence summarizing network and its extension using simple attention.

1) I-Vectors: A common way to represent speaker-related information in speech data is the i-vector, which has been used extensively for e.g. speaker recognition [24], speaker adaptation [19], [37] or speaker diarization [38]. I-vectors are fixed-length low-dimensional representations of speech segments of variable length. For more information on i-vector extraction, we refer the reader to [24], [39]. In our work, we extract the i-vector from the adaptation utterance and post-process it with an auxiliary network to obtain the vector $\boldsymbol{\lambda}$ used in one of the three schemes presented in the previous section.

2) Sequence Summarizing Network: Although i-vector extraction is designed to preserve speaker variability, it uses a separate step, which is not optimized for the target speaker extraction task we are addressing. Therefore, some information important for speaker extraction may be lost. The second method we propose applies the adaptation utterance directly to the input of the auxiliary network. To convert from frame-wise features to an utterance-wise vector, we employ average pooling after the last layer in the auxiliary network. This way, the extraction of speaker information from the adaptation utterance may be learned jointly with the speaker extraction:

$$\bar{\boldsymbol{\lambda}}_t = z(|\mathbf{A}|), \tag{9}$$

$$\boldsymbol{\lambda} = \frac{1}{T_a} \sum \bar{\boldsymbol{\lambda}}_t, \tag{10}$$

where z is the transformation performed by the auxiliary neural network, $\bar{\boldsymbol{\lambda}}_t$ is the frame-wise vector extracted by the auxiliary network for frame t, which is then averaged over the T frames in the adaptation utterance to obtain the final $\boldsymbol{\lambda}$.

3) Sequence Summarizing Network With Attention: The average pooling at the end of the auxiliary network weighs all frames equally. This may be detrimental when some of the frames are silence or for example, corrupted by noise. To make the scheme more flexible, we extend it with a simple attention mechanism. Here, the output of the auxiliary network is extended with one value, $\bar{a}_t$. This predicted value for each frame is then used, after a softmax operation, to weigh the contribution of the individual frames to the averaging operation:

$$(\bar{\boldsymbol{\lambda}}_t, \bar{a}_t) = z(|\mathbf{A}|), \tag{11}$$

$$\mathbf{a} = \mathrm{softmax}(\bar{\mathbf{a}}), \tag{12}$$

$$\boldsymbol{\lambda} = \sum a_t \bar{\boldsymbol{\lambda}}_t, \tag{13}$$

where $\bar{\mathbf{a}} = [\bar{a}_1, \ldots, \bar{a}_{T_a}]$ denotes the attention energies (before the $\mathrm{softmax}$), and $\mathbf{a} = [a_1, \ldots, a_{T_a}]$ is the final attention vector, after the $\mathrm{softmax}$ normalization.

D. Training Objective

The neural network in SpeakerBeam estimates a T-F mask corresponding to the target speech. Different choices for the objective function for training the mask estimation networks have been previously explored in the literature. Here, we follow the findings in [40], which show that a good choice for the objective function is the mean square error between the magnitude of the STFT of the desired speech and the magnitude of the STFT of the observation, masked by the estimated mask. In addition, we also weigh the different time-frequency points using the phase differences between the clean and observed signals as suggested in [27]. This leads to an objective function with the form:

$$J_{\mathrm{spkbeam}} = ||\mathbf{M} \odot |\mathbf{Y}| - |\mathbf{S}_0| \odot \max(0, \cos(\boldsymbol{\theta}_y - \boldsymbol{\theta}_{s_0}))||^2, \tag{14}$$

where $\boldsymbol{\theta}_y$ and $\boldsymbol{\theta}_{s_0}$ are the $T \times F$ matrices of the phases of observed speech and target speaker speech, respectively.

We also explore the multi-task training of SpeakerBeam together with the Deep clustering method, in a similar fashion to that employed for Chimera networks [41], where the DC objective serves as a regularizer in a singing voice separation task. In this case, the neural network has two output layers, one predicting a mask for SpeakerBeam and the other predicting embeddings for Deep clustering

$$(\mathbf{M}, \mathbf{E}) = g(|\mathbf{Y}|, |\mathbf{A}|), \tag{15}$$

where $\mathbf{E}$ is the matrix of the embeddings. The objective function of the training is then computed as the average of the SpeakerBeam and Deep clustering objective functions

$$J_{\mathrm{spkbeam+dc}} = \alpha J_{\mathrm{spkbeam}} + \beta J_{\mathrm{dc}}, \tag{16}$$

where α, β are interpolation weights. In this paper, we set α and β so that both objectives are in approximately the same range ($\alpha = 0.5$, $\beta = 0.5e^{-5}$). For details on the computation of the objective function for deep clustering J_{dc} from the estimated embeddings $\mathbf{E}$, please refer to [10], [25].

IV. Integration With Beamforming and ASR

In this section, we describe how to integrate the SpeakerBeam method with beamforming and an ASR-level objective criterion.

A. Multi-Channel Extraction

Following the single-channel procedure, the neural network estimates a mask corresponding to the target speech in the mixture. The mask is estimated for each channel separately and the overall mask is obtained as a median across all the channels. The resulting mask is then used to accumulate statistics about the target signal and compute statistically optimal beamforming filters. Finally, to obtain the estimate of the target speech, the filters are applied to the multi-channel signal. The procedure for estimating the statistics of the target signal can be described as

$$\mathbf{M}^{(m)} = g(|\mathbf{Y}^{(m)}|, |\mathbf{A}|), \tag{17}$$

$$\mathbf{M} = \text{median}(\mathbf{M}^{(m)}), \tag{18}$$

$$\Phi_{SS}[f] = \frac{\sum_t M[t,f]\mathbf{y}[t,f]\mathbf{y}^H[t,f]}{\sum_t M[t,f]}, \tag{19}$$

$$\Phi_{NN}[f] = \frac{\sum_t (1-M[t,f])\mathbf{y}[t,f]\mathbf{y}^H[t,f]}{\sum_t (1-M[t,f])}, \tag{20}$$

where $\mathbf{M}^{(m)}$ is the estimated mask for channel m and Φ_{SS}, Φ_{NN} are the spatial co-variance matrices (SCM) corresponding to the target speech and interference, respectively. $\mathbf{y}[t,f] = [Y^{(1)}[t,f], \ldots, Y^{(M)}[t,f]]$ is a vector comprising the observed signal at time-frequency point $[t,f]$ for all microphones. Different beamforming filters, such as the Generalized Eigenvalue beamformer (GEV) [42] or Minimum variance distortionless response (MVDR) beamformer [43], can then be computed using the estimated Φ_{SS}, Φ_{NN}. In this work, we use the GEV beamformer defined as

$$\mathbf{h}_{\text{GEV}}[f] = \underset{\mathbf{h}}{\text{argmax}} \frac{\mathbf{h}^H[f]\mathbf{\Phi}_{SS}[f]\mathbf{h}[f]}{\mathbf{h}^H[f]\mathbf{\Phi}_{NN}[f]\mathbf{h}[f]}, \tag{21}$$

where $\mathbf{h}$ is a beamforming filter. The computed beamforming filters then can be used to estimate the target signal as

$$\hat{S}[t,f] = \mathbf{h}^H[f]\mathbf{y}[t,f]. \tag{22}$$

This procedure for neural network mask-based beamforming was proposed for speech denoising [44], [45] and has been shown to be very effective. Estimating the mask by using the neural network from each channel separately ensures independence from microphone array configuration, and averaging across time and channels when computing the statistics provides robustness against small errors made by the network. Additionally, speech produced by the linear filtering process is better suited for processing by automatic speech recognition systems than signals produced by masking as in Eq. (4).

B. Joint Training With ASR

For a case where SpeakerBeam is used in a chain with beamforming and the ASR acoustic model, we also explore the option of training it jointly with the acoustic model, using the cross-entropy between the estimated tied-state distribution and the true tied-state labels, J_{asr}. To train the SpeakerBeam network, this objective is then back-propagated through the acoustic model, feature extraction and the beamforming process:

$$\frac{\partial J_{asr}}{\partial \boldsymbol{\psi}} = \frac{\partial J_{asr}}{\partial \hat{\mathbf{s}}_{\text{fbank}}} \frac{\partial \hat{\mathbf{s}}_{\text{fbank}}}{\partial \hat{\mathbf{S}}} \frac{\partial \hat{\mathbf{S}}}{\partial \mathbf{M}} \frac{\partial \mathbf{M}}{\partial \boldsymbol{\psi}}, \tag{23}$$

where $\hat{\mathbf{s}}_{\text{fbank}}$ are the features extracted from the estimated signal, $\hat{\mathbf{S}}$ is the STFT of the estimated signal, $\mathbf{M}$ are the estimated masks and $\boldsymbol{\psi}$ is the vector of the parameters of the SpeakerBeam neural network. Most of the gradients can be computed using backpropagation through standard neural network blocks. For gradient $\partial\hat{\mathbf{S}}/\partial\mathbf{M}$, we need to backpropagate through a GEV beamformer, in particular through complex eigenvalue decomposition. This step was thoroughly covered in [46].

V. Datasets

For our evaluation, we chose the dataset introduced in [10], which has been used in many previous studies [10], [25]–[27]. It consists of simulated mixtures based on utterances taken from the Wall Street Journal (WSJ0) corpus [47]. For different experiments, we report results for four different versions of the dataset, namely *WSJ0-2mix* [10] for single-channel 2-speaker experiments, *WSJ0-3mix* [10] for single-channel 3-speaker experiments, *WSJ0-2mix-MC* [48] for multi-channel experiments and our own modification of WSJ0-2mix, *WSJ0-2mix-long*, which consists of single-channel 2-speaker mixtures with a longer duration and more complicated overlapping pattern and *WSJ0-2mix-noisy*, where we mixed additional noise into the mixtures. In the following, we describe these sets in detail. With all datasets, the adaptation utterances are randomly chosen. In evaluation set, for each mixture and each speaker in the mixture, we randomly choose one utterance from the same speaker, different than the utterance in the mixture, to be the adaptation utterance. For training set, for each mixture and each speaker, we randomly choose 100 adaptation utterances which we iterate through over the training epochs (the same adaptation utterance may be repeated). The choice for both evaluation and training is fixed for all experiments.

A. WSJ0-2mix and WSJ0-3mix

The WSJ0-2mix [10] contains mixtures of two speakers at signal-to-noise ratios between 0 dB and 5 dB. It consists of a training set, a cross validation set and an evaluation set of 30, 10 and 5 hours, respectively. For training and cross-validation sets, the mixed utterances were randomly selected from the `si_tr_s`, while for evaluation set, the utterances were taken from `si_dt_05` and `si_et_05` parts of WSJ0. In total, the training set contains 20000 mixtures from 101 speakers, the cross-validation set contains 5000 mixtures from the same 101 speakers and the evaluation set contains 3000 utterances from 18 speakers (unseen in the training). The WSJ0-3mix [25] contains three-speaker mixtures analogous to WSJ0-2mix in terms of the amounts of data, number of speakers and WSJ0 sets from which the utterances are selected. All data are used at an 8 kHz sampling rate for consistency with previous studies. In experiments evaluating only signal-based measures, we use "min" versions

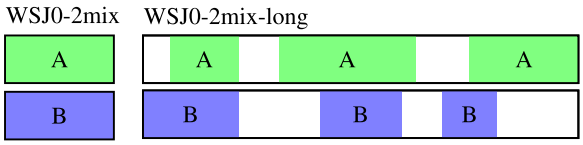


Fig. 3. Type of mixtures in datasets WSJ0-2mix and WSJ0-2mix-long. The first row corresponds to speech from speaker A, the second row to speech from speaker B.

of the datasets, where the mixture is cut to the length of the shortest utterance (for consistency with previous work). However, to be able to evaluate ASR accuracy in Sections VIII.D, VIII.E, VIII.F, we use the "max" version, where the shorter utterance in the mixture is padded with zeros.

B. WSJ0-2mix-long

In addition to the original WSJ0-2mix, we created a dataset that aims to model more realistic overlapping conditions, similar to those occurring in natural conversations. The mixing process followed the procedure used to create WSJ0-2mix, but for each of the speakers in the mixture, we selected 3 random utterances and placed them in sequence with random pauses in between (sampled uniformly in the 0-10 seconds range). This resulted in a dataset of mixtures with an average length of 45 seconds and an average overlap of 20%. Figure 3 shows a schematic comparison of the types of mixtures in WSJ0-2mix and WSJ0-2mix-long.

C. WSJ0-2mix-MC

The WSJ0-2mix-MC [48] dataset is a spatialized version of WSJ0-2mix. It is created by convolving the data with room impulse responses generated with the image method [49], [50] to simulate an 8-channel microphone array. The room characteristics, speaker locations and microphone array geometry are randomly generated — microphone array sizes range from 15 to 25 cm, T60 is drawn from 0.2-0.6 seconds. The average distance of a speaker from the array is 1.3 m with a 0.4 m standard deviation.

D. WSJ0-2mix-noisy

The WSJ0-2mix-noisy dataset is equivalent to WSJ0-2mix, but with additional noises added to the mixtures. The noises were randomly selected from the CHiME-1 [51] and CHiME-3 [52] corpora. The CHiME-1 noises were recorded in a living room, and thus contain noises from typical domestic environments and often children's speech. The CHiME-3 noises are from four environments — buses, streets, cafes and pedestrian areas. We split the noises into training and test subsets and mixed them into the mixtures at SNRs of 20 dB to 0 dB (with respect to the mixture signal).

VI. System Configuration

A. Speaker Extraction Neural Network Settings

In the experiments, we used two different neural network configurations. The first and smaller configuration, is used to compare the different techniques of informing the neural network about the speaker in the SpeakerBeam scheme. For all the following experiments, we used the larger configuration. The small configuration consisted of one BLSTM and three fully connected layers. All the layers used ReLU activations and batch normalization, except for the output layer with logistic sigmoid activation. The numbers of neurons in the layers were 300-1024-1024-257. The larger configuration consisted of 3 BLSTM layers, each followed by a linear projection layer and one linear output layer. The BLSTM layers had 512 units per direction and their output of dimensionality 1024 (512 forward + 512 backward) was then transformed by the projection layer back to dimension 512. Each projection layer was followed by tanh nonlinearity. The larger configuration is depicted in Figure 4. For both the factorized layer and scaled activations methods, the second layer was used as speaker adaptive layer. With the factorized layer, it was split into 30 sub-layers. For the input-bias method, the dimension of the appended speaker vector (extracted by the auxiliary network) was 100. The networks were trained with an Adam optimizer with a learning rate $1e-4$. With the larger configuration, we did not use dropout, or batch normalization (in contrast with the smaller configuration where batch normalization was used). The network parameters were initialized using the Glorot initialization [53]. The neural networks used for comparison with DC and PIT had the same architecture apart from the last layer, which was $I \times 257$ for PIT (predicting masks for all the speakers) and $D \times 257$ for DC, where $D = 30$ is the embedding size.

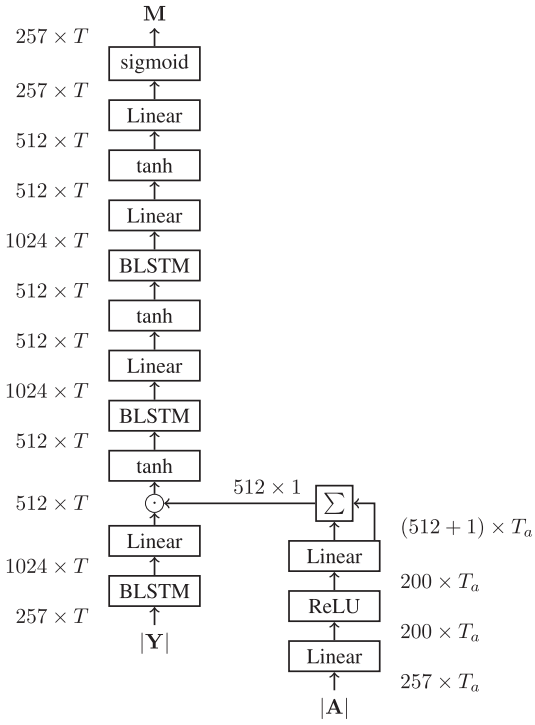


Fig. 4. Final configuration of the neural network for SpeakerBeam with the scaled activations method and sequence summarization with attention. For more details, see Equations (8), (11)–(13).

B. Speaker Information Extraction Settings

For i-vectors, we used a Kaldi i-vector extractor [54], trained on clean data. The Universal Background model we used consisted of 2048 Gaussians, and the i-vectors were 100-dimensional. The i-vectors were computed per utterance.

As the auxiliary network, we used a network with 2 fully connected layers with 200 units per layer and ReLU activations. The output layer had linear activation. Its size was determined by the method used (see Figure 2). The auxiliary network was trained jointly with the main network.

C. Beamforming Settings

The beamforming was undertaken in the STFT domain with 20 ms windows and a 10 ms shift. We used a GEV beamformer as specified in IV-A. We regularized the noise spatial co-variance matrix by adding $1e^{-3}$ to its diagonal to stabilize its inversion. The output signal was post-processed with a single-channel post-filter [42].

D. ASR Settings

The input acoustic features were 40-dimensional log Mel filterbanks with a context window extension of 11 frames. The features were mean-normalized per utterance. For the acoustic model, we used a simple DNN with 5 fully connected hidden layers of 2048 units each and ReLU activation functions. For training, we used HMM tied-state alignments obtained from single-channel clean data using a GMM-HMM system.

VII. Experiments

This section provides an experimental evaluation of our approach. We compare the different methods used to inform the neural network about the target speaker and compare the performance with DC and PIT. We also explore the effectiveness with mixtures containing three speakers. Then, we explore the performance with noisy and multichannel data. All the experiments are evaluated using the signal-to-distortion ratio (SDR) (as defined by [55] and computed using [56]) or the frequency-weighted signal-to-noise ratio (fw-SNR) computed using tools provided with the REVERB challenge [57]. For automatic speech recognition experiments, we provide word error rates (WER).

A. Methods for Informing the Network

We compared different methods for informing the network about the target speaker and for extracting speaker information from the adaptation utterance as described in Sections III-B, III-C. These experiments were performed on the WSJ0-2mix dataset and used the smaller architecture of the network, as some methods do not scale well to a larger architecture. Table I shows the results of the experiments.

We can observe that the input bias adaptation (*input-bias*) method performs rather poorly. In this case, the neural network does not learn to make proper use of the additional input features and keeps extracting all speakers present in the mixture.

TABLE I
Comparison of Different Methods for Informing the Network About the Speaker and Extracting the Speaker Information. Results Show SDR and FW-SNR Improvements for the 2mix Dataset. IBM Stands for Ideal Binary (Oracle) Mask

method	speaker representation	2mix ΔSDR [dB]	2mix Δfw-SNR [dB]
input-bias	i-vec	-3.8	-1.4
	seqsum	-2.2	-0.8
	seqsum+att	-2.2	-0.8
fact-layer	i-vec	5.7	3.5
	seqsum	6.1	3.7
	seqsum+att	6.2	3.7
scaled-act	i-vec	5.2	2.8
	seqsum	5.6	3.5
	seqsum+att	5.7	3.5
IBM	-	12.8	7.3

Although adapting the bias is a very successful approach to ASR acoustic models adaptation, for our task, it is arguably insufficiently powerful. We confirmed that the poor results are not a consequence of the smaller architecture of the network by repeating the *input-bias + seqsum + att* experiment with the larger architecture. This lead to -1.7 dB SDR degradation and -0.9 dB fw-SNR degradation.

The factorized layer (*fact-layer*) and scaled activations (*scaled-act*) both yield notably better extraction. The factorized layer approach tends to be slightly better, however, this is at the cost of increased computation and memory demands due to the many sub-layers. Therefore, for experiments described in the following sections, we used the scaled activations method, which constitutes a compromise between performance and computational cost.

Comparing the different methods of extracting the speaker information, we find that all three methods (ivec, seqsum, seqsum-att) lead to similar results. Training the speaker representation jointly with the network performs slightly better. Although the attention does not significantly improve the performance, we observed that the learned attention weights properly detect the non-silent parts of the adaptation utterance, which could be helpful when the adaptation utterances contain larger amounts of silence or noise. We therefore retained the attention mechanism for the following experiments.

B. Comparison With DC and PIT

To better evaluate the ability of SpeakerBeam to extract a target speaker, we compare its performance with Deep clustering and Permutation invariant training. For these experiments, we use the larger architecture, which is similar to settings used in previously published work on DC [10], [25] and PIT [27]. Note that with PIT and DC, the outputs are assigned to individual speakers in an oracle way, i.e. we choose an assignment that minimizes the error. With SpeakerBeam, we extract each of the speakers by providing the network with the speaker information, and the assignment is thus decided within the method. For a fairer comparison, we could consider coupling DC and PIT with

TABLE II
COMPARISON OF THE SDR IMPROVEMENTS [DB] WITH WSJ0-2MIX AND WSJ0-2MIX-LONG DATASETS FOR SPEAKERBEAM, DC AND PIT. FOR DC AND PIT, WE USE ORACLE PERMUTATIONS OF THE SEPARATED SOURCES FOR EVALUATION

	2mix ΔSDR[dB]	2mix-long ΔSDR[dB]
SpeakerBeam	9.7	14.1
PIT	9.2	11.0
DC	8.7	9.8
PIT + DC	9.9	11.8
SpeakerBeam + DC	10.9	15.8
IBM	12.8	17.1

a speaker identification module, possibly introducing additional errors. However, we adhere to using the oracle assignment to inspect the upper bound of such an extraction.

The first set of experiments compares performance for the 2mix dataset, which is commonly used to evaluate speech separation methods. Table II shows that the results for this dataset are comparable, with Deep clustering performing slightly worse than the other two methods. Previously published work on DC [58] achieved an SDR improvement of 9.4 dB with a similar network architecture. The main differences between [58] and our setup are the optimization schedule and dropout regularization. Tuning these training settings could thus lead to improved accuracy.

The last experiment in the first part of Table II shows that we can combine Deep clustering and SpeakerBeam. For this experiment, the SpeakerBeam architecture was extended by an additional output layer with a Deep clustering objective. This additional loss can serve to better train the network, while during evaluation, this output is discarded. The results show that the combination indeed helps with training, and the accuracy surpasses both SpeakerBeam and DC when used individually.

The second part of Table II shows the performance with the WSJ0-2mix-long dataset with longer, less overlapped mixtures. For these mixtures, we used networks trained for the WSJ0-2mix and refined them using random 10-second excerpts from the WSJ0-2mix-long training data. The network could thus learn to process segments with no or a partial overlap. The results show that for these data, SpeakerBeam performs better. The degradation of DC and PIT compared with SpeakerBeam originates from the errors in tracing the speaker correctly over time; in some mixtures, the speakers on the output are switched in the middle of the utterance as shown in the example in Figure 5. The outputs of DC and PIT would require further processing for tracing the speakers over the utterance, whereas SpeakerBeam does this jointly with the extraction. We can speculate that such behavior would appear more frequently with even longer mixtures or more speakers. Combining the DC and SpeakerBeam objectives during training again leads to a performance gain.

C. Three Speaker Experiments

Table III shows the results of the extraction when applied to mixtures with 3 speakers. Since the neural network in

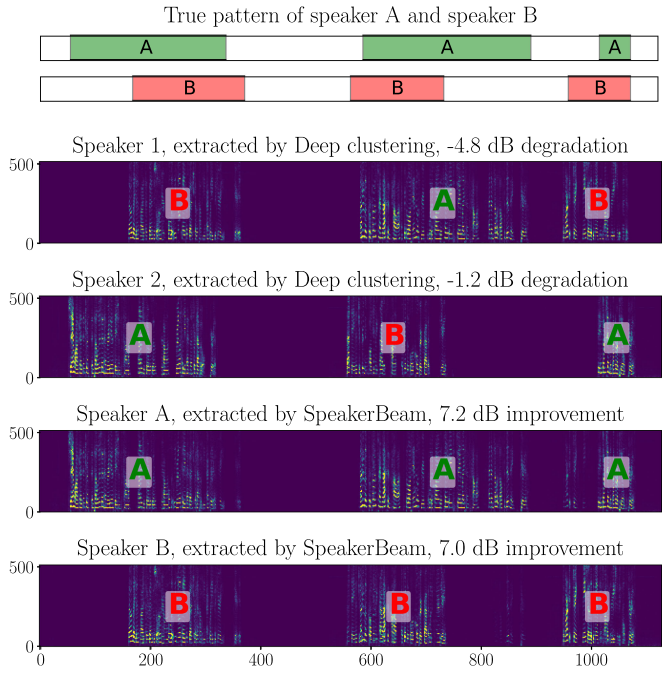


Fig. 5. Example of a mixture from the WSJ0-2mix-long dataset as processed by SpeakerBeam and Deep Clustering methods.

TABLE III
RESULTS OF PERFORMING EXTRACTION ON MIXTURES OF TWO AND THREE SPEAKERS, USING WSJ0-2MIX AND WSJ0-3MIX DATASETS. THE RESULTS ARE IN TERMS OF SDR IMPROVEMENTS [DB]

	Training data	2mix ΔSDR[dB]	3mix ΔSDR[dB]
SpeakerBeam	2mix	9.7	3.9
	3mix	7.9	7.4
	2mix + 3mix	9.7	7.7
DC	2mix	8.7	1.9
	3mix	7.2	6.2
	2mix + 3mix	8.9	6.3
PIT	2mix	9.0	0.1
	3mix	6.5	6.9
	2mix + 3mix	9.1	7.0

SpeakerBeam is independent of the number of speakers in the mixture, we can train the same network for both 2-speaker and 3-speaker data. Table III compares the performance for both 2-speaker and 3-speaker mixtures with different training sets. The results show that a network trained only on 2-speaker mixtures does not generalize very well to 3-speaker mixtures. If we train only on 3-speaker mixtures, the network can extract speakers from both 2- and 3-speaker mixtures with a reasonable level of performance. For the 2-speaker mixtures, there is still a gap in accuracy compared with matched training. The use of all the data for training leads to good performance with both 2 and 3 speaker mixtures. We performed the same set of experiments with DC and PIT. For DC, we used the oracle number of speakers during the clustering step. For PIT, we used a network with 3 outputs. For 2-speaker mixtures, during the training, we considered one

TABLE IV
RESULTS OF AUTOMATIC SPEECH RECOGNITION WITH WSJ0-2MIX IN TERMS OF WORD ERROR RATE USING SINGLE-CHANNEL RECORDINGS

	2mix WER [%]
single speaker	12.2
mixtures	73.4
SpeakerBeam	30.6
PIT	31.2
DC	31.5

TABLE V
RESULTS OF AUTOMATIC SPEECH RECOGNITION ON WSJ0-2MIX IN TERMS OF WORD ERROR RATE USING MULTI-CHANNEL RECORDINGS AND BEAMFORMING

	2mix WER [%]
single speaker	16.2
mixtures	85.2
SpeakerBeam + GEV	22.5
SpeakerBeam + GEV (joint)	20.7

of the outputs to be silent channel, and during testing, we kept only two outputs with the most energy. This follows the procedure described in [27]. The results of both PIT and DC show similar trend as with SpeakerBeam, with even slightly worse generalization from network trained on 2 speakers to 3-speaker mixtures, especially with PIT.

D. Automatic Speech Recognition Experiments

Table IV shows the results we obtained for the automatic speech recognition of the extracted speech in the WSJ0-2mix dataset. Note that in these experiments, to allow for ASR evaluation, we used the *max* version of the dataset, where the length of the mixture corresponds to the length of the longer of the two utterances. By contrast, in the previous experiments the mixtures were cut to the length of the shorter utterance. The *single speaker* and *mixtures* results show the lower and upper bounds of the error. In all the experiments, the speech recognition system is trained on matched training data (single-speaker, mixture or processed with SpeakerBeam, PIT or DC). We can see that SpeakerBeam significantly reduces the error compared with the original mixtures and can thus work as a front-end for an ASR system. Additionally, we also processed single speaker data with SpeakerBeam, to see how much the processing degrades the speech when there is no overlap. The result shows degradation from 12.2% to 15.3% WER. In this case, SpeakerBeam was not trained on single-speaker data, such training could possibly reduce the performance gap.

E. Multi-Channel Experiments

The experimental results in Table V show the ASR performance with the multi-channel dataset WSJ0-2mix-MC. Note that these results cannot be directly compared with Table IV as WSJ0-2mix-MC contains much more reverberation. SpeakerBeam is used here in combination with a GEV beamformer as described in Section IV-A. The use of the beamformer, which employs the SpeakerBeam output, improves the accuracy of the ASR system to 22.5% WER.

In addition, training the front-end jointly with the ASR using the cross-entropy objective further improves the results. In this case, the SpeakerBeam network is initialized with the network trained with the mask objective (Eq. (14)) and the acoustic model with the network trained on the data enhanced by SpeakerBeam. Both networks are then jointly fine-tuned with the final ASR objective. The masks extracted with the front-end tend to be sparser when trained for the ASR objective which may be more convenient for further processing with beamforming.

TABLE VI
RESULTS OF EXTRACTION FOR DATASET WITH ADDITIONAL NOISE IN TERMS OF SDR IMPROVEMENTS

noise level	mixture SDR [dB]	PIT ΔSDR [dB]	DC ΔSDR [dB]	SpkBeam ΔSDR [dB]
∞	0.1	9.6	7.9	10.0
20 dB	0.0	9.4	7.7	9.8
15 dB	-0.2	9.3	6.9	9.7
10 dB	-0.8	9.0	5.8	9.2
5 dB	-2.3	8.6	4.5	8.9
0 dB	-5.0	8.6	3.8	8.9

TABLE VII
RESULTS OF EXTRACTION FOR DATASET WITH ADDITIONAL NOISE IN TERMS OF SPEECH RECOGNITION ERRORS

noise level	mixture WER [%]	SpkBeam WER [%]
∞	72.5	30.5
20 dB	69.9	28.8
15 dB	73.4	30.5
10 dB	80.7	34.4
5 dB	91.4	42.6
0 dB	115.2	55.0

F. Noisy Data

Tables VI,VII show the results of experiments on WSJ0-2mix-noisy. For all the experiments (PIT, DC, SpkBeam), the networks are trained on a training set, where each mixture contains additional noise with a randomly selected SNR as described in Section V-D. For testing, we created several copies of the test-set with various levels of noise ranging from 20 to 0 dB. We can see that even with quite high levels of unstationary noises, SpeakerBeam still succeeds in extracting the target speaker and improves both the signal-level measure and the ASR performance. For more results on noisy and reverberant mixtures, reader can also refer to our study in [59] or our demo video [60]. Note that although the presented experiments use noises recorded in real environments, the mixtures consist of fully overlapped speech, thus may not well reflect the nature of real conversations. The application of speech extraction methods in real conditions is an important issue which we plan to investigate in future work.

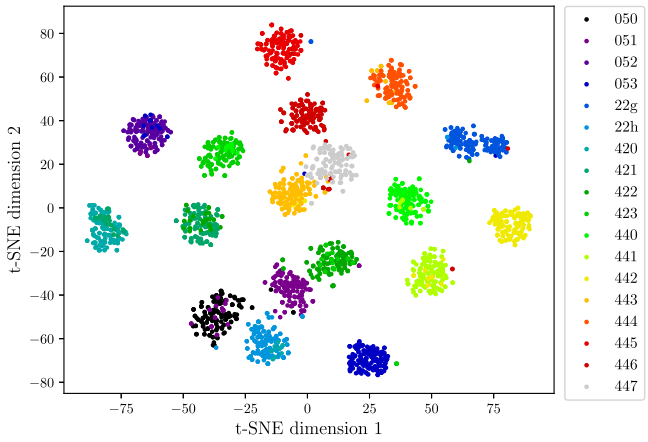


Fig. 6. t-SNE of derived speaker representations. The clusters correspond to speakers in the test data.

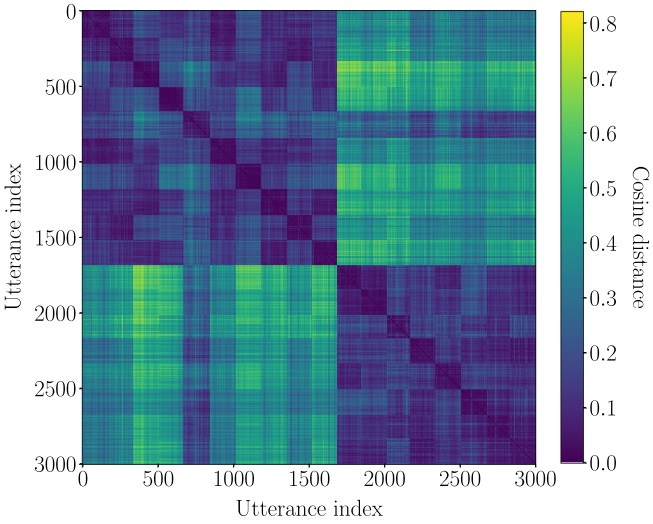


Fig. 7. Matrix of pairwise distances of derived speaker representations.

VIII. Analysis of Learned Behavior

A. Learned Speaker Embeddings

The auxiliary network in the SpeakerBeam architecture should convey information about the speaker from the adaptation utterance to the main network performing the speaker extraction. However, the auxiliary network is never trained with a direct speaker-related objective, only with the final objective of the speaker extraction. In this section, we explore how the learned vectors for the output of the auxiliary network capture the speaker information. Figure 6 shows the embeddings obtained from the adaptation utterances in test data, projected into two dimensions by means of t-SNE [61]. We can see that the vectors form 18 clusters corresponding to the 18 speakers in the test data. Note that there is no overlap between the speakers in the training and evaluation sets. The auxiliary network thus seems to generalize well to unseen speakers. The same conclusions can also be drawn from Figure 7, which shows the pair-wise Euclidean distances of the embeddings. Apart from the distinct speaker clusters, we can also see two main categories of speakers corresponding to males and females; the gender represents important variability in the embedding space.

B. Analysis of Performance Per Speaker

The speaker characteristics are arguably a big factor in the performance of speaker extraction. In this Section, we inspect more closely the accuracy of the method for different speakers. First, we examine whether the accuracy varies greatly for different target speakers in the dataset. We used the WSJ0-2mix dataset and the larger neural network architecture for this analysis (corresponds to the 9.7 dB improvement in Table II). Figure 8 shows that the mean SDR improvement does not vary very significantly for different target speakers, with a minimum of 8.0 dB mean SDR improvement for speaker '423' and a maximum of 11.2 dB mean SDR improvement for speaker '442'. A greater variation can be observed in the results, if we consider the impact of the combination of two speakers in the mixture. In Figure 9, we show the mean SDR improvements for different combinations of target and interfering speakers for SpeakerBeam, PIT and DC. Again, we can see two main groups of speakers corresponding to gender. Mixtures of same-gender speakers tend to be much more difficult to separate. Overall, the mean SDR improvement on same-gender mixtures is 7.2 dB, while for different-gender mixtures, it is 11.9 dB (For PIT, the SDR improvements are 6.3 dB and 11.8 dB and for DC, 5.9 dB and 10.9 dB for same-gender and different-gender mixtures respectively). We can see a few speaker pairs where the method is unable to differentiate between the speakers sufficiently well and the improvements are close to zero. By comparison with Figure 7, these correspond to cases where the extracted speaker embeddings are very similar. We believe that the ability to differentiate between these speakers would improve by training SpeakerBeam with a larger speaker variability in the training set (the WSJ0-2mix dataset we used comprises 101 training speakers).

C. Impact the Adaptation Utterance Length

In all of our experiments, the average length of an adaptation utterance was about 6 seconds. However, for some applications, it might be more convenient to use shorter utterances. In Figure 10, we thus further analyze the impact the length of the adaptation utterance has on the accuracy of the separation. For this analysis, we used the WSJ0-2mix dataset and assigned each test utterance an adaptation utterance of longer than 8.5 seconds. All the adaptation utterances were then cut to different lengths of 0.5 to 8 seconds and used as an input to the auxiliary network. During the cutting, we also removed the initial 0.5 seconds of the utterances to avoid an initial silence. The plot shows the average SDR improvements achieved using these shortened adaptation data. For an adaptation utterance of longer than 2.5 seconds, the performance saturates. Already at 1 second, the accuracy of the extraction is fairly close to that of the longer utterances. With less speech, the performance deteriorates, however even with 0.5 seconds of adaptation data, SpeakerBeam manages to improve the SDR compared with the mixtures. Note that these tendencies may be highly dependent on the training data.

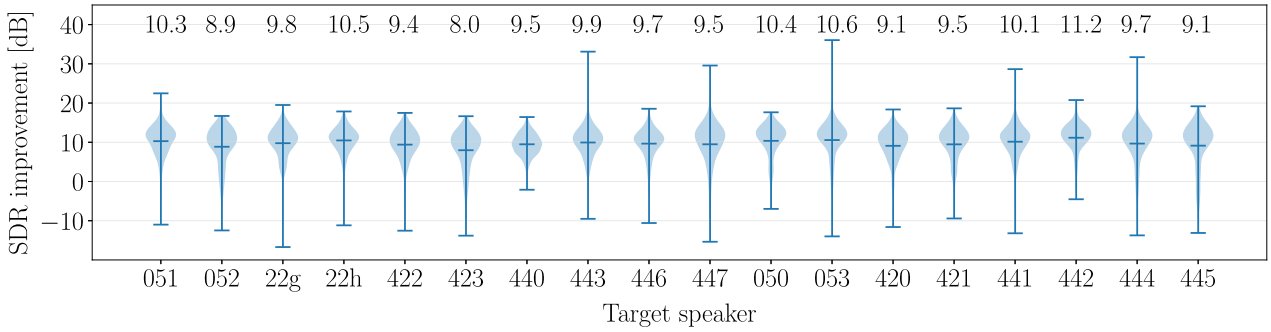
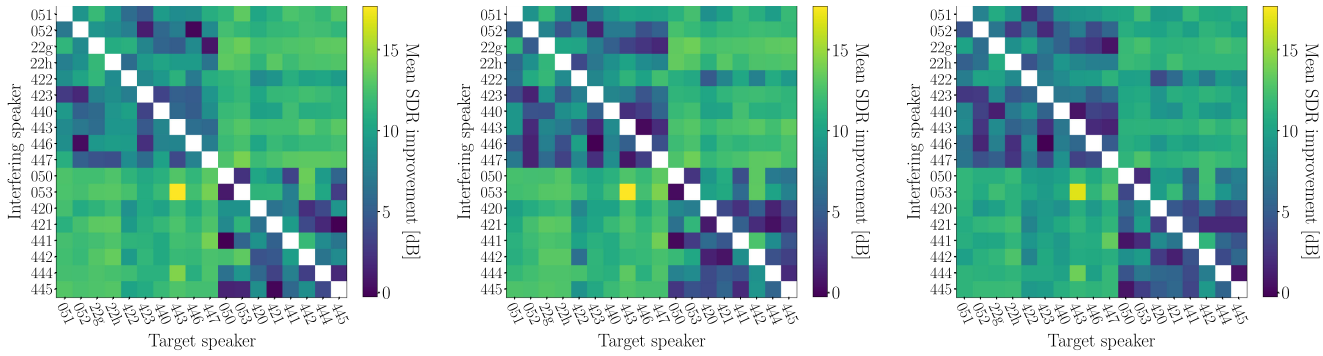


Fig. 8. SDR improvements for different speakers in the WSJ0-2mix testing set. The numbers at the top of the figure are the mean SDR improvements for each speaker. The violin plot shows the distribution shape, maximum, minimum and mean SDR improvement over the utterances from the target speaker.



(a) SpeakerBeam. (b) PIT. (c) DC.

Fig. 9. Mean SDR improvements for different target-interfering speaker combinations in the WSJ0-2mix testing set. Speakers are sorted by gender. Speakers 051 to 447 are male, speakers 050 to 445 are female.

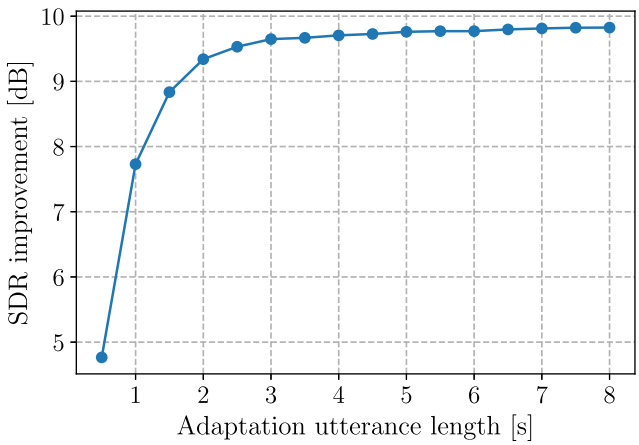


Fig. 10. Impact of the adaptation utterance length on SDR improvement.

IX. Conclusion

In this paper, we introduced the SpeakerBeam method for extracting a target speaker from a mixture of multiple overlapping speakers based on informing the neural network about the target speaker using additional speaker information. We compared different methods for informing the neural network. The results show that the scaled-activations and factorized-layer methods are more suitable than simply appending the speaker information to the input. We compared the method to Deep Clustering and Permutation invariant training, where we observed comparable performance for short, fully overlapped mixtures and the advantage of SpeakerBeam for longer mixtures with more complicated overlapping patterns. This is due to the ability of SpeakerBeam to better track the speaker over time. Furthermore, the method can be also combined with Deep Clustering for further gains. In addition to using our method with single-channel 2-speaker mixtures, we also showed its ability to handle 3-speaker mixtures and the possibility of extending the method to multi-channel processing and joint training with an automatic speech recognition system.

In future work, we plan to explore the effect of using larger datasets, especially with higher numbers of speakers, to further improve learned speaker representations and extraction accuracy. Another possible direction involves combining SpeakerBeam with existing speaker diarization approaches and testing its performance on speaker diarization tasks.

References

[1] S. Makino, T.-W. Lee, and H. Sawada, *Blind Speech Separation*, vol. 615. Berlin, Germany: Springer, 2007.

[2] Y.-M. Qian, C. Weng, X.-K. Chang, S. Wang, and D. Yu, "Past review, current progress, and challenges ahead on the cocktail party problem," *Frontiers Inf. Technol. Electron. Eng.*, vol. 19, no. 1, pp. 40–63, Jan. 2018.

[3] G. J. Brown and M. Cooke, "Computational auditory scene analysis," *Comput. Speech Lang.*, vol. 8, no. 4, pp. 297–336, 1994.

[4] D. P. W. Ellis, "Prediction-driven computational auditory scene analysis," Ph.D. dissertation, Dept. Elect. Eng. Comput., Massachusetts Inst. Technol., Cambridge, MA, USA, 1996.

[5] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Adv. Neural Inf. Process. Syst.*, 2001, pp. 556–562.

[6] T. Virtanen, "Monaural sound source separation by nonnegative matrix factorization with temporal continuity and sparseness criteria," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 15, no. 3, pp. 1066–1074, Mar. 2007.

[7] J. R. Hershey, S. J. Rennie, P. A. Olsen, and T. T. Kristjansson, "Super-human multi-talker speech recognition: A graphical modeling approach," *Comput. Speech Lang.*, vol. 24, no. 1, pp. 45–66, 2010.

[8] S. J. Rennie, J. R. Hershey, and P. A. Olsen, "Single-channel multitalker speech recognition," *IEEE Signal Process. Mag.*, vol. 27, no. 6, pp. 66–80, Nov. 2010.

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

[10] J. R. Hershey, Z. Chen, J. L. Roux, and S. Watanabe, "Deep clustering: Discriminative embeddings for segmentation and separation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2016, pp. 31–35.

[11] D. Yu, M. Kolbæk, Z.-H. Tan, and J. Jensen, "Permutation invariant training of deep models for speaker-independent multi-talker speech separation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2017, pp. 241–245.

[12] R. Maas, S. H. K. Parthasarathi, B. King, R. Huang, and B. Hoffmeister, "Anchored speech detection," in *Proc. Interspeech*, 2016, pp. 2963–2967.

[13] B. King *et al.*, "Robust speech recognition via anchor word representations," in *Proc. Interspeech*, 2017, pp. 2471–2475.

[14] S. H. Mallidi, R. Maas, K. Goehner, A. Rastrow, A. Matsoukas, and B. Hoffmeister, "Device-directed utterance detection," in *Proc. Interspeech*, 2018, pp. 1225–1228.

[15] J. Wang *et al.*, "Deep extractor network for target speaker recovery from single channel speech mixtures," in *Proc. Interspeech*, 2018, pp. 307–311.

[16] J. Du, Y. Tu, Y. Xu, L. Dai, and C.-H. Lee, "Speech separation of a target speaker based on deep neural networks," in *Proc. IEEE 12th Int. Conf. Signal Process.*, 2014, pp. 473–477.

[17] X.-L. Zhang and D. Wang, "A deep ensemble learning method for monaural speech separation," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 24, no. 5, pp. 967–977, May 2016.

[18] J. Du, Y. Tu, L.-R. Dai, and C.-H. Lee, "A regression approach to single-channel speech separation via high-resolution deep neural networks," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 24, no. 8, pp. 1424–1437, Aug. 2016.

[19] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny, "Speaker adaptation of neural network acoustic models using i-vectors," in *Proc. IEEE Workshop Autom. Speech Recognit. Understanding*, 2013, pp. 55–59.

[20] A. Senior and I. Lopez-Moreno, "Improving DNN speaker independence with i-vector inputs," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2014, pp. 225–229.

[21] K. Vesely, S. Watanabe, K. Zmolikova, M. Karafiat, L. Burget, and J. H. Cernocky, "Sequence summarizing neural network for speaker adaptation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2016, pp. 5315–5319.

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

[23] P. Swietojanski and S. Renals, "Learning hidden unit contributions for unsupervised speaker adaptation of neural network acoustic models," in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2014, pp. 171–176.

[24] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 19, no. 4, pp. 788–798, May 2011.

[25] Y. Isik, J. L. Roux, Z. Chen, S. Watanabe, and J. R. Hershey, "Single-channel multi-speaker separation using deep clustering," in *Proc. Interspeech*, 2016, pp. 545–549.

[26] Z. Chen, Y. Luo, and N. Mesgarani, "Deep attractor network for single-microphone speaker separation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process*, 2017, pp. 246–250.

[27] M. Kolbæk *et al.*, "Multitalker speech separation with utterance-level permutation invariant training of deep recurrent neural networks," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 25, no. 10, pp. 1901–1913, Oct. 2017.

[28] 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. Interspeech*, Aug. 2017, pp. 2655–2659.

[29] K. Žmolíková, M. Delcroix, K. Kinoshita, T. Higuchi, A. Ogawa, and T. Nakatani, "Learning speaker representation for neural network based multichannel speaker extraction," in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, 2017, pp. 8–15.

[30] K. Zmolikova, M. Delcroix, K. Kinoshita, T. Higuchi, T. Nakatani, and J. Černocký, "Optimization of speaker-aware multichannel speech extraction with ASR criterion," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2018, pp. 6702–6706.

[31] M. Delcroix, K. Zmolikova, K. Kinoshita, A. Ogawa, and T. Nakatani, "Single channel target speaker extraction and recognition with speaker beam," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2018, pp. 5554–5558.

[32] L. Drude, T. von Neumann, and R. Haeb-Umbach, "Deep attractor networks for speaker re-identification and blind source separation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2018, pp. 11–15.

[33] Q. Wang *et al.*, "Voicefilter: Targeted voice separation by speaker-conditioned spectrogram masking," arXiv preprint arXiv:1810.04826, Oct. 2018.

[34] L. Wan, Q. Wang, A. Papir, and I. L. Moreno, "Generalized end-to-end loss for speaker verification," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2018, pp. 4879–4883.

[35] M. Delcroix, K. Kinoshita, T. Hori, and T. Nakatani, "Context adaptive deep neural networks for fast acoustic model adaptation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2015, pp. 4535–4539.

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

[37] M. Karafiát, L. Burget, P. Matějka, O. Glembek, and J. Černocký, "ivector-based discriminative adaptation for automatic speech recognition," in *Proc. IEEE Workshop Autom. Speech Recognit. Understanding*, 2011, pp. 152–157.

[38] P. Kenny, D. Reynolds, and F. Castaldo, "Diarization of telephone conversations using factor analysis," *IEEE J. Sel. Topics Signal Process.*, vol. 4, no. 6, pp. 1059–1070, Dec. 2010.

[39] O. Glembek, L. Burget, P. Matějka, M. Karafiát, and P. Kenny, "Simplification and optimization of i-vector extraction," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2011, pp. 4516–4519.

[40] Y. Wang, A. Narayanan, and D. Wang, "On training targets for supervised speech separation," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 22, no. 12, pp. 1849–1858, Dec. 2014.

[41] Y. Luo, Z. Chen, J. R. Hershey, J. L. Roux, and N. Mesgarani, "Deep clustering and conventional networks for music separation: Stronger together," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2017, pp. 61–65.

[42] E. Warsitz and R. Haeb-Umbach, "Blind acoustic beamforming based on generalized eigenvalue decomposition," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 15, no. 5, pp. 1529–1539, Jul. 2007.

[43] K. Kumatani, J. McDonough, and B. Raj, "Microphone array processing for distant speech recognition: From close-talking microphones to far-field sensors," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 127–140, Nov. 2012.

[44] J. Heymann, L. Drude, and R. Haeb-Umbach, "Neural network based spectral mask estimation for acoustic beamforming," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2016, pp. 196–200.

[45] H. Erdogan, J. R. Hershey, S. Watanabe, M. Mandel, and J. L. Roux, "Improved MVDR beamforming using single-channel mask prediction networks," in *Proc. Interspeech*, 2016, pp. 1981–1985.

[46] C. Böddeker, P. Hanebrink, L. Drude, J. Heymann, and R. Haeb-Umbach, "Optimizing neural-network supported acoustic beamforming by algorithmic differentiation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Mar. 2017, pp. 171–175.

[47] J. Garofolo, "CSR-I (WSJ0) complete LDC93S6A," 1993. [Online]. Available: https://catalog.ldc.upenn.edu/ldc93s6a

[48] Z.-Q. Wang, J. L. Roux, and J. R. Hershey, "Multi-channel deep clustering: Discriminative spectral and spatial embeddings for speaker-independent speech separation," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2018, pp. 1–5.

[49] J. B. Allen and D. A. Berkley, "Image method for efficiently simulating small-room acoustics," *J. Acoust. Soc. Am.*, vol. 65, no. 4, pp. 943–950, 1979.

[50] E. A. P. Habets, "Room impulse response generator," Technische Universiteit Eindhoven, Eindhoven, The Netherlands, Tech. Rep. 2.2.4, 2010. [Online]. Available: http://home.tiscali.nl/ehabets/rir_generator/rir_generator.pdf
[51] J. Barker, E. Vincent, N. Ma, H. Christensen, and P. Green, "The PASCAL CHiME speech separation and recognition challenge," *Comput. Speech Lang.*, vol. 27, no. 3, pp. 621–633, May 2013.
[52] J. Barker, R. Marxer, E. Vincent, and S. Watanabe, "The third CHiME speech separation and recognition challenge: Dataset, task and baselines," in *Proc. IEEE Workshop Autom. Speech Recognit. Understanding*, 2015, pp. 504–511.
[53] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 249–256.
[54] D. Povey *et al.*, "The Kaldi speech recognition toolkit," in *Proc. IEEE Workshop Autom. Speech Recognit. Understanding*, 2011, Paper EPFL-CONF-192584.
[55] E. Vincent, R. Gribonval, and C. Févotte, "Performance measurement in blind audio source separation," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 14, no. 4, pp. 1462–1469, Jul. 2006.
[56] C. Raffel *et al.*, "mir_eval: A transparent implementation of common MIR metrics," in *Proc. 15th Int. Soc. Music Inf. Retrieval Conf.*, 2014, pp. 367–372.
[57] K. Kinoshita *et al.*, "The REVERB challenge: A common evaluation framework for dereverberation and recognition of reverberant speech," in *Proc. IEEE Workshop Appl. Signal Process. Audio Acoust.*, 2013, pp. 1–4.
[58] Y. Isik, J. L. Roux, Z. Chen, S. Watanabe, and J. R. Hershey, "Single-channel multi-speaker separation using deep clustering," in *Proc. Interspeech*, 2016, pp. 545–549.
[59] M. Delcroix, K. Zmolikova, T. Ochiai, K. Kinoshita, S. Araki, and T. Nakatani, "Compact network for speakerbeam target speaker extraction," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2019, pp. 6965–6969.
[60] "SpeakerBeam." [Online]. Available: https://youtu.be/7FSHgKip6vI (English), https://youtu.be/BM0DXWgGY5A (Japanese).
[61] L. V. D. Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, 2008.

Kateřina Žmolíková received the B.Sc. degree in information technology in 2014 and the Ing. degree in mathematical methods in information technology in 2016 from the Faculty of Information Technology, Brno University of Technology (BUT), Brno, Czech Republic, where she is currently working toward the Ph.D. degree. Since 2013, she has been part of the Speech@FIT research group at BUT. She is the Ph.D. Talent scholarship holder. Her research interests include robust speech recognition, speech separation, and deep learning.

Marc Delcroix (M'05–SM'16) received the M.Eng. degree from the Free University of Brussels, Brussels, Belgium, and Ecole Centrale Paris, Paris, France, in 2003, and the Ph.D. degree from the Graduate School of Information Science and Technology, Hokkaido University, Sapporo, Japan, in 2007. He is currently a Distinguished Researcher with the Media Information Laboratory, Signal Processing Group, NTT Communication Science Laboratories, NTT Corporation, Kyoto, Japan. From 2007 to 2008 and 2010 to 2012, he was a Research Associate with the NTT Communication Science Laboratories, where he became a permanent Research Scientist in 2012. He was also a Visiting Lecturer with the Faculty of Science and Engineering, Waseda University, Tokyo, Japan, from 2015 to 2018. His research interests include robust speech recognition, acoustic model adaptation, and speech enhancement. He took an active part in the development of NTT robust speech recognition systems for the REVERB and the CHiME 1 and 3 challenges. He was the recipient of several research awards including the 2006 Sato Paper Award from ASJ, the 2015 IEEE-ASRU Best Paper Award Honorable Mention, and the 2016 ASJ Awaya Young Researcher Award. He was part of the organizing committee of the REVERB Challenge/Workshop 2014 and ASRU 2017 and is a Member of the IEEE SPS Speech and Language Processing Technical Committee. He is a member of ASJ.

Keisuke Kinoshita (M'05) received the M.Eng. and Ph.D. degrees from Sophia University, Tokyo, Japan, in 2003 and 2010, respectively. He is currently a Distinguished Researcher with the NTT Communication Science Laboratories, NTT Corporation, Kyoto, Japan. After joining the NTT Communication Science Laboratories in 2003, he has been engaged in the research on various types of speech, audio, and music signal processing, including speech enhancement such as 1ch/multichannel blind dereverberation, noise reduction, source separation, distributed microphone array processing, and robust speech recognition. He has authored or coauthored more than 100 technical papers in refereed journals and conference proceedings. He also contributed to five book chapters. He was a Chief Coordinator of the REVERB Challenge 2014, an Associate Editor of IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences from 2013 to 2015, and has been a Member of IEEE AASP TC since 2018. He was the recipient of the 2006 IEICE Paper Award, the 2009 ASJ Outstanding Technical Development Prize, the 2011 ASJ Awaya Prize, the 2012 Japan Audio Society Award, the 2015 IEEE-ASRU Best Paper Award Honorable Mention, and 2017 Maeshima Hisoka Award. He is a member of ASJ and IEICE.

Tsubasa Ochiai received the B.E., M.E., and Ph.D. degrees in information engineering from Doshisha University, Kyoto, Japan, in 2013, 2015, and 2018, respectively. Since 2018, he has been a Researcher with the Communication Science Laboratories, NTT Corporation, Kyoto, Japan. His research interests include speech recognition, speech enhancement, and deep learning. He is a member of the IEICE and ASJ.

Tomohiro Nakatani (M'03–SM'06) received B.E., M.E., and Ph.D. degrees from Kyoto University, Kyoto, Japan, in 1989, 1991, and 2002, respectively. He is currently a Senior Distinguished Researcher with NTT Communication Science Laboratories, NTT Corporation, Kyoto, Japan. Since joining NTT Corporation as a Researcher in 1991, he has been investigating audio signal processing technologies for intelligent human-machine interfaces, including dereverberation, denoising, source separation, and robust ASR. He was the recipient of the 2005 IEICE Best Paper Award, the 2009 ASJ Technical Development Award, the 2012 Japan Audio Society Award, the 2015 IEEE ASRU Best Paper Award Honorable Mention, the 2017 Maejima Hisoka Award, and the 2018 IWAENC Best Paper Award. He was a Visiting Scholar with the Georgia Institute of Technology for a year from 2005 and he was a Visiting Assistant Professor with the Department of Media Science, Nagoya University, from 2008 to 2017. He served as an Associate Editor for the IEEE TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING from 2008 to 2010. He was a member of the IEEE Signal Processing Society (SPS) Audio and Acoustics Technical Committee (AASP-TC) from 2009 to 2014 and served as the Review Subcommittee Chair for the TC from 2013 to 2014. He has been an associate member of the AASP-TC since 2015 and a member of the IEEE SPS Speech and Language Processing Technical Committee since 2016. He was also a member of IEEE CAS Society Blind Signal Processing Technical Committee from 2007 to 2009. He served as the Chair of the IEEE Kansai Section Technical Program Committee from 2011 to 2012, and he has been serving as the Chair of the IEEE SPS Kansai Chapter since 2019. He was a Technical Program Co-Chair of the IEEE WASPAA-2007, a Co-Chair of the 2014 REVERB Challenge Workshop, and a General Co-Chair of the IEEE ASRU-2017. He is a member of IEICE and ASJ.

Lukáš Burget is currently an Assistant Professor with the Faculty of Information Technology, Brno University of Technology, Brno, Czech Republic, and the Research Director of the BUT Speech@FIT group. He was a Visiting Researcher with OGI Portland, USA, and with SRI International, Menlo Park, CA, USA. His scientific interests are in the field of speech data mining, concentrating on acoustic modeling for speech, speaker, and language recognition, including their software implementations. Dr. Burget was on numerous EU- and US-funded projects, was the PI of US-Air Force EOARD project and BUTs PI in IARPA BEST.

Jan (Honza) Černocký (M'2001–SM'2008) is currently an Associate Professor and the Head of the Department of Computer Graphics and Multimedia, BUT Faculty of Information Technology (FIT). He also serves as a Managing Director of BUT Speech@FIT research group. His research interests include artificial intelligence, signal processing, and speech data mining (speech, speaker, and language recognition). He is responsible for signal and speech processing courses at FIT BUT. In 2006, he cofounded Phonexia. He is the General Chair of Interspeech 2021 in Brno.