CONTINUOUS SPEECH SEPARATION WITH RECURRENT SELECTIVE ATTENTION NETWORK

Yixuan Zhang[1,2,*], *Zhuo Chen*[1], *Jian Wu*[1], *Takuya Yoshioka*[1], *Peidong Wang*[1], *Zhong Meng*[1], *Jinyu Li*[1]

[1]Microsoft, Redmond, WA, USA
[2]The Ohio State University, Columbus, OH, USA
zhang.7388@osu.edu, {zhuc, wujian, tayoshio, peidongwang, zhong.meng, jinyli}@microsoft.com

ABSTRACT

While permutation invariant training (PIT) based continuous speech separation (CSS) significantly improves the conversation transcription accuracy, it often suffers from speech leakages and failures in separation at "hot spot" regions because it has a fixed number of output channels. In this paper, we propose to apply recurrent selective attention network (RSAN) to CSS, which generates a variable number of output channels based on active speaker counting. In addition, we propose a novel block-wise dependency extension of RSAN by introducing dependencies between adjacent processing blocks in the CSS framework. It enables the network to utilize the separation results from the previous blocks to facilitate the current block processing. Experimental results on the LibriCSS dataset show that the RSAN-based CSS (RSAN-CSS) network consistently improves the speech recognition accuracy over PIT-based models. The proposed block-wise dependency modeling further boosts the performance of RSAN-CSS.

Index Terms— Continuous speech separation, recurrent selective attention network, meeting transcription

1. INTRODUCTION

In recent years, considerable efforts [1, 2] have been made to improve the automatic transcription of real conversations. Two key challenges in this task are the overlapped speech and quick speaker turns, which often occur during the interactions between multiple speakers. Speech separation, as a front-end solution for these problems, has been explored in different ways. Methods such as deep clustering (DC) [3] and permutation invariant training (PIT) [4, 5], were proposed to address the permutation issue, and they have been extended by various methods to further improve the separation quality [6, 7, 8, 9, 10, 11]. To process realistic long conversation recordings, which are continuous and usually partially overlapped, Yoshioka et al. [12, 2] proposed a continuous speech separation (CSS) framework. With CSS, the long recording is processed in a segment-wise manner, with a simple stitching mechanism to make the output signals consistent across the segments. Thus, the long-form input audio stream can be separated into several output channels with the same length, each containing continuous overlap-free utterances.

Utterance-level PIT (uPIT) [4], which addresses the permutation issue by searching for the best utterance-level permutation during training, is usually used for CSS [12]. However, since uPIT requires a fixed number of output channels, the maximum number of separable sources is limited. This would be a problem for the "hot spot" region during a conversation, i.e., the region where the simultaneously

talking speakers exceeds the pre-assumed channel number. Also, with the fixed output channel design, in the single speaker region, the speech energy sometimes leaks from the active output channel to other output channels that should have zero energies. This so-called "leakage issue" could lead to insertion errors during speech recognition.

Recently, a recursive selective attention network (RSAN) [13] was proposed and showed a comparable performance to uPIT for fully overlapped speech. Unlike conventional uPIT that separates each mixing source simultaneously, the recursive approach separates one source at a time and decides when to stop the recursion based on the detection of the remaining speakers in the input. Compared with uPIT, in theory, this method generates a variable number of sources, and thus it can handle an arbitrary number of overlapped speakers. In addition, this approach generates a single output channel for one active speaker region, which could help with the leakage issue.

Inspired by the promising properties of the recursive speech separation [13, 14, 15, 16, 17] and iterative speaker adaptation for speech recognition [18, 19, 20, 21, 22], in our work, we examine the application of RSAN to the CSS task. In addition, we propose a novel block-wise dependency modeling in the RSAN framework for tighter integration with the CSS pipeline. In the original setup, the audio stream is separated into overlapped blocks, and each block is processed independently. Instead, we consider dependencies between the calculation in the neighboring blocks, allowing each block to reuse the previous separation results. The experiments on LibriCSS data set show that the RSAN-based CSS outperforms the uPIT baseline, and better performance can be achieved with the proposed block-wise dependency method. To further explore the "hot spot" issue, we retrain RSAN with the three-speaker overlap dataset. We show that with the speaker-counting ability, the RSAN-CSS pipeline can take advantage of those more challenging overlapped training data while maintaining the performance for 1 and 2 speaker regions. In contrast, a similarly trained uPIT model suffers from significant performance degradation on the test set with fewer speech overlaps.

The remainder of the paper is organized as follows. Sections 2 and 3 describe the the architecture of recurrent selective attention network and its block-wise dependency extension, respectively. In Section 4, our experimental setup and evaluation results are described. Finally, the concluding remarks are given in Section 5.

2. RECURSIVE SPEECH SEPARATION

2.1. Recurrent Selective Attention Network

Our RSAN model is modified based on the original recursive selective hearing network [13]. We follow the idea of performing speech

---

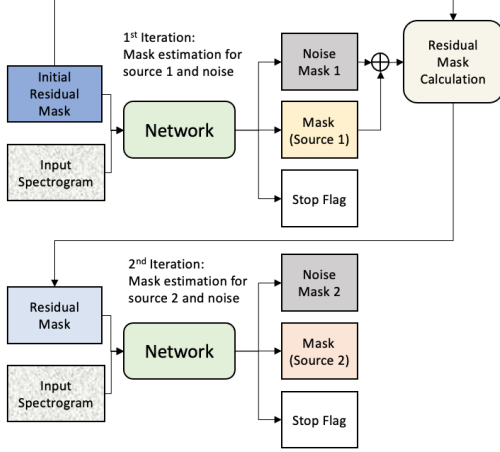*This work was done during an internship at Microsoft.

Fig. 1: Illustration of speech separation with RSAN for two-speaker case.

separation recursively and estimating one speech source at a time. Compared with [13], instead of estimating the noise at the first iteration and speech at the following ones, we estimate speech and remaining noise at every iteration. This reduces the computation without negatively impacting the performance in our observation.

Figure 1 explains the overall framework. The network takes the magnitude spectrogram of the input signal, and a residual mask as input. The residual mask has the same size of the mixture spectrum, serving as an indicator of the presence of any remaining unseparated speakers in the input signal. It starts with an all-one mask, and gets updated at each iteration by subtracting the estimated masks from the current residual mask values. In each iteration, the network generates three outputs: one mask for the extracted speaker, one mask for the remaining noise, and a stop flag. The stop flag is a scalar within the range of $[0, 1]$. The value of the stop flag indicates the likelihood of all sources being already separated out. In other words, it indicates when the iteration should stop. During inference, a threshold is set for the stop flag. When the value of the stop flag exceeds the threshold, the iteration would stop.

An example of the two-speaker separation procedure is illustrated in Figure 1, which consists of two iterations. At the first iteration, the residual mask is initialized with ones. The network thus pays attention to all regions of the input spectrogram. Before the next iteration, the residual mask is updated by subtracting the initial residual mask with the estimated masks from the first iteration. The resulting negative values in the updated residual mask, if any, are replaced with 0. When the separation is done, the noise mask from each iteration, $\mathbf{N}_i$, will be added together to generate the final noise mask estimate: $\mathbf{N} = \sum_i \mathbf{N}_i$.

The training is similar to the inference procedure. One difference is that the number of iterations for each utterance is predetermined since the total number of speakers is known for each training sample. The cost function for training is defined as,

$$\mathcal{L} = \mathcal{L}^{\text{mse}} + \alpha \cdot \mathcal{L}^{\text{flag}}, \tag{1}$$

where $\mathcal{L}^{\text{mse}}$ is the MSE loss between the estimated masks and the reference masks of the target sources and the noise. $\mathcal{L}^{\text{flag}}$ is the loss regarding the stop flag. α is set to 0.05 for all related experiments
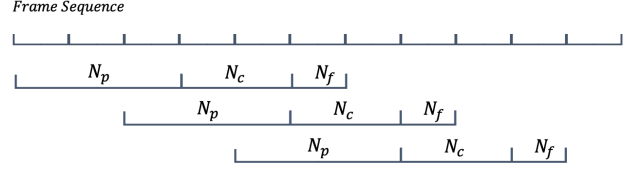

Fig. 2: Streaming processing with continuous speech separation

described in Section 4. $\mathcal{L}^{\text{mse}}$ in the cost function is defined as,

$$\mathcal{L}^{\text{mse}} = \frac{1}{STF} \sum_{i=1}^{S} ||\hat{\mathbf{M}}_i \odot \mathbf{Y} - \mathbf{A}_{\phi^*}||^2$$
$$+ \frac{1}{TF} ||\hat{\mathbf{M}}_N \odot \mathbf{Y} - \mathbf{N}||^2, \tag{2}$$

where $\mathbf{Y}, \mathbf{A}, \mathbf{N}$ are the magnitude spectrograms of the input, target sources and background noise, respectively. ϕ^* indicates the preferred permutation, which is determined by exhaustive search of the best permutation. $\hat{\mathbf{M}}_i$ is a mask for the speaker estimated at the i-th iteration. $\hat{\mathbf{M}}_N$ is the estimated mask of the noise. S, T, F correspond to the number of target sources, time frames and frequency bins. $\mathcal{L}^{\text{flag}}$ in the cost function is defined as

$$\mathcal{L}^{\text{flag}} = -\sum_{i=1}^{S} [\mathbf{z}_i \ln \hat{\mathbf{z}}_i + (1 - \mathbf{z}_i) \ln (1 - \hat{\mathbf{z}}_i)], \tag{3}$$

where $\mathbf{z}$ is a vector that contains S elements, in which the first $S-1$ elements are 0, and the last element is 1. $\hat{\mathbf{z}}$ is the estimated vector of the stop flags.

2.2. Continuous Speech Separation with RSAN

In CSS, we first segment the long input with a sliding window and perform separation within segmented blocks. As depicted in Figure 2, the input is divided into several blocks by a sliding window which consists of three parts, including N_p, N_c, N_f, each representing the past, current and future frames. Each time, the sliding window shifts by N_c frames. During separation, the whole window is fed into the separation network to generate masks, and only the mask within N_c's region is used. A stitching processing is then applied to align the permutation between adjacent windows, by comparing the mask similarity between their shared region, i.e. $N_p + N_f$. The final output is obtained by combing each aligned window.

When integrating RSAN into CSS framework, RSAN serves as a local separator for each window. As RSAN generates variable number of outputs for each block, we zero-pad the local separation result to align them with the same number of output channel.

3. BLOCK-WISE DEPENDENCY MODELING

Despite the shared content among the adjacent blocks, each block in the original CSS framework is processed independently, resulting in potentially redundant computation and sub-optimal separation due to the limited observation for each block, e.g. 1.6s in [23]. In the recursive separation framework, the residual mask conveys the separation information from earlier iterations. Such message passing property can be naturally extended to sequential modeling. In this section, we propose a block-wise dependency mechanism to take advantage of the overlapping information between neighboring blocks by passing
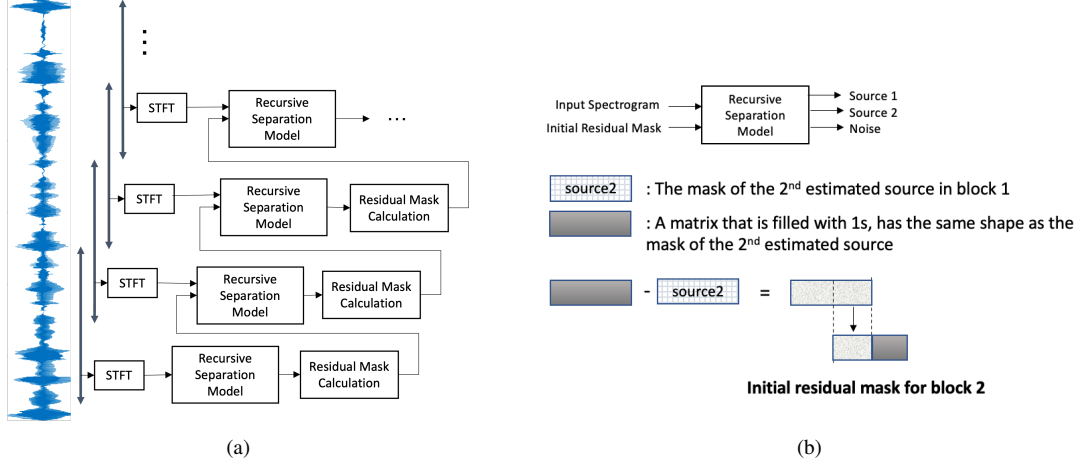
Fig. 3: Illustration of the proposed block-wise dependency modeling mechanism. (a) Overview of the dependencies between blocks. (b) An example of initial residual mask calculation. For simplicity, we plot a scenario where the previous block generates two source outputs.

the separation information from the previous block to the next. This will facilitate the separation in the subsequent block.

Figure 3 shows the proposed block-wise dependency modeling mechanism. Figure 3 (a) gives an overview of how the inter-block dependencies are added to the overall CSS processing. The separated source masks from each block are used to calculate the initial residual mask for the next block. Regarding the first block of the long input signal, since no earlier information is available, the initial residual mask is filled with 1s.

Figure 3 (b) shows an example of how the initial residual mask is calculated based on the separated speech signals from the previous block. In this example, we assume that the overlap ratio between blocks is 50%. To form the initial residual mask for block 2, we firstly aggregate the estimated masks from block 1. Specifically, we sum the masks from the second recursion to the end, i.e. $M_{B1} = \sum_i^{2:} M_i^{B_1}$, where B_1 and B_2 refer to block 1 and 2, i indexes the recursion. A new mask D is then constructed by subtracting M_{B1} from an all one mask. Finally, the initial residual mask for block 2 is formed by passing the values from D for the overlapped region between blocks 1 and 2, as shown below :

$$\mathbf{R}_1^{B2}[i,j] = \mathbf{D}[\frac{T}{2} + i, j], \quad \forall i = 1, \ldots, \frac{T}{2}, \forall j,$$
$$\mathbf{R}_1^{B2}[i,j] = \mathbf{1}, \qquad \forall i = \frac{T}{2} + 1, \ldots, T, \forall j, \tag{4}$$

where T is the length of the block. The newly formed residual mask R_1^{B2} contains strong bias information for the first extracted speaker in block1, thus providing a warm start for subsequent separation. When only 1 speaker exists in the previous block, the mask of the later estimated sources $\hat{\mathbf{M}}_{2:}^{B1}$ is filled with 0s, making $\mathbf{D}$ an all-one matrix.

4. EXPERIMENTS

4.1. Experimental Setup

4.1.1. Training set

We use the same training dataset as in [24], which contains 219 hours of mixed speech. The source utterances in each mixed sample are randomly picked from WSJ1 [25]. The number of speakers in a mixture can be one or two. (We later show the experimental results with more overlapping speakers in Section 4.3.) The sampled utterances are artificially reverberated by convolving with room impulse responses (RIRs) that are simulated with the image method [26]. The reverberated utterances are then rescaled and mixed to form the overlapped speech, with a random energy ratio between -5 dB to 5 dB. A simulated isotropic noise [27] clip is added to each mixture with a signal to noise ratio between 0 and 10 dB. During training, each utterance is segmented into 4s clips before feeding into the network.

4.1.2. Evaluation details

We evaluate the models on the LibriCSS [23] dataset, which contains 10 hours of multi-speaker recordings. The dataset consists of 60 10-minute sessions, and is obtained by recording the replays of the concatenated and mixed utterances from LibriSpeech [28] with a circular microphone array in the meeting room. We use the the first channel for evaluation. Session 0 from the LibriCSS dataset is applied as the development set for hyper-parameter tuning, e.g., the the stop flag threshold for RSAN.

The trained models are evaluated on the conversation transcription pipeline described in [2]. The asclite speaker agnostic word error rate (SA-WER) is used as the evaluation metric. During separation, the long mixture audio is segmented to 2.4s blocks with 0.8s hop size.

4.1.3. Network training

The RSAN model contains 16 Conformer encoder layers [24], each having 4 attention heads, 256 attention dimensions and 1024 feed forward network dimensions. The output from the last Conformer decoder layer is converted to two masks and a flag value through three corresponding sigmoid projection layers. The block-wise dependency model is initialized with a well trained RSAN network. Note that we use 2.4s chunk size with 0.8s hop size for block-wise dependency model training as it shows the best performance.

For comparison, the uPIT-based model from [24] is employed as the baseline, which has the same amount of parameters. All models are trained using the AdamW [29] optimizer with an initial learning rate of 0.0001.

Table 1: WER (%) comparison of uPIT, RSAN model with or without block-wise dependency mechanism

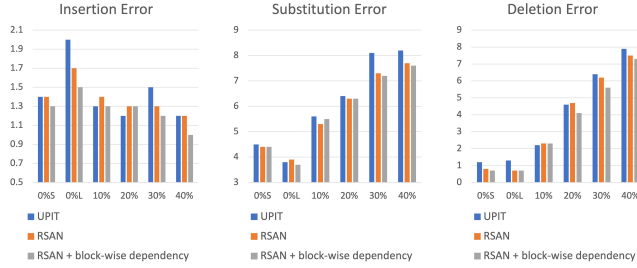| Methods	Overlap Ratio in %					
	0S	0L	10	20	30	40
uPIT	7.0	7.1	9.2	12.2	16.1	17.2
RSAN	6.6	6.3	**9.0**	12.4	14.8	16.4
+ Dependency	**6.3**	**6.0**	9.1	**11.8**	**14.1**	**15.9**

Fig. 4: Analysis on the WER results of three methods: uPIT, RSAN and RSAN with block-wise dependency

4.2. Results

The evaluation results are shown in Table 1. We observe that RSAN consistently outperforms uPIT. On average, in comparison with uPIT, the RSAN model reduces the WER by 5% relatively. The improvements are mainly observed in low/high overlap scenarios such as 0L, 0S, 30% and 40%. Figure 4 shows an analysis of the WERs of different models. Apart from the expected decrease of the insertion error due to the likely improvement on the leakage issue, it is also observed that the substitution error is reduced especially in the scenarios with high overlap ratios. A possible explanation is that the recursive separation brings better separation quality in the overlapped region, which is also observed in [13].

The average WER is further reduced with the dependency mechanism. The main improvements are from sessions that have higher overlap ratios such as 20%, 30% and 40%, which shows the effectiveness of the block-wise dependency mechanism in the overlapped regions.

4.3. CSS with more output channels

In previous experiments, the evaluation is performed with 2.4s processing window, assuming that each block contains at most two active speakers. This assumption is not always true, especially for the "hot spot" regions. In this experiment, we relax the above assumption to having at most three speakers in a block. Accordingly, we retrain the RSAN model on a new training set which is built on the training set described in Section 4.1.1 by adding a third speaker (also picked from WSJ1) to some 2-speaker mixtures. Note that the retrained RSAN model doesn't have block-wise dependency. The resulting training set contains 32% 3-speaker mixtures, 48% 2-speaker mixtures and 20% 1-speaker mixtures. A three speaker uPIT-based model is trained for comparison. The uPIT model has the same network architecture in Section 4.1.3, except that it generates 4 estimated masks, three for speakers and one for noise. As LibriCSS dataset only contains overlap speech from two simultaneously talking speaker, to explore the modeling with more participants in each window, a longer window is used in the experiment. We perform the

Table 2: WER (%) results of uPIT and RSAN models trained on three-speaker mixture dataset. S, B indicate stop flag threshold and block size, respectively.

Methods	S	B	Overlap Ratio in %					
			0S	0L	10	20	30	40
uPIT	×	2.4s	13.1	11.4	14.9	17.1	18.6	20.3
		4.8s	10.8	8.0	12.9	15.4	17.6	18.7
RSAN	0.6	2.4s	6.4	6.1	**8.9**	11.5	14.6	15.9
	[0.6, 0.1]	2.4s	**6.3**	**5.7**	9.0	**11.4**	14.7	15.3
	0.6	4.8s	6.8	5.8	9.2	11.8	**14.5**	**15.2**
	[0.6, 0.1]	4.8s	7.0	5.8	9.1	12.0	14.6	15.9

evaluation on 2.4 and 4.8s (double-sized) audio blocks. CSS with three output channel is used in this experiment, where the empty outputing channels are zero padded for 1 speaker and 2 speaker regions.

Previously, we set one stop flag threshold for all iterations. This makes the model "equally likely" to stop in any iterations. Since the appearance of three or more active speakers is less likely, it is reasonable to set a lower threshold for the later iterations. In this experiment, two sets of the stop flag thresholds are used. One is setting a consistent threshold 0.6 for all iterations. The other is setting the threshold as 0.6 only for the first iteration, and use 0.1 for the following iterations. In this case, the third and later iterations are not encouraged.

Experimental results are shown in Table 2. Compared with the result in Table 1, the three speaker uPIT model suffers from significant performance drop for 2.4s window. Under 4.8s window setting, the uPIT model has improved performance, but is still much worse than two speaker uPIT model. This proves our hypothesis that fixed outputting channel design negatively impacts the model generalization.

In contrast, the three speaker RSAN has the same performance for 2.4s window evaluation as its two speaker version. With longer window, a better performance is observed. A better generalization has been shown for RSAN model thanks to its speaker counting capability.

In the evaluation of RSAN model on 2.4s audio blocks, we observe improved WERs with decreased stop flag thresholds in the later iterations. The reduction of the threshold encourages the network to stop early. In other words, the network will only estimate the third source when it's very confident, which is good since three active speakers rarely exist in small blocks. We then evaluate the model on 4.8s audio blocks, where we find that reducing the threshold for the second and later iterations increases the WERs, especially for the scenario with the highest overlap ratio. A possible explanation is that 3 speakers appear more frequently in 4.8s blocks. Decreasing the stop flag threshold for the later iterations may cause incorrect early stopping, thus results in worse WERs.

5. CONCLUSIONS

In this paper, we explore the use of recurrent selective attention network for continuous speech separation. From the experimental results, it is observed that RSAN consistently outperforms the uPIT-based model. In addition, a novel block-wise dependency mechanism is proposed to introduce tighter connections between consecutive blocks and brings further improvement. Finally, to deal with the "hot spot" regions, we also retrain RSAN and uPIT model on a training set with 3-speaker mixtures.

6. REFERENCES

[1] S. Watanabe, M. Mandel, J. Barker, E. Vincent, A. Arora, X. Chang, S. Khudanpur, V. Manohar, D. Povey, D. Raj *et al.*, "Chime-6 challenge: Tackling multispeaker speech recognition for unsegmented recordings," *arXiv preprint arXiv:2004.09249*, 2020.

[2] T. Yoshioka, I. Abramovski, C. Aksoylar, Z. Chen, M. David, D. Dimitriadis, Y. Gong, I. Gurvich, X. Huang, Y. Huang *et al.*, "Advances in online audio-visual meeting transcription," in *Proc. ASRU*, 2019, pp. 276–283.

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

[4] 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. ICASSP*, 2017, pp. 241–245.

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

[6] Y. Luo and N. Mesgarani, "Conv-TasNet: Surpassing ideal time–frequency magnitude masking for speech separation," *IEEE/ACM TASLP*, vol. 27, no. 8, pp. 1256–1266, 2019.

[7] Y. Liu and D. Wang, "Divide and conquer: A deep CASA approach to talker-independent monaural speaker separation," *IEEE/ACM TASLP*, vol. 27, no. 12, pp. 2092–2102, 2019.

[8] Y. Luo, Z. Chen, and T. Yoshioka, "Dual-path RNN: efficient long sequence modeling for time-domain single-channel speech separation," in *Proc. ICASSP*, 2020, pp. 46–50.

[9] C. Subakan, M. Ravanelli, S. Cornell, M. Bronzi, and J. Zhong, "Attention is all you need in speech separation," in *Proc. ICASSP*, 2021, pp. 21–25.

[10] N. Zeghidour and D. Grangier, "Wavesplit: End-to-end speech separation by speaker clustering," *IEEE/ACM TASLP*, vol. 29, pp. 2840–2849, 2021.

[11] T. Yoshioka, H. Erdogan, Z. Chen, and F. Alleva, "Multi-microphone neural speech separation for far-field multi-talker speech recognition," in *Proc. ICASSP*, 2018, pp. 5739–5743.

[12] T. Yoshioka, H. Erdogan, Z. Chen, X. Xiao, and F. Alleva, "Recognizing overlapped speech in meetings: A multichannel separation approach using neural networks," *arXiv preprint arXiv:1810.03655*, 2018.

[13] K. Kinoshita, L. Drude, M. Delcroix, and T. Nakatani, "Listening to each speaker one by one with recurrent selective hearing networks," in *Proc. ICASSP*, 2018, pp. 5064–5068.

[14] P. Wang, Z. Chen, X. Xiao, Z. Meng, T. Yoshioka, T. Zhou, L. Lu, and J. Li, "Speech separation using speaker inventory," in *Proc. ASRU*, 2019, pp. 230–236.

[15] P. Wang, Z. Chen, D. L. Wang, J. Li, and Y. Gong, "Speaker separation using speaker inventories and estimated speech," *IEEE/ACM TASLP*, vol. 29, pp. 537–546, 2021.

[16] Z. Q. Wang, P. Wang, and D. L. Wang, "Complex spectral mapping for single- and multi-channel speech enhancement and robust asr," *IEEE/ACM TASLP*, vol. 28, pp. 1778–1787, 2020.

[17] ——, "Multi-microphone complex spectral mapping for utterance-wise and continuous speech separation," *IEEE/ACM TASLP*, vol. 29, pp. 2001–2014, 2021.

[18] P. Wang and D. L. Wang, "Utterance-wise recurrent dropout and iterative speaker adaptation for robust monaural speech recognition," in *Proc. ICASSP*, 2018, pp. 4814–4818.

[19] ——, "Filter-and-convolve: A CNN based multichannel complex concatenation acoustic model," in *Proc. ICASSP*, 2018, pp. 5564–5568.

[20] P. Wang, K. Tan, and D. L. Wang, "Bridging the gap between monaural speech enhancement and recognition with distortion-independent acoustic modeling," in *Proc. Interspeech*, 2019, pp. 471–475.

[21] P. Wang and D. L. Wang, "Enhanced spectral features for distortion-independent acoustic modeling." in *Proc. Interspeech*, 2019, pp. 476–480.

[22] P. Wang, K. Tan, and D. L. Wang, "Bridging the gap between monaural speech enhancement and recognition with distortion-independent acoustic modeling," *IEEE/ACM TASLP*, vol. 28, pp. 39–48, 2019.

[23] Z. Chen, T. Yoshioka, L. Lu, T. Zhou, Z. Meng, Y. Luo, J. Wu, X. Xiao, and J. Li, "Continuous speech separation: Dataset and analysis," in *Proc. ICASSP*, 2020, pp. 7284–7288.

[24] S. Chen, Y. Wu, Z. Chen, J. Wu, J. Li, T. Yoshioka, C. Wang, S. Liu, and M. Zhou, "Continuous speech separation with conformer," in *Proc. ICASSP*, 2021, pp. 5749–5753.

[25] Linguistic Data Consortium Philadelphia, "CSR-II (WSJ1) Complete," 1994, https://catalog.ldc.upenn.edu/LDC94S13A.

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

[27] E. A. Habets and S. Gannot, "Generating sensor signals in isotropic noise fields," *JASA*, vol. 122, no. 6, pp. 3464–3470, 2007.

[28] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an ASR corpus based on public domain audio books," in *Proc. ICASSP*, 2015, pp. 5206–5210.

[29] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.