Interspeech 2024
1-5 September 2024, Kos, Greece

SpeakerBeam-SS: Real-time Target Speaker Extraction with Lightweight Conv-TasNet and State Space Modeling

Hiroshi Sato[1], *Takafumi Moriya*[1], *Masato Mimura*[1], *Shota Horiguchi*[1], *Tsubasa Ochiai*[1], *Takanori Ashihara*[1], *Atsushi Ando*[1], *Kentaro Shinayama*[1], *Marc Delcroix*[1]

[1]NTT Corporation, Japan

hrs.sato@ntt.com

Abstract

Real-time target speaker extraction (TSE) is intended to extract the desired speaker's voice from the observed mixture of multiple speakers in a streaming manner. Implementing real-time TSE is challenging as the computational complexity must be reduced to provide real-time operation. This work introduces to Conv-TasNet-based TSE a new architecture based on state space modeling (SSM) that has been shown to model long-term dependency effectively. Owing to SSM, fewer dilated convolutional layers are required to capture temporal dependency in Conv-TasNet, resulting in the reduction of model complexity. We also enlarge the window length and shift of the convolutional (TasNet) frontend encoder to reduce the computational cost further; the performance decline is compensated by over-parameterization of the frontend encoder. The proposed method reduces the real-time factor by 78% from the conventional causal Conv-TasNet-based TSE while matching its performance.

Index Terms: speech enhancement, streaming, lightweight, SpeakerBeam, TasNet, state space modeling, S4

1. Introduction

While recent advances in machine learning technology have drastically improved speech processing, it is still difficult to handle overlapping speech. Target speaker extraction (TSE) deals with such overlapping speech. It extracts only the target speaker's voice from the observed mixtures using an enrollment speech to identify the target speaker [1–5]. TSE is especially promising for personalized applications such as smartphones and smartwatches that are typically required to respond to just their owner. TSE is also effective in improving teleconferencing by removing background noise and interference speakers from the microphone signals.

Considering real-time speech understanding by machines and telecommunication by humans, it is essential to apply TSE in a causal and streaming manner. Streaming TSE requires not only low algorithmic latency suitable for streaming applications, but also sufficiently low computational cost at inference time to complete processing in real-time, i.e. real-time factor (RTF) < 1. Due to the general trade-off between computational efficiency and performance, constructing methods that simultaneously maintain both aspects is a challenge.

Despite the difficulty, some preceding works attempted to fulfill the requirements [6–9]. Wang *et al.* first proposed a real-time TSE by devising a lightweight TSE architecture called VoiceFilter-Lite [6]. Since VoiceFilter-Lite enhances filterbank features for ASR, it is not suitable for communication applications. Subsequently, Eskimez *et al.* proposed personalized deep complex convolution recurrent network (pDCCRN) [7]. pDCCRN efficiently models complex-valued spectrograms by using convolutional neural networks (CNN) to model local dependency and recurrent neural networks (RNN) to model global temporal dependency. More recently, Liu *et al.* proposed the end-to-end enhancement network (E3Net) utilizing a frontend encoder that maps a waveform into a latent representation sequence with 1-d convolutional filters to perform TSE on waveforms [8]. E3Net has better TSE performance and lower computation cost compared to pDCCRN under conditions when the target speaker is close to the interfering speaker. Another causal algorithm that performs TSE in the waveform domain is causal ConvTasNet [10]. Different from E3Net, which is mainly composed of Long Short-Term Memory (LSTM) layers, the separation network of ConvTasNet consists of multiple blocks of stacked dilated 1-D convolutional layers to attain a sufficiently wide receptive field. Although Conv-TasNet-based TSE demonstrates superior enhancement performance in causal TSE algorithms compared with VoiceFilter-Lite, pDCCRN, and E3Net [11], it cannot achieve real-time inference on general CPUs. This is due to 1) a high number of 1-D convolutional blocks that must be stacked to ensure a long-enough receptive field, and 2) the short window size and shift of the frontend encoder results in a large number of frames to be processed.

State space modeling (SSM) has recently been introduced to neural sequence modeling and shown to have superior performance in modeling long-range dependency, compared with traditional deep sequence models such as RNN, CNN, and transformers [12–17]. Structured State Space Sequence model (S4) [15] and the Diagonal version of S4 (S4D) [16], S4ND [17] are common SSM variants. Several speech processing studies use SSM in place of RNNs or CNNs and demonstrated successfully reduced model footprint or increased throughput while maintaining performance, such as in speech recognition [18, 19], speech generation [20], and denoising [21]. However, it has yet to be applied to the TSE task.

In this work, we propose a new TSE architecture, SpeakerBeam with State Space modeling (SpeakerBeam-SS); it achieves real-time TSE with high computational efficiency. By introducing S4D layers to Conv-TasNet-based TSE [10, 22], fewer dilated convolutional layers are required to capture temporal dependency, resulting in the reduction of model complexity. To further reduce computational complexity, we adopt a wider window size and shift of the frontend encoder to reduce the number of frames processed, which generally causes a performance decline. To compensate for this, we adopt over-parameterization of the frontend encoder following [8].

An experiment on simulated mixtures of two speakers and noise showed that SpeakerBeam-SS reduced RTF by 78% compared with the conventional Conv-TasNet-based TSE architecture, while maintaining performance in terms of signal-to-distortion ratio (SDR) and DNSMOS.

10.21437/Interspeech.2024-413

2. Related work

One preceding work utilizing SSM for speech enhancement is S4ND U-Net [21]. It uses the S4ND [17] to model multidimensional patterns with SSM. The major difference from this study, apart from the task (denoising vs TSE), is that we focus on causal and real-time inference whereas S4ND focuses on reducing the model footprint and not measuring inference speed. Despite the small footprint, S4ND-UNet was computationally intensive in our experiments; it requires approximately ten times more training time per epoch than the proposed method[1].

3. Background

3.1. Conv-TasNet-based TSE

In this work, we assume that the observed mixture $\boldsymbol{y} \in \mathbb{R}^T$ is a single channel signal modeled as $\boldsymbol{y} = \boldsymbol{s} + \boldsymbol{i} + \boldsymbol{n}$ where $\boldsymbol{s}, \boldsymbol{i}, \boldsymbol{n} \in \mathbb{R}^T$ are clean target speech, interfering speech, and noise, respectively; T denotes the number of samples in the signal. TSE aims at extracting only the target speaker's speech from the observed mixture by using the enrollment speech of the target speaker $\boldsymbol{c}^{\mathrm{S}} \in \mathbb{R}^{T_c}$, as in $\hat{\boldsymbol{s}} = \texttt{TSE}(\boldsymbol{y}, \boldsymbol{c}^{\mathrm{S}}; \boldsymbol{\Lambda})$ where $\texttt{TSE}(\cdot,\cdot)$ represents the TSE network, $\hat{\boldsymbol{s}} \in \mathbb{R}^T$ is the enhanced signal, T_c is the number of samples in the enrollment speech, and $\boldsymbol{\Lambda}$ denotes learnable parameters of the model. In Conv-TasNet-based TSE model [22], the input signal is first projected into a latent space by a frontend encoder consisting of a 1-D convolutional layer as follows:

$$\boldsymbol{Z}_{\text{enc}} = \texttt{Encoder}(\boldsymbol{y}; \boldsymbol{\theta}_{\text{enc}}), \tag{1}$$

where $\texttt{Encoder}(\cdot)$ represents the encoder network, $\boldsymbol{Z}_{\text{enc}} \in \mathbb{R}^{N \times T'}$ is the latent representation of the input signal, and $\boldsymbol{\theta}_{\text{enc}}$ is learnable parameters of the encoder. N is the number of filters in the encoder. T' is the number of the frames, which is determined by the number of samples in the signal T and the kernel size L and shift $L/2$ of the 1-D convolutional layer. To identify the target speaker, speaker vector $\boldsymbol{e}^{\mathrm{S}}$ is extracted from enrollment speech $\boldsymbol{c}^{\mathrm{S}}$ by a learnable speaker encoder. Based on $\boldsymbol{e}^{\mathrm{S}}$, the separator network extracts the target speaker as follows:

$$\boldsymbol{Z}^{\mathrm{S}}_{\text{sep}} = \texttt{Separator}(\boldsymbol{Z}_{\text{enc}}, \boldsymbol{e}^{\mathrm{S}}; \boldsymbol{\theta}_{\text{sep}}), \tag{2}$$

where $\texttt{Separator}(\cdot,\cdot)$ represents the separator network, $\boldsymbol{Z}^{\mathrm{S}}_{\text{sep}} \in \mathbb{R}^{N \times T'}$ is the extracted representation of the target speaker's speech, and $\boldsymbol{\theta}_{\text{sep}}$ is the learnable parameters of the separator. The separator network in Conv-TasNet consists of multiple stacked dilated 1-D convolutional blocks with increasing dilation from $d = 1$ to $d = 2^{X-1}$ to ensure a sufficient receptive field where X is the number of repetitions of dilated 1-D convolutional blocks. Representation $\boldsymbol{Z}^{\mathrm{S}}_{\text{sep}}$ is then transformed back into a waveform signal, $\hat{\boldsymbol{s}}$, by a 1-D transposed convolution layer-based decoder as follows:

$$\hat{\boldsymbol{s}} = \texttt{Decoder}(\boldsymbol{Z}^{\mathrm{S}}_{\text{sep}}; \boldsymbol{\theta}_{\text{dec}}), \tag{3}$$

where $\texttt{Decoder}(\cdot)$ represents the decoder network, and $\boldsymbol{\theta}_{\text{dec}}$ represents the learnable parameters of the decoder. All parameters $\boldsymbol{\Lambda} = \{\boldsymbol{\theta}_{\text{enc}}, \boldsymbol{\theta}_{\text{sep}}, \boldsymbol{\theta}_{\text{dec}}, \boldsymbol{\theta}_{\text{spkenc}}\}$ are jointly trained on the minimization criterion between the enhanced speech $\hat{\boldsymbol{s}}$ and target speech $\boldsymbol{s}$, where $\boldsymbol{\theta}_{\text{spkenc}}$ denotes the learnable parameters of the speaker encoder.

Although Conv-TasNet-based TSE shows prominent performance among TSE networks, its computation cost is too high to attain real-time inferencing on general CPUs. One cause of this is the high number of frames processed by the separator network; the kernel size L and shift $L/2$ of the frontend encoder are generally as short as 20 and 10 samples, corresponding 1.25 and 0.625 ms at sampling rates of 16 kHz, respectively [22]. Another cause is the large number of repetitions of stacked dilated 1-D convolutional blocks in the separator network needed to attain a sufficient receptive field.

[1] We investigated the extension of S4ND-UNet to TSE tasks by feeding d-vector or speaker vector extracted by ECAPA-TDNN, but training was unstable and competitive performance was not achieved.

3.2. State Space Modeling

S4 [15] and S4D [16] are variants of SSM. The start point of SSM is the introduction of ordinary differential equations (ODEs) that map one-dimensional signal $u(t) \longmapsto v(t)$ as follows:

$$\boldsymbol{x}'(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}u(t), \;\; v(t) = \boldsymbol{C}\boldsymbol{x}(t) + \boldsymbol{D}u(t), \tag{4}$$

where $\boldsymbol{A} \in \mathbb{C}^{D \times D}$, $\boldsymbol{B} \in \mathbb{C}^{D \times 1}$, $\boldsymbol{C} \in \mathbb{C}^{1 \times D}$, and $\boldsymbol{D} \in \mathbb{C}^{1 \times 1}$ are state transition matrices that are learnable parameters. $\boldsymbol{x}(t) \in \mathbb{C}^{D \times 1}$ is the internal state of the state space equation. The ODEs are discretized by either Bilinear or zero-order hold (ZOH) discretization to permit the use of sequence-to-sequence modeling. The discretized form of Eq. (4) by ZOH discretization is expressed as follows:

$$\boldsymbol{x}_k = \overline{\boldsymbol{A}}\boldsymbol{x}_{k-1} + \overline{\boldsymbol{B}}u_k, \;\; v_k = \boldsymbol{C}\boldsymbol{x}_k + \boldsymbol{D}u_k, \tag{5}$$

where $\overline{\boldsymbol{A}} = \exp(\Delta\boldsymbol{A}), \overline{\boldsymbol{B}} = (\Delta\boldsymbol{A})^{-1}(\exp(\Delta \cdot \boldsymbol{A}) - \boldsymbol{I}) \cdot \Delta\boldsymbol{B}$ and $\Delta \in \mathbb{R}$ is a learnable parameter representing time-step size. According to Eq.(5), S4 and S4D can be calculated causally by a simple linear recurrent neural network at inference, which improves computation efficiency.

By setting initial state $\boldsymbol{x}_0$ to $\boldsymbol{0}$, Eq. (5) can be expanded as follows:

$$v_k = \boldsymbol{C}\overline{\boldsymbol{A}}^k\overline{\boldsymbol{B}}u_0 + \boldsymbol{C}\overline{\boldsymbol{A}}^{k-1}\overline{\boldsymbol{B}}u_1 + \cdots + \boldsymbol{C}\overline{\boldsymbol{B}}u_k + \boldsymbol{D}u_k. \tag{6}$$

Although the S4 layer processes the signal recurrently in a sample-by-sample manner at inference time, its training and batch inference can be parallelized by introducing SSM convolutional kernel $\bar{\boldsymbol{K}} := (\boldsymbol{C}\overline{\boldsymbol{B}}, \boldsymbol{C}\overline{\boldsymbol{A}\boldsymbol{B}}, ..., \boldsymbol{C}\overline{\boldsymbol{A}}^{K-1}\overline{\boldsymbol{B}})$ where $\bar{\boldsymbol{K}}$ is the signal length [13]. Output signal $\boldsymbol{v} = \{v_0, ..., v_{K-1}\}$ can be written in the form of the convolution of the kernel $\bar{\boldsymbol{K}}$ and the input signal $\boldsymbol{u} = \{u_0, ..., u_{K-1}\}$ as follows:

$$\boldsymbol{v} = \boldsymbol{u} * \bar{\boldsymbol{K}} + \boldsymbol{D}u_k, \tag{7}$$

It is known that fast Fourier transform (FFT) can calculate Eq. (7) efficiently. Since the input and the output to the S4 layer are actually the two-dimensional feature, Eq. (5) and Eq. (7) are independently applied to each feature dimension.

For S4, state matrix $\boldsymbol{A}$ is parameterized as a diagonal plus low rank (DPLR) form, which theoretically allows the internal state to memorize the history of the input, resulting in the ability to model long-range dependency. S4D further simplifies the parameterization of $\boldsymbol{A}$ to diagonal form, which is also empirically effective in temporal modeling. Moreover, S4D parameterizes $\boldsymbol{B}$ as the fixed term $\boldsymbol{B} = \boldsymbol{1}$.

4. Proposed method

In this work, we propose a new TSE architecture, SpeakerBeam-SS. Figure 1 (a) shows an overview.

SpeakerBeam-SS basically follows the architecture of the Conv-TasNet-based TSE proposed in [22]. For causal implementation, channel-wise layer normalization and causal convolutional layers are adopted in place of global layer normalization and non-causal convolutional layers.

To reduce computational complexity, we introduce the S4D layer to the Conv-TasNet-based architecture to model long-

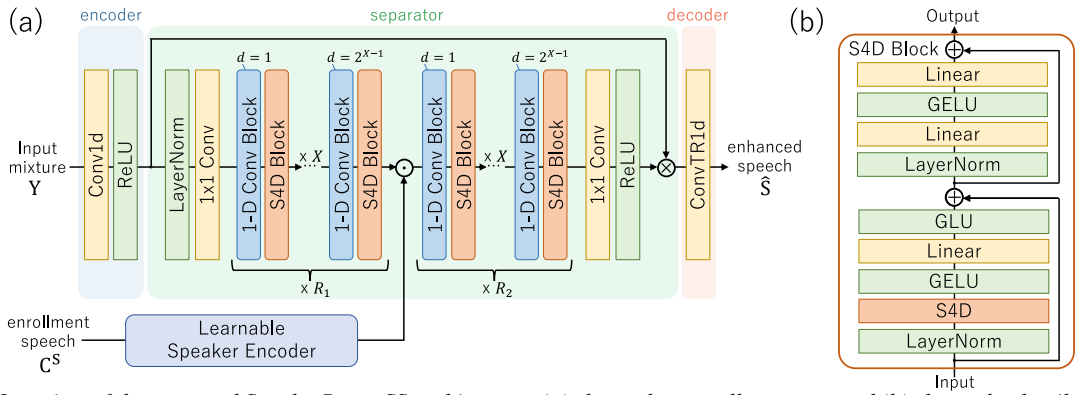

Figure 1: *Overview of the proposed SpeakerBeam-SS architecture. (a) shows the overall structure and (b) shows the details of the S4D block. The dropout layer is omitted from the figure. d refers to the dilation of 1-D convolutional blocks.*

range dependency more efficiently. The 1-D convolution blocks of the original Conv-TasNet architecture can only see the context provided by the receptive field of the layers below them. Here we augment the conv-blocks with S4D blocks to allow access to the long context. Consequently, fewer repetitions of blocks are required to model enough length of the temporal context. The detailed structure of the S4D block is shown in Figure 1 (b). The S4D block structure is based on the S4 block in [20] with the modification of replacing S4 layer with S4D layer. A preliminary experiment on the Conv-TasNet-based architecture showed that introducing the S4D layer yielded better performance than S4 or DSS [14]. The design of the 1-D convolutional blocks follows [22].

To further reduce the computational complexit to meet the requirement of real-time processing, we increased the window size and shift. Although this incurs a performance penalty, which we compensate by increasing the number of the filters of the frontend encoder as found in [8]. This also largely reduces the RTF while maintaining the performance (see Figure 2).

Since the S4D block works in a causal manner, the maximum algorithmic latency of the proposed SpeakerBeam-SS is the same as the window length of the frontend encoder. We also implemented lookahead for the proposed SpeakerBeam-SS for the applications where lower latency constraints are acceptable, by modifying several 1-D convolutional blocks to be non-causal.

5. Experiments

5.1. Experimental setup

5.1.1. Dataset

Training and evaluation were performed on simulated mixtures generated by speech recordings from the LibriSpeech corpus [23] and noise samples from the DNS4 challenge dataset [24]. For training, we prepared mixtures of two speakers and noise, where the noise was added at signal-to-noise ratio (SNR) values randomly sampled between 0 and 25 dB. We evaluated two conditions: 1) target and interference speakers and noise mixtures generated with SNR values between 10 and 20 dB and 2) target speaker and noise mixtures generated at SNR values between 0 and 10 dB. The signal-to-interference ratio (SIR) between the target and interfering speech was set to randomly sampled values between -5 to 5 dB for training and evaluation data. We generated 50,000, 3,000, and 2,000 mixtures for training, development, and evaluation datasets, respectively, with no duplicate speakers between each dataset.

5.1.2. System configuration and training procedure

For the baseline Conv-TasNet-based TSE and the proposed SpeakerBeam-SS, we set the hyperparameter B=256, H=512 and $P=3$ following the notation in [10]. We set R_1 and R_2 in Figure 1 to 3 and 1, respectively. We set the number of repetitions of the blocks X to 8 in the baseline Conv-TasNet-based TSE and 2 in SpeakerBeam-SS. We increased the number of filters in the frontend encoder, N, from 256 to 4096, and the kernel width, L, and shift $L/2$ from 20 and 10 to 320 and 160, respectively. We implemented SpeakerBeam-SS based on the public Conv-TasNet implementation in [25] and S4D implementation in [26]. The learnable speaker encoder consisted of one dilated convolution block and frontend encoder following [22]. We implemented 40 ms and 120 ms lookahead for SpeakerBeam-SS by modifying the first or first and second 1-D convolutional blocks to be non-causal. We set the state space dimension D to 32 and the input dimension of the SSM to 256. The hidden dimension of the fully connected layer in S4D blocks was set to 512. We used the reduce-on-plateau learning rate scheduler with a peak learning rate of 5e-4.

We evaluated pDCCRN and E3Net for comparison[2]. We implemented pDCCRN based on the public implementation [27] by modifying it to receive speaker embeddings. Each CNN's kernel size and number of kernels were set to 5 and $\{16, 32, 64, 128, 256, 256\}$, respectively. We adopted a 2-layer RNN with 128 units. For STFT, we set the window length to 32 ms and the hop to 16 ms. We used the same learning rate scheduler as mentioned above with peak learning rate of 5e-4. We implemented E3Net based on the description in [8]. We set the number of filters in the frontend encoder to 2048, kernel width to 320, and shift to 160, respectively. We adopted a 4-layer LSTM with 256 units. The hidden dimension of the fully connected block was set to 1024. We used a cosine annealing scheduler with peak learning rate of 1e-3. As the speaker encoder of pDCCRN and E3Net, we adopted a publicly available d-vector extractor trained on LibriSpeech [23] and VoxCeleb 1 and 2 [28, 29] dataset [30][3]. All models were trained with Adam optimizer [31] for 200 epochs with early stopping if the loss did not decrease within 20 epochs. We evaluated the averaged model against the three models that performed the best on the development set.

[2]It has been reported that E3Net set the restriction that the target speaker should be closer to the microphone than the interference speaker [8], yet we examine the TSE performance without this restriction to gain a broader range of insights.

[3]Preliminary experiments showed that the d-vector extractor performed better than ECAPA-TDNN and the learnable speaker encoder model trained from scratch.

Table 1: *Performance and model complexity of Conv-TasNet based and SpeakerBeam-SS systems. The results were obtained on mixtures of two speakers and noise. 'OP' stands for the over-parameterization of the frontend encoder, corresponding to $N = 2048$. L and N represent the window shift and the number of filters in the frontend encoder, respectively. X is the number of repetitions of the blocks in the separator network. 95% confidence interval (CI) is calculated by the bootstrapping method. Values in the bold letter indicate the best performance within systems with delay of 20 ms or less.*

		Model architecture				Latency [ms]	Complexity		SDR [dB]		DNSMOS		
	Method	L	N	X	S4D		#param	RTF	mean	(95% CI)	OVRL	SIG	BAK
	Mixture	-	-	-	-	-	-	-	−0.45	-	2.44	3.95	2.17
(a)	Baseline-noncausal	20	256	8		∞	8.69 M	-	13.63	(13.39, 13.86)	3.40	3.71	4.24
(b1)	Baseline	20	256	8		1.25	8.69 M	1.67	11.10	(10.88, 11.30)	2.82	3.22	3.76
(b2)		320	256	8		20	8.84 M	0.40	9.86	(9.64, 10.07)	2.76	3.14	3.82
(c1)	Baseline + OP	320	2048	8		20	10.91 M	0.54	11.41	(11.19, 11.59)	2.91	3.29	3.87
(c2)		320	2048	2		20	6.87 M	0.22	9.42	(9.21, 9.59)	2.64	3.04	3.68
(d1)	Baseline + OP + S4D	320	2048	2	✓	20	7.93 M	0.36	**11.58**	(11.37, 11.77)	**2.95**	**3.33**	**3.89**
(d2)	+ lookahead 40 ms	320	2048	2	✓	60	7.93 M	0.36	11.88	(11.67, 12.06)	3.14	3.51	4.03
(d3)	+ lookahead 120 ms	320	2048	2	✓	140	7.93 M	0.35	12.12	(11.89, 12.32)	3.21	3.58	4.05

Table 2: *Performance comparison with existing real-time TSE architectures in SDR [dB] and DNSMOS OVRL values.*

	Method	Latency [ms]	2 speakers + noise		1 speaker + noise	
			SDR	OVRL	SDR	OVRL
	Mixture	-	−0.45	2.44	3.84	2.28
(e)	pDCCRN [7]	32	10.70	2.83	10.66	2.78
(f)	E3Net [8]	20	9.83	2.83	10.37	2.77
(d1)	Ours	20	**11.58**	**2.95**	**11.10**	**2.89**

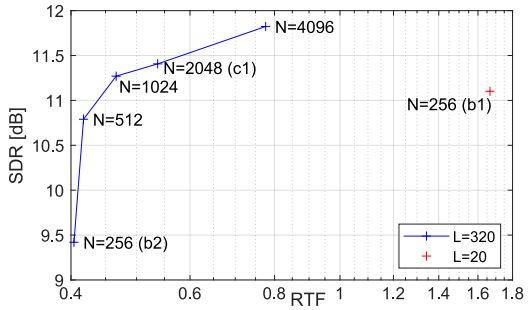

Figure 2: *The relationship between RTF and the enhancement performance with various numbers of filters N and window size L, in the frontend encoder.*

5.1.3. Evaluation details

As performance metrics, we report SDR [32] and DNSMOS P.835 [33] values. Moreover, to precisely measure the practical RTF, we implement the inference algorithm for each system on C++. We measured the RTF of frame-by-frame inferencing achieved when using one core of an AMD EPYC 7502P based on our C++ implementation, as averaged over 100 audio clips.

5.2. Experimental results

Table 1 shows the complexity and the TSE performance for each system. Comparing systems (b1) and (b2), enlarging window size L and shift $L/2$ can significantly lower the RTF by reducing the number of frames processed; however, this also results in a significant decrease in performance. Note that the window size of $L = 320$ corresponds to 20 ms of maximum algorithmic delay. The increase in the number of convolution filters N from 256 to 2048 (c1) successfully compensates the performance drop while reducing the RTF by about 68% compared with the baseline system (b1).

The reduction in the repetition number of 1-D convolutional block X from 8 to 2 (c2) lowers the RTF by reducing the computation cost, yet at the same time, the performance significantly drops due to the narrowing of the receptive field. By introducing S4D blocks, which can efficiently model long sequences, system (d1) attained enhancement performance equal to that of system (c1) with fewer repetitions. Compared with system (c1), system (d1) achieved about a 27% reduction in model parameters and a 33% reduction in RTF while maintaining the enhancement performance. This result indicates the effectiveness of the S4D layer in constructing real-time TSE. As a whole, compared to the initial setup of the Conv-TasNet-based TSE models in [22] (b1), SpeakerBeam-SS (d1) offers a 78% reduction in RTF. Rows (d2) and (d3) show that if lower latency constraints are acceptable, the introduction of lookahead within SpeakerBeam-SS can improve performance.

Table 2 shows a performance comparison of the proposed SpeakerBeam-SS with other architectures. In both 2 speaker and noise and 1 speaker and noise conditions, the proposed method outperforms conventional architectures. Although E3Net was reported in [8] to have better performance than pDCCRN, this is not the case in our experimental conditions. This is probably because the target speaker is not necessarily closer than the interfering speaker as reported in [8].

To further understand the effect of the number of filters in the frontend encoder, we investigated the relationship between RTF and performance with various numbers of filters in frontend encoder N and window size L, shown in Figure 2. With the conventional setup of the number of filters $N = 256$, performance drops substantially when the length of the window is widened from $L = 20$ to $L = 320$. However, the increase in the number of filters N recovers the performance while only modestly impacting the RTF.

6. Conclusion

In this work, we investigated reducing the complexity of the Conv-TasNet-based TSE model while maintaining its performance to realize real-time TSE. The over-parameterization of the frontend encoder while increasing the size of the convolutional kernel and the introduction of state space modeling both successfully reduced RTF without hurting performance. As a result, a 78% reduction in RTF has been achieved overall compared with the baseline, without hurting the SDR or DNSMOS values. The proposed SpeakerBeam-SS also shows superior performance to conventional architectures. The introduction of SSM to other TSE architectures such as pDCCRN and E3Net than ConvTasNet will be part of our future work.

7. References

[1] M. Delcroix, K. Žmolíková, K. Kinoshita, A. Ogawa, and T. Nakatani, "Single channel target speaker extraction and recognition with speaker beam," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, 2018, pp. 5554–5558.

[2] J. Wang, J. Chen, D. Su, L. Chen, M. Yu, Y. Qian, and D. Yu, "Deep extractor network for target speaker recovery from single channel speech mixtures," in *Proc. Annual Conference of the International Speech Communication Association*, 2018, pp. 307–311.

[3] K. Žmolíková, M. Delcroix, K. Kinoshita, T. Ochiai, T. Nakatani, L. Burget, and J. Černockỳ, "SpeakerBeam: Speaker aware neural network for target speaker extraction in speech mixtures," *IEEE Journal of Selected Topics in Signal Processing*, vol. 13, no. 4, pp. 800–814, 2019.

[4] Q. Wang, H. Muckenhirn, K. Wilson, P. Sridhar, Z. Wu, J. R. Hershey, R. A. Saurous, R. J. Weiss, Y. Jia, and I. L. Moreno, "VoiceFilter: Targeted voice separation by speaker-conditioned spectrogram masking," in *Proc. Annual Conference of the International Speech Communication Association*, 2019, pp. 2728–2732.

[5] K. Zmolikova, M. Delcroix, T. Ochiai, K. Kinoshita, J. Černocký, and D. Yu, "Neural target speech extraction: An overview," *IEEE Signal Processing Magazine*, vol. 40, no. 3, pp. 8–29, 2023.

[6] Q. Wang, I. L. Moreno, M. Saglam, K. Wilson, A. Chiao, R. Liu, Y. He, W. Li, J. Pelecanos, M. Nika, and A. Gruenstein, "VoiceFilter-Lite: Streaming targeted voice separation for on-device speech recognition," in *Proc. Annual Conference of the International Speech Communication Association*, 2020, pp. 2677–2681.

[7] S. E. Eskimez, T. Yoshioka, H. Wang, X. Wang, Z. Chen, and X. Huang, "Personalized speech enhancement: New models and comprehensive evaluation," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, 2022, pp. 356–360.

[8] M. Thakker, S. E. Eskimez, T. Yoshioka, and H. Wang, "Fast real-time personalized speech enhancement: End-to-end enhancement network (E3Net) and knowledge distillation," in *Proc. Annual Conference of the International Speech Communication Association*, 2022, pp. 991–995.

[9] H. Dubey, V. Gopal, R. Cutler, A. Aazami, S. Matusevych, S. Braun, S. E. Eskimez, M. Thakker, T. Yoshioka, H. Gamper, and R. Aichner, "ICASSP 2022 deep noise suppression challenge," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, 2022, pp. 9271–9275.

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

[11] X. Liu, X. Li, and J. Serrà, "Quantitative evidence on overlooked aspects of enrollment speaker embeddings for target speaker separation," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2023, pp. 1–5.

[12] A. Gu, T. Dao, S. Ermon, A. Rudra, and C. Ré, "HiPPO: Recurrent memory with optimal polynomial projections," in *Proc. Advances in neural information processing systems*, vol. 33, 2020, pp. 1474–1487.

[13] A. Gu, I. Johnson, K. Goel, K. Saab, T. Dao, A. Rudra, and C. Ré, "Combining recurrent, convolutional, and continuous-time models with linear state space layers," in *Proc. Advances in neural information processing systems*, vol. 34, 2021, pp. 572–585.

[14] A. Gupta, A. Gu, and J. Berant, "Diagonal state spaces are as effective as structured state spaces," in *Proc. Advances in Neural Information Processing Systems*, vol. 35, 2022, pp. 22 982–22 994.

[15] A. Gu, K. Goel, and C. Ré, "Efficiently modeling long sequences with structured state spaces," in *Proc. International Conference on Learning Representations*, 2022.

[16] A. Gu, K. Goel, A. Gupta, and C. Ré, "On the parameterization and initialization of diagonal state space models," in *Proc. Advances in Neural Information Processing Systems*, vol. 35, 2022, pp. 35 971–35 983.

[17] E. Nguyen, K. Goel, A. Gu, G. Downs, P. Shah, T. Dao, S. Baccus, and C. Ré, "S4ND: Modeling images and videos as multidimensional signals with state spaces," in *Proc. Advances in neural information processing systems*, vol. 35, 2022, pp. 2846–2861.

[18] K. Miyazaki, M. Murata, and T. Koriyama, "Structured state space decoder for speech recognition and synthesis," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, 2023, pp. 1–5.

[19] H. Shan, A. Gu, Z. Meng, W. Wang, K. Choromanski, and T. Sainath, "Augmenting conformers with structured state space models for online speech recognition," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, 2024 (to appear).

[20] K. Goel, A. Gu, C. Donahue, and C. Ré, "It's raw! audio generation with state-space models," in *International Conference on Machine Learning*, 2022, pp. 7616–7633.

[21] P.-J. Ku, C.-H. H. Yang, S. Siniscalchi, and C.-H. Lee, "A Multidimensional Deep Structured State Space Approach to Speech Enhancement Using Small-footprint Models," in *Proc. Annual Conference of the International Speech Communication Association*, 2023, pp. 2453–2457.

[22] M. Delcroix, T. Ochiai, K. Zmolikova, K. Kinoshita, N. Tawara, T. Nakatani, and S. Araki, "Improving speaker discrimination of target speech extraction with time-domain SpeakerBeam," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, 2020, pp. 691–695.

[23] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "LibriSpeech: An ASR corpus based on public domain audio books," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, 2015, pp. 5206–5210.

[24] C. K. Reddy, H. Dubey, K. Koishida, A. Nair, V. Gopal, R. Cutler, S. Braun, H. Gamper, R. Aichner, and S. Srinivasan, "INTERSPEECH 2021 deep noise suppression challenge," in *Proc. Annual Conference of the International Speech Communication Association*, 2021, pp. 2796–2800.

[25] "conv-tasnet," https://github.com/funcwj/conv-tasnet, Cited February 28 2024.

[26] "state-spaces," https://github.com/state-spaces/s4, Cited February 28 2024.

[27] "DeepComplexCRN," https://github.com/huyanxin/DeepComplexCRN, Cited February 28 2024.

[28] A. Nagrani, J. S. Chung, and A. Zisserman, "VoxCeleb: A Large-Scale Speaker Identification Dataset," in *Proc. Annual Conference of the International Speech Communication Association*, 2017, pp. 2616–2620.

[29] J. S. Chung, A. Nagrani, and A. Zisserman, "VoxCeleb2: Deep speaker recognition," in *Proc. Annual Conference of the International Speech Communication Association*, 2018, pp. 1086–1090.

[30] "Resemblyzer," https://github.com/resemble-ai/Resemblyzer/tree/master, Cited February 28 2024.

[31] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. International Conference on Learning Representations*, 2015.

[32] E. Vincent, R. Gribonval, and C. Févotte, "Performance measurement in blind audio source separation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 4, pp. 1462–1469, 2006.

[33] C. K. Reddy, V. Gopal, and R. Cutler, "Dnsmos p. 835: A non-intrusive perceptual objective speech quality metric to evaluate noise suppressors," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, 2022, pp. 886–890.