INTERSPEECH 2023
20-24 August 2023, Dublin, Ireland

Real-Time Personalised Speech Enhancement Transformers with Dynamic Cross-attended Speaker Representations

Shucong Zhang, Malcolm Chadwick, Alberto Gil C. P. Ramos, Titouan Parcollet, Rogier van Dalen, Sourav Bhattacharya

Samsung AI Centre, Cambridge, UK

{s1.zhang,malcolm.c,a.gilramos,t.parcollet,r.vandalen,sourav.b1}@samsung.com

Abstract

Personalised speech enhancement (PSE) extracts only the speech of a target user and removes everything else from corrupted input audio. This can greatly improve on-device streaming audio processing, such as voice calls and speech recognition, which has strict requirements on model size and latency. To focus the PSE system on the target speaker, it is conditioned on a recording of the user's voice. This recording is usually summarised as a single static vector. However, a static vector cannot reflect all the target user's voice characteristics. Thus, we propose using the full recording. To condition on such a variable-length sequence, we propose fully Transformer-based PSE models with a cross-attention mechanism which generates target speaker representations dynamically. To better reflect the on-device scenario, we carefully design and publish a new PSE dataset. On the dataset, our proposed model significantly surpasses strong baselines while halving the model size and reducing latency.

Index Terms: personalised speech enhancement, speech separation, Transformers

1. Introduction

Audio source separation can greatly improve voice calls and speech recognition. Deep neural networks have lately shown great performance improvements in audio source separation tasks [1, 2, 3]. In many cases, such as on a mobile phone, data from the target user is available to improve performance further. The process of removing all other audio components from the corrupted input utterance, both environmental noise and other voices than the target user, is called personalised speech enhancement (PSE). For the mobile phone scenario, response time and application size are critical factors. Thus, streaming-based inference, latency, and model size are of paramount importance when designing on-device PSE systems.

For PSE systems, personalisation relies on a previously recorded enrolment clip of the target user. Previous works typically use a separate embedding neural network to summarise the entire enrolment clip into a static embedding vector, and this embedding vector is viewed as a voice profile of the user [4, 5]. A widely-used method of utilising the voice profile is to concatenate the embedding with the intermediate features of the PSE network [4, 5, 6, 7, 8, 9]. Even previous works, which utilise attention mechanisms, execute the attention between the corrupted input utterance and the single static embedding vector [10, 11, 12]. However, the single static vector may fail to capture the variability of the target user's speech.

To represent the variability in the enrolment audio, we propose to use a sequence of hidden states instead of a single static embedding vector. As shown in related fields, such as text-to-speech (TTS) and voice cloning, a sequence of hidden states captures the speaker characteristics better than a single vector [13, 14]. To allow a PSE system to rely on an entire sequence of vectors, we propose a fully Transformer-based model with a novel cross-attention mechanism. For each input frame, the proposed method dynamically constructs a suitable speaker representation for the separation task by choosing the most relevant enrolment audio frames. Thus, compared to [10, 11, 12], this leads to more flexible and relevant speaker representations.

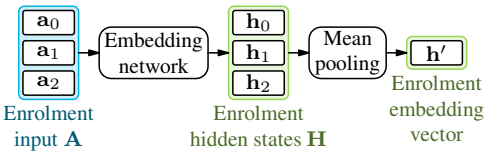


Figure 1: *Speaker Embeddings. The enrolment clip A is transformed into a sequence of hidden states H though an embedding network. Then, mean pooling is applied to produce a single vector representation h' of A.*

To better match real-world applications than existing datasets, such as WSJ0-2mix [15] and LibriMix [16], we build a dataset to reflect the on-device close-talk microphone scenarios, where there are numerous types of environmental ambient noise and both of the ambient and babble noise can happen anywhere in the audio input. Through extensive experiments, we compare our proposed models with strong baselines. Thanks to the flexibility of the proposed cross-attention, our model achieves 1–2 dB absolute gain in terms of signal-to-distortion ratio (SDR) and 10–30 % relative word error rate (WER) reduction in the downstream automatic speech recognition (ASR) tasks. Even with only half the model size, our proposed model still surpasses the baselines consistently and significantly. Further, due to the halved model size, our online model gives similar or smaller latency compared to the baselines. Thus, the key contributions of our work include: (1) a novel cross-attention mechanism which utilises the dynamics of both the enrolment audio and the input audio; (2) a fully Transformer-based online streaming PSE model which outperforms strong baselines in the three critical aspects (i.e., PSE and downstream ASR performance, model size, and latency) of the real-world on-device scenarios; (3) a dataset for real-world on-device PSE tasks. [1]

2. Background

2.1. Single Speaker Embedding Vector Extraction

PSE systems typically condition on a single embedding vector for the speaker, which is usually extracted from an enrolment

[1] Dataset available at: https://github.com/shucongzhang/CrossAttnPse

10.21437/Interspeech.2023-1066

utterance A. For instance, a normalised i-vector [17] or a d-vector [18] is commonly used as the single embedding vector. Fig. 1 shows the work-flow of using a d-vector network to extract a single enrolment embedding vector. Once the enrolment audio has been recorded, the single embedding vector does not change.

2.2. PSE as Mask Prediction

Given a corrupted input utterance $\boldsymbol{X}' = (\boldsymbol{x}'_1, \boldsymbol{x}'_2, ..., \boldsymbol{x}'_t)$ and the enrolment utterance $\boldsymbol{A} = (\boldsymbol{a}_1, \boldsymbol{a}_2, ..., \boldsymbol{a}_p)$ of a target speaker, where t and p are the sequence length for $\boldsymbol{X}'$ and $\boldsymbol{A}$ respectively, a PSE task is to extract the target speaker's speech $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_t)$ accurately. Since our focus is on-device, we do not use convolutional layers to extract features from the raw wave [1], in order to reduce model size, overall computations, and latency. Here, we consider PSE as a spectral-subtraction task [19]. Thus, in the scope of the paper, both $\boldsymbol{X}'$ and $\boldsymbol{X}$ are spectrograms. For the corrupted input $\boldsymbol{X}'$, the PSE model generates a mask whose elements are in $[0, 1]$ by conditioning on $\boldsymbol{A}$. Then, the mask is applied on $\boldsymbol{X}'$ and the output should be as close to $\boldsymbol{X}$ as possible.

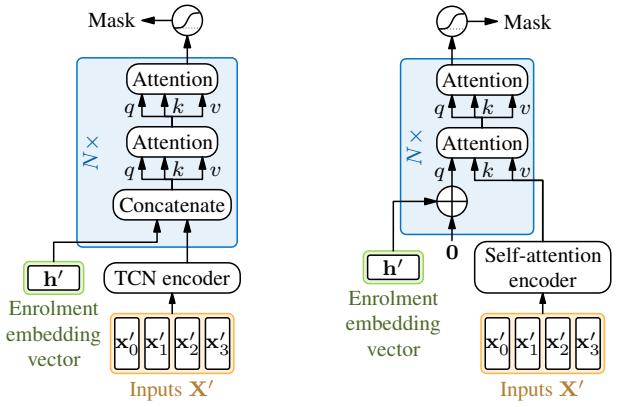
3. Related Work

For PSE systems, the main trend to combine a static vector with the speech enhancement model is to simply concatenate it with the hidden representations originating from the corrupted input signal [4, 5, 6, 7, 8, 9, 20, 21, 22, 23, 24, 25]. Fig. 2(a) gives an example of a previous work [8] that uses concatenation. Although the authors of [8] claim to use cross-attention, as shown in Fig. 2(a), the input to the attention network is the concatenation of corrupted audio hidden states and a single static vector. Other approaches consist of integrating factorized [6] or conditioning layers [26, 27] such as FiLM [28] or learnable activation [29] to the PSE model to blend the static speaker information in a learnable way. For previous works that use attention mechanism to inject the speaker profile [10, 11, 12], the attention is still between the corrupted input and the single static vector. Fig. 2(b) illustrates the architecture of a previous work [12] which applies attention on the single static vector. In our work however, we propose to use cross-attention to attend the full sequence of enrolment utterance hidden states, since a sequence of hidden states typically better captures the speaker characteristics than a single static vector [13, 14].

A different but related use case is where a voice profile is available for not just the target speaker, but also for an interfering speaker. [30, 31] use an attention layer to attend both the target and the interference enrolment audio once, but otherwise use only recurrent neural networks. The scenario where interfering speakers are not only known but have also enrolled could be relevant for home smart speakers, but it is not considered plausible for this work, which focuses on mobile phones.

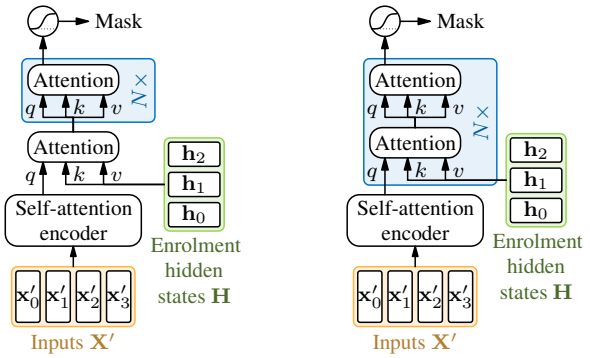
[32] uses attention mechanisms to soft-select a single static embedding for the current speaker from multiple enrolled users. This is not suitable for our mobile phone use case, where the device is typically used by a single user, and the motivation in [32] is not to produce dynamic speaker representations.

[6] applies an attention mechanism only within the enrolment utterance itself to generate a single static speaker embedding. In this work however, we propose to execute attention between the enrolment clip and the corrupted input clip to generate dynamic speaker representations.



(a) *SpEX$_{pc}$* (b) *AvaTr*

Figure 2: *Baselines which use on a single static embedding vector. We implement SpEX$_{pc}$ following [8] and AvaTr following [12]. TCN refers to temporal convolutional networt [7, 8] and attention refers to multi-head attetion layer [33].*



(a) *Cross-Seq* (b) *Cross-Seq+*

Figure 3: *Proposed models which utilise cross-attention to attend the full sequence of enrolment hidden states.*

4. Method

To better capture the dynamics of both the enrolment utterance and the corrupted input audio, we propose two variations of Transformer-based encoder-decoder models. Fig. 3 presents an architectural overview of our proposed models. Considering the on-device use case, we further make these models streaming.

4.1. Self-attention Encoder

We use a self-attention encoder to transform a corrupted input utterance $\boldsymbol{X}' = (\boldsymbol{x}'_1, \boldsymbol{x}'_2, ..., \boldsymbol{x}'_t)$ to a sequence of hidden states $\boldsymbol{Z} = (\boldsymbol{z}_1, \boldsymbol{z}_2, ..., \boldsymbol{z}_t)$, where $\boldsymbol{X}' \in \mathbb{R}^{t \times d_{fft}}$ and $\boldsymbol{Z} \in \mathbb{R}^{t \times d_{model}}$ for the proposed architectures (see Fig. 3). d_{fft} refers to the number of features after a fast Fourier transform (FFT). Denoting $\boldsymbol{X}'$ or the output of the previous encoder layer as $\boldsymbol{S}$, then each encoder layer can be described as:

$$\boldsymbol{S}' = \mathrm{LN}(\boldsymbol{S} + \mathrm{MHA}(Q = \boldsymbol{S}, K = \boldsymbol{S}, V = \boldsymbol{S})) \quad (1)$$

$$\boldsymbol{S}'' = \mathrm{LN}(\boldsymbol{S}' + \mathrm{FFN}(\boldsymbol{S}')). \quad (2)$$

where MHA, LN, FFN denote multi-headed attention [33], layer normalisation [34] and the feed-forward network in [33], respectively. To achieve real-time streaming, we apply a relative positional encoding (RPE) [35, 36] and a mask for the MHA component such that at each time step, each encoder layer only looks back k frames.

4.2. Cross-attention Decoder

As discussed in the previous sections, a single static speaker embedding may fail to capture all of the speaker's characteristics. Rather, the full sequence of hidden states of the enrolment utterance better reflects the variability of the speaker's voice features [13, 14]. Therefore, instead of using a single static embedding vector, we propose to utilise the full enrolment audio hidden states. For the scenario where only one enrolment clip of the target user is available, we apply a pre-trained d-vector model [18] to $\boldsymbol{A}$ to generate a sequence of hidden states $\boldsymbol{H}$ (e.g., in Fig. 1 $\boldsymbol{H} = [\boldsymbol{h}_0, \boldsymbol{h}_1, \boldsymbol{h}_2]$). In this case, $\boldsymbol{H}$ reflects the variability of the speaker's voice over time.

We further consider the case where q different enrolment utterances $\boldsymbol{A}_{1:q}$ of the same target user are available. In this setting, we apply the pre-trained d-vector model to each enrolment utterance to get $\boldsymbol{H}_{1:q}$. Since processing q full sequences $\boldsymbol{H}_{1:q}$ could be too time-consuming for the on-device use case, we propose to use $\boldsymbol{h}_{1:q,p}$ as the hidden state sequence, where p denotes the last time step of the enrolment clip. $\boldsymbol{h}_{1:q,p}$ presents the variability of the speaker's voice over different utterances.

By a slight abuse of notation, we denote both $\boldsymbol{H}$ and $\boldsymbol{h}_{1:q,p}$ as $\boldsymbol{H}$. To utilise $\boldsymbol{H}$, motivated by attention-based neural machine translation models [33, 37], we propose to apply cross-attention between the corrupted input frames and the enrolment hidden states. As shown in Fig. 3, to execute the cross-attention, we view the hidden representation of the current corrupted input as the Query (q) and $\boldsymbol{H}$ as the Key and Value (k and v). Therefore, based on the current input frame, the cross-attention learns to do either a "soft selection" of suitable enrolment hidden states along the time axis of the single enrolment audio, or a "soft selection" of the most suitable enrolment audio from multiple enrolment utterances. Thus, compared to a static vector, the speaker representation produced by the proposed cross-attention is more flexible and accurate.

We denote as $\boldsymbol{Y}$, both the encoder hidden states $\boldsymbol{Z}$ and the output of the previous decoder layer. To apply the dot-product in cross-attention, we map $\boldsymbol{H}$ into $\boldsymbol{H}'$ through a linear transformation to match the feature dimension of $\boldsymbol{Y}$. Then, each cross-attention layer in the decoder can be described as:

$$\boldsymbol{Y}' = \mathrm{LN}(\boldsymbol{Y} + \mathrm{MHA}(Q = \boldsymbol{Y}, K = \boldsymbol{H}', V = \boldsymbol{H}')) \quad (3)$$

$$\boldsymbol{Y}'' = \mathrm{LN}(\boldsymbol{Y}' + \mathrm{MHA}(Q = \boldsymbol{Y}', K = \boldsymbol{Y}', V = \boldsymbol{Y}')) \quad (4)$$

$$\boldsymbol{Y}''' = \mathrm{LN}(\boldsymbol{Y}'' + \mathrm{FFN}(\boldsymbol{Y}'')). \quad (5)$$

We apply masking and RPE to the MHA component in Equation (4) to enable streaming. However, for the cross-attention MHA in Equation (3), since $\boldsymbol{H}'$ is pre-computed and available locally, we do not need to apply masking to support streaming. We propose two decoder architectures. The first one is "Cross-Seq" as shown in Fig. 3(a). It only has one cross-attention layer. Then, it uses self-attention to extract further hidden representations. The second one is "Cross-Seq+" as shown in Fig. 3(b), where the decoder is fully cross-attention based.

4.3. Baselines

Among the related works in Section 3, which utilise a single static embedding vector through concatenation or attention mechanisms, [4, 8, 12] make a comprehensive set of different model architectures which includes recurrent networks [4], convolutional networks [8] and MHA networks [8, 12]. Thus, we choose [4, 8, 12] as baselines for comparisons. [8, 12] are the most relevant to the proposed work since they use MHA networks in the model backbone, and Fig. 2 shows the architectures of [8, 12].

4.4. Time Complexity

One decoder layer in [8] takes $O(t^2 \cdot d_{model})$ time, while one proposed cross-attention layer takes $O(t \cdot p \cdot d_{model})$ time. Typically, $p \leq t$ and especially in the multiple utterance case, usually $p \ll t$. Thus, for the decoder part, our method has a lower time complexity. For the encoder, one encoder convolutional sub-layer of [8] has a lower time-complexity than one encoder MHA sub-layer of our proposed model. Nevertheless, in Section 5, we will show that with only about half of the parameters, our proposed models can give superior results than [8]. Due to reduced model size and number of layers, the actual run-time of our model is similar to [8]. [12] is a Transformer-based model and has a same level of complexity compared to our model. However, as we will show in Section 5, our model can surpass [12] with only half of the parameters, resulting in lower latency.

5. Experiments

Dataset. Since public datasets such as WSJ0-2mix [15] and LibriMix [16] do not perfectly match our use case, we build our dataset to reflect the on-device close-talk microphone scenario: (1) we add periods of silence; (2) we consider overlapping and non-overlapping noise; (3) the noise can start or end at any point of the input clip; (4) The types of environmental noise are enormous. For ground-truth clean speech, we use LibriSpeech [38], which is already split into train, dev and eval sets. For training, we use all 460 hours available. For each clean audio, we pair it with either an ambient noise (45%) from FreeSoundDataset [39] or a babble noise (45%) from a different speaker in LibriSpeech. The remaining 10% are left as clean samples, with a small amount of white noise added. The target speech and noise are then added to create the corrupted data, with the noise amplitude adjusted to create varying SNR values (sampled uniformly between -3 dB and 10 dB inclusive). Each corrupted audio is then matched with 1 or 5 enrolment utterances, which are random audio clips from the same speaker in the LibriSpeech dataset. For each utterance, a random 3 second chunk (after removing silence) is taken from the longer clip to act as a short enrolment audio. The corrupted clips are then segmented into 3 second chunks, to allow for efficient batching and training. For calculating word error rates (WERs), the non chunked audio is used instead to avoid transcription issues. The corrupted audio is then converted to a spectrogram via the short-time Fourier transform. The spectrograms are made up of overlapping windows, with a step of 10ms and length 25ms. Due to the 16kHz sample rate, the number of samples in each window is 400 and the number of subsequent features is 201.

Experimental Setup. All the models are implemented via Tensorflow 2.6 [40] and Horovod 0.22 [41]. We follow [18] to build a d-vector model and use it to generate the sequence of enrolment hidden states. To investigate the performance of proposed models at different scales, we built base models and large models, where both the encoder and decoder have 3 and 6 layers respectively. All MHA components have 8 heads and each head has dimension 32. The FFN has one hidden layer of dimension 1024. We apply dropout [42] with a probability 0.1 to the output of each sub-component in each layer. For streaming, we restrict the look back step to 100 frames. We train the base/larger models for 100/200 epochs with batch size 320/640 on 4/8 NVIDIA V100 GPUs. Each epoch contains 350 hours of training data. We follow the optimiser and the Noam learning rate scheduler in [33] with 16k warm-up steps. The training objective is to reduce the power-law compressed spectral distance [43] between the cleaned spectrograms produced by the model

Table 1: *SDR (dB, higher is better) of models with offline and online scenarios. Cross-Seq+ 5R indicates the cross-attention is applied among 5 different enrolment clips. For the online Cross-Seq+ models, we train each model 3 to 9 times and report the mean SDR along with the standard deviation.*

Model	Dev SDR	Eval SDR	#Param	Speed
offline				
VoiceFilter	16.71	16.09	7.8M	N/A
SpEX$_{pc}$	17.95	17.86	11.7M	N/A
AvaTr	18.02	17.42	12.3M	N/A
Cross-Seq large	18.23	17.54	10.8M	N/A
Cross-Seq+ base	18.70	18.16	6.1M	N/A
Cross-Seq+ large	18.96	18.41	12.0M	N/A
online				
SpEX$_{pc}$	15.45	15.14	11.7M	1.03x
AvaTr	15.57	15.02	12.3M	0.88x
Cross-Seq large	15.70	15.00	10.8M	0.95x
Cross-Seq+ base	15.71±0.08	15.29±0.06	6.1M	1.00x
Cross-Seq+ large	16.07±0.07	15.62±0.09	12.0M	0.91x
Cross-Seq+ 5R	16.23±0.10	15.84±0.05	12.0M	0.95x

and the ground-truth. For each training process, the final model is selected based on the SDR on the development set.

Baseline Setup. We implement the baseline models VoiceFilter [4], SpEX$_{pc}$ [8], and AvaTr [12]. For VoiceFilter and SpEX$_{pc}$ we use the original model architectures in [4, 8]. Since AvaTr is a Transformer-based model, for a fair comparison, we implement it using the same hyper-parameters as our proposed Cross-Seq+. In the online streaming setting, we change the TCN blocks in SpEX$_{pc}$ to causal and restrict the look back step to 96 frames (the closest frame count possible to 100 due to the dilation in [8]) for the convolutional and MHA sub-layer. As in our proposed cross-attention models, we set the look back steps to 100 for AvaTr. In [8, 12], the embedding networks of SpEX$_{pc}$ and AvaTr are different to each other. In this work, for a fair comparison, we use the d-vector network as the embedding network for all baselines.

Experimental Results. Although we primarily focus on streaming scenarios, we also provide non-streaming offline PSE models whose performance can be viewed as a soft upper bound. Table 1 shows the SDR results for both of the offline and online scenarios. For the offline models, our proposed Cross-Seq outperforms the VoiceFilter and AvaTr baselines, but it does not surpass SpEX$_{pc}$ on the eval set. Nevertheless, our Cross-Seq+ models outperform all the baselines by a large margin.

For the online streaming scenario, we do not consider VoiceFilter or its streaming variant [5] since the performance gap between VoiceFilter and other offline models are significant. Table 1 gives the experimental results of the streaming models. Our streaming Cross-Seq outperforms baselines on the development set but this does not generalise to the evaluation dataset. However, our streaming Cross-Seq+ models outperform the baselines consistently. We train each streaming Cross-Seq+ model 3 to 9 times and this set of extensive experiments demonstrates that the better performance of Cross-Seq+ models over baselines is statistically significant. The inferior SDR values given by the baseline systems with a single static embedding vector indicate the usefulness of the dynamic speaker representations, as well as the benefits of attending the full sequence of enrolment utterance hidden states. Further, Cross-Seq+ 5R, which attends 5 different enrolment clips, gives the best results among the streaming models. This indicates the variability of the speaker is best captured across different recordings.

Table 2: *WER (lower is better) of models with online streaming settings. The WER is calculated using an open-source STT engine [44]. Cross-Seq+ 5R indicates the cross-attention is applied among 5 different enrolment clips. For Cross-Seq+, we report the mean WER with one standard deviation of 3 to 9 independently trained models.*

Model (online)	Ambient WER	Babble WER	#Param
Ground Truth	8.34	8.03	N/A
No PSE	18.32	72.32	N/A
SpEX$_{pc}$	17.18	32.70	11.7M
AvaTr	16.20	25.71	12.3M
Cross-Seq large	16.36	30.09	10.8M
Cross-Seq+ base	15.66±0.15	26.03±0.51	6.1M
Cross-Seq+ large	14.98±0.07	22.24±0.67	12.0M
Cross-Seq+ 5R	15.51±0.08	22.65±0.08	12.0M

In addition to the PSE performance, we also measure model size and inference speed, which are critical factors for on-device systems. Impressively, as shown in Table 1, the Cross-Seq+ base model outperforms SpEX$_{pc}$, AvaTr, and Cross-Seq large, even with approximately half of the model size. For the streaming models, we record their speed of processing 5000 corrupted utterances through a single NVIDIA A40 GPU. We view the speed factor of streaming Cross-Seq+ base as 1.00. Table 1 shows the inference speed of streaming Cross-Seq+ base is similar to streaming SpEX$_{pc}$ and faster than all other streaming models. It is worth noting that the encoder of SpEX$_{pc}$ is a convolutional network, which is typically less computational expensive than a MHA network with a similar amount of parameters. However, due to the expressiveness of our proposed cross-attention, compared to streaming SpEX$_{pc}$, streaming Cross-Seq+ base gives higher SDRs with halved model size, and this results in a similar decoding speed between Cross-Seq+ base and SpEX$_{pc}$.

We further focus on the downstream ASR task. Table 2 shows the WERs. First, we apply the ASR system on the clean utterances to have a performance upper bound. To have a lower bound, we apply the ASR engine to the corrupted clips without any PSE module. Then, we apply an inverse fast Fourier transform (IFFT) to the output of the PSE models and feed the output of the IFFT to the ASR engine. In terms of WERs, the SpEX$_{pc}$ baseline gives the highest WERs among all models, although its PSE performance measured by SDR is not the worst. With halved model size, our proposed Cross-Seq+ base has significantly lower ambient WERs compared to all the baselines and similar babble WERs compared to AvarTr. When the model sizes are similar, the Cross-Seq+ large models outperform all models by a large margin.

6. Conclusion

We proposed fully Transformer-based streaming PSE models, which utilise on a novel cross-attention approach to generate dynamic target speaker representations. Compared to existing PSE systems, which uses a single static speaker embedding vector, our proposed approach better captures the variability of the speech of the target speaker. The improved expressiveness of the model leads to better performance in both PSE and downstream ASR tasks as well as reduced model size and latency when compared to baselines. Our proposed cross-attention can also be integrated into PSE models with backbone architectures other than Transformers, which we left as a future work.

7. References

[1] Y. Luo and N. Mesgarani, "Conv-tasnet: Surpassing ideal time–frequency magnitude masking for speech separation," *TASLP*, 2019.

[2] J. Chen, Q. Mao, and D. Liu, "Dual-path transformer network: Direct context-aware modeling for end-to-end monaural speech separation," in *INTERSPEECH*, 2020.

[3] C. Subakan, M. Ravanelli, S. Cornell, M. Bronzi, and J. Zhong, "Attention is all you need in speech separation," in *ICASSP*, 2021.

[4] Q. Wang, H. Muckenhirn, K. Wilson, P. Sridhar, Z. Wu, J. Hershey, R. A. Saurous, R. J. Weiss, Y. Jia, and I. L. Moreno, "Voicefilter: Targeted voice separation by speaker-conditioned spectrogram masking," in *INTERSPEECH*, 2019.

[5] Q. Wang, I. L. Moreno, M. Saglam, K. Wilson, A. Chiao, R. Liu, Y. He, W. Li, J. Pelecanos, M. Nika *et al.*, "Voicefilter-lite: Streaming targeted voice separation for on-device speech recognition," in *INTERSPEECH*, 2020.

[6] 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," *JSTSP*, 2019.

[7] M. Ge, C. Xu, L. Wang, E. S. Chng, J. Dang, and H. Li, "Spex+: A complete time domain speaker extraction network," in *INTERSPEECH*, 2020.

[8] W. Wang, C. Xu, M. Ge, and H. Li, "Neural speaker extraction with speaker-speech cross-attention network." in *INTERSPEECH*, 2021.

[9] P. Shen, S. He, and X. Zhang, "Exarn: self-attending rnn for target speaker extraction," *arXiv preprint arXiv:2212.01106*, 2022.

[10] T. Ochiai, M. Delcroix, K. Kinoshita, A. Ogawa, and T. Nakatani, "A unified framework for neural speech separation and extraction," in *ICASSP*, 2019.

[11] J. Han, W. Rao, Y. Long, and J. Liang, "Attention-based scaling adaptation for target speech extraction," in *ASRU*, 2021.

[12] S. X. Hu, M. R. Arefin, V.-N. Nguyen, A. Dipani, X. Pitkow, and A. S. Tolias, "Avatr: One-shot speaker extraction with transformers," in *INTERSPEECH*, 2021.

[13] H. Zhan, H. Zhang, W. Ou, and Y. Lin, "Improve cross-lingual text-to-speech synthesis on monolingual corpora with pitch contour information." in *INTERSPEECH*, 2021.

[14] F. Lux, J. Koch, and N. T. Vu, "Exact prosody cloning in zero-shot multispeaker text-to-speech," in *IEEE SLT*, 2023.

[15] J. R. Hershey, Z. Chen, J. Le Roux, and S. Watanabe, "Deep clustering: Discriminative embeddings for segmentation and separation," in *ICASSP*, 2016.

[16] J. Cosentino, M. Pariente, S. Cornell, A. Deleforge, and E. Vincent, "Librimix: An open-source dataset for generalizable speech separation," 2020.

[17] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *TASLP*, 2010.

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

[19] S. V. Vaseghi, *Advanced Digital Signal Processing and Noise Reduction*. New York: John Wiley & Sons, 2008.

[20] C. Xu, W. Rao, E. S. Chng, and H. Li, "Time-domain speaker extraction network," in *ASRU*, 2019.

[21] T. Li, Q. Lin, Y. Bao, and M. Li, "Atss-net: Target speaker separation via attention-based neural network," in *INTERSPEECH*, 2020.

[22] X. Ji, M. Yu, C. Zhang, D. Su, T. Yu, X. Liu, and D. Yu, "Speaker-aware target speaker enhancement by jointly learning with speaker embedding extraction," in *ICASSP*, 2020.

[23] S. E. Eskimez, T. Yoshioka, H. Wang, X. Wang, Z. Chen, and X. Huang, "Personalized speech enhancement: New models and comprehensive evaluation," in *ICASSP*, 2022.

[24] R. Giri, S. Venkataramani, J.-M. Valin, U. Isik, and A. Krishnaswamy, "Personalized percepnet: Real-time, low-complexity target voice separation and enhancement," in *INTERSPEECH*, 2021.

[25] 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 *INTERSPEECH*, 2022.

[26] T. O'Malley, A. Narayanan, Q. Wang, A. Park, J. Walker, and N. Howard, "A conformer-based asr frontend for joint acoustic echo cancellation, speech enhancement and speech separation," in *ASRU*, 2021.

[27] B. Gfeller, D. Roblek, and M. Tagliasacchi, "One-shot conditional audio filtering of arbitrary sounds," in *ICASSP*, 2021.

[28] E. Perez, F. Strub, H. de Vries, V. Dumoulin, and A. Courville, "Film: Visual reasoning with a general conditioning layer," in *AAAI*, 2018.

[29] A. G. C. P. Ramos, A. Mehrotra, N. D. Lane, and S. Bhattacharya, "Conditioning sequence-to-sequence networks with learned activations," in *ICLR*, 2022.

[30] X. Xiao, Z. Chen, T. Yoshioka, H. Erdogan, C. Liu, D. Dimitriadis, J. Droppo, and Y. Gong, "Single-channel speech extraction using speaker inventory and attention network," in *ICASSP*, 2019.

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

[32] R. Rikhye, Q. Wang, Q. Liang, Y. He, and I. McGraw, "Closing the gap between single-user and multi-user voicefilter-lite," *arXiv preprint arXiv:2202.12169*, 2022.

[33] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *NeurIPS*, 2017.

[34] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[35] P. Shaw, J. Uszkoreit, and A. Vaswani, "Self-attention with relative position representations," *CoRR*, 2018.

[36] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. V. Le, and R. Salakhutdinov, "Transformer-xl: Attentive language models beyond a fixed-length context," in *ACL*, 2019.

[37] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *ICLR*, 2015.

[38] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *ICASSP*, 2015.

[39] E. Fonseca, M. Plakal, F. Font, D. P. W. Ellis, X. Favory, J. Pons, and X. Serra, "General-purpose tagging of freesound audio with audioset labels: Task description, dataset, and baseline. in proceedings of dcase 2018 workshop," 2018. [Online]. Available: https://arxiv.org/abs/1807.09902

[40] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.

[41] A. Sergeev and M. Del Balso, "Horovod: fast and easy distributed deep learning in tensorflow," *arXiv preprint arXiv:1802.05799*, 2018.

[42] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *JMLR*, 2014.

[43] S. Braun and I. Tashev, "A consolidated view of loss functions for supervised deep learning-based speech enhancement," in *ICTSP*, 2021.

[44] S. Team, "Silero models: pre-trained enterprise-grade stt / tts models and benchmarks," https://github.com/snakers4/silero-models, 2021.