ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 978-1-7281-6327-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICASSP49357.2023.10095369

CONDITIONAL CONFORMER: IMPROVING SPEAKER MODULATION FOR SINGLE AND MULTI-USER SPEECH ENHANCEMENT

Tom O'Malley, Shaojin Ding, Arun Narayanan, Quan Wang
*Rajeev Rikhye, Qiao Liang, Yanzhang He, Ian McGraw**

Google LLC, U.S.A

ABSTRACT

Recently, Feature-wise Linear Modulation (FiLM) has been shown to outperform other approaches to incorporate speaker embedding into speech separation and VoiceFilter models. We propose an improved method of incorporating such embeddings into a VoiceFilter frontend for automatic speech recognition (ASR) and text-independent speaker verification (TI-SV). We extend the widely-used Conformer architecture to construct a FiLM Block with additional feature processing before and after the FiLM layers. Apart from its application to single-user VoiceFilter, we show that our system can be easily extended to multi-user VoiceFilter models via element-wise max pooling of the speaker embeddings in a projected space. The final architecture, which we call *Conditional Conformer*, tightly integrates the speaker embeddings into a Conformer backbone. We improve TI-SV equal error rates by as much as 56% over prior multi-user VoiceFilter models, and our element-wise max pooling reduces relative WER compared to an attention mechanism by as much as 10%.

***Index Terms*—** Noise robust ASR, Speaker embedding, VoiceFilter

1. INTRODUCTION

Better neural networks [1–4], larger training data [5–7], and improved data augmentation strategies [8–10] have in the recent years significantly increased the robustness of automatic speech recognition (ASR) systems. In particular, Conformer-based architectures [11] have been shown to perform very well in speech applications. However, there are a number of factors that still significantly deteriorate performance [12, 13]. Competing speech, which is the focus of the current work, is one such factor that poses a unique set of challenges to an ASR system, and has therefore attracted a lot of research focus over the last few years [14–16].

Competing speech can come from one or more non-target speakers. In multitalker scenarios, it is assumed that target speech co-occurs with competing speech. Multitalker condition poses a special challenge for traditional ASR systems. Without additional context, it is generally ambiguous as to which speaker the model should attend. Therefore, typical ASR models are trained on utterances containing only a single speaker. Multi-speaker ASR models [15] are sometimes applicable, but they must decide how many speakers to support *a priori*. Furthermore, it is usually a challenge to retain the best performance in single talker scenarios when training multitalker models. This is especially true with the recent focus on building large scale multidomain [7] and multilingual ASR models [17]. Nonetheless, improving performance in competing speech remains an important challenge for ASR systems.

One way to solve this problem is to make use of a speaker embedding vector [18]. This vector is computed from a number of enrollment utterances provided by the user, and serves to identify the target speaker. While these speaker embedding vectors are usually trained for speaker verification tasks, they have shown to be beneficial when used for speech enhancement for ASR [19–21]. Such models are typically referred to as VoiceFilter models [19]. The speaker embedding vector is usually combined with the acoustic features either by concatenation [19] or through feature-wise linear modulation (FiLM) [22], with FiLM generally outperforming concatenation [21, 23]. FiLM is often performed multiple times, for example at the beginning of each processing block like a Conformer or an LSTM [21, 24].

Recent work has shown that multiple enrolled users can also be supported by a single model [24, 25]. In this case, multiple speaker embedding vectors are computed, and the model is trained to retain speech from any of the enrolled speakers. Rikhye *et al.* combine acoustic features with the speaker embeddings via an attention-based mechanism to select the relevant speaker embedding vector for each utterance [24]. The output of this attention-mechanism is then used as the modulator in FiLM, in the same way that a single speaker embedding is used in the single-user case.

In this work, we propose a simple speaker modulation processing block that improves upon previous approaches for both the single-user and multi-user case. We tightly integrate our improvements into a Conformer backbone to create the *Conditional Conformer* architecture. Our design is motivated by the structure of the other components of a Conformer, and aims to improve how the speaker embedding is incorporated. The design is applicable to any Conformer application that accepts a contextual vector as a side input. Our model significantly improves speech recognition and speaker verification performance in overlapped speech conditions compared to prior works. It can handle an arbitrary number of speaker embeddings from enrolled users. Even though the focus of the current work is speech enhancement for ASR and speaker verification, the same mechanism can be applied to personalized VAD, personalized ASR, or any Conformer-based application that accepts one or more conditioning vectors as a side input.

The rest of the paper is organized as follows. Sec. 2 describes the overall architecture and modeling improvements. Experimental design is presented in Sec. 3, and Sec. 4 contains detailed results and analysis. We summarize the work and discuss future work in Sec. 5.

2. SYSTEM

Previous VoiceFilter models have predominantly used LSTMs as the neural architecture [19, 24]. Unlike these works, our work uses a

*We thank A. Gruenstein, A. Park, J. Walker, N. Howard, and S. Panchapagesan for several useful discussions.

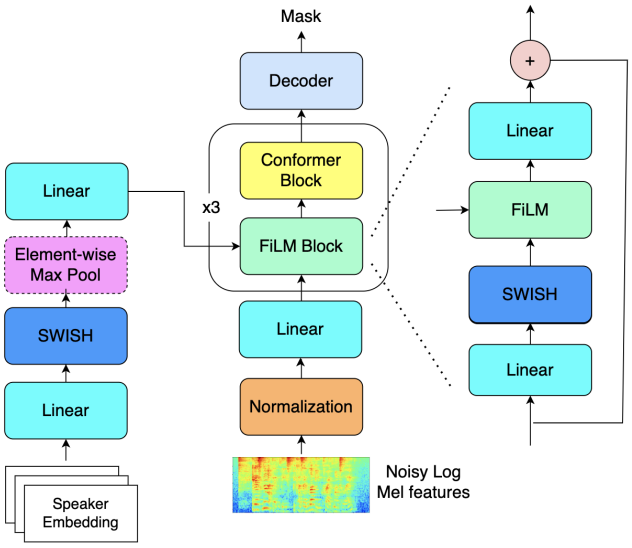

Fig. 1: *Conditional Conformer. Speaker embeddings are first processed and their features are pooled. The resulting vector is then used as the modulation vector for the FiLM Block. Note that in the single-user case, the element-wise max pooling operation reduces to a no-op.*

Conformer-based backbone as in [21]. We use conformers since they have been shown to be well-suited for speech and audio tasks in prior works, outperforming LSTMs [11,26]. Fig. 1 shows a block diagram of the overall system. In addition to Conformer blocks, we introduce 2 new processing blocks in Conditional Conformer. The first block preprocesses the speaker embeddings. This preprocessing is done in such a way that makes it trivial to extend it to handle multiple speakers' embedding vectors. The preprocessing combines one or more enrolled speaker embeddings into a single conditioning vector that can be used for subsequent processing. Next, we add a *FiLM Block* to the start of each conformer layer. This block improves upon the simpler FiLM layer used in [21], and closely mirrors the other components of a Conformer architecture. These blocks are discussed in detail in the subsequent subsections.

2.1. Preprocessing of Speaker Embedding(s)

We propose a simple mechanism for preprocessing speaker embedding vectors, that generalizes to both the single-user and multi-user use case. Previous systems have relied on attention-based mechanisms to select the appropriate speaker embedding vector for the target utterance in the case of multiple enrolled users [19]. However, these models can be difficult to train [24] and performance in multi-user conditions significantly lags behind single-user conditions at low signal-to-noise ratios (SNRs).

Preprocessing consists of an initial projection, followed by a Swish activation and an element-wise max-pooling operation across speakers, and a final projection:

$$\begin{aligned} x' &= Linear(x), \\ x'' &= SWISH(x'), \\ x''' &= MaxPool(x''), \\ x'''' &= Linear(x''') \end{aligned} \tag{1}$$

During the first projection, we project each speaker embedding vector to a higher dimension. For our models, the speaker embedding vectors are 256-dimensional, which we project to a 512-dimensional representation. We use a Swish activation function [27] after this projection and then perform an element-wise max pooling across speakers in the projected space. Finally, we project the resulting vector back to 256 dimensions. It is worth noting that in the single-user case, the max pooling operation becomes a no-op, and so this mechanism is applicable for both the single and multi-user models.

Typically, speaker embedding vectors are optimized on speaker verification tasks, and reused for applications like VoiceFilter. This makes them suboptimal, since the goals of speaker verification and speech separation are not always aligned; the goal of speaker verification is to verify a speaker given a segment of audio, but the goal of enhancement is to extract all audio components of a target speaker from a multitalker mixture. The preprocessing that we use tries to partly address this mismatch. It allows the model to focus on the most distinctive characteristics of each speaker, and retain any speech that matches those distinctive characteristics. It is also optimized for enhancement, thereby allowing it to focus on the task at hand. The initial projection to a higher-dimensional space allows max pooling to happen in a richer representation that is trained specifically for speech enhancement, instead of the initial speaker embedding representation that is trained for speaker verification. The max pooling aggregates salient features from all target speakers. This is similar to the methodology VoxelNet uses to aggregate across a voxel grid [28]. The final projection compresses the resultant vector to help keep the model size small, and transforms the vector into a representation that can be more readily used by the subsequent FiLM blocks.

After this processing, the resultant vector becomes the conditioning vector used in the subsequent FiLM Blocks. This processing is only performed once, and the same vector is used as input to each FiLM Block.

2.2. FiLM Block

In our previous work, we used FiLM with a residual connection to incorporate the speaker embedding. Motivated by the design of conformer [11] and Macron-Net [29], we create a dedicated residual FiLM Block at the beginning of each conformer layer. Similar to how convolution and self-attention blocks are sandwiched between linear layers in a conformer, the FiLM Block adds linear transforms before and after the FiLM layer. Specifically, we first apply a projection and Swish activation to the input features. This allows the modulation to occur in a different space from the main Conformer encoding, one more suited to feature-wise modulation. We then apply FiLM using the conditioning vector from the preprocessing block, and project the resulting features back to the main encoding space before adding a residual connection. The FiLM block thus tightly integrates the FiLM layer into the overall Conformer architecture. We present ablations in Sec. 4 that show how various components in the FiLM Block contribute to the final performance.

2.3. Handling Missing Speaker Embeddings

With this system, a model can be trained to handle up to N enrolled users. However during runtime, there may be fewer than N enrolled users on a device. Our system must be robust to the case where fewer than the maximum number of supported users are enrolled on the device. To handle this, during training we simulate a number of enrolled users between 1 and N. When the number of enrolled

Table 1: *WER results for single-user VoiceFilter ablation study.*

Model	Clean	Speech		
		-5 dB	0 dB	5 dB
Baseline	7.1	69.2	46.4	29.3
Proposed Model	7.3	**34.7**	**25.1**	**18.8**
- FiLM changes	7.3	35.3	25.3	**18.8**
- Embedding processing	7.3	35.7	25.6	**18.8**
- Both	7.3	35.7	25.7	18.9

users is less than N, we use Signal Dropout [30] to fill-in the missing speaker embedding vector with all-zeros.

2.4. Additional details

Our models are trained in the log Mel domain, to match the feature representation used by the ASR and speaker verification models. The model takes noisy log Mel features as input. The input features are normalized and projected to a dimension that matches the number of units in each Conformer layer. The projected features are passed to our Conditional Conformer encoder module, which consists of stacked layers of alternating FiLM Blocks and Conformer layers. The FiLM blocks also receive the preprocessed conditional vector containing information about all the target speaker. The output of the final encoder is then passed to a mask decoder, which is a simple linear layer with a sigmoid activation that estimates a time-frequency ideal ratio mask (IRM) [31]. The decoder additionally predicts mask scalars to limit speech distortion to improve ASR [32]. The scaled estimated IRM is applied to the noisy log Mel features to produce enhanced features that are used by pretrained ASR and speaker verification models.

For the loss functions, our models use a combination of $\mathcal{L}_1$ and $\mathcal{L}_2$ distance between estimated and ideal masks, and an ASR loss that minimizes the $\mathcal{L}_2$ distance between the ouptut activations of a pretrained ASR encoder when the clean and enhanced log Mel features are used as input. This follows the same formulation in [21].

3. EXPERIMENTS

3.1. Datasets

We train on datasets derived from Librispeech [33], LibriVox [34], and internal vendor-collected utterances. Together, the training set has ∼50.8k hours of audio. We treat these data sources as 'clean' and add either background noise or competing speech to each utterance. We evaluate on simulated noisy test sets, created using the clean subset of these datasets.

Background noise: The datasets are created using a room simulator [8]. The room simulator adds reverberation with T60 between 0 msec and 900 msec, and noise with an SNR in the range [-10 dB, 30 dB]. The noise snippets represent common noise sources such as cars, kitchen, etc. as well as publicly available noise sources from Getty Audio[1] and YouTube Audio Library[2]. Test sets are also created in a similar fashion, but the mixing conditions used are disjoint from the training conditions.

Competing Speech: In order to simulate multi-speaker conditions, we mix the the training utterances with random competing speech chosen from the training datasets, using similar reverberation and SNR conditions as the background noise datasets. Our tests sets are constructed in a similar fashion.

[1] https://www.gettyimages.com/about-music
[2] https://youtube.com/audiolibrary

3.2. Training details

All models use a Conformer backbone with a hidden dimension of 196. We use 3 Conformer layers in each model. 128-dimensional log Mel features are used for all audio signals, computed for 32 msec windows with 10 msec hop. 4 frames of features are stacked together with a stride of 3 before being passed to the model. The speaker embedding vectors are 256-dimensional and precomputed using a text-independent speaker recognition model. Causal convolution and local attention with 31 frames of left context are used in the Conformer layers to allow the model to be streaming.

The ASR model is a recurrent neural net transducer trained with ∼400k hours of English speech covering domains like Search, YouTube and Telephony [4]. The utterances from Search and Telephony domains are anonymized and hand-transcribed; YouTube utterances are transcribed in a semi-supervised fashion. The models also make use of data augmentation strategies such as SpecAug and multistyle-training. The speaker verification model used during evaluation is the same model used to generate the speaker embedding vectors. All models are trained in TensorFlow [35], using the Lingvo toolkit [36].

3.3. Models

We perform an ablation study on the modeling improvements described above for the single-user case. We train 4 models: a baseline model with no speaker embedding preprocessing and with simple FiLM modulation, a model with the speaker embedding preprocessing presented in this paper, a model with our proposed FiLM block, and a model that contains both changes. We evaluate the WER when these models are used as an enhancement frontend to a pre-trained ASR system.

For the multi-user case, we train models designed to handle a maximum of either 2 or 4 enrolled users. For each number of enrolled users, we evaluate two models: a model that preprocesses the speaker embedding vectors using an attention-based mechanism, and a model that uses our proposed element-wise max module. For the attention mechanism, we use a design similar to [25], except we replace the ScorerNet architecture used in that paper with a single-layer Conformer. We evaluate the performance of these models using two metrics: the WER when the resulting features are passed to a pre-trained ASR model, and the equal error rate (EER) when the resulting features are passed to a pre-trained text-independent speaker verification (TI-SV) model. All models have ∼3.5M parameters.

To demonstrate that our approach is robust to the number of users enrolled on the device, we also report TI-SV model EER for cases in which the number of actual enrolled users is less than the maximum number that the model is designed to handle.

4. RESULTS

4.1. Single-user VoiceFilter

4.1.1. ASR

Table 1 shows the WER results of our single-user ablation study when the enhanced features are passed to the downstream ASR model. The results are on simulated noisy test sets created using the Librispeech test-clean subset. As can be seen, the proposed model significantly improves performance in all noisy conditions, without significantly deteriorating performance in clean conditions. Replacing the FiLM Block with the simpler FiLM layer and removing speaker preprocessing, each results in larger WERs, especially at -5 dB and 0 dB. Having neither performs the worst. Overall, the proposed model consistently outperforms the alternatives.

Table 2: *ASR WER results for 2 and 4 enrolled user models*

Supported Users	Processing Type	Clean	Noise			Speech		
			-5 dB	0 dB	5 dB	-5 dB	0 dB	5 dB
-	Baseline	16.1	36.8	26.3	20.8	117.3	94.1	68.5
2	Attention	16.6	34.4	25.4	20.6	60.8	50.5	37.0
2	Max	**16.0**	**33.7**	**25.1**	**20.2**	**56.1**	**45.3**	**34.1**
4	Attention	16.4	34.6	25.7	20.4	69.1	56.0	41.0
4	Max	**16.1**	**34.1**	**25.3**	**20.3**	**61.6**	**49.4**	**36.9**

Table 3: *EER results for 1, 2, and 4 supported-user models with varying number of enrolled users*

Supported Users	Enrolled Users	Processing Type	Clean	Additive Speech			Reverberant Speech		
				-5 dB	0 dB	5 dB	-5 dB	0 dB	5 dB
-	-	Baseline	0.71	12.3	8.2	5.2	17.1	10.9	6.5
1	1	Max	0.77	1.7	1.4	1.3	2.8	2.1	1.8
2	2	Rikhye *et al.* [25]	**0.74**	8.9	5.2	3.1	12.0	7.1	4.2
2	2	Attention	1.3	5.0	3.9	3.2	7.6	5.7	4.4
2	2	Max	0.85	**2.9**	**2.3**	**1.8**	**5.2**	**3.6**	**2.7**
2	1	Rikhye *et al.* [25]	**0.71**	3.9	2.3	1.7	6.4	3.5	2.2
2	1	Attention	**0.71**	**2.0**	**1.5**	**1.3**	**3.2**	2.9	**1.8**
2	1	Max	0.79	**2.0**	1.6	**1.3**	**3.2**	**2.4**	1.9
4	4	Attention	1.6	8.0	6.3	5.1	11.6	8.8	7.0
4	4	Max	**0.9**	**5.1**	**3.7**	**2.8**	**8.6**	**5.9**	**4.3**
4	2	Attention	1.2	5.0	3.9	3.1	7.7	5.8	4.4
4	2	Max	**0.9**	**3.4**	**2.6**	**2.0**	**5.9**	**4.1**	**3.0**
4	1	Attention	**0.71**	2.2	**1.6**	**1.3**	3.6	**2.5**	**1.9**
4	1	Max	0.74	**2.1**	**1.6**	1.4	**3.5**	2.6	2.0

4.2. Multi-user VoiceFilter

4.2.1. ASR

For evaluating ASR performance using multi-user VoiceFilter, we construct noisy sets using vendor collected test data. This data has ∼15k utterances. Apart from the clean condition, we construct test sets using speech and non-speech background at -5 dB, 0 dB and 5 dB.

For both the 2-speaker and 4-speaker models, our proposed element-wise max pooling significantly outperforms the attention mechanism used in prior work [25]. Most notably, the 2-speaker model reduces the relative WER by 7.7% at -5 dB competing speech, and the 4-speaker model reduces the relative WER by 10.8% in -5 dB competing speech.

4.2.2. TI-SV

For TI-SV evaluation, we reuse the test sets from [24, 25] to make the results comparable to those presented in prior work. Test sets are constructed using vendor collected data, either by adding speech noise in reveberant ('Reverberant' in Table 3) and non-reverberant ('Additive' in Table 3)) conditions at -5 dB, 0 dB and 5 dB.

Table 3 shows the EER results when the enhanced features are passed to a downstream TI-SV model. Compared to Rikhye *et al.* [25], we show up to a 56% relative reduction in EER for the 2-speaker models when 2 users are enrolled. For these same conditions, holding all else equal, the element-wise max pooling processing shows an improvement of up to 31% over the use of an attention-based mechanism. With only 1 enrolled speaker, the 3 approaches work similarly, with the proposed model working slightly better in reverberant speech noise conditions at 0 dB. For the 4-speaker model with 4 enrolled users, element-wise max pooling consistently outperforms the attention mechanism. For example, at -5 dB in reveberant conditions, EER improves by 26%. The element-wise max and attention mechanisms perform similarly when only a single user is enrolled, which is to be expected as these mechanisms are designed to select salient features from multiple users. Interestingly, when there are multiple enrolled users, the element-wise max performs better in clean conditions, but when there is only a single user, the attention mechanism tends to perform better. This is likely because the attention mechanism does not have to choose a speaker, when there is only a single enrolled user. Rikhye *et al.* also performs slightly better in clean conditions. There is also a small degradation for the 4-speaker model in clean conditions, when there are 2 or 4 enrolled speakers. We intend to address these regressions in future work.

5. CONCLUSION

In this work, we presented Conditional Conformer, which is a VoiceFilter frontend that can seamlessly handle single and multiple enrolled users. It does that using a preprocessing block for speaker embeddings that uses element-wise max pooling, and an improved FiLM block for incorporating the preprocessed embedding. Our system reduces EER for text-independent speaker verification in noisy multitalker conditions by as much as 56% compared to existing work. It also shows improvements in WER compared to using the typical FiLM layers for incorporating speaker embeddings. Conditional Conformer provides a useful architecture for tightly integrating contextual information into a Conformer backbone. Future work will focus on incorporating additional contextual information into this framework, and improving TI-SV performance in clean conditions with multiple enrolled speakers.

6. REFERENCES

[1] R. Prabhavalkar, K. Rao, T. N. Sainath, and B. Li et al., "A Comparison of Sequence-to-Sequence Models for Speech Recognition," in *Proc. Interspeech*, 2017.

[2] E. Battenberg, J. Chen, R. Child, A. Coates, and Y. G. Y. Li et al., "Exploring Neural Transducers for End-to-end Speech Recognition," in *Proc. of ASRU*, 2017.

[3] T. Hori, S. Watanabe, Y. Zhang, and W. Chan, "Advances in Joint CTC-Attention Based End-to-End Speech Recognition with a Deep CNN Encoder and RNN-LM," in *Proc. Interspeech*, 2017.

[4] B. Li, A. Gulati, J. Yu, T. Sainath, C.-C. Chiu, A. Narayanan, S.-Y. Chang, et al., "A better and faster end-to-end model for streaming asr," in *Proc. ICASSP*. IEEE, 2021.

[5] S. Mirsamadi and J. Hansen, "On multi-domain training and adaptation of end-to-end rnn acoustic models for distant speech recognition," in *Proc. Interspeech*, 2017.

[6] D. Hakkani-Tür, G. Tür, A. Celikyilmaz, and Y.-N. Chen et al., "Multi-domain joint semantic frame parsing using bi-directional rnn-lstm.," in *Proc. Interspeech*, 2016.

[7] A. Narayanan, A. Misra, K. C. Sim, G. Pundak, and A. Tripathi et al., "Toward domain-invariant speech recognition via large scale training," in *Proc. of SLT*, 2018.

[8] C. Kim, A. Misra, K. Chin, T. Hughes, and A. Narayanan et al, "Generation of large-scale simulated utterances in virtual rooms to train deep-neural networks for far-field speech recognition in Google Home," in *Proc. Interspeech*, 2017.

[9] D. Park, W. Chan, Y. Zhang, C.-C. Chiu, and B. Zoph et al., "Specaugment: A simple data augmentation method for automatic speech recognition," in *Proc. Interspeech*, 2019.

[10] I. Medennikov, Y. Khokhlov, A. Romanenko, and D. Popov et al., "An investigation of mixup training strategies for acoustic models in asr.," in *Proc. Interspeech*, 2018.

[11] A. Gulati, J. Qin, and C.-C. Chiu et al, "Conformer: Convolution-augmented transformer for speech recognition," *arXiv preprint arXiv:2005.08100*, 2020.

[12] J. Barker, R. Marxer, E. Vincent, and S. Watanabe, "The third CHiME speech separation and recognition challenge: Analysis and outcomes," *Computer Speech & Language*, vol. 46, pp. 605–626, 2017.

[13] J. Barker, S. Watanabe, E. Vincent, and J. Trmal, "The fifth 'CHiME' speech separation and recognition challenge: Dataset, task and baselines," in *Proc. Interspeech*, 2018.

[14] X. Chang, Y. Qian, and K. et al Yu, "End-to-end monaural multi-speaker asr system without pretraining," in *Proc. IEEE ICASSP*, 2019.

[15] A. Tripathi, H. Lu, and H. Sak, "End-to-end multi-talker overlapping speech recognition," in *Proc. ICASSP*. IEEE, 2020.

[16] Ilya Sklyar, Anna Piunova, and Yulan Liu, "Streaming multi-speaker asr with rnn-t," in *Proc. IEEE ICASSP*, 2021.

[17] O. Adams, M. Wiesner, S. Watanabe, and D. Yarowsky, "Massively multilingual adversarial speech recognition," in *Proc. of NAACL*, 2019.

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

[19] Kevin Wilson et al. Quan Wang, Hannah Muckenhirn, "VoiceFilter: Targeted voice separation by speaker-conditioned spectrogram masking," in *Proc. Interspeech*, 2019, pp. 2728–2732.

[20] Q. Wang, I.L. Moreno, and M. Saglam et al., "VoiceFilter-Lite: Streaming targeted voice separation for on-device speech recognition," in *Proc. Interspeech*, 2020, pp. 2677–2681.

[21] 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 *IEEE ASRU*, 2021.

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

[23] Q. Liang et al S. Ding, R. Rikhye, "Personal VAD 2.0: Optimizing personal voice activity detection for on-device speech recognition," in *Proc. Interspeech*, 2022.

[24] Qiao Liang et al. Rajeev Rikhye, Quan Wang, "Multi-user VoiceFilter-Lite via attentive speaker embedding," in *Proc. IEEE ASRU*, 2021.

[25] Rajeev Rikhye, Quan Wang, Qiao Liang, Yanzhang He, and Ian McGraw, "Closing the gap between single-user and multi-user VoiceFilter-Lite," in *Odyssey: The Speaker and Language Recognition Workshop*, 2022.

[26] 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. IEEE ICASSP*, 2021.

[27] Prajit Ramachandran, Barret Zoph, and Quoc V Le, "Searching for activation functions," *arXiv preprint arXiv:1710.05941*, 2017.

[28] Yin Zhou and Oncel Tuzel, "Voxelnet: End-to-end learning for point cloud based 3d object detection," in *Proc. IEEE CVPR*, 2018.

[29] Y. Lu, Z. Li, and D. et al. He, "Understanding and improving transformer from a multi-particle dynamic system point of view," *arXiv preprint arXiv:1906.02762*, 2019.

[30] T. O'Malley, A. Narayanan, and Q. Wang, "A universally-deployable asr frontend for joint acoustic echo cancellation, speech enhancement, and voice separation," *arXiv preprint arXiv:2209.06410*, 2022.

[31] A. Narayanan and D. L. Wang, "Ideal ratio mask estimation using deep neural networks for robust speech recognition," in *Proc. ICASSP*, 2013, pp. 7092–7096.

[32] A. Narayanan, J. Walker, S. Panchapagesan, N. Howard, and Y. Koizumi, "Learning mask scalars for improved robust automatic speech recognition," in *Proc. IEEE SLT*, 2022.

[33] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *Proc. ICASSP*. IEEE, 2015.

[34] J. Kearns, "Librivox: Free public domain audiobooks," *Reference Reviews*, 2014.

[35] M. Abadi, P. Barham, J. Chen, Z. Chen, and A. Davis et al., "Tensorflow: a system for large-scale machine learning.," in *OSDI*, 2016, vol. 16, pp. 265–283.

[36] Jonathan Shen, Patrick Nguyen, Yonghui Wu, Zhifeng Chen, et al., "Lingvo: a modular and scalable framework for sequence-to-sequence modeling," 2019.