A Universally-Deployable ASR Frontend for Joint Acoustic Echo Cancellation, Speech Enhancement, and Voice Separation

Tom O'Malley, Arun Narayanan, Quan Wang

Google LLC, U.S.A

{omalleyt, arunnt, quanw@google}.com

Abstract

Recent work has shown that it is possible to train a single model to perform joint acoustic echo cancellation (AEC), speech enhancement, and voice separation, thereby serving as a unified frontend for robust automatic speech recognition (ASR). The joint model uses contextual information, such as a reference of the playback audio, noise context, and speaker embedding. In this work, we propose a number of novel improvements to such a model. First, we improve the architecture of the *Cross-Attention Conformer* that is used to ingest noise context into the model. Second, we generalize the model to be able to handle varying lengths of noise context. Third, we propose *Signal Dropout*, a novel strategy that models missing contextual information. In the absence of one or more signals, the proposed model performs nearly as well as task-specific models trained without these signals; and when such signals are present, our system compares well against systems that require all context signals. Over the baseline, the final model retains a relative word error rate reduction of 25.0% on background speech when speaker embedding is absent, and 61.2% on AEC when device playback is absent.

Index Terms: Noise robust ASR, Speaker embedding, Neural AEC, VoiceFilter

1. Introduction

Neural network based end-to-end models [1, 2, 3, 4], large-scale training data [5, 6, 7], and improved data augmentation strategies [8, 9, 10] have significantly improved the robustness of automatic speech recognition (ASR) systems. However, factors such as device echo from smart speakers, harsher background noise, and competing speech still significantly deteriorate performance [11, 12]. While it is possible to train separate ASR models that address each adverse condition, in practice, it is hard to maintain multiple task-specific ASR models and dynamically pick the model that perfectly suits a use case. Furthermore, with large scale multi-domain [7] and multi-lingual modeling [13, 14] gaining more interest, it is increasingly infeasible to optimize the ASR model to address varying use cases and adverse noise conditions simultaneously. Therefore, it is often convenient to use separate frontend feature-processing modules that address these adverse conditions, and to train and maintain them separately from the ASR models.

Adverse noise conditions can broadly be classified into 3 distinct categories: background noise, competing speech, and device echo.

Background noise: Non-speech background noise is handled via data augmentation strategies such as multi-style training (MTR) [15, 8]. This technique is relatively straight-forward

to apply during ASR training, and as a result, large-scale ASR models are generally robust to moderate levels of non-speech noise. However, background noise can still degrade performance in low signal-to-noise ratio (SNR) conditions [16].

Competing speech: Competing speech refers to the presence of one or more non-target speakers in the utterance. ASR models are typically trained on a single speaker, since in multi-speaker conditions it is generally unclear as to which speaker the model should attend to, without additional contextual information. Models that separate out multiple speakers [17] are generally sub-optimal since it is difficult to determine how many speakers to support. As a consequence, competing speech conditions are quite challenging for typical ASR models.

Device echo: With interactive devices such as smart home speakers, ASR performance degrades severely when the device is playing back audio, especially if that audio contains discernible speech, which is typical for voice assistants. Acoustic echo cancellation (AEC) techniques [18, 19, 20] address this issue. Signal processing [18, 19, 21, 22] and neural network [20, 23, 24, 25] based solutions have both been proposed for AEC. What makes the task distinct from the others is that the reference signal of the device playback is usually available and can be used for noise suppression.

The aforementioned noise conditions have been addressed in the literature, typically, using task-specific models. A recent work has shown that it is possible to address all 3 types of interference simultaneously with a single model [26]. Such a model uses multiple contextual signals:

- A noise context signal, which is a noise-only segment prior to the utterance, to help remove background noise.
- A speaker embedding vector for improving performance in conditions with competing speech.
- The playback reference signal for removing device echo.

The model proposed in [26] makes the simplifying assumption that the necessary context signals are always present during training and inference. But in practice, not all contextual signals are available at all times. For example, speaker embedding vectors are usually obtained via an optional speaker enrollment process, and many device users choose to opt-out of this enrollment. Similarly, the reference signal for echo cancellation may not be available on certain devices, or are badly aligned and, therefore, unusable. In this work, we generalize this previous work to handle missing contextual information. Modeling missing information is a well studied area, especially in the audio-visual literature [27, 28, 29]. These works typically deal with occlusion and corrupted signals, which is more common to visual signals. Unlike these earlier works, we specifically address completely missing contextual information and strictly for an acoustic frontend.

A related problem is the availability and the length of the noise context before the utterance. This depends on the ex-

10.21437/Interspeech.2022-11140

istence and size of the audio buffer on the device, which can also vary widely. Apart from modeling missing information, we therefore also propose modifications to noise context modeling to allow for varying lengths of noise context.

Finally, we propose improvements to the architecture of the Cross-Attention Conformer [30] that was used in [26] for noise context modeling.

The rest of this paper is laid out as follows: Sec. 2 provides a detailed description of the proposed system. Sec. 3 describes our general experimental settings. Experiments and results are presented in Sec. 4 and conclusions in Sec. 5.

2. System

We start by giving a brief overview of the system in [26], which is what we build on, and then focus on the improvements that allow this system to generalize better.

2.1. Overview

Fig. 1 shows a block diagram of the overall system. The model is composed of a primary encoder, a noise context encoder, and a cross-attention encoder that combines the outputs of the two encoders. Each of these encoders consists of modified conformer blocks [31] that use a speaker embedding vector to modulate their input. Conformers have been shown to be well-suited for speech and audio tasks in prior work [31, 32]. Modulation via a speaker embedding vector allows the model to focus on the speaker of interest during enhancement.
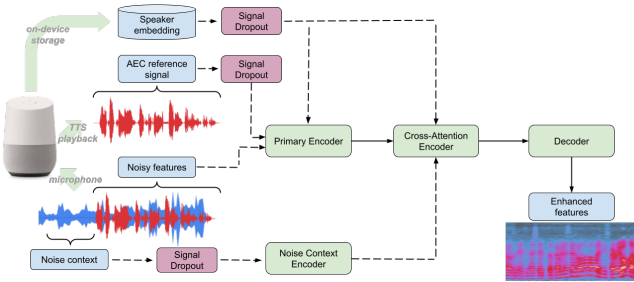

Figure 1: *System overview. The system receives noisy features, and 3 optional signals: the device playback, noise context, and speaker embedding. Signal Dropout is performed only during training.*

Primary Encoder: The primary encoder consists of N modified conformer blocks. The inputs to the primary encoder are log Mel-filterbank energy (LFBE) features from the noisy signal and the AEC reference signal, stacked together in the frame dimension. The encoder also receives a speaker embedding of the target speaker as a side input. At the start of each conformer block, the speaker embedding is combined with the block's inputs using feature-wise linear modulation (FiLM) [33]. The modulated inputs are then passed to a standard conformer encoder.

Noise Context Encoder: The noise context encoder is composed of N standard conformer blocks, and is executed in parallel with the primary encoder. The encoder uses LFBE features from the noise context as input.

Cross-Attention Encoder: The outputs of the primary encoder and the noise-context encoder are fed to the cross-attention encoder [30] as the input feature and the auxiliary feature, respectively. The cross-attention encoder is composed of M modified cross-attention conformer blocks. Before each block, we repeat the process of combining the speaker embed-

ding with the input features using FiLM, similar to the primary encoder. The cross-attention encoder summarizes encoded noise-context features with a cross-attention layer that uses the modulated input features as *queries*. The summarized noise context is subsequently merged with the inputs using FiLM and a second cross-attention layer. In prior work, we show that the cross attention conformer is able to summarize highly non-stationary noise types, summarizing it separately for each input frame to be enhanced, thereby overcoming the limitations of using a single noise embedding to represent noise context [30].

2.2. Proposed improvements

2.2.1. Improvements to Cross-Attention Conformer

We modify the cross attention blocks to better summarize and merge the noise context with input features. After processing encoded input features and noise context features through feed-forward and convolutional layers, we summarize the processed noise context features using cross-attention. Similar to [26], for cross-attention, the processed input features are used as queries, and the noise context features are used for deriving keys and values. But unlike [26], we skip the residual connection after cross-attention, relying on the subsequent FiLM layer to combine the noise-context summary with the input features. Following FiLM, we use a self-attention layer instead of the cross-attention layer used in [26].

Mathematically, if x, m, and n are the encoded input, d-vector and the encoded noise context from the previous layer, the modified cross-attention encoder does the following:

$$
\begin{aligned}
\hat{x} &= x + r(m) \odot x + h(m), \\
\tilde{x} &= \hat{x} + \frac{1}{2}\text{FFN}(\hat{x}), \tilde{n} = n + \frac{1}{2}\text{FFN}(n), \\
x' &= \tilde{x} + \text{Conv}(\tilde{x}), n' = \tilde{n} + \text{Conv}(\tilde{n}), \\
x'' &= \text{MHCA}(x', n'), \\
x''' &= x' + x' \odot r(x'') + h(x''), \\
x'''' &= x''' + \text{MHSA}(x'''), \\
y &= \text{LayerNorm}(x'''' + \frac{1}{2}\text{FFN}(x'''')).
\end{aligned} \tag{1}
$$

Here, MHCA and MHSA stand for multi-headed cross attention and multi-headed self-attention, respectively. The modifications, compared to [26], are marked in blue. With these modifications, x'' now strictly corresponds to a noise summary, x''' merges input features and noise summary, and x'''' adds self-attention processing block on the combined features. y is the final output of the processing block, which is passed on to the next layer, along with the d-vector, m, and the encoded noise context, n.

2.2.2. Generalized Noise-context Modeling

In the previous work, it was assumed that 6 seconds of background noise context was always available to the frontend model. In this work, we relax that assumption by training our model on randomly trimmed noise contexts, with the context length uniformly sampled from 0 to 6 seconds. The random length of the resulting noise context poses a challenge to the absolute positional embedding, which is commonly used in transformer / conformer blocks when modeling sequences. As the distance of each noise context frame from the start of the target utterance is now also dependent on the length of the noise

context, adding the positional embedding [34, 31] during the pre-processing stage of the noise context can provide misleading information to both the noise context encoder and the cross attention encoder. For example, the last noise context frame will have different absolute positional embedding depending on the length of the noise context. Intuitively, it is the actual spectral information carried in the noise context (especially for non-stationary noise types) that helps improve enhancement. Furthermore, it is the distance of a noise context frame from the start of the target utterance that is more important than their absolute position within the noise context.

Based on these intuitions, we consider two alternatives to absolute positional embedding. First, we drop the positional embedding in the noise context encoder, but use a reversed absolute positional embedding in the cross-attention encoder. That is, instead of using a positional embedding, $p(t)$, for the noise context frame, $n(t)$, the reversed positional embedding uses $p(T_N - t)$ as the embedding, where T_N is the number of frames in the noise context. Therefore, the positional embedding encodes how long before the target utterance a frame of noise context occurred, fixing the misalignment problem otherwise introduced by the random trim. The second approach drops positional embedding altogether in both the noise context encoder and the cross-attention encoder.

2.2.3. Modeling Missing Context Signals

To allow the system to generalize to environments in which one or more context signals are missing, we propose using *signal dropout* during training, wherein context signals are dropped out with a given probability. The idea is similar to dropout regularization [35], but applied at an input level. Our goal with signal dropout is to encourage the model to not just rely on the most relevant context signal, but to also utilize alternative context signals that may provide useful information for enhancing the input. For example, for AEC, even when the reference signal is missing, the speaker embedding vector will provide some information to separate the target signal from the noisy input.

Since the model architecture is typically static, the dropped-out context signal still needs to be presented in some fashion to the model. Therefore, when a context signal is dropped, we replace the features that would have been generated for that signal with all-zeros. For the reference signal for AEC, we thus create an all-zero feature of the same length and feature dimension as the utterance. The feature dimension is the same as that of the LFBE features used for the reference signal, if it were not missing. For the noise context signal, we create an all-zero feature with a length of 6 seconds, and the same dimension as the LFBE features. The speaker embedding is replaced by a 256-dimensional all-zero vector, which matches the length of the speaker embedding vector. All features are then fed, as before, to the joint ASR frontend.

We also experimented with replacing the missing signals with a frame-level learned representation, rather than all-zeroes. However, we found that this did not improve performance over using all-zeros.

3. Experimental settings

3.1. Datasets

Similar to [26], we train on datasets derived from LibriSpeech [36] as well as internal vendor-collected utterances. LibriSpeech consists of 281k utterances. The vendor-collected sets consist of 1,916k utterances. We treat these data sources

Table 1: *Ablation study of the proposed Cross-Attention Conformer changes. WER are reported.*

Model	Noise
Proposed Model	**13.6**
- Residual changes	13.7
- Self-Attention changes [30]	14.2

Table 2: *Results using generalized noise context modeling. Reversed refers to reversed positional embedding for noise and None refers to no positional embedding. WER are reported.*

Positional Embeddings	Speech		Noise	
	-5 dB	5 dB	-5 dB	5 dB
Reversed	41.0	21.0	31.8	12.9
None	**39.4**	**20.4**	**30.2**	**12.3**

as 'clean' and add device echo, background noise, or competing speech to each utterance. The test sets are created using the test-clean subset of Librispeech.

Background noise: The speech enhancement training set is created using a room simulator [8]. This first adds reverberation with T60s between 0 msec and 900 msec, then noise with an SNR in the range [-10 dB, 30 dB]. The noise snippets represent typical noise conditions such as kitchen, cars, etc. as well as publicly available noises from Getty[1] and YouTube Audio Library[2]. Test sets are also created in a similar fashion, but using mixing conditions that are disjoint from training.

Competing Speech: In order to simulate multi-speaker conditions, we mix the training utterances with competing speech from the training datasets, chosen randomly. Test sets are also constructed in a similar fashion.

Device Echo: We use two types of device echoes [37]. The first type consists of entirely synthetic echoes, created using a reference signal played through a reverberant room simulator, with the speaker configured to be close to the microphone. Reference signals are drawn from Librispeech. The second type of training data contains re-recorded echoes using an internal TTS dataset. These utterances are re-recorded after being played back in a room on Google Home devices at varying signal levels. The goal is to capture microphone non-linearities, which are harder to model using a room simulator. The echoes are added to reverberant target speech signal at SNRs in the range $[-20 \text{ dB}, 5 \text{ dB}]$. The AEC evaluation sets are created using only these re-recorded echoes, since that subset is both more challenging and closer to real-world conditions.

3.2. Training details

All models are trained in TensorFlow [38], using the Lingvo [39] toolkit. All joint frontend models contain approximately 15M parameters. We use 2 conformer layers for each of the primary encoder, noise context encoder, and cross-attention encoder. Each conformer layer has 256 units, with the kernel size for convolution set to 15. 128-dimensional LFBE features are used for all signals, computed for 32 msec windows with 10 msec hop. The dvector is 256-dimensional. All attention layers use causal self-attention with a window of 65 frames in the past.

The frontend model estimates an ideal ratio mask [40], which is used to enhance the noisy LFBE features. The enhanced features are then passed to a pretrained recurrent neural

---

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

Table 3: *Missing Context Signal Results. Word error rates are reported for -5 db SNR background speech, -5 dB SNR background noise, and -10 db SNR AEC.*

Model	All context			No dvector	No noise context	No playback
	Speech	Noise	AEC	Speech	Noise	AEC
Baseline	69.2	36.5	80.5	69.2	36.5	80.5
0% Signal Dropout	**44.6**	**31.9**	**23.9**	61.7	34.9	65.5
20% Signal Dropout	45.1	32.0	24.2	51.9	**33.6**	31.2
50% Signal Dropout	45.0	32.2	26.1	**50.9**	34.3	30.6
Dedicated Model	-	-	-	51.3	34.5	**29.0**

net transducer based ASR model [41]. As losses, the frontend models use a combination of $\mathcal{L}_1$ and $\mathcal{L}_2$ distance between estimated and ideal masks, and an ASR loss that encourages the activations of a pretrained ASR encoder generated using clean and enhanced LFBE features to be close to each other.

4. Results

All experiments are evaluated by passing noisy data through the enhancement frontend, and then passing the enhanced features to a pre-trained ASR model. We report the ASR model's word error rate (WER).

4.1. Modified Cross-Attention Conformer

We perform a few ablations on the improvements to the Cross-Attention Conformer. The results are shown in Table 1. We only report results on the evaluation data with background noise, since the Cross-Attention Conformer is primarily used to integrate noise context into the model. WERs are reported on an evaluation set consisting of background noise added to the dev-clean subset of Librispeech at a uniform range of SNRs from -5 dB to 15 dB. As can be seen, our proposed system achieves a reduction in relative WER of 4.2% compared to the original Cross-Attention Conformer. Most of these gains are coming from replacing the last cross-attention with self-attention; moving the residual connection from the first cross-attention to the subsequent FiLM module improves WER by only ~1%.

4.2. Generalized Noise-Context Modeling

Table 2 shows the WERs using the various strategies to encode positional information for noise context. For these experiments, we train a model with no signal dropout on the training data with background noise and background speech. Although using reversed positional embedding to encode the distance in time from the beginning of the target utterance is helpful, we find that removing the positional embedding entirely outperforms such a method, across all conditions. This is likely because it encourages the model to summarize noise context entirely based on the spectral information, as opposed to the position of the frame in the noise context.

4.3. Modeling Missing Context Signals

We train models with 2 different probabilities of dropping each context signal: 20%, and 50%. The same probability is used for each context signal. We also train dedicated models without speaker embedding, a model without reference signal, and a model without noise context, to use as strong comparison points for evaluating performance when a signal is missing. Similarly, we train a model with all contextual signals always present, to use as a strong comparison point for evaluating performance when all signals are present. The "Baseline" corresponds to using the noisy features directly for ASR without any enhance-

ment.

For evaluations, we separate each noise type into its own evaluation set, and compare the performance of the model on each set at different SNRs. All results shown are for -5 dB SNR background speech, -5 dB SNR background noise, and -10 dB SNR acoustic echo settings. Additionally, to measure performance when a signal is absent, we create evaluation sets without speaker embedding, without noise context, and without the reference signal.

Table 3 shows the WER of the missing context signal experiments. These results show that, when all context signals are present, the 20% and 50% signal dropout method performs nearly as well as a model trained without any signal dropout (0% signal dropout). Additionally, when a context signal is missing, the models trained with signal dropout significantly outperform such a model and achieve comparable performance to a dedicated model trained entirely without that context signal.

Training with a 20% signal dropout rate appears sufficient to achieve the majority of these benefits. When all signals are present, the 20% signal dropout model achieves a relative WER within 1.1% on the background speech dataset, 0.3% on the background noise dataset, and 1.2% on the AEC dataset compared to the 0% signal dropout model. Compared to dedicated models, the 20% signal dropout model performs within 1.2% relative WER on the background speech dataset when speaker embedding is missing, slightly better on the background noise dataset when noise context is missing, and within 7.6% relative WER on the AEC dataset when device playback is missing.

Compared to baseline, when all signals are present the 20% signal dropout model reduces relative WER in low SNR environments by 35.5%, 12.3%, and 70.0%, on background speech, background noise, and AEC datasets, respectively. The model retains a relative reduction over baseline of 25.0% on background speech when speaker embedding is absent, 7.9% on background noise when noise context is absent, and 61.2% on AEC when device playback is absent. We note that the model performs surprisingly well on AEC even when the device playback is missing, presumably because it is able to make use of the speaker embedding; this shows that the models trained with signal dropout learns to make use of the available information even when the most relevant context signal is missing.

5. Conclusion

In this work, we presented an improved ASR frontend for joint AEC, speech enhancement, and voice separation. Our system can handle the presence or absence of multiple context signals, allowing it to be deployed in a wide variety of environments. In addition, we adapted the system to better incorporate noise context of varying lengths. Finally, we showed general-purpose architectural improvements to the originally proposed Cross-Attention Conformer. In future work, we plan to explore extending our approach to multi-channel inputs.

6. References

[1] R. Prabhavalkar, K. Rao, T. N. Sainath, B. Li, L. Johnson, and N. Jaitly, "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] 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.

[12] 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.

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

[14] Vineel Pratap, Anuroop Sriram, Paden Tomasello, Awni Hannun, Vitaliy Liptchinsky, Gabriel Synnaeve, and Ronan Collobert, "Massively multilingual ASR: 50 languages, 1 model, 1 billion parameters," *Proc. Interspeech*, 2020.

[15] R. Lippmann, E. Martin, and D. Paul, "Multi-style training for robust isolated-word speech recognition," in *Proc. ICASSP*. IEEE, 1987.

[16] T. Ko, V. Peddinti, D. Povey, M. L. Seltzer, and S. Khudanpur, "A study on data augmentation of reverberant speech for robust speech recognition," in *Proc. ICASSP*. IEEE, 2017.

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

[18] E. Hänsler and G. Schmidt, *Acoustic echo and noise control: A practical approach*, vol. 40, John Wiley & Sons, 2005.

[19] J. Benesty, T. Gänsler, D. Morgan, M. Sondhi, and S. Gay, *Advances in network and acoustic echo cancellation*, Springer, 2001.

[20] H. Zhang and D. Wang, "Deep learning for acoustic echo cancellation in noisy and double-talk scenarios," in *Proc. Interspeech*, 2018.

[21] J. Benesty, C. Paleologu, T. Gänsler, and S. Ciochină, *A perspective on stereophonic acoustic echo cancellation*, vol. 4, Springer Science & Business Media, 2011.

[22] G. Enzner, H. Buchner, A. Favrot, and F. Kuech, "Acoustic echo control," in *Academic Press Library in Signal Processing*, vol. 4, pp. 807–877. Elsevier, 2014.

[23] A. Fazel, M. El-Khamy, and J. Lee, "Deep multitask acoustic echo cancellation.," in *Proc. Interspeech*, 2019.

[24] Q. Lei, H. Chen, J. Hou, L. Chen, and L. Dai, "Deep neural network based regression approach for acoustic echo cancellation," in *Proc. of the 2019 4th International Conference on Multimedia Systems and Signal Processing*, 2019.

[25] S. Ding, Y. Jia, K. Hu, and Q. Wang, "Textual echo cancellation," *arXiv preprint arXiv:2008.06006*, 2020.

[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 *IEEE ASRU*, 2021.

[27] Triantafyllos Afouras, Joon Son Chung, and Andrew Zisserman, "My lips are concealed: Audio-visual speech enhancement through obstructions," *Proc. Interspeech*, 2019.

[28] Shiliang Zhang, Ming Lei, Bin Ma, and Lei Xie, "Robust audio-visual speech recognition using bimodal dfsmn with multi-condition training and dropout regularization," in *ICASSP 2019-2019 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2019, pp. 6570–6574.

[29] Jianwei Yu, Shi-Xiong Zhang, Bo Wu, Shansong Liu, Shoukang Hu, Mengzhe Geng, Xunying Liu, Helen Meng, and Dong Yu, "Audio-visual multi-channel integration and recognition of overlapped speech," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 2067–2082, 2021.

[30] A. Narayanan, C.-C. Chiu, T. O'Malley, Q. Wang, and Y. He, "Cross-attention conformer for context modeling in speech enhancement for ASR," in *IEEE ASRU*, 2021.

[31] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, et al., "Conformer: Convolution-augmented transformer for speech recognition," *arXiv preprint arXiv:2005.08100*, 2020.

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

[33] E. Perez, F. Strub., H. de Vries, V. Dumoulin, and A. Courville, "FiLM: Visual reasoning with a general conditioning layer," in *Proc. of the AAAI Conference on Artificial Intelligence*, 2018.

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

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

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

[37] N. Howard, A. Park, T. Z. Shabestary, A. Gruenstein, and R. Prabhavalkar, "A neural acoustic echo canceller optimized using an automatic speech recognizer and large scale synthetic data," in *Proc. ICASSP*. IEEE, 2021.

[38] 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.

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

[40] 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.

[41] T. N. Sainath, Y. He, B. Li, A. Narayanan, and R. Pang et al., "A streaming on-device end-to-end model surpassing server-side conventional model quality and latency," in *Proc. ICASSP*. IEEE, 2020.