Deep model with built-in self-attention alignment for acoustic echo cancellation

Evgenii Indenbom, Nicolae-Cătălin Ristea, Ando Saabas, Tanel Pärnamaa, Jegor Gužvin

Microsoft Corporation
{eindenbom, nristea, ando.saabas, tanel.parnamaa, jeguzvin}@microsoft.com

Abstract

With recent research advances, deep learning models have become an attractive choice for acoustic echo cancellation (AEC) in real-time teleconferencing applications. Since acoustic echo is one of the major sources of poor audio quality, a wide variety of deep models have been proposed. However, an important but often omitted requirement for good echo cancellation quality is the synchronization of the microphone and far end signals. Typically implemented using classical algorithms based on cross-correlation, the alignment module is a separate functional block with known design limitations. In our work we propose a deep learning architecture with built-in self-attention based alignment, which is able to handle unaligned inputs, improving echo cancellation performance while simplifying the communication pipeline. Moreover, we show that our approach achieves significant improvements for difficult delay estimation cases on real recordings from AEC Challenge data set.

Index Terms: aligning, echo cancellation, noise suppression, speech enhancement, real-time processing, transformers

1. Introduction

In recent years, the use of teleconferencing systems, e.g. Microsoft Teams, Skype, Zoom etc., has increased significantly, being indispensable for remote working. For these systems it is mandatory to ensure good call quality in order to provide a productive and pleasant experience to end-users. Degradations caused by acoustic echo are one of the major sources of poor call quality in voice and video calls. This problem becomes even more challenging in fully-duplex communication when echo interferes with double talk scenario [1]. Formally, acoustic echo appears when a loudspeaker and a microphone are coupled in a communication system such that the microphone picks up the loudspeaker signal, usually distorted by the room impulse response. If not properly handled, the far end user will hear his or her own voice delayed by the round trip time of the system (i.e., an echo), mixed with the target signal from the near end. To solve this issue, AEC algorithms are employed.

Conventionally, AEC methods based on digital signal processing (DSP) have been used to remove echoes during calls, but the performance of such approaches can degrade given devices with poor physical acoustics design or environments outside their lab-based tests [1]. Mainly, the echo cancellation is accomplished by identifying an acoustic impulse response between the loudspeaker and the microphone using a finite impulse response filter [2,3]. To increase the convergence rate of adaptive filtering algorithms and improve their robustness against double-talk, many efforts have been made in the past decades [2,3]. However, due to the advance of deep learning techniques, deep AEC models progressively replaced their classical counterparts, showing superior performance [4].

The application of deep learning models for audio related tasks started with combining classical DSP-based methods with neural networks. For example, [7,8,9] have demonstrated that the combination of adaptive filters and recurrent neural networks (RNNs) provide good performance in the AEC task. Other researchers have tried to develop a pure deep learning solution for echo cancellation problem and obtained convincing results [1,10,11,12] on complex data sets. The ability of recurrent neural networks (RNNs), e.g., gated recurrent units (GRU) [13], long short term memory (LSTM) [13] to model time varying functions play an important role in addressing AEC problems [15,16]. Braun et al. [15] proposed CRUSE model for noise suppression, based on a U-Net architecture, with a middle recurrent block. The model offers a good trade-off between computational complexity and speech quality, measured on real recordings using an accurate mean opinion score (MOS) estimator. Motivated by these advances in applying deep learning models to echo cancellation, we seek to provide a model which can simplify the communication pipeline, while improving the overall performance.

One major issue in AEC systems is the delay between the microphone and the far end reference signals, which drastically affects performance. In most cases, it is assumed that the acoustic echo path is linear and the time delay is limited to a known prior. Under this assumption, the acoustic echo signal can be cancelled effectively using traditional methods [17]. However, the performance of the existing linear AEC algorithms may be greatly degraded in many practical applications, because of the hardware-related latency or software buffering mechanism, which leads to a larger range for delays. Moreover, situations when the acoustic echo path may be time variant (e.g., the speaker changes location during a call) or the microphone signal could contain some environmental noise and reverberation, also impacts the AEC performance. Figure 1 shows the distribution of estimated delays between microphone and far-end and loopback signals on real recordings from the AEC Challenge data set [11]. The delays are estimated using a cross-correlation based algorithm over the entire clip. The plot shows a long tail for the delay distributions, but also outliers in both ends of the distributions, i.e. estimates of negative delay or delay over 800 ms. The outliers are typically mis-estimates due to distorted or noisy signals. In real-time call scenarios, such mis-estimates typically entail significant echo leaks when conventional signal alignment is used.

Many works use already aligned data or compensate the delay by a separate block [18, 19], a procedure which can work poorly in practice. Instead, being inspired by the success of self-attention architectures [20], lately adopted in audio related tasks [10, 21], we propose a real-time deep neural network architecture with built-in alignment module based on self-attention, named Align-CRUSE, capable of handling non-aligned microphone and far end signals in linear and non-linear echo path scenarios. Our model eliminates the necessity of an alignment block, which is conventionally performed with DSP algorithms (e.g., cross-correlation [22]), by including a built-in
self-attention module, which synchronizes the microphone and far end signals in a latent space. Instead of hard alignment of self-attention module, which synchronizes the microphone and real-time applications. Inference time and latency, our model is a viable solution for thermore, considering low computational complexity and low

800

A generic acoustic echo cancellation system could be formally signal an acoustic echo path (emulated by the room impulse response). \( \hat{z}_t \) is received by the far end user as \( \tilde{s}_t \). We highlight that the echo \( e(t) \) component from the microphone signal \( m(t) \) is a delayed version of the received reference far end signal \( f(t) \), because of the echo propagation path (from loudspeaker to microphone) and hardware or software related latency. Therefore, between the far end and near end users an AEC system is integrated. Hence, our goal is to remove undesired echoes, having the microphone and reference far end signals.

2.2. Feature extraction

All audio signals are sampled at 16 kHz and the preprocessing is performed identically for reference far end and microphone signals. The input features to the network are log power spectra computed with a squared root Hann window.

2.3. Network architecture

The network structure is derived from [15], with AEC adaptation and self-attention mechanism for built-in deep alignment. The model is composed by an encoder, decoder and a recurrent module, as illustrated in Figure 2. In this section we will use \( c, c_1, c_2, p, t, f \in \mathbb{N} \) to denote arbitrary axis lengths.

Encoder. In the encoding stage, the network consists of two branches, having as input the far end and microphone features. Each conv block is built by a convolutional layer, a batch-norm layer and an exponential linear unit (ELU) [23] activation function. The far end branch is composed by two conv blocks, followed by the align block, which takes as input the far end features and the depth-corresponding feature maps from the microphone branch. Consequently, the aligned far end maps are concatenated to the microphone branch and processed by two more conv blocks. The numbers of kernels for the microphone conv blocks are 16, 40, 72, 32. The far end branch is composed by 8 and 24 filters. All convolution kernels have size \( 4 \times 3 \) and a stride of \( 1 \times 2 \), reducing the number of bins along the frequency axis. Each convolution is causal, meaning that the padding is performed such that no look-ahead is used.

Recurrent block. Between the encoder and decoder sits a recurrent layer, which is fed with feature maps flattened along the channel and frequency dimensions. Formally, the input \( X \in \mathbb{R}^{c \times t \times f} \) is flattened into \( X \in \mathbb{R}^{c \times (t \times f)} \). Afterwards, \( X \) is fed into the recurrent layer and the output is reshaped back to \( X \in \mathbb{R}^{c \times t \times f} \). Following [15], considering that an LSTM does not bring significant performance improvements over a GRU, we use a GRU layer to reduce the model complexity.

Skip block. Replacing the classical skip connection, based on concatenation or summing, with a trainable channel-wise scaling and bias, improves the performance at small additional cost [15]. Moreover, it allows us to use asymmetric encoder-decoder blocks, by adapting the number of encoding channels to the number of corresponding decoding channels. Considering \( X \in \mathbb{R}^{c_1 \times t \times f} \) the decoder input and \( Y \in \mathbb{R}^{c_1 \times t \times f} \) the corresponding encoder input, the skip block is defined as follows:

\[
Z = X + \text{Conv}(Y),
\]

where \( Z \in \mathbb{R}^{c_2 \times t \times f} \) and \( \text{Conv}() \) has \( c_2 \) kernels of size \( 1 \times 1 \) applied with stride of \( 1 \times 1 \).

Decoder. The decoding stage consists of 3 alternating skip and conv transpose blocks, followed by a last skip and mask block. The reshaped output from GRU, \( Z \in \mathbb{R}^{c_2 \times t \times f} \) is combined with the corresponding features from the encoder (into the skip block) and fed into the transpose conv block. Each transpose conv block is composed by a transposed convolutional layer, followed by a batch norm and an ELU activation function. For each convolutional layer, we used a non causal kernel, having size of \( 1 \times 3 \), which computes features along the frequency dimension. The stride is identical to the encoding part, while the number of filters for convolutional blocks are 32, 48 and 48. Subsequently, the output of the last skip block is processed by the mask block. It consists of a convolutional layer with a single filter of \( 1 \times 3 \), followed by a sigmoid activation. Moreover, to compensate potential over suppression by applying masking, we added a learnable parameter in the mask block, which multiply the output mask. This allows the model to have an internal learnable control gain.

Align block. Let \( X_m \in \mathbb{R}^{c \times t \times f} \) be the microphone features and \( X_f \in \mathbb{R}^{c \times t \times f} \) the far end features. Firstly, the feature maps are reduced with a max-pooling layer (having a kernel size of \( 1 \times 4 \)) along the frequency dimension to reduce the computation cost brought by the alignment module. Further, the features are reshaped such that \( X_m \in \mathbb{R}^{c \times (t \times f)} \) and \( X_f \in \mathbb{R}^{c \times (t \times f)} \). Next, they are projected into queries \( Q \in \mathbb{R}^{c \times f} \) and keys \( K \in \mathbb{R}^{c \times f} \). Secondly, the \( K \) tensor is zero-padded at the beginning and cropped at the end with the same \( d \) value, generating a synthetic delay. Afterwards, a time axis dot product is performed for delayed \( K \) and \( Q \). This procedure is composed for each delay index \( d \) from a specific interval, given by the maximum supported delay \( d_{\text{max}} \), conducting to a result vector of length

Figure 1: Delay distribution for far end and loopback signals compared to the microphone. Delays smaller than \(-100\)ms and bigger than \(800\)ms are truncated.
Prediction. The model’s output is a suppression mask used to enhance the microphone signal by removing undesired components. The inference process starts with the microphone and far-end signals, which are transformed into time-frequency features with STFT and fed into the network. The output suppression mask is applied to the complex spectrum of the microphone signal, resulting in the enhanced complex signal. Further, the inverse STFT is computed to obtain the time domain enhanced signal. The process is illustrated in the Figure 2 top-middle.

2.4. Loss function

We train the networks with STFT consistency enforcement by propagating the time domain complex enhanced output again through STFT. In the training stage, the reconstructed complex spectrum of the enhanced signal is fed into the loss function with the time domain target signal processed by STFT. The training procedure is shown in Figure 2 top-right. The network is optimized by minimizing the complex compressed mean-squared error loss, which blends the magnitude with a phase-aware term, which we found to be superior to other losses. Formally, the loss function is given by:

$$L = \beta \sum_{n,k} |S|^2 - |\hat{S}|^2 + (1 - \beta) \sum_{n,k} |S|^2 e^{j\phi_n} - |\hat{S}|^2 e^{j\phi_n}|^2$$

where \(c = 0.3\) is a compression factor, \(\beta = 0.7\) is a weighting factor between complex and magnitude-based losses. We omitted the dependency of \(S(k, n)\) and \(\hat{S}(k, n)\) on the time and frequency indices \(n, k\) for brevity.

Figure 2: Align-CRUSE architecture for the AEC task. The output is a magnitude mask, used to predict the enhanced microphone signal. In the right-top, the training procedure is illustrated. Samples marked with * are complex valued. Best viewed in colors.

Figure 3: The aligning block used to synchronize microphone and far end latent features.

3. Experimental setup

3.1. Data sets

To ensure the generalization ability, the training data are synthesized online with random parameters for each sample (e.g., signal-to-noise ratio, distortion, gain, signal-to-echo ratio).

Training sets. We trained the network on data from the AEC challenge. The training set contains more than 10,000 real scenarios of audio in diverse environments collected with different devices.

Test sets. We tested our approach on the blind test set from challenge, which contain real world recordings in diverse scenarios. We split the far end single talk blind test set into two: FEST-HD, which contains 27 samples with difficult delay estimation cases (e.g., long delays or variable delays, as indicated by the authors), and FEST-GEN, which contains 273 samples with other types of scenarios (e.g., non-linear-distortions, stationary-noise). Moreover, we generated two synthetic data sets, containing 500 samples each, to specifically address long delay cases. In LD-300-500 we randomly distributed the delays uniformly in 300 – 500ms, while for LD-500-1000 in 500 – 1000ms.

3.2. Evaluation metrics

We employed AECMOS to test the removal capacity of echos. The metric reflects a pseudo-subjective quality of samples, being highly correlated with human subjective opinion. To address the echo cancellation ability in far end single talk scenarios, we also employed the echo return loss enhancement
Table 1: Results of our Align-CRUSE model on the synthetically generated test sets LD-300-500, LD-500-1000 and AEC challenge [11] single talk test set, split into FEST-HD and FEST-GEN. The baseline CRUSE model was tested on non-aligned data (CRUSE), online aligned data (CRUSE†) and globally aligned data (CRUSE‡). The inference time is in milliseconds per frame. For both AECMOS and ERLE metrics, the bigger the better.

| Method          | LD-300-500 | LD-500-1000 | FEST-HD [11] | FEST-GEN [11] | Inference Time (ms) | #Params |
|-----------------|------------|------------|--------------|---------------|---------------------|---------|
|                 | AECMOS     | ERLE       | AECMOS       | ERLE          |                     |         |
| CRUSE†          | 2.31       | 7.63       | 2.21         | 3.93          | 2.91                | 0.216   | 0.74M         |
| CRUSE‡          | 3.89       | 26.30      | 3.57         | 17.57         | 3.78                | 0.216   | 0.74M         |
| CRUSE‡‡         | 3.99       | 35.84      | 3.84         | 33.50         | 4.30                | 0.216   | 0.74M         |
| Align-CRUSE     | 4.54       | 42.88      | 4.44         | 39.37         | 4.54                | 0.196   | 0.75M         |

Table 2: Results for our Align-CRUSE model on the AEC (double talk) challenge test set, against CRUSE† baseline method. For all metrics the bigger the better.

| Method          | AEC [11] | AECMOS | MOS |
|-----------------|----------|--------|-----|
| CRUSE†          | 4.49     | 4.35   |     |
| Align-CRUSE     | 4.56     | 4.40   |     |

Figure 4: The alignment heat-map and spectrograms for an audio sample are illustrated.

3.3. Hyper-parameters tuning

For feature generation we used a squared root Hann window of length 20ms, a hop length of 10ms and a discrete Fourier transform length of 320. We trained both CRUSE and Align-CRUSE models in the same fashion. We used Adam [24] optimizer for all networks, with batches of 400 samples for 150 epochs, with a learning rate of $1.5 \cdot 10^{-4}$ and a weight decay of $5 \cdot 10^{-6}$. We considered $d_{\text{max}} = 100$, equivalent to a maximum delay of 1 second.

3.4. Results

In Table 1 we highlight the importance of alignment by conducting experiments on the CRUSE architecture tested on un-aligned, real-time aligned (CRUSE†) and globally aligned data (CRUSE‡). Even though the global alignment method is based on the entire audio sample, being unsuitable for real-time applications, we included it as a stronger baseline. In Table 2 we show that Align-CRUSE significantly surpasses the baseline models, including the globally aligned CRUSE‡‡ model.

On the synthetic LD-300-500 and LD-500-1000 data sets, our model surpasses CRUSE‡ by up to 0.6 AECMOS and 7 dB ERLE. Against other baselines models, the difference is even more significant.

To measure the generalization capacity of our model, we tested it on the real world recordings from FEST-HD and FEST-GEN data sets. For both FEST-GEN and FEST-HD data sets, we obtain better results than the baselines. Compared to CRUSE†, the improvement is 7.22 dB ERLE and 0.24 AECMOS for FEST-HD and 2.44 dB ERLE and 0.08 AECMOS for FEST-GEN. The results show that having a robust delay estimator is mandatory for hard delay estimation cases, a fact also observed in production. Our self-aligning approach obtains superior results in all AEC data sets, consistently surpassing the CRUSE† baseline in hard delay estimation cases, while also improving the overall performance.

In addition to far-end single talk tests, we compared the models on the blind double-talk data set from the AEC Challenge [11], measuring AECMOS and MOS. For this test, we included only CRUSE† as baseline, since CRUSE tested on un-aligned data is considerably worse (see Table 1) and CRUSE‡‡ is not feasible for real-time applications. As shown in Table 3, Align-CRUSE model surpasses the CRUSE† network in the double talk scenario for both AEC performance metrics.

In Figure 4 an example of alignment is illustrated. We observe that, mainly, our model predicts a constant delay (approximately 5 frames, corresponding to 100ms), but there are regions where a soft distribution is preferred. Additionally, Table 1 shows the number of parameters and the inference time per frame on a CPU Intel Core i7 10600K@3.8 GHz. Our model is faster than the baseline with approximately 10% per frame, while having with only 0.01M parameters more. Thus, our model provides significantly better results, especially in long and hard delay estimation cases, while assuring a lower inference time, critical aspect for real-time processing. Overall, Align-CRUSE improves the AEC in both single talk and double talk scenarios. Considering that the performance boost comes with a communication pipeline simplification and a 10% smaller inference time, Align-CRUSE is a great candidate for real-time applications.

4. Conclusion

We proposed a novel architecture with a built-in aligning module, based on self-attention, which is able to handle unaligned far end and microphone signals in complex echo scenarios. We empirically demonstrated the alignment importance in AEC systems for both synthetic and real world audio samples. Our novel model significantly outperforms the baseline for AEC, having a smaller inference time. Considering that Align-CRUSE simplifies the communication pipeline, while reducing the inference time and improving the performance, we conclude that it is a great candidate for real-time applications.
5. References

[1] K. Sridhar, R. Cutler, A. Saabas, T. Parnamaa, M. Loide, H. Gamper, S. Braun, R. Aichner, and S. Strinivasan, “Icassp 2021 acoustic echo cancellation challenge: Datasets, testing framework, and results,” in ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2021, pp. 151–155.

[2] J. Benesty, T. Gansler, D. R. Morgan, M. M. Sondhi, S. L. Gay et al., “Advances in network and acoustic echo cancellation,” 2001.

[3] S.-S. Lin and W.-R. Wu, “A low-complexity adaptive echo canceller for xDSL applications,” IEEE Transactions on Signal Processing, vol. 52, no. 5, pp. 1461–1465, 2004.

[4] K. Eneman and M. Moonen, “Iterated partitioned block frequency-domain adaptive filtering for acoustic echo cancellation,” IEEE Transactions on Speech and Audio Processing, vol. 11, no. 2, pp. 143–158, 2003.

[5] M. M. Halimeh and W. Kellermann, “Efficient multichannel nonlinear acoustic echo cancellation based on a cooperative strategy,” in ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2020, pp. 461–465.

[6] J. Casebeer, N. J. Bryan, and P. Smaragdis, “Auto-dsp: Learning to optimize acoustic echo cancellers,” in 2021 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA). IEEE, 2021, pp. 291–295.

[7] H. Zhang, K. Tan, and D. Wang, “Deep learning for joint acoustic echo and noise cancellation with nonlinear distortions.” in Interspeech, 2019, pp. 4255–4259.

[8] L. Ma, H. Huang, P. Zhao, and T. Su, “Acoustic echo cancellation by combining adaptive digital filter and recurrent neural network,” arXiv preprint arXiv:2005.09237, 2020.

[9] H. Zhang and D. Wang, “A deep learning approach to multi-channel and multi-microphone acoustic echo cancellation,” Proc. Interspeech 2021, pp. 1139–1143, 2021.

[10] A. Fazel, M. El-Khamy, and J. Lee, “Cad-aec: Context-aware deep acoustic echo cancellation,” in ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2020, pp. 6919–6923.

[11] R. Cutler, A. Saabas, T. Parnamaa, M. Loide, S. Sootla, H. Gamper et al., “Interspeech 2021 acoustic echo cancellation challenge,” in Proc. Interspeech, 2021.

[12] C. Zhang and X. Zhang, “A robust and cascaded acoustic echo cancellation based on deep learning,” in INTERSPEECH, 2020, pp. 3940–3944.

[13] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, “Empirical evaluation of gated recurrent neural networks on sequence modeling,” in NIPS 2014 Workshop on Deep Learning, December 2014, 2014.

[14] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural computation, vol. 9, no. 8, pp. 1735–1780, 1997.

[15] S. Braun, H. Gamper, C. K. Reddy, and I. Tashev, “Towards efficient models for real-time deep noise suppression,” in ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2021, pp. 656–660.

[16] N. L. Westhausen and B. T. Meyer, “Acoustic echo cancellation with the dual-signal transformation lstm network,” in ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2021, pp. 7138–7142.

[17] B. Farhang-Boroujeny, Adaptive filters: theory and applications. John Wiley & Sons, 2013.

[18] R. Peng, L. Cheng, C. Zheng, and X. Li, “Icassp 2021 acoustic echo cancellation challenge: Integrated adaptive echo cancellation with time alignment and deep learning-based residual echo plus noise suppression,” in ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2021, pp. 146–150.

[19] K. N. Watcharasupat, T. N. T. Nguyen, W.-S. Gan, S. Zhao, and B. Ma, “End-to-end complex-valued multidilated convolutional neural network for joint acoustic echo cancellation and noise suppression,” arXiv preprint arXiv:2110.00745, 2021.

[20] A. Vasznavi, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” Advances in neural information processing systems, vol. 30, 2017.

[21] J.-H. Kim and J.-H. Chang, “Attention wave-u-net for acoustic echo cancellation,” in Interspeech, 2020, pp. 3969–3973.

[22] J. Ianniello, “Time delay estimation via cross-correlation in the presence of large estimation errors,” IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 30, no. 6, pp. 998–1003, 1982.

[23] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, “Fast and accurate deep network learning by exponential linear units (elus),” arXiv preprint arXiv:1511.07289, 2015.

[24] S. Wisdom, J. R. Hershey, K. Wilson, J. Thorpe, M. Chinen, B. Patton, and R. A. Saurous, “Differentiable consistency constraints for improved deep speech enhancement,” in ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2019, pp. 900–904.

[25] A. Ephrat, I. Mosseri, O. Lang, T. Dekel, K. Wilson, A. Hassidim, W. T. Freeman, and M. Rubinstein, “Looking to listen at the cocktail party: a speaker-independent audio-visual model for speech separation,” ACM Transactions on Graphics (TOG), vol. 37, no. 4, pp. 1–11, 2018.

[26] M. Purin, S. Sootla, M. Sponza, A. Saabas, and R. Cutler, “Aecmos: A speech quality assessment metric for echo impairment,” arXiv preprint arXiv:2110.03010, 2021.

[27] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in ICLR (Poster), 2015.