IMPROVING THE LATENCY AND QUALITY OF CASCADED ENCODERS

Tara N. Sainath, Yanzhang He*, Arun Narayanan*, Rami Botros, Weiran Wang, David Qiu, Chung-Cheng Chiu, Rohit Prabhavalkar, Alexander Gruenstein, Anmol Gulati, Bo Li, David Rybach, Emmanuel Guzman, Ian McGraw, James Qin, Krzysztof Choromanski, Qiao Liang, Robert David, Ruoming Pang, Shuo-yiin Chang, Trevor Strohman, W. Ronny Huang, Wei Han, Yonghui Wu, Yu Zhang*

Google LLC, USA

{tsainath,yanzhanghe,arunnt}@google.com

ABSTRACT

In this paper, we explore reducing computational latency of the 2-pass cascaded encoder model [1]. Specifically, we experiment with reducing the size of the causal 1st-pass and adding capacity to the non-causal 2nd-pass, such that the overall latency can be reduced without loss of quality. In addition, we explore using a confidence model for deciding to stop 2nd-pass recognition if we are confident in the 1st-pass hypothesis. Overall, we are able to reduce latency by a factor of 1.7X, compared to the baseline cascaded encoder from [1]. Secondly, with the added capacity in the non-causal 2nd-pass, we find that we can improve WER by up to 7% relative using wav2vec and minimum word-error-rate (MWER) training.

Index Terms— end-to-end ASR, rnnt, conformer, long-form ASR, two-pass ASR

1. INTRODUCTION

End-to-end (E2E) models have gained popularity over the past few years, particularly for on-device speech recognition, as they can achieve similar recognition performance compared to conventional hybrid systems [2] at a fraction of the size. One area of particular interest across many research groups [3, 4, 5, 6, 7, 8] is to develop an E2E model that surpasses conventional models in both quality and latency in diverse test conditions.

Recently, we presented one such on-device E2E model that outperformed a conventional model in terms of word error rate (WER) on both search and long-tail queries, as well as endpointer latency [9] metrics. This model consists of a 2-pass cascaded conformer encoder [1] (100M parameter 1st-pass causal conformer encoder, 50M parameter 2nd-pass non-causal conformer encoder), an embedding decoder [10], and a conformer language model to rescore N-best hypotheses. However, these quality wins do not come for free, since they result in an overall increase in computational latency, which we seek to improve in this work.

One of the concerns with a conformer encoder in a streaming ASR setting is that the number of internal *states* to maintain per layer is much larger than an LSTM. For example, an LSTM with 8 layers and 640 dimensions per layer has roughly 5,120 states corresponding to the hidden layer activations that have to be propagated from one time-step to the next. In comparison, a conformer encoder that has 12 layers, 23 frames of left context per layer and 512 dimensions per layer has roughly 141,312 states, almost 30X as much. These states correspond to the "key" and "value" tensors for self-attention.

*Equal contribution

While specialized hardware with a large number of cores, such as edge tensor processing units (TPUs) [11], can significantly speed up overall computation on a single utterance, cloud TPUs, which must process requests across multiple users, have additional bandwidth constraints compared to edge TPUs. As described in [12], conformer processing is often slow due to the memory-bandwidth cost of repeatedly loading in large key and value tensors. Empirically, we have also found that due to this memory-bandwidth issue, the per-step inference latency for a conformer encoder on cloud TPUs is almost 6 times as large as an LSTM encoder with a similar size. Even though conformers allow batching across time-frames to improve computational efficiency, because the 1st-pass of the cascaded conformer encoder is streaming, batching across too many frames can increase user-perceived latency. In this work, we seek a solution to improve computational latency for both edge and cloud TPUs.

To address the aforementioned issues, we change our cascaded encoder so that the causal 1st-pass is small (50M parameters) and the non-causal 2nd-pass is large (100M parameters). Having a smaller 1st-pass reduces the number of internal states significantly, which runs without time batching to maintain low user-perceived streaming latency. Adding more capacity to the 2nd-pass, which can be batched across longer 1-second chunks, further improves the issues surrounding memory-bandwidth and latency. Finally, we explore only completing 2nd-pass recognition if we are not confident in the output from the 1st-pass, based on a confidence model [13], which results in additional sentence-end latency savings.

Various previous work has looked at improving the computational efficiency of transformer or conformer [14, 15, 16, 17, 18]. Much of this work looks at different methods to reduce the number of operation to compute self-attention. For example, [17] use so-called grouped attention, which reduces attention complexity by reshaping its input and stacking neighboring frames together into the depth dimension. Many of these papers target a much smaller model (10-13M parameters), compared to the 100M conformer model in this work. In addition, they are able to reduce latency by 10-20%, for example [14], which is much smaller then the 6X latency difference between LSTM and conformer we have observed. Furthermore, the importance of displaying words to the screen as soon as the model can emit them is important for our application. In contrast, [18] looks to improve self-attention computation and memory usage by chunking the utterance into segments and parallelizing computation in these segments. Since we are looking for a much bigger latency improvement in the context of streaming constraints, we explore reducing size of the first pass significantly and allowing a larger 2nd-pass which does not suffer as much from conformer memory-bandwith and latency issues.

The proposed small 1st/large 2nd pass model allows for quality improvements as well. Since self-supervised techniques like Wav2Vec models have shown to do better on non-streaming models compared to streaming models [19, 20, 21], by adding more capacity to the 2nd-pass – which is non-causal with 900 msec of right context – we achieve further quality improvements while still fitting into a streaming production setting. We also improve MWER [22] training of the cascaded encoder model by turning off SpecAug [23] when generating the N-best list from the model during MWER training to more closely match the inference setting.

We conduct our experiments on a large scale Voice Search task. Our baseline model is a large 1st pass and small 2nd pass cascaded encoder, presented in [9]. The proposed smaller 1st pass model degrades WER by 10% relative while improving per-step latency on cloud TPUs by ∼3.5x. Adding more capacity to the 2nd pass improves quality such that it is on par with [9], while improving the overall latency of the 1st and 2nd pass by approximately 1.7x.

2. MODELING IMPROVEMENTS

2.1. Latency Improvements

2.1.1. Smaller, Faster 1st Pass

First, we explore making the causal 1st-pass conformer encoder smaller. A smaller 1st-pass implies fewer internal states that have to be maintained during streaming, which translates to reduced latency. To make the model smaller, instead of using 12 conformer layers, we reduce the number of layers to 7. In addition, since the output of the feed-forward module of the conformer does not have to be saved across time-steps and is computationally cheaper, we increase each layer's feed forward dimension by a factor of 2. This partly addresses quality loss due to smaller 1st-pass.

Our conformer encoder uses a stacking layer after its 2nd conformer block, which downsamples the input by a factor of 2 [24]. The first 2 conformer blocks, therefore, operate at twice the rate as the remaining layers and are the most computationally expensive layers during inference. Hypothesizing that the low-level features learned by the first N blocks can be captured by simpler layers, we remove the self-attention layers from them, relying just on the convolution and feed-forward operations. Removing self-attention reduces the number of internal states to be propagated between time-steps, and also reduces the number of computations. Unless stated otherwise, we use $N = 3$, which has given the best balance between speed, size and accuracy.

It is expected that a smaller 1st-pass causes a small degradation in 1st-pass WER. As discussed next, if we add enough capacity to the 2nd-pass model so that it can recoup this loss, we can still have a significant reduction in overall latency.

2.1.2. Cascaded Encoder - Small 1st Pass, Large 2nd Pass

The *Cascaded Encoder* model [1] is shown in Figure 1. It consists of 2 encoders – the causal 1st-pass and the non-causal 2nd-pass encoder – connected in cascade. The latter takes in both left-context and an additional 900 msec of right context of the 1st-pass encoder's outputs. A single RNN-T decoder is shared between the 1st and 2nd-passes for better long-form performance, smaller model size and on-device benefits. The decoder is separately run for both passes.

The output of the 1st-pass model is used to emit words to the screen as quickly as possible, in order to provide a low-latency streaming experience. The outputs from the 1st pass conformer encoder, therefore, must be computed at every frame. However, the

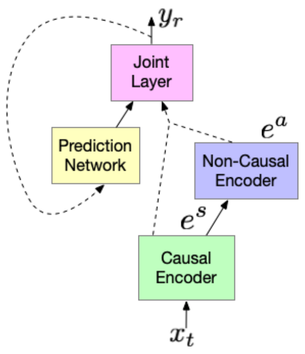

Fig. 1. The cascaded encoder for joint modeling of two passes.

2nd-pass conformer is delayed by 900 msec and does not have to emit words as quickly. This allows us to batch the computation of the 2nd-pass conformer across multiple frames, utilizing the TPU more efficiently. More importantly, since multiple output frames are computed in parallel, the total number of internal states that have to be propagated for an utterance is also smaller, addressing memory-bandwidth issues [12]. Section 4 shows how computing the non-causal conformer over 1 second chunks improves TPU latency.

In [9], the causal encoder was roughly 100M parameters, while the non-causal 2nd-pass encoder was roughly 50M parameters. As discussed in Section 2.1.1, reducing capacity of the 1st-pass to 50M parameters improves latency but at the cost of quality. Since the non-causal conformer can be batched across multiple time frames to improve latency, we look to add capacity to the 2nd-pass and make it 100M parameters to make up for the 1st-pass quality regression.

2.1.3. Confidence Model

Following [9], the 1st-pass model is responsible for generating the streaming result and the endpointing decision [25], after which the final result is updated to the 2nd-pass output once it finishes. To further improve latency, we explore stopping the 2nd-pass when we are confident of the 1st-pass hypothesis at the end of the utterance, since the 2nd-pass operates 900 msec behind the 1st-pass. Specifically, for 'easy' utterances (e.g., acoustically clean, linguistically simple), using the 1st-pass alone achieves good WER. To improve latency, the system should only wait for 2nd-pass recognition on the difficult utterances. To perform this system combination, a word-level confidence estimation module [13] is trained and the output scores are averaged to form an utterance-level confidence. The overall system trusts the 1st-pass output when the confidence is higher than a preset threshold, and only waits for the 2nd-pass recognition otherwise.

2.2. Quality Improvements

2.2.1. Wav2Vec

One issue with the end-to-end models is their quality on long-tail words [9]. Since the model learns a direct mapping from the audio to the text tokens, words composed of a rare sequence of tokens in the training data often receive lower probability and are therefore more likely to be misrecognized. One potential solution for the issue is to encourage the encoder to learn a generalized audio representation and train the model with a larger amount of unsupervised data. Recently, wav2vec 2.0 [19] has been shown to learn an effective discrete representation of the audio and incorporate untranscribed utterances to improve the quality of the downstream ASR

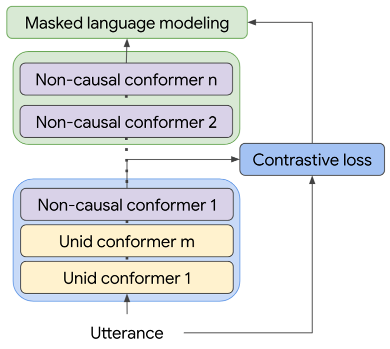

Fig. 2. Applying w2v-BERT on the cascaded encoder. The approach utilizes the non-causal layers from the non-causal encoder to provide past and future context for both the contrastive learning and the masked language modeling.

system. A following work, w2v-BERT [21], further improves the pre-training effectiveness by incorporating a masked language modeling loss. These approaches, however, utilize a BERT-style pre-training and are designed for non-causal models. This fundamental property makes it challenging to achieve similar improvements on streaming models as those observed in non-causal models.

The architecture of the cascaded encoder allows us to better utilize the benefit of the BERT-style pre-training and reduce the quality gap between streaming and non-streaming models. In particular, with the non-causal components in the cascaded encoder, the encoder has access to the future context, which is critical for the BERT-style pre-training. With the benefit of the non-causal component, we incorporate the w2v-BERT approach with the cascaded encoder and utilize it to improve the quality on long-tail words. The architecture of w2v-BERT with the cascaded encoder is shown in Figure 2.

2.2.2. *MWER Training*

Our model is first trained with the RNN-T loss to convergence, and is then fine-tuned for a small number of steps with the MWER training criterion [22], following our previous work [26]. We use SpecAug [23] as a data augmentation strategy during training. However, SpecAug is typically turned off during inference. In order to better match this setting during MWER training, we generate the N-best hypotheses required during training by turning off SpecAug. Note that SpecAug is still applied to the input features before they are passed through the encoders for the final loss computation. These modifications allow the model to get the benefits of SpecAug-based data augmentation, while allowing MWER to learn from N-best hypotheses that more closely capture what happens during inference. As in prior work, we sample paths from both causal and non-causal decoders during MWER training, taking care to ensure that hypotheses are generated from the same path that is used to compute the loss: e.g., if an utterance is fed to the computational path for the non-causal decoder – $x_t \rightarrow e^s \rightarrow e^a \rightarrow y_r$ (see Fig. 1) – for generating beam search hypotheses, then the loss is back-propagated through the same path (rather than $x_t \rightarrow e^s \rightarrow y_r$).

3. EXPERIMENTAL SETTINGS

3.1. Datasets

Similar to [26], all E2E models are trained on ∼300M multidomain audio-text utterance pairs [27]. Utterances from all domains are anonymized and hand-transcribed, except for YouTube where the transcriptions are obtained in a semi-supervised fashion [28]. In addition to the diverse training sets, multi-condition training (MTR) [29], random data down-sampling to 8kHz [30] and SpecAug [23] are used to further increase data diversity.

The *Search* test set includes around 12K Voice Search utterances with an average length of 5.5 seconds. They are anonymized and hand-transcribed, and are representative of Google's Voice Search traffic. To measure performance on long-tail words, we create a synthetic *Rare-word* test set, described in more detail in [31]. Specifically, we look for words in a 100B language model text corpora that are rare in the multi-domain training data (occurring less than 5 times). Around 1k utterances are synthesized [32] across the News domain to create the final test-set.

3.2. Modeling

The cascaded encoder model architecture is very similar to [33, 9]. Specifically, all models are trained on 400k hours of data. A 128D log-mel feature frontend with a 16-D one-hot domain-id vector appended to it [27] is passed to the first encoder layer. All layers use causal convolution; causal conformer layers use limited left-context attention to strictly restrict the model to use no future inputs; non-causal conformer encoder additionally uses limited right context. 8-head attention is used in the self-attention layer and the convolution kernel size used is 15. The baseline model consists of 12 causal conformer layers, and 5 non-causal conformer layers that process a total of 900 milliseconds of speech into the future. Both causal and non-causal layers feeds into the shared RNN-T decoder. Unless otherwise indicated, all cascaded encoder models in this paper are trained with FastEmit [34] in both passes, which is a change from [1, 33, 9] which only apply FastEmit to the causal 1st pass.

The RNN-T decoder consists of a prediction network and a joint network with a single feed-forward layer with 640 units. The embedding prediction network [10] uses an embedding dimension of 320, and has 9M parameters. All E2E models are trained with the HAT factorization to predict 4,096 word pieces [35]. The cascaded encoder with embedding decoder has 150M parameters in total.

Following [4], the LSTM encoder baseline has 8 layers, with each layer having 2,048 units and a projection layer with 640 units.

The confidence model is built around one transformer decoder block consisting of one self-attention layer attending to the causal decoder's hypothesized tokens' embeddings, one cross-attention layer attending to the causal encoder outputs, and two feedforward layers. The transformer has 8 heads, each with 40-dimensional outputs. Accordingly, the inputs are first projected down to 320-dimensional vectors. The outputs are processed with a final linear layer followed by a sigmoid layer, and trained to make the binary prediction of whether each hypothesized word is correct with the word-level loss in [13]. The model has 3M parameters in total.

4. RESULTS

4.1. Reducing 1st-pass Latency

Table 1 shows the WER and the per-frame encoder TPU computational latency of different 1st-pass models. The latency issue with conformer vs. LSTM is highlighted by comparing $B0$ and $B1$. Our goal here is to improve the latency of the conformer encoder ($B1$). First, we make the encoder smaller, through optimizations discussed in Section 2.1.1 to get a large latency reduction ($E0$). Replacing the bottom 3 conformer layers with convolution ($E1$) reduces latency down to 5.1ms. Overall, with the proposed optimizations we can reduce conformer latency by 3.5X compared to $B1$. As expected, reducing the model size comes with a quality hit, which we address in the next section.

Table 1. WER and Latency of 1st-pass models

Exp	Model	WER	Cloud TPU latency (ms)	Model Size (M)
B0	LSTM Enc.	6.7	2.4	120
B1	Conformer Enc.	6.5	12.5	120
E0	Smaller Conf. Enc.	7.3	8.0	62
E1	E0 + 3 Conv	7.5	5.1	56

4.2. Smaller 1st-pass, Larger 2nd-pass

To make up for the quality degradation in the 1st pass, we explore adding more capacity to the 2nd-pass non-causal conformer. The benefit of the 2nd-pass is that because it is delayed by 900 msec, we can batch all frames together when computing the non-causal layers. Since this minimizes the overhead of maintaining internal states in the conformer, latency is significantly reduced as shown for both 2nd-pass models in Table 2. Notice that from $E3$, by adding more capacity to the 2nd-pass, the final WER is on par with the smaller 2nd-pass in $E2$. In addition, the overall cloud TPU per-frame latency of both passes is around 8.0 msec (5.1 + 2.9), which has a 1.7X speedup compared to $E2$. We also measure the E2E median real-time factor (RTF, measured as the ratio between computation time and speech audio time) for the whole ASR system on the Pixel 6 phone Edge TPU, since cloud and on-device have different senarios and constraints, as is explained in Section 1. $E3$ also shows a 1.5X speedup compared to $E2$ on E2E RTF. Thus, by having a smaller 1st and larger 2nd-pass ($E3$), we can maintain quality and improve latency over larger 1st and smaller 2nd ($E2$).

Table 2. WER and Latency of 1st+2nd-pass models

Exp	Model	Model size (M)	WER	Cloud TPU latency (ms)	Edge TPU E2E RTF
E2	1st-pass	100M	7.4	12.5	0.30
	2nd-pass	50M	5.9	1.3	
E3	1st-pass	56M	7.7	5.1	0.20
	2nd-pass	100M	5.8	2.9	

4.3. Wav2Vec

Another benefit of small 1st and large 2nd-pass is that wav2vec pretraining can have more benefits. Table 3 shows the WER on both Search and Long-tail sets for both large-1st/small-2nd ($E2$) and small 1st/large-2nd ($E3$). Notice that WER degrades with wav2vec for $E2$ but improves for $E3$ on the Rare-word test set. This confirms our hypothesis that wav2vec should work better when more capacity is added to non-causal conformer layers.

Table 3. wav2vec Experiments

Exp	Test Set	No wav2vec	wav2vec
E2	Search	5.9	6.5
E3	Search	5.8	5.8
	Rare-word	9.7	9.5

4.4. MWER Improvements

Table 4 shows the difference in performance obtained with MWER training when generating N-best hypotheses with/without SpecAug as described in Section 2.2.2. Generating N-best hypotheses without SpecAug ($E5$) during MWER improves performance relative to $E4$, which decodes with SpecAug, for both 1st- and 2nd-passes.

Table 4. WER after MWER Training

Exp	Model	1st-pass WER	2nd-pass WER
E3	no MWER	7.7	5.8
E4	MWER, SpecAug	7.3	5.6
E5	MWER, no SpecAug	7.0	5.5

4.5. E2E Final 2nd-Pass Latency

Finally, we show the improvement on the E2E final 2nd-pass latency in Table 5, which is measured from the end-of-utterance time determined by the 1st-pass E2E endpointer [25], to the time we finalize the recognition result. It is benchmarked on a Pixel 6 phone with Edge TPU on 100 utterances that are representative of voice search and Assistant usage; we report the median latency. $E6$ is exactly the cascaded encoder model in [9], which does not turn on FastEmit [34] in the 2nd-pass. By comparing $E5$ vs. $E6$, we demonstrate that FastEmit training, especially on the 2nd-pass, is important for reducing the final latency (by 31ms), although it comes with a slight WER regression (4% relative). In $E7$, we apply a confidence model on top of E5 to conditionally stop waiting for the 2nd-pass to update the final result based on the confidence score of the 1st-pass hypothesis, as is described in Section 2.1.3. We end up having 26% of the utterances trusting the 1st-pass result based on a confidence threshold we tune to balance between quality and latency. It leads to an additional 22ms reduction to the median final latency with only a tiny WER regression (2% relative). With both improvements, we are able to meet our goal to maintain the median latency within 200ms to minimize the user-perceived latency at the end before query execution.

Table 5. WER and Latency for edge TPU

Exp	Model	2nd-pass WER	E2E 2nd-pass final latency (ms)
E6	E5 without FastEmit	5.3	239
E5	Small 1st/Large 2nd + MWER	5.5	208
E7	E5 + confidence	5.6	186

5. CONCLUSION

In this paper, we propose a cascaded encoder architecture with small 1st-pass causal layers running in a streaming fashion and large 2nd-pass non-causal layers batching multiple frames over time, so that we can improve latency of our previous cascaded encoder architecture by up to 1.7X, but still maintain low streaming latency and high quality. Using FastEmit on both 1st-pass and 2nd-pass, and by using a confidence model to decide whether or not to wait for the 2nd-pass to finish, we reduce overall latency to under 200 msec. Finally, by using wav2vec pretraining and improving MWER training, we can further improve the quality of cascaded encoders.

6. REFERENCES

[1] A. Narayanan, T. N. Sainath, R. Pang, et al., "Cascaded encoders for unifying streaming and non-streaming ASR," in *Proc. ICASSP*, 2021.

[2] G. Pundak and T. N. Sainath, "Lower frame rate neural network acoustic models," in *Proc. Interspeech*, 2016.

[3] J. Li, Y. Wu, Y. Gaur, et al., "On the Comparison of Popular End-to-End Models for Large Scale Speech Recognition," in *Proc. Interspeech*, 2020.

[4] Y. He, T. N. Sainath, R. Prabhavalkar, et al., "Streaming End-to-end Speech Recognition For Mobile Devices," in *Proc. ICASSP*, 2019.

[5] C.-C. Chiu, T. N. Sainath, Y. Wu, et al., "State-of-the-art Speech Recognition With Sequence-to-Sequence Models," in *Proc. ICASSP*, 2018.

[6] S. Kim, T. Hori, and S. Watanabe, "Joint CTC-attention based end-to-end speech recognition using multi-task learning," in *Proc. ICASSP*, 2017.

[7] J. Li, R. Zhao, H. Hu, and Y. Gong, "Improving RNN transducer modeling for end-to-end speech recognition," in *Proc. ASRU*, 2019.

[8] A. Zeyer, A. Merboldt, R. Schlüter, and H. Ney, "A new training pipeline for an improved neural transducer," in *Proc. Interspeech*, 2020.

[9] T. N. Sainath, Y. He, A. Narayanan, et al., "An Efficient Streaming Non-Recurrent On-Device End-to-End Model with Improvements to Rare-Word Modeling," in *Proc. of Interspeech*, 2021.

[10] R. Botros, T.N. Sainath, Robert David, Emmanuel Guzman, Wei Li, and Yanzhang He, "Tied & reduced rnn-t decoder," in *Proc. Interspeech*, 2021.

[11] Norman P Jouppi, Cliff Young, Nishant Patil, David Patterson, Gaurav Agrawal, Raminder Bajwa, Sarah Bates, Suresh Bhatia, Nan Boden, Al Borchers, et al., "In-datacenter performance analysis of a tensor processing unit," in *Proc. international symposium on computer architecture*, 2017.

[12] Noam Shazeer, "Fast transformer decoding: One write-head is all you need," *arXiv preprint arXiv:1911.02150*, 2019.

[13] David Qiu, Qiujia Li, Yanzhang He, Yu Zhang, Bo Li, Liangliang Cao, Rohit Prabhavalkar, et al., "Learning word-level confidence for subword end-to-end asr," in *Proc. ICASSP*, 2021.

[14] Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang, "Fastformer: Additive attention can be all you need," *arXiv preprint arXiv:2108.09084*, 2021.

[15] Peidong Wang and DeLiang Wang, "Efficient end-to-end speech recognition using performers in conformers," *arXiv preprint arXiv:2011.04196*, 2020.

[16] K. Choromanski, V. Likhosherstov, D. Dohan, X. Song, A. Gane, T. Sarlós, P. Hawkins, J. Quincy Davis, A. Mohiuddin, L. Kaiser, D. B. Belanger, L. J. Colwell, and A. Weller, "Rethinking attention with performers," in *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*, 2021.

[17] Maxime Burchi and Valentin Vielzeuf, "Efficient conformer: Progressive downsampling and grouped attention for automatic speech recognition," *arXiv preprint arXiv:2109.01163*, 2021.

[18] Y. Shi, Y. Wang, C. Wu, C. Yeh, J. Chan, F. Zhang, D. Le, and M. Seltzer, "Emformer: Efficient Memory Transformer Based Acoustic Model for Low Latency Streaming Speech Recognition," in *Proc. ICASSP*, 2021.

[19] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," in *NeurIPS*, 2020.

[20] Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli, "wav2vec: Unsupervised pre-training for speech recognition," in *Interspeech*, 2019.

[21] Y. Chung, Y. Zhang, W. Han, C.C. Chiu, J. Qin, R. Pang, and Y. Wu, "w2v-bert: Combining contrastive learning and masked language modeling for self-supervised speech pre-training," in *ASRU*, 2021.

[22] R. Prabhavalkar, T. N. Sainath, Y. Wu, P. Nguyen, Z. Chen, C. C. Chiu, and A. Kannan, "Minimum Word Error Rate Training for Attention-based Sequence-to-sequence Models," in *Proc. ICASSP*, 2018.

[23] D. S. Park, W. Chan, Y. Zhang, C. Chiu, B. Zoph, E.D. Cubuk, and Q.V. Le, "SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition," in *Proc. Interspeech*, 2019.

[24] A. Gulati, J. Qin, C.-C. Chiu, et al., "Conformer: Convolution-augmented Transformer for Speech Recognition," in *Proc. Interspeech*, 2020.

[25] B. Li, S.-Y. Chang, T. N. Sainath, et al., "Towards fast and accurate streaming end-to-end ASR," in *Proc. ICASSP*, 2020.

[26] T. N. Sainath, Y. He, B. Li, et al., "A Streaming On-Device End-To-End Model Surpassing Server-Side Conventional Model Quality and Latency," in *Proc. ICASSP*, 2020.

[27] A. Narayanan, R. Prabhavalkar, C.-C. Chiu, et al., "Recognizing Long-Form Speech Using Streaming End-to-End Models," in *Proc. ASRU*, 2019.

[28] H. Liao, E. McDermott, and A. Senior, "Large Scale Deep Neural Network Acoustic Modeling with Semi-supervised Training Data for YouTube Video Transcription," in *Proc. ASRU*, 2013.

[29] C. Kim, A. Misra, K. Chin, 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.

[30] J. Li, D. Yu, J. Huang, and Y. Gong, "Improving Wideband Speech Rcognition using Mixed-bandwidth Training Data in CD-DNN-HMM," in *Proc. SLT*, 2012.

[31] T.N. Sainath J. Apfel R. Pang S. Kumar C. Peyser, S. Mavandadi, "Improving Tail Performance of a Deliberation E2E ASR Model Using a Large Text Corpus," in *Proc.Interspeech*, 2020.

[32] X. Gonzalvo, S. Tazari, C. Chan, M. Becker, A. Gutkin, and H. Silen, "Recent Advances in Google Real-time HMM-driven Unit Selection Synthesizer," in *Proc. Interspeech*, 2016.

[33] B. Li, A. Gulati, J. Yu, et al., "A Better and Faster End-to-End Model for Streaming ASR," in *Proc. ICASSP*, 2021.

[34] J. Yu, C.-C. Chiu, B. Li, et al., "FastEmit: Low-latency Streaming ASR with Sequence-level Emission Regularization," in *Proc. ICASSP*, 2021.

[35] M. Schuster and K. Nakajima, "Japanese and Korean voice search," in *Proc. ICASSP*, 2012.