INTERSPEECH 2020
October 25–29, 2020, Shanghai, China

Conv-Transformer Transducer: Low Latency, Low Frame Rate, Streamable End-to-End Speech Recognition

Wenyong Huang, Wenchao Hu, Yu Ting Yeung, Xiao Chen

Huawei Noah's Ark Lab

{wenyong.huang, huwenchao, yeung.yu.ting, chen.xiao2}@huawei.com

Abstract

Transformer has achieved competitive performance against state-of-the-art end-to-end models in automatic speech recognition (ASR), and requires significantly less training time than RNN-based models. The original Transformer, with encoder-decoder architecture, is only suitable for offline ASR. It relies on an attention mechanism to learn alignments, and encodes input audio bidirectionally. The high computation cost of Transformer decoding also limits its use in production streaming systems. To make Transformer suitable for streaming ASR, we explore Transducer framework as a streamable way to learn alignments. For audio encoding, we apply unidirectional Transformer with interleaved convolution layers. The interleaved convolution layers are used for modeling future context which is important to performance. To reduce computation cost, we gradually downsample acoustic input, also with the interleaved convolution layers. Moreover, we limit the length of history context in self-attention to maintain constant computation cost for each decoding step. We show that this architecture, named Conv-Transformer Transducer, achieves competitive performance on LibriSpeech dataset (3.6% WER on test-clean) without external language models. The performance is comparable to previously published streamable Transformer Transducer and strong hybrid streaming ASR systems, and is achieved with smaller look-ahead window (140 ms), fewer parameters and lower frame rate.

Index Terms: speech recognition, Transformer Transducer, end-to-end, RNN-T

1. Introduction

Transformer [1] models have achieved state-of-the-art results in many natural language processing tasks [2, 3]. Recently, Transformer is gaining popularity in speech recognition research community [4–7]. Transformer-based speech recognition models have achieved comparable or better performance than state-of-the-art models, which are mostly based on recurrent neural network (RNN). Transformer is more parallelizable than RNN, thus is significantly faster to train [1, 7].

Despite its success, the original Transformer with encoder-decoder architecture is only suitable for offline speech recognition. There are a number of challenges to apply the original Transformer for streaming speech recognition. First, the original Transformer relies on an attention mechanism over full encoder output to learn alignments between input and output sequences [8]. Second, the original Transformer encodes input audio in a bidirectional way, thus requires a full utterance as input. Moreover, computation of Transformer increases in quadratic complexity with the length of input sequences. Naive implementation usually leads to much slower decoding than RNN-based models [7].

For alignment learning, there exist many streamable ways for speech recognition, e.g., Connectionist Temporal Classification (CTC) [9], Transducer [10], Monotonic Chunkwise Attention (MoChA) [11], Triggered Attention [12]. All of these alignment learning methods can be combined with Transformer. In this work, we focus on Transducer, as previous works [13,14] suggest that Transducer models outperform traditional hybrid models for streaming speech recognition in production settings. Several research groups have been working on combining Transformer with Transducer for speech recognition, which is usually referred to as Transformer Transducer [15–17].

To make Transformer encoder streamable, previous works of Transformer Transducer [16, 17] tried to limit future context (right-context) in self-attention for input audio encoding. Although each layer only requires a little future context, the overall future context aggregates over all Transformer layers. The overall look-ahead window is still large. In this work, we explore using unidirectional Transformer with interleaved convolution layers for audio encoding. Unidirectional Transformer requires no future context. Thus it is streamable and does not introduce latency into the model. However, future context is important to speech recognition performance. Therefore, we add interleaved convolutions between Transformer layers to model future context. We carefully control number of layers and filter size of convolution layers. In our model, the look-ahead window introduced by convolution layers is 140 ms.

We explore a number of options for practical streaming speech recognition with Transformer. First, we apply interleaved convolutions to gradually downsample input audio sequence. This enables our model to run in a low frame rate of 80 ms, thus reduces computation cost significantly [18]. Second, we limit the length of history context (left-context) of self-attention in Transformer layers to maintain constant computation cost for each decoding step. Finally, we apply relative position encoding [3], which enables hidden state reuse for Transformer. Hidden state reuse improves decoding speed of Transformer significantly [3].

We name our design as Conv-Transformer Transducer. We show that this architecture achieves competitive performance on LibriSpeech dataset [19], with word error rate (WER) of 3.5% in test-clean and 8.3% in test-other without external language models. When we further limit history context of self-attention, our proposed model still achieves WER of 3.6% and 8.9% respectively. The performance is comparable to previously published streamable Transformer-Transducer [17], and strong hybrid speech recognition systems, with smaller look-ahead window, fewer parameters and lower frame rate.

This paper is organized as follows. In the next Section, we introduce the Transducer framework and detailed architecture of Conv-Transformer Transducer. In Section 3, we present our experimental results on LibriSpeech dataset. Finally, we conclude our work in Section 4.

http://dx.doi.org/10.21437/Interspeech.2020-2361

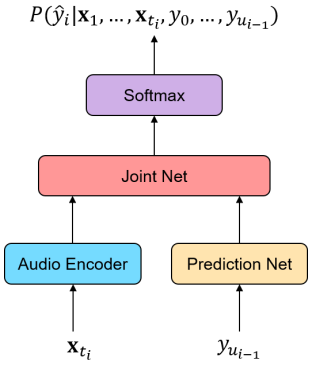

Figure 1: *Transducer model architecture.*

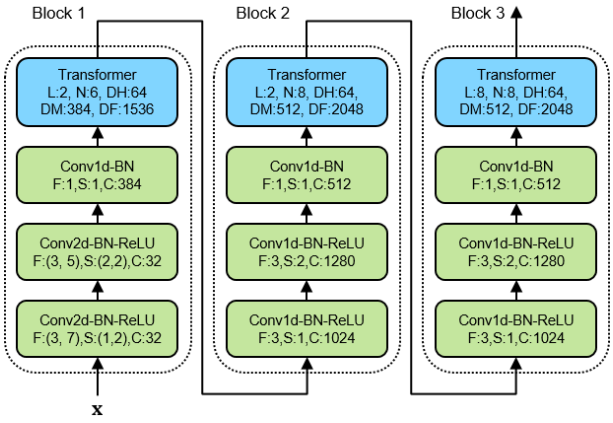

Figure 2: *Audio encoder of Conv-Transformer Transducer:* ***ConvND-BN-RELU*** *denotes N-Dimensional convolution layers followed by batch normalization and Rectified Linear Unit (ReLU) activation. For convolution layers,* ***F*** *denotes number of filters,* ***S*** *denotes the value of stride,* ***C*** *denotes number of output channels. For 2D convolutions, the first values of* ***F*** *and* ***S*** *correspond to time dimension, the second values correspond to frequency dimension. For Transformer,* ***L*** *denotes number of layers,* ***N*** *denotes number of heads,* ***DH*** *denotes dimension of each head,* ***DM*** *denotes input and output dimension of self-attention layers and feed-forward network (FFN) layers,* ***DF*** *denotes dimension of intermediate hidden layer in the FFN.*

2. Conv-Transformer Transducer

2.1. Transducer

Transducer, proposed in [10], is a neural sequence transduction framework which can be used for end-to-end speech recognition, i.e., transduction of an input acoustic sequence into an output transcription label sequence.

We denote an input acoustic sequence with T frames as $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_T)$, and a transcription label sequence of length U as $\mathbf{y} = (y_1, \ldots, y_U)$, where $y_u \in \mathcal{Z}$ and $\mathcal{Z}$ is vocabulary of output labels. As depicted in Figure 1, a Transducer model first encodes an input acoustic sequence with an audio encoder network, producing a sequence of encoder states denoted as $\mathbf{h} = (\mathbf{h}_1, \ldots, \mathbf{h}_T)$. For each encoder state $\mathbf{h}_t$, the model first predicts either a label or a special blank symbol $\langle \mathtt{b} \rangle$ with a joint net. When the model predicts a label, the model continues to predict the next output. When the model predicts a blank symbol, which indicates that no more labels can be predicted, the model proceeds to the next encoder state. This process is similar to CTC [9], but there is a difference. A Transducer model makes use of previously predicted non-blank labels as input condition to predict the next output. The previously predicted labels are encoded with another network referred to as prediction net.

During the process described above, the Transducer model defines a conditional distribution,

$$P(\hat{\mathbf{y}}|\mathbf{x}) = \prod_{i=1}^{T+U} P(\hat{y}_i|\mathbf{x}_1, \cdots, \mathbf{x}_{t_i}, y_0, \ldots, y_{u_{i-1}}) \quad (1)$$

where $\hat{\mathbf{y}} = (\hat{y}_1, \ldots, \hat{y}_{T+U}) \subset \{\mathcal{Z} \cup \langle \mathtt{b} \rangle\}^{T+U}$ is an alignment path with T blank symbols and U labels such that removing all blank symbols in $\hat{\mathbf{y}}$ yields $\mathbf{y}$, and y_0 represents start of sentence. To obtain probability of target sequence, Transducer computes a marginalized distribution,

$$P(\mathbf{y}|\mathbf{x}) = \sum_{\hat{y} \in \mathcal{A}_{\text{ALIGN}}(\mathbf{x}, \mathbf{y})} P(\hat{\mathbf{y}}|\mathbf{x}) \quad (2)$$

where $\mathcal{A}_{ALIGN}(\mathbf{x}, \mathbf{y})$ is the set containing all valid alignment paths such that removing the blank symbols in $\hat{\mathbf{y}}$ yields $\mathbf{y}$. The summation of probabilities of all alignment paths is computed efficiently with forward-backward algorithm [10].

In most of previous works in Transducer framework, audio encoder and prediction net are composed of RNN [10]. Such type of Transducers are often referred to as Recurrent Neural Network Transducer (RNN-T). Other types of networks are allowed in Transducer model.

2.2. Architecture of Conv-Transformer Transducer

We now describe the architecture of our Conv-Transformer Transducer. The audio encoder is illustrated in Figure 2, which is composed of three blocks. Each block is composed of three convolution layers followed by unidirectional Transformer. Unidirectional Transformer is similar to encoder of the original Transformer, except that self-attention is masked, preventing the self-attention from attending to subsequent positions beyond current position. Unidirectional Transformer does not require future context, and is widely used in language modeling [3, 20] and text generation [21].

However, future acoustic context is helpful to improve recognition accuracy. Therefore, we apply convolution layers before Transformer in each block. In our model, all future context comes from convolution layers. The size of look-ahead window required by current input frame is illustrated in Figure 3. In our 3-block setting, the size of look-ahead window is 140 ms, which is acceptable in streaming speech recognition. We also downsample input representation with the convolution layers. As shown in Figure 2, at the second convolution layer of each block, the stride is 2 along time dimension. With an input frame rate of 10 ms, the frame rate becomes 20 ms in the first block, 40 ms in the second block, and finally 80 ms in the third block, as shown in Figure 3. We find that this progressive downsampling scheme causes no loss in accuracy, and reduces memory requirement for training significantly. We can use bigger batch size for efficient training. Moreover, downsampling greatly reduces computation cost of inference. To further reduce computation complexity of audio encoder, we place most of Transformer layers in the third block with 80 ms frame rate. The idea of using interleaved convolution for gradual downsampling and incorporating future context is inspired by [22].

The prediction net is illustrated in Figure 4. An embedding layer converts previously predicted non-blank labels into vec-

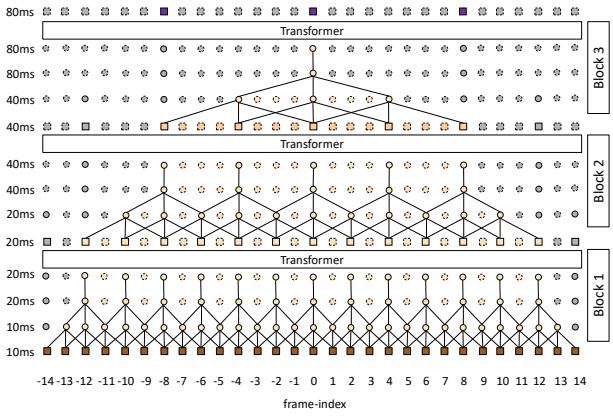

Figure 3: *Illustration of context window and frame rate change of convolution layers in audio encoder of Conv-Transformer Transducer at current input frame (frame-index = 0). Input and output of each block are represented by squares. Convolution layers are represented by circles. Only time dimension is shown. For each layer, activation with dashed outline is skipped in computation.*

tor representations. Then a linear layer projects the embedding vectors to match input dimension of unidirectional Transformer layers.

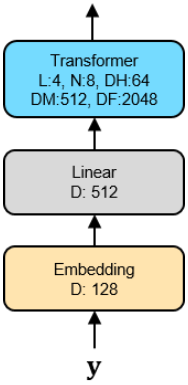

Figure 4: *Prediction net of Conv-Transformer Transducer. We denote the output dimensions of embedding layer and linear layer as* ***D****.*

The joint net is a fully-connected feed-forward neural network with single hidden layer. There are 512 units in the hidden layer, with Rectified Linear Unit (ReLU) as activation function. We concatenate outputs of audio encoder and prediction net as input of joint net.

2.3. Limiting history context of self-attention

During decoding, self-attention of unidirectional Transformer attends to all history context. Computation cost continues to grow with the length of history context as decoding goes on, which is undesirable for a streaming system. We limit history context of self-attention with a fixed-size window. Computational cost of each step becomes constant. The idea of limiting history context of self-attention for streamable speech recognition has been explored in [16, 17, 23].

2.4. Relative position encoding

We apply relative position encoding to model sequential order in self-attention [3]. Relative position encoding is proposed in [24], and leads to significant improvement over absolute position encoding in the original Transformer. Moreover, relative position encoding is necessary for reusing hidden states when applying self-attention with limited attention context [3].

3. Experiments

We perform our experiments with LibriSpeech dataset [19], which is a publicly available English read speech corpus of audiobooks from the LibriVox project. The corpus consists of 960-hour training data, 10.7 hours of development data, and 10.5 hours of test data. The audio is sampled at 16 kHz and quantized at 16 bit. The development and test data are further divided into `clean` and `other` subsets according to the quality, assessed with a speech recognition system. We train our models with the entire training set. For clarity, we only report the results of test sets (test-clean and test-other), with the best-performed decoding configuration in the development sets.

3.1. Training setup

All the Conv-Transformer Transducer models are trained using 8 GPUs with per-GPU batch bucketing configuration as shown in Table 1. We use NovoGrad [25] as our optimizer. The learning rate is warmed up from 0 to 0.01 in the first 10k steps, then is decayed to 5×10^{-6} in the following 200k steps polynomially.

We use 128-dimensional log-mel filterbank as acoustic feature, calculated with 20 ms window and 10 ms stride. We use 4k subwords as output target units, which are generated from training transcripts of LibriSpeech using SentencePiece [26]. We apply several techniques to avoid overfitting. We add additive Gaussian noise and apply speed perturbation [27] in time domain. After feature extraction, we apply SpecAugment as described in [28]. We do not apply time-warping for SpecAugment as we have already applied speed perturbation. Transformer layers are regularized with 10% dropout. Due to limited computation resources, we choose hyperparameters without intensive tuning.

Table 1: *Batch bucketing configuration*

SampleLength	BatchSize
$0 \sim 7s$	22
$7 \sim 12s$	18
$12 \sim 14s$	18

3.2. Baseline conventional streamable model

Our baseline system is based on Kaldi's [29] chain-model recipe [30]. The TDNN-LSTM based acoustic model (AM) [22] is trained with LibriSpeech, plus additional CommonVoice (CV) data (`en_1488h_2019-12-10`, validated set excluding dev and test sets) [31]. The AM is monophone-based, and contains 33.5M parameters. The look-ahead window size is 170 ms, which is comparable to our Conv-Transformer Transducer. We apply SpecAugment during AM training. We perform language model (LM) training with normalized training text of the official LibriSpeech LM [32], plus transcription of CV training data. Vocabulary size of lexicon is about 200k. We deploy a two-pass decoding strategy, with a smaller 4-gram LM with 5.8M n-grams in first pass. We utilize a bigger 4-gram LM with

191M n-grams for second-pass rescoring. Note that we train the baseline system with additional CommonVoice data.

3.3. Results

We compare the results of our Conv-Transformer Transducer (ConvT-T) with the results of a previously published streamable Transformer Transducer model (T-T) [17], a hybrid model with Transformer AM [23] and our hybrid TDNN-LSTM AM baseline. We use beam search without an external language model for Conv-Transformer Transducer decoding. As shown in Table 2, Conv-Transformer Transducer achieves the lowest WER on both test-clean and test-other. Our model operates with fewer parameters, smaller look-ahead window, and lower frame rate. We do not list the number of parameters of the hybrid models as they both rely on large external n-gram language models.

We also evaluate if low frame rate of 80 ms leads to performance degradation. We train a model with 40 ms frame rate (HFR ConvT-T), by setting the stride to 1 instead of 2 in the second convolution layer of the third block. As this model requires more memory, we train the 40-ms model with only half of the original batch size, but with twice more steps to maintain the same number of epochs. As shown in Table 2, the performance difference between the 40-ms model and the 80-ms model is insignificant.

Table 2: *WER comparison with previously published streamable Transformer Transducer model, hybrid streamable model with a Transformer-based AM, our conventional hybrid baseline model and our streamable Conv-Transformer Transducer models on LibriSpeech test sets.*

Model	Params	Look ahead	Frame rate	WER (%) clean	WER (%) other
Hybrid TDNN-LSTM	-	170ms	30ms	4.0	8.9
Hybrid Transformer [23]	-	2480ms	20ms	3.65	9.01
T-T [17]	139M	1080ms	30ms	3.6	10.0
ConvT-T (Ours)	67M	140ms	**80ms**	**3.5**	8.3
HFR ConvT-T (Ours)	67M	140ms	40ms	**3.5**	**8.2**

We then perform experiments to analyze effects of varying the number of Transformer layers over different blocks of various frame rates. The results are shown in Table 3. Recall that in the original model, there are 2 layers of Transformer in the first (20 ms) block, 2 layers of Transformer in the second (40 ms) block and 8 layers of Transformer in the third (80 ms) block. Moving Transformer layers from the third to the second block does not affect recognition performance significantly, but increases computation cost as more Transformer layers operate at higher frame rate. However, when we move all the Transformer layers in the second block to the third block as shown in Row 4 of Table 3, small performance degradation begins. We further move all the Transformer layers in the first two blocks to the third block as shown in Row 5. This is equivalent to running a convolutional neural network (CNN) followed by multi-layer Transformer. There is further performance degradation. Note that there are fewer parameters for the Transformer layers in the first block. We reduce total number of Transformer layers from 12 to 11 to maintain similar number of parameters. The experimental results support our strategies of placing most of Transformer layers in the third block with the lowest frame rate, and applying interleaved convolution in the model.

We further evaluate the effects of limiting the amount of history context (left-context) in self-attention of Transformer layers. The results are shown in Table 4. The size of attention window in prediction net does not have a significant impact on recognition performance of test-clean, even when the window size is only 2. With an attention window size of 16, we get the same accuracy as the original model. A bigger window helps to improve accuracy of the more challenging test-other slightly. For the evaluation of audio encoder, we fix the window size of self-attention to 16 in prediction net. Unlike prediction net, a smaller window in self-attention of audio encoder leads to more significant performance degradation. With self-attention window size of 32 in audio encoder, although there is performance degradation, overall results are still competitive to those of hybrid models and Transformer Transducer.

Table 3: *WER of different number of Transformer layers over the three blocks in audio encoder. Layer distribution B1-B2-B3 corresponds to number of Transformer layers in the first, the second and the third block respectively*

Layer distribution	WER (%) clean	WER (%) other
2-2-8	3.5	**8.3**
2-4-6	3.5	**8.3**
2-6-4	**3.4**	8.4
2-0-10	3.6	8.5
0-0-11	3.7	8.9

Table 4: *WER of different window sizes of history context (left-context) in self-attention of Transformer layers. Inf means that all history context is used.*

Left-context window size audio encoder	Left-context window size prediction net	WER (%) clean	WER (%) other
Inf	Inf	3.5	**8.3**
Inf	2	3.5	8.6
Inf	8	3.6	8.5
Inf	16	**3.4**	**8.3**
8	16	3.8	9.3
16	16	3.6	9.1
32	16	3.6	8.9

4. Conclusions

We propose Conv-Transformer Transducer model, which is suitable for streaming speech recognition. Our model is end-to-end trainable using Transducer framework. By applying unidirectional Transformer with interleaved convolution for audio encoding, our model requires only a 140 ms look-ahead window. A smaller look-ahead window generally leads to lower latency. We gradually downsample audio input into frame rate of 80 ms to reduce computation cost. Recognition accuracy is still maintained. Experimental results also support our strategies of applying interleaved convolution, and placing most of Transformer layers in the block with the lowest frame rate. We also limit the size of history context window in self-attention of Transformer layers. Although there is performance degradation, the results are still comparable to published LibriSpeech results. Our model achieves competitive results to previously published streamable Transformer transducer model, and state-of-the-art hybrid models, with lower latency, lower frame rate and fewer parameters.

5. References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, pp. 5998–6008.

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota, Jun. 2019, pp. 4171–4186.

[3] Z. Dai, Z. Yang, Y. Yang, J. G. Carbonell, Q. Le, and R. Salakhutdinov, "Transformer-XL: Attentive language models beyond a fixed-length context," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, Jul. 2019, pp. 2978–2988.

[4] L. Dong, S. Xu, and B. Xu, "Speech-transformer: A no-recurrence sequence-to-sequence model for speech recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2018)*, 2018, pp. 5884–5888.

[5] S. Karita, N. E. Y. Soplin, S. Watanabe, M. Delcroix, A. Ogawa, and T. Nakatani, "Improving transformer-based end-to-end speech recognition with connectionist temporal classification and language model integration," in *Interspeech 2019*, 2019, pp. 1408–1412.

[6] A. Zeyer, P. Bahar, K. Irie, R. Schlüter, and H. Ney, "A comparison of transformer and lstm encoder decoder models for asr," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2019, pp. 8–15.

[7] S. Karita, N. Chen, T. Hayashi, T. Hori, H. Inaguma, Z. Jiang, M. Someki, N. E. Y. Soplin, R. Yamamoto, X. Wang, S. Watanabe, T. Yoshimura, and W. Zhang, "A comparative study on transformer vs RNN in speech applications," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2019, pp. 449–456.

[8] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *International Conference on Learning Representations 2015 (ICLR 2015)*, 2015, arXiv preprint arXiv:1409.0473.

[9] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd International Conference on Machine learning (ICML)*, 2006, pp. 369–376.

[10] A. Graves, "Sequence transduction with recurrent neural networks," in *International Conference of Machine Learning 2012 (ICML) Workshop on Representation Learning*, 2012, arXiv preprint arXiv:1211.3711.

[11] C.-C. Chiu and C. Raffel, "Monotonic chunkwise attention," in *International Conference on Learning Representations 2018 (ICLR 2018)*, 2018, arXiv preprint arXiv:1712.05382.

[12] N. Moritz, T. Hori, and J. Le Roux, "Triggered attention for end-to-end speech recognition," in *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2019)*, 2019, pp. 5666–5670.

[13] Y. He, T. N. Sainath, R. Prabhavalkar, I. McGraw, R. Alvarez, D. Zhao, D. Rybach, A. Kannan, Y. Wu, R. Pang *et al.*, "Streaming end-to-end speech recognition for mobile devices," in *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2019)*, 2019, pp. 6381–6385.

[14] T. N. Sainath, Y. He, B. Li, A. Narayanan, R. Pang, A. Bruguier, S. Chang, W. Li, R. Alvarez, Z. Chen, C. Chiu, D. Garcia, A. Gruenstein, K. Hu, A. Kannan, Q. Liang, I. McGraw, C. Peyser, R. Prabhavalkar, G. Pundak, D. Rybach, Y. Shangguan, Y. Sheth, T. Strohman, M. Visontai, Y. Wu, Y. Zhang, and D. Zhao, "A streaming on-device end-to-end model surpassing server-side conventional model quality and latency," in *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2020)*, 2020, pp. 6059–6063.

[15] Z. Tian, J. Yi, J. Tao, Y. Bai, and Z. Wen, "Self-attention transducers for end-to-end speech recognition," in *Interspeech 2019*, 2019, pp. 4395–4399.

[16] C.-F. Yeh, J. Mahadeokar, K. Kalgaonkar, Y. Wang, D. Le, M. Jain, K. Schubert, C. Fuegen, and M. L. Seltzer, "Transformer-Transducer: End-to-end speech recognition with self-attention," *arXiv preprint arXiv:1910.12977*, 2019.

[17] Q. Zhang, H. Lu, H. Sak, A. Tripathi, E. McDermott, S. Koo, and S. Kumar, "Transformer Transducer: A streamable speech recognition model with transformer encoders and RNN-T loss," in *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2020)*, 2020, pp. 7829–7833.

[18] G. Pundak and T. N. Sainath, "Lower frame rate neural network acoustic models," in *Interspeech 2016*, 2016, pp. 22–26.

[19] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2015)*. IEEE, 2015, pp. 5206–5210.

[20] R. Al-Rfou, D. Choe, N. Constant, M. Guo, and L. Jones, "Character-level language modeling with deeper self-attention," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 3159–3166.

[21] P. J. Liu, M. Saleh, E. Pot, B. Goodrich, R. Sepassi, L. Kaiser, and N. Shazeer, "Generating Wikipedia by summarizing long sequences," *arXiv preprint arXiv:1801.10198*, 2018.

[22] V. Peddinti, Y. Wang, D. Povey, and S. Khudanpur, "Low latency acoustic modeling using temporal convolution and LSTMs," *IEEE Signal Processing Letters*, vol. 25, no. 3, pp. 373–377, 2017.

[23] Y. Wang, A. Mohamed, D. Le, C. Liu, A. Xiao, J. Mahadeokar, H. Huang, A. Tjandra, X. Zhang, F. Zhang, C. Fuegen, G. Zweig, and M. L. Seltzer, "Transformer-based acoustic modeling for hybrid speech recognition," in *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2020)*, 2020, pp. 6874–6878.

[24] P. Shaw, J. Uszkoreit, and A. Vaswani, "Self-attention with relative position representations," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, New Orleans, Louisiana, Jun. 2018, pp. 464–468.

[25] B. Ginsburg, P. Castonguay, O. Hrinchuk, O. Kuchaiev, V. Lavrukhin, R. Leary, J. Li, H. Nguyen, and J. M. Cohen, "Stochastic gradient methods with layer-wise adaptive moments for training of deep networks," *arXiv preprint arXiv:1905.11286*, 2019.

[26] T. Kudo and J. Richardson, "SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Brussels, Belgium, Nov. 2018, pp. 66–71.

[27] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," in *Interspeech 2015*, 2015, pp. 3586–3589.

[28] D. S. Park, Y. Zhang, C. Chiu, Y. Chen, B. Li, W. Chan, Q. V. Le, and Y. Wu, "SpecAugment on large scale datasets," in *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2020)*, 2020, pp. 6879–6883.

[29] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The Kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2011.

[30] D. Povey, V. Peddinti, D. Galvez, P. Ghahremani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur, "Purely sequence-trained neural networks for ASR based on lattice-free MMI," in *Interspeech 2016*, 2016, pp. 2751–2755.

[31] "Common Voice," https://voice.mozilla.org, accessed: 2020-05-15.

[32] "LibriSpeech language models, vocabulary and G2P models," http://www.openslr.org/11/, accessed: 2020-05-15.