On the Comparison of Popular End-to-End Models for Large Scale Speech Recognition

Jinyu Li[1], *Yu Wu*[2], *Yashesh Gaur*[1], *Chengyi Wang*[2], *Rui Zhao*[1], *Shujie Liu*[2]

[1]Microsoft Speech and Language Group
[2]Microsoft Research Asia

{jinyli, wu.yu, yagaur, v-chengw, ruzhao, shujiliu}@microsoft.com

Abstract

Recently, there has been a strong push to transition from hybrid models to end-to-end (E2E) models for automatic speech recognition. Currently, there are three promising E2E methods: recurrent neural network transducer (RNN-T), RNN attention-based encoder-decoder (AED), and Transformer-AED. In this study, we conduct an empirical comparison of RNN-T, RNN-AED, and Transformer-AED models, in both non-streaming and streaming modes. We use 65 thousand hours of Microsoft anonymized training data to train these models. As E2E models are more data hungry, it is better to compare their effectiveness with large amount of training data. To the best of our knowledge, no such comprehensive study has been conducted yet. We show that although AED models are stronger than RNN-T in the non-streaming mode, RNN-T is very competitive in streaming mode if its encoder can be properly initialized. Among all three E2E models, transformer-AED achieved the best accuracy in both streaming and non-streaming mode. We show that both streaming RNN-T and transformer-AED models can obtain better accuracy than a highly-optimized hybrid model.

Index Terms: end-to-end, RNN-transducer, attention-based encoder-decoder, transformer

1. Introduction

Recently, the speech community is seeing a significant trend of moving from deep neural network based hybrid modeling [1] to end-to-end (E2E) modeling [2, 3, 4, 5, 6, 7, 8, 9, 10] for automatic speech recognition (ASR). While hybrid models require disjoint optimization of separate constituent models such as acoustic and language model, E2E ASR systems directly translate an input speech sequence into an output token (subwords, or even words) sequence using a single network.

Some widely used contemporary E2E approaches for sequence-to-sequence transduction are: (a) Connectionist Temporal Classification (CTC) [11, 12], (b) recurrent neural network Transducer (RNN-T)[13], and (c) Attention-based Encoder-Decoder (AED) [14, 15, 3]. Among these three approaches, CTC was the earliest and can map the input speech signal to target labels without requiring any external alignments. However, it also suffers from the conditional frame-independence assumption. RNN-T extends CTC modeling by changing the objective function and the model architecture to remove the frame-independence assumption. Because of its streaming nature, RNN-T has received a lot of attention for industrial applications and has also managed to replace traditional hybrid models for some cases [9, 16, 17, 18].

AED is a general family of models that was initially proposed for machine translation [19] but has shown success in other domains (including ASR [14, 15, 3]) as well. These models are not streaming in nature by default but there are several studies towards that direction, such as monotonic chunkwise attention [20] and triggered attention [21]. The early AED models used RNNs as a building block for its the encoder and decoder modules. We refer to them as RNN-AED in this study. More recently, the transformer architecture with self attention [22] has also become prevalent and is being used as a fundamental building block for encoder and decoder modules [23, 24, 25]. We refer to such a model as Transformer-AED in this paper.

Given the fast evolving landscape of E2E technology, it is timely to compare the most popular and promising E2E technologies for ASR in the field, shaping the future research direction. This paper focuses on the comparison of current most promising E2E technologies, namely RNN-T, RNN-AED, and Transformer-AED, in both non-streaming and streaming modes. All models are trained with 65 thousand hours of Microsoft anonymized training data. As E2E models are data hungry, it is better to compare its power with such a large amount of training data. To our best knowledge, there is no such a detailed comparison. In a recent work [16], the streaming RNN-T model was compared with the non-streaming RNN-AED. In [26], streaming RNN-AED is compared with streaming RNN-T for long-form speech recognition. In [25], RNN-AED and Transformer-AED are compared in a non-streaming mode, with training data up to 960 hours. As the industrial applications usually requires the ASR service in a streaming mode, we further put more efforts on how to develop these E2E models in a streaming mode. While it has been shown in [27] that combining RNN-T and RNN-AED in a two-pass decoding configuration can surpass an industry-grade state-of-the-art hybrid model, this study shows that a single streaming E2E model, either RNN-T or Transformer-AED, can also surpass a state-of-the-art hybrid model [28, 29].

In addition to performing a detailed comparison of these promising E2E models for the first time, other contributions of this paper are 1) We propose a multi-layer context modeling scheme to explore future context with significant gains; 2) The cross entropy (CE) initialization is shown to be much more effective than CTC initialization to boost RNN-T models; 3) For streaming Transformer-AED, we show chunk-based future context integration is more effective than the lookahead method; 4) We release our Transformer related code with reproducible results on Librispeech at [30] to facilitate future research.

2. Popular End-to-End Models

In this section, we give a brief introduction of current popular E2E models: RNN-T, RNN-AED, and Transformer-AED. These models have an acoustic encoder that generates high level representation for speech and a decoder, which autoregressively generates output tokens in the linguistic domain. While the acoustic encoders can be same, the decoders of RNN-T and

AED are different. In RNN-T, the generation of next label is only conditioned on the label outputs at previous steps while the decoder of AED conditions the next output on acoustics as well. More importantly, RNN-T works in a frame-synchronized way while AED works in a label-synchronized fashion.

2.1. RNN transducer

The encoder network converts the acoustic feature $x_{1:T}$ into a high-level representation $h^{enc}_{1:T}$. The decoder, called prediction network, produces a high-level representation h^{pre}_{u} by consuming previous non-blank target y_{u-1}. Here u denotes output label index. The joint network is a feed-forward network that combines the encoder network output h^{enc}_{t} and the prediction network output h^{pre}_{u} to generate the joint matrix $h_{t,u}$, which is used to calculate softmax output. Here t denotes time index.

The encoder and prediction networks are usually realized using RNN with LSTM [31] units. When the encoder is a uni-directional LSTM-RNN as Eq. (1), RNN-T works in streaming mode by default.

$$h^{enc}_{t} = LSTM(x_t, h^{enc}_{t-1}) \tag{1}$$

However, when the underlying LSTM-RNN encoder is a bi-directional model as Eq. (2), it is a non-streaming E2E model.

$$h^{enc}_{t} = [LSTM(x_t, h^{enc}_{t-1}), LSTM(x_t, h^{enc}_{t+1})] \tag{2}$$

When implemented with LSTM-RNN, the prediction network formulation is

$$h^{pre}_{u} = LSTM(y_{u-1}, h^{pre}_{u-1}). \tag{3}$$

With the advantage of Transformer models, there is a recent work to replace the LSTM-RNN in the encoder with the Transformer model to construct Transformer transducer [32] and Conformer transducer [33].

2.2. Attention-based Encoder-Decoder

While RNN-T has received more attention from the industry due to its streaming nature, the Attention-based Encoder-Decoder (AED) models attracts more research from academia because of its powerful attention structure. RNN-AED and Transformer-AED differ at the realization of encoder and decoder by using LSTM-RNN and Transformer, respectively.

2.2.1. RNN-AED

The encoder of RNN-AED can have the same structure as RNN-T like Eq. (1) and Eq. (2). However, the attention-enhanced decoder operates differently as below:

$$h^{dec}_{u} = LSTM(c_u, y_{u-1}, h^{dec}_{u-1}). \tag{4}$$

here c_u is the context vector obtained by weighted combination of the encoder output. c_u is supposed to contain the acoustic information necessary to emit the next token. It is calculated using the help of the attention mechanism [14, 34].

2.2.2. Transformer-AED

Even though RNNs can capture long term dependencies, Transformer [22] based models can do it more effectively given the attention mechanism sees all context directly. Specifically, the encoder is composed of a stack of Transformer blocks, where each block has a multi-head self-attention layer and a feed-forward layer. Suppose that the input of a Transformer block can be linearly transformed to $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$. Then, the output of a multi-head self-attention layer is

$$\text{Multihead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = [\mathbf{H_1} \dots \mathbf{H}_{d_{head}}]\mathbf{W}^{head} \tag{5}$$
$$\text{where } \mathbf{H_i} = \text{softmax}(\frac{\mathbf{Q_i K_i^T}}{\sqrt{d_k}})\mathbf{V_i},$$
$$\mathbf{Q_i} = \mathbf{Q}W^{Q_i}, \mathbf{K_i} = \mathbf{K}W^{K_i}, \mathbf{V_i} = \mathbf{V}W^{V_i}.$$

Here d_{head} is the number of attention heads and d_k is the dimension of the feature vector for each head. This output is fed to the feed-forward layer. Residual connections [35] and layer normalization (LN) [36] are indispensable when we connect different layers and blocks. In addition to the two layers in an encoder block, the Transformer decoder also has an additional third layer that performs multi-head attention over the output of the encoder. This is similar to the attention mechanism in RNN-AED.

3. Our Models

3.1. Model building block

The encoder and decoder of E2E models are constructed as the stack of multiple building blocks described in this section. For the models using LSTM-RNN, we explore two structures. The first one, LSTM_cuDNN, directly calls Nvidia cuDNN library [37] for the LSTM implementation. We build every block by concatenating a cuDNN LSTM layer, a linear projection layer to reduce model size, and then followed by LN. Calling Nvidia cuDNN implementation enables us for fast experiment of comparing different models.

The second structure, LSTM_Custom, puts LN and projection layer inside LSTM, as it was indicated in [9] that they are important for better RNN-T model training. Hence, we only use this structure for RNN-T by customizing the LSTM function. The detailed formulations are in [17]. However, this slows down the model training speed by 50%.

For the Transformer-AED models, we remove the position embedding part [38] and use a VGG-like convolution module [39] to pre-process the speech feature $x_{1:T}$ before the Transformer blocks. The LN is put before multi-head attention layer (Pre-LN), which makes the gradients well-behaved at the early stage in training.

3.2. Non-streaming models

We achieve non-streaming behavior in RNN-T by adding bidirectionality in the encoder. The encoder of this RNN-T is composed of multiple blocks of bi-directional LSTM_cuDNN as described in Section 3.1. The prediction network is realized with multiple uni-directional blocks of LSTM_cuDNN.

Similar to RNN-T, the non-streaming RNN-AED investigated in this study also uses multiple blocks of bi-directional LSTM_cuDNN in the encoder and uni-directional LSTM_cuDNN in the decoder. This decoder works together with a location-aware softmax attention [15]. No multi-task training or joint-decoding with CTC is used for RNN-AED.

Following [25], the Transformer-AED model uses the multi-task training and the joint decoding of CTC/attention. The training objective function is

$$\mathcal{L} = -\alpha \log p_{ctc}(y|x_{1:T}) - (1-\alpha) \log p_{att}(y|x_{1:T}). \tag{6}$$

The log-likelihood of the next subword $\log p(y_u|x_{1:t}, y_{1:u})$ in the joint decoding is formulated as

$$\log p_{ctc}(y_u|x_{1:t}, y_{1:u}) + \beta_1 \log p_{att}(y_u|x_{1:t}, y_{1:u}). \tag{7}$$

In practice, we first use the attention model to select top-k candidates and then re-rank them with Eq. 7.

3.3. Streaming models

Streaming RNN-T model has a uni-directional encoder. While we can directly incorporate a standard LSTM as the building block with either LSTM_cuDNN or LSTM_Custom as described in Section 3.1, incorporating the future context into encoder structure can significantly improve the ASR accuracy, as shown in [17]. However, different from [17] which explores future context frames together with the layer trajectory structure, in this study we propose to only use context modeling. We do this to save model parameters. Future context is modelled using the simple equation below.

$$\zeta_t^l = \sum_{\delta=0}^{\tau} q_\delta^l \odot g_{t+\delta}^l . \tag{8}$$

Because $\odot$ is element-wise product, Eq. (8) only increases the number of model parameters very slightly. It transfers a lower layer vector g_t^l together with its future vectors $g_{t+\delta}^l$ into a new vector ζ_t^l, where δ is future frame index. We modify the block of LSTM_cuDNN or LSTM_Custom with the context modeling.

- LSTM_cuDNN_Context: the block is constructed with a Nvidia cuDNN LSTM layer, followed by a linear projection layer, then the context modeling layer, and finally a LN layer.
- LSTM_Custom_Context: the block is constructed with the layer normalized LSTM layer with projection, and then followed by the context modeling layer.

A similar concept of context modeling was applied to RNN in [40] as Lookahead convolution layer. However, it was only applied to the top layer of a multi-layer RNN. In contrast, in this study we apply context modeling to every block of LSTM_cuDNN or LSTM_Custom, and also investigate its effectiveness in the context of E2E modeling. For RNN-T, we also investigate initializing the encoder with either CTC [6] or CE training [41].

RNN-AED models use blocks of LSTM_cuDNN_Context as encoder. Experiments with LSTM_Custom_Context will be a part of future study. The streaming mechanism we have chosen for this study is Monotonic Chunkwise Attention (MoChA) [42]. MoChA consists of a monotonic attention mechanism [43] which scans the encoder output in a left to right order and selects a particular encoder state when it decides to trigger the decoder. This selection probability is selected by sampling from a parameterized Bernoulli random variable. Once a trigger point is detected, MoChA also uses an additional lookback window and applies a regular softmax attention over that. Note that we have a sampling operation here, which precludes the use of standard backpropagation. Therefore we train with respect to the expected values of the context vectors. Please refer to [42] for more details.

To enable streaming scenario in Transformer-AED models, we borrow the idea in trigger-attention (TA) [21], where the CTC conducts frame-synchronized decoding to select top-k candidates for each frame and then the attention model is leveraged to jointly re-rank the candidates using Eq. 7 once a new subword is triggered by the CTC. Since the Transformer encoder is deeper than LSTM, the lookahead method may not be the best solution. We compare the chunk-based method and the lookahead-based method. The former segments the entire input into several fixed-length chunks and then feeds them into the model chunk by chunk, while the latter is exactly the same with the method in RNN-T and RNN-AED. For the chunk-based encoder, the decoder can see the end of a chunk. For the lookahead based encoder, we set a fixed window size for decoder.

Table 1: *Average WER of all non-streaming E2E models on 13 test sets containing 1.8 M words.*

non-streaming models	WER
RNN-T (cuDNN)	9.25
RNN-AED (cuDNN)	8.05
Transformer-AED	7.83

4. Experiments

In this section, we evaluate the effectiveness of all models by training them with 65 thousand (K) hours of transcribed Microsoft data. The test sets cover 13 application scenarios such as Cortana and far-field speech, containing a total of 1.8 million (M) words. We report the word error rate (WER) averaged over all test scenarios. All the training and test data are anonymized with personally identifiable information removed.

For fair comparison, all E2E models built for this study have around 87 M parameters. The input feature is 80-dimension log Mel filter bank with a stride of 10 milliseconds (ms). Three of them are stacked together to form a 240-dimension super-frame. This is fed to the encoder networks for RNN-T and RNN-AED, while Transformer-AED directly consumes the 10 ms feature. All E2E models use the same 4 K word piece units as the output target.

4.1. Non-streaming E2E models

As described in Section 3.1, the non-streaming RNN-T model uses bi-directional LSTM with Nvidia cuDNN library in its encoder. The LSTM memory cell size is 780. The LSTM outputs from the forward and backward direction are concatenated with the total dimension of 1560 then linearly projected to dimension 780, followed by a LN layer. There are total 6 stacked blocks of such operation. The prediction network has 2 stacked blocks, each of which contains a uni-directional cuDNN LSTM with memory cell size of 1280, followed by a linear projection layer to reduce the dimension to 640, and then with a LN layer.

The non-streaming RNN-AED model uses exactly the same encoder and decoder structures as the non-streaming RNN-T model. Similar to [34], a location-aware attention mechanism is used. In addition to the encoder and decoder hidden states, this mechanism also takes alignments from previous decoder step as inputs. The attention dimension is 512.

The Transformer-AED model has 18 Transformer blocks in encoder and 6 Transformer blocks in decoder. Before Transformer blocks in encoder, we use a 4 layers VGG network to pre-process the speech feature with total stride 4. The number of attention head is 8 and the attention dimension of each head is 64. The dimension of the feed-forward layer is 2048 in Transformer blocks. The combination weights of joint training and decoding (i.e. α, β) are both 0.3.

As shown in Table 1, the non-streaming AED models have a clear advantage over the non-streaming RNN-T model due to the power of attention modeling. Transformer-AED improves RNN-AED by 2.7% relative WER reduction.

4.2. Surpassing hybrid model with streaming E2E models

In [28] we reported results from our best hybrid model called the contextual layer trajectory LSTM (cltLSTM) [29]. The cltLSTM was trained with a three-stage optimization process. This model was able to obtain a 16.2% relative WER reduction over the CE baseline. Introducing 24 frames of total future-context further yields an 18.7% relative WER reduction. The encode latency is only 480 ms (24*20ms=480 ms; stride-per-frame is 20 ms due to frame skipping [44]). Hence, this cltLSTM model (Table 2) presents a very challenging streaming hybrid model to beat. This model has 65 M parameters, and is decoded with 5 gigabytes 5gram decoding graph.

We list the results for all streaming E2E models in Table 2. The baseline RNN-T implementation uses unidirectional cuDNN LSTMs in both the encoder and the decoder. The encoder has 6 stacked blocks of LSTM_cuDNN. Each block has a unidirectional cuDNN LSTM with 1280 memory cells which projected to 640 dimension and followed by LN. The prediction and the joint network is the same as in the non-streaming RNN-T model. This RNN-T model obtains 12.16% test WER. The second RNN-T model inserts the context modeling layer (Eq. (8)) after the linear projection layer in each block. The context modeling has 4 frames lookahead at each block, and therefore the encoder has $4 * 6 = 24$ frames lookahead. Because the frame shift is 30 ms, the total encoder lookahead is 720ms. The lookahead brings great WER improvement, obtaining 10.65% WER. This is 12.4% relative WER reduction from the first RNN-T model without any lookahead. We also followed lookahead convolution proposed in [40] by using 24 frames lookahead only on the top most RNN block. This model gives 11.19% WER, showing that our proposed context modeling, which allocates lookahead frames equally at each block, is better than lookahead convolution [40], which simply puts all lookahead frames on the top layer only.

Next, we look at the impact of encoder initialization for RNN-T. Shown in Table 2, the CTC initialization of RNN-T encoder doesn't help too much while the CE initialization significantly reduces WER to 9.80. This is 8.0% relative WER reduction from the randomly initialized model. The CTC initialization makes the encoder emit token spikes together with lots of blanks while CE initialization enables the encoder to learn time alignment. Given the gain with CE initialization, we believe the encoder of RNN-T functions more like an acoustic model in the hybrid model. Note the CE pre-training needs time alignments, which is hard to get for word piece units as many of them don't have phoneme realisation. However, the time alignment for words is still accurate. We make an approximation and obtain alignments for a word piece by simply segmenting the duration of its word equally into its constituent word pieces.

For the last RNN-T model, we put projection layer and LN inside the LSTM cell (Custom_LSTM), and then insert the context modeling layer after it. Putting projection layer inside allows us to use larger number of memory cells while keeping similar model size as the cuDNN_LSTM setup. This LSTM has 2048 memory cells and the project layer reduces the output size to 640. This model finally gives 9.27% WER, which is slightly better than our best hybrid model.

With the same encoder architecture as the cuDNN RNN-T, the MoChA-based streaming RNN-AED model gives impressive results. Unlike RNN-T, it does not need any initialization and is still able to slightly outperform it in an apple-to-apple comparison (9.61% vs 9.80%). To the best of our knowledge, this is the first time a streaming RNN-AED has outperformed RNN-T on a large scale task. Note that our previous study didn't observe accuracy improvement for RNN-AED with CE initialization [45]. We will investigate whether RNN-AED can also benefit from customized LSTM function in future study.

Table 2: *Average WERs of streaming models on 13 test sets containing 1.8 M words.*

streaming models	WER	encoder lookahead
hybrid		
cltLSTM	9.34	480 ms
RNN-T		
cuDNN	12.16	0 ms
cuDNN+Context	10.65	720 ms
cuDNN+convolution [40]	11.19	720 ms
cuDNN+Context+CTC init.	10.62	720 ms
cuDNN+Context+CE init.	9.80	720 ms
Custom+Context+CE init.	9.27	720 ms
RNN-AED		
cuDNN+Context	9.61	720 ms
Transformer-AED		
Lookahead method	10.26	720 ms
Chunk-based method	9.16	720 ms

The architecture of the streaming Transformer-AED model is the same as the non-streaming one. For lookahead context-modeling method, each encoder block looks ahead 1 frame. Considering the total stride of VGG is 4 and the speech sampling rate is 10ms, the encoder has $1 * 18 * 4 * 10ms = 720ms$ latency. The decoder of the lookahead method introduces an extra 240ms latency. The chunk-based method considers future context with a fixed-chunk. The latency of each frame is in the range of [480ms, 960ms], resulting in a 720ms averaged latency without extra decoder latency. The chunk-based method obtains 9.16% WER, significantly outperforming the lookahead method, mainly because the bottom Transformer blocks of the lookahead approach cannot enjoy the full advantages provided by the right context.

5. Conclusions

This work presents the first large-scale comparative study of three popular E2E models (RNN-T, RNN-AED, and Transformer-AED). The models are compared in both streaming and non-streaming modes. All models are trained with 65K hours of Microsoft's internal anonymized data. We observe that with the same encoder structure, AED is better than RNN-T for both non-streaming and streaming models. With customized LSTM and CE initialization for encoder, the RNN-T model becomes better than RNN-AED. Among all models, Transformer-AED obtained the best WERs in both streaming and non-streaming modes.

In this study, both streaming RNN-T and Transformer-AED outperformed a highly-optimized hybrid model. There are several significant factors contributing to this success. For streaming RNN-T, the proposed context modeling reduces the WER by 12.4% relative from the one without any lookahead. The CE initialization for RNN-T improves over the random initialization baseline by 8.0% relative WER reduction. This shows pretraining is helpful even on a large scale task. To utilize future context for streaming Transformer-AED, we show that the chunk-based method is better than the lookahead method by 10.7% relative.

6. References

[1] G. Hinton, L. Deng, D. Yu *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[2] Y. Miao, M. Gowayyed, and F. Metze, "EESEN: End-to-end speech recognition using deep RNN models and WFST-based decoding," in *Proc. ASRU*. IEEE, 2015, pp. 167–174.

[3] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *Proc. ICASSP*, 2016, pp. 4960–4964.

[4] 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, pp. 939–943.

[5] E. Battenberg, J. Chen, R. Child, A. Coates, Y. G. Y. Li, H. Liu, S. Satheesh, A. Sriram, and Z. Zhu, "Exploring neural transducers for end-to-end speech recognition," in *Proc. ASRU*. IEEE, 2017, pp. 206–213.

[6] K. Rao, H. Sak, and R. Prabhavalkar, "Exploring architectures, data and units for streaming end-to-end speech recognition with RNN-transducer," in *Proc. ASRU*, 2017.

[7] C.-C. Chiu, T. N. Sainath, Y. Wu, R. Prabhavalkar, P. Nguyen, Z. Chen, A. Kannan, R. J. Weiss, K. Rao, K. Gonina *et al.*, "State-of-the-art speech recognition with sequence-to-sequence models," in *Proc. ICASSP*, 2018.

[8] J. Li, G. Ye, A. Das, R. Zhao, and Y. Gong, "Advancing acoustic-to-word CTC model," in *Proc. ICASSP*, 2018.

[9] 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 *Proc. ICASSP*, 2019, pp. 6381–6385.

[10] J. Li, , R. Zhao, Z. Meng *et al.*, "Developing RNN-T models surpassing high-performance hybrid models with customization capability," in *Proc. Interspeech*, 2020.

[11] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of ICML*. ACM, 2006, pp. 369–376.

[12] A. Graves and N. Jaitley, "Towards end-to-end speech recognition with recurrent neural networks," in *PMLR*, 2014, pp. 1764–1772.

[13] A. Graves, "Sequence transduction with recurrent neural networks," *CoRR*, vol. abs/1211.3711, 2012.

[14] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[15] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *NIPS*, 2015, pp. 577–585.

[16] T. Sainath, R. Pang, and et. al., "Two-pass end-to-end speech recognition," in *Proc. Interspeech*, 2019.

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

[18] M. Jain, K. Schubert, J. Mahadeokar *et al.*, "RNN-T for latency controlled ASR with improved beam search," *arXiv preprint arXiv:1911.01629*, 2019.

[19] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[20] C.-C. Chiu and C. Raffel, "Monotonic chunkwise attention," *arXiv preprint arXiv:1712.05382*, 2017.

[21] N. Moritz, T. Hori, and J. Le Roux, "Triggered attention for end-to-end speech recognition," in *Proc. ICASSP*, 2019, pp. 5666–5670.

[22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, pp. 6000–6010.

[23] L. Dong, S. Xu, and B. Xu, "Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition," in *Proc. ICASSP*, 2018, pp. 5884–5888.

[24] S. Zhou, L. Dong, S. Xu, and B. Xu, "Syllable-based sequence-to-sequence speech recognition with the transformer in Mandarin Chinese," in *Proc. Interspeech*, 2018.

[25] S. Karita, N. Chen, T. Hayashi *et al.*, "A comparative study on transformer vs RNN in speech applications," in *Proc. ASRU*, 2019.

[26] C.-C. Chiu, W. Han, Y. Zhang *et al.*, "A comparison of end-to-end models for long-form speech recognition," in *Proc. ASRU*, 2019.

[27] 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, pp. 6059–6063.

[28] J. Li, R. Zhao, E. Sun, J. H. Wong, A. Das, Z. Meng, and Y. Gong, "High-accuracy and low-latency speech recognition with two-head contextual layer trajectory LSTM model," in *Proc. ICASSP*, 2020.

[29] J. Li, L. Lu, C. Liu, and Y. Gong, "Improving layer trajectory LSTM with future context frames," in *Proc. ICASSP*, 2019, pp. 6550–6554.

[30] C. Wang, *Streaming Transformer*, 2020. [Online]. Available: https://github.com/cywang97/StreamingTransformer

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

[32] Q. Zhang, H. Lu, H. Sak *et al.*, "Transformer transducer: A streamable speech recognition model with transformer encoders and RNN-T loss," in *Proc. ICASSP*, 2020.

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

[34] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, "End-to-end attention-based large vocabulary speech recognition," in *Proc. ICASSP*. IEEE, 2016, pp. 4945–4949.

[35] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *arXiv preprint arXiv:1512.03385*, 2015.

[36] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[37] S. Chetlur, C. Woolley, P. Vandermersch, J. Cohen, J. Tran, B. Catanzaro, and E. Shelhamer, "cuDNN: Efficient primitives for deep learning," *arXiv preprint arXiv:1410.0759*, 2014.

[38] C. Wang, Y. Wu, Y. Du, J. Li, S. Liu, L. Lu, S. Ren, G. Ye, S. Zhao, and M. Zhou, "Semantic mask for transformer based end-to-end speech recognition," in *Proc. Interspeech*, 2020.

[39] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. ICLR*, 2015.

[40] C. Wang, D. Yogatama, A. Coates, T. Han, A. Hannun, and B. Xiao, "Lookahead convolution layer for unidirectional recurrent neural networks," in *Proc. ICLR Workshop*, 2016.

[41] H. Hu, R. Zhao, J. Li, L. Lu, and Y. Gong, "Exploring pre-training with alignments for RNN transducer based end-to-end speech recognition," in *Proc. ICASSP*, 2020.

[42] C.-C. Chiu* and C. Raffel*, "Monotonic chunkwise attention," in *International Conference on Learning Representations*, 2018.

[43] C. Raffel, D. Eck, P. Liu, R. J. Weiss, and T. Luong, "Online and linear-time attention by enforcing monotonic alignments," in *Thirty-fourth International Conference on Machine Learning*, 2017.

[44] Y. Miao, J. Li, Y. Wang, S. Zhang, and Y. Gong, "Simplifying long short-term memory acoustic models for fast training and decoding," in *Proc. ICASSP*, 2016.

[45] H. Inaguma, Y. Gaur, L. Lu, J. Li, and Y. Gong, "Minimum latency training strategies for streaming sequence-to-sequence asr," in *Proc. ICASSP*, 2020.