Developing RNN-T Models Surpassing High-Performance Hybrid Models with Customization Capability

Jinyu Li, Rui Zhao, Zhong Meng, Yanqing Liu, Wenning Wei, Sarangarajan Parthasarathy, Vadim Mazalov, Zhenghao Wang, Lei He, Sheng Zhao, and Yifan Gong

Microsoft Speech and Language Group

{jinyli, ruzhao, zhme, yanqliu, wennwei, sarangp, vadimma, zhwang, helei, szhao, ygong}@microsoft.com

Abstract

Because of its streaming nature, recurrent neural network transducer (RNN-T) is a very promising end-to-end (E2E) model that may replace the popular hybrid model for automatic speech recognition. In this paper, we describe our recent development of RNN-T models with reduced GPU memory consumption during training, better initialization strategy, and advanced encoder modeling with future lookahead. When trained with Microsoft's 65 thousand hours of anonymized training data, the developed RNN-T model surpasses a very well trained hybrid model with both better recognition accuracy and lower latency. We further study how to customize RNN-T models to a new domain, which is important for deploying E2E models to practical scenarios. By comparing several methods leveraging text-only data in the new domain, we found that updating RNN-T's prediction and joint networks using text-to-speech generated from domain-specific text is the most effective.

Index Terms: end-to-end, RNN-T, LSTM, customization, context modeling

1. Introduction

Recently, one of the most significant trends in speech community is to replace hybrid models [1] with end-to-end (E2E) models [2, 3, 4, 5, 6, 7, 8, 9] for automatic speech recognition (ASR). Different from hybrid systems which have the limitation that many components such as acoustic model (AM) and language model (LM) are optimized separately, E2E ASR systems directly translate an input speech sequence into an output token sequence using a single network.

Currently, the most predominant E2E approaches for sequence-to-sequence transduction in ASR are recurrent neural network transducer (RNN-T) [10] and attention-based encoder-decoder (AED) [11, 12] (or LAS: Listen, Attend and Spell [3]). Because of its streaming nature, RNN-T has become a very promising E2E model in industry to replace the traditional hybrid models [8, 13, 14, 15]. In contrast, AED is more powerful and popular in academia. In [13, 16], a 2-pass E2E system was proposed to beat the hybrid model by leveraging the RNN-T's streaming capability in the first pass and the AED's modeling power in the second pass with rescoring.

Different from the 2-pass E2E system, this study focuses on developing a single RNN-T model surpassing a high-performance hybrid model [17] which was developed by integrating 3-stage training and advanced acoustic modeling [18]. Our contributions include GPU memory saving strategies for training, better initialization and advanced modeling which significantly improve the recognition accuracy.

Customization is another important requirement for deploying models into a new scenario which has only *text* data available. There are few ways of leveraging text-only data. A straightforward method is to interpolate the RNN-T model with an external LM built from the domain-specific text data as shallow fusion [19]. The second way is to generate synthetic speech data using text-to-speech (TTS), and use the text and speech pair to update the E2E models [20]. Spelling correction [21] uses TTS data to train a separate translation model to correct the errors made by E2E ASR models. To our best knowledge there is no comparison between these methods, especially for customization. In this study, we use RNN-T as an example to explore and compare these methods.

2. Improving RNN-T Models

In this section, we first briefly introduce RNN-T models. Then, we elaborate our efforts on GPU memory saving during training and on improving the accuracy of RNN-T models with a better initialization strategy and an advanced model structure.

2.1. RNN-T

RNN-T contains an encoder network, a prediction network, and a joint network. The encoder network converts the acoustic feature x_t into a high-level representation h_t^{enc}, where t is time index. The prediction network produces a high-level representation h_u^{pre} by conditioning on the previous non-blank target y_{u-1} predicted by the RNN-T model, where u is output label index. The joint network is a feed-forward network that combines the encoder network output h_t^{enc} and the prediction network output h_u^{pre} to generate $h_{t,u}$ which is used to calculate softmax output.

In [8], layer normalization and projection layer for long short-term memory (LSTM) [22] were reported important to the success of RNN-T modeling. We denote the layer-normalized LSTM function with projection layer as

$$h_t^l = LSTM(h_{t-1}^l, x_t^l), \tag{1}$$

where h_t^l is the lth layer output at time t. For the multi-layer LSTM, $x_t^l = h_t^{l-1}$. We use the last hidden layer output h_t^L and h_u^M of the encoder and prediction networks as h_t^{enc} and h_u^{pre}, where L and M denote the number of layers in encoder and prediction networks respectively.

2.2. Saving GPU memory

A practical challenge when we train RNN-T with large-scale data is that we cannot fit too many speech frames in a minibatch, because there are several 3-dimension tensors which consume large amount of GPU memories . In [14], we proposed several ways of reducing GPU memory usage by effectively organizing the encoder and prediction networks in memory and merging several network functions.

In this study, we further improve tokenization of word-piece units (WPUs) [23]. Some studies treated space ($) as an output token and used it as the delimiter of words [24, 25]. For example, a transcription “hey cortana i love gardening” is decomposed as “$ hey $ cor tana $ i $ love $ garden ing $”, which has 13 WPUs. This tokenization works well for CTC or AED training. However, the 3-dimension tensors in RNN-T always has one dimension as the total number of WPUs decomposed from the transcription. It is ideal to reduce this decomposition number. Although some RNN-T work [7] used `<space>` as the delimiter, we remove the `<space>` token and use “_” as the word beginning marker instead of a separate token. The example transcription can now be decomposed as: “_hey _cor tana _i _love _garden ing”, which has only 7 WPUs. This tokenization method significantly reduced the GPU memory consumption of those 3-dimension tensors, hence speeding up RNN-T training by using larger minibatch size.

2.3. Improving initialization

In this study, we initialize the encoder with either connectionist temporal classification (CTC) or cross entropy (CE) training. We don’t initialize the prediction network with a pre-trained LM as it has proven ineffective [26].

Using WPUs as output units facilitate the initialization with CTC because no alignment is needed. However, the CE training needs the time alignment information which is hard to get for WPUs which don’t have phoneme realisation. Because the time alignment for words is accurate, we just evenly segment the audio features and assign equal number of frames aligned to each word piece [27]. For example, if a word has starting time S and ending time E with K word-piece units, the time alignment of the kth WPU in the word is: $[S + \frac{k-1}{K}(E-S), S + \frac{k}{K}(E-S)], k = 1......K$.

2.4. Improving encoder

Incorporating the future context into RNN-T’s encoder structure can significantly improve the ASR accuracy, as shown in [14]. However, instead of consuming future context frames with a layer trajectory structure [14] which almost doubles the parameters of LSTM, in this study, we propose to only use context modeling to save model size as

$$g_t^{l-1} = \sum_{\delta=0}^{\tau} v_\delta^{l-1} \odot h_{t+\delta}^{l-1} \tag{2}$$

$$h_t^l = LSTM(h_{t-1}^l, g_t^{l-1}). \tag{3}$$

In Eq. (2), the output of LSTM with projection layer at current frame (h_t^{l-1}) and future τ frames ($h_{t+\delta}^{l-1}, \delta = 1......\tau$) are transferred to a new vector g_t^{l-1}, which is used as the input in Eq. (3) to calculate next layer’s LSTM output h_t^l. Because $\odot$ is element-wise product, Eq. (2) only increases the number of model parameters very slightly. g_t^L is used as the encoder network output. Because of context expansion, the number of total lookahead frames with context modeling is $Lx\tau$.

3. Customizing RNN-T models with text-only data

In this section, we study RNN-T customization with text-only data from a new domain. While we can directly build an external LM using domain-specific text data, we can also generate TTS data from this text, and then either adapt the RNN-T model or add an additional spelling correction model on top of the RNN-T model.

3.1. LM rescoring

We train an LSTM-LM [28] with target-domain text-only data to rescore each hypothesis generated by the RNN-T model through a log-linear interpolation between RNN-T and LSTM-LM scores. The hypothesis with the highest interpolated score is selected as the final output as follows

$$\hat{n} = \arg\max_n \left[\log P_{\text{RNN-T}}(\mathbf{y}^{(n)}|\mathbf{x}) + \lambda \log P_{\text{LM}}(\mathbf{y}^{(n)})\right], \tag{4}$$

where $\mathbf{x} = \{x_1, ..., x_T\}$ is a test utterance, $\mathbf{y}^{(n)} = \{y_1^{(n)}, ..., y_{U(n)}^{(n)}\}$ is n^{th} hypothesis in the N-best list from RNN-T beam search decoding, $n = 1, \ldots, N$, $\mathbf{y}^{(\hat{n})}$ is final output of LM rescoring, and λ is the weight for LM score.

3.2. Adapting RNN-T with TTS data

We use a multi-speaker neural TTS system [29] to generate TTS data. The TTS system consists of a spectrum predictor with speaker embeddings and a parallel WaveNet vocoder [30]. The spectrum predictor with speaker embeddings was trained with in-house data containing 7000 speakers. Then we use this TTS system to generate audio from the text-only data in the new domain. The TTS audio is used to adapt RNN-T models.

3.3. Spelling correction

A spelling correction model corrects the error patterns in the output hypotheses of a speech recognizer. [21] first proposes an attention-based spelling correction model with RNN structure to correct the output of AED model using TTS data. [31] introduces a transformer model [32] to correct ASR model output into grammatically and semantically correct text and uses the weights of a pre-trained BERT [33] to initialize the model. In this work, we use the transformer with encode-decoder architecture [32] for spelling correction. We generate synthetic audio signals using neural TTS models from text-only data and decode them using the baseline RNN-T speech recognizer to generate an erroneous hypotheses to pair with the ground-truth text at the TTS input. We then train a spelling correction model on these text pairs to correct potential recognizer errors. To compensate for the limited target-domain text data, we extract word-piece embeddings of the erroneous hypotheses from an RoBERTa [34] pre-trained with a large amount of external text, and add these embeddings to each layer of the encoder and decoder through multi-head self-attention to further improve the spelling correction. To incorporate local information, we insert a LocalRNN [35] at the input of the encoder and decoder.

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 set covers 13 application scenarios such as Cortana and far-field speech, using 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 data with personally identifiable information removed.

The feature is 80-dimension log Mel filter bank for every 10 milliseconds (ms) speech. Three of them are stacked together

Table 1: *Comparison of hybird models with average WERs on 1.8 M words test sets. The LM decoding graph size is 5 Gb.*

acoustic models	CE WER	MMI WER	T/S WER	parameter number	lookahead
LSTM	14.75	13.01	11.49	30 M	0
cltLSTM	11.15	10.36	9.34	63 M	480ms

to form a frame of 240-dimension input acoustic feature to the encoder network. The output targets are 4 K word-piece units.

4.1. Hybrid models

In [17], we reported our best hybrid model which was developed by integrating 3-stage training and an advanced acoustic model. We showed WERs of two hybrid models in Table 1. The first one is with a standard LSTM and the second one is a contextual layer trajectory LSTM (cltLSTM) [18] which 1) decouples the temporal modeling and target classification tasks with time and depth LSTMs respectively, 2) incorporates future context frames to get more information for accurate acoustic modeling. The input feature is 80-dimension log Mel filter bank for every 20 milliseconds (ms) of speech by using frame skipping [36]. The softmax layer has 9404 nodes to model the senone labels. Runtime decoding is performed using a 5-gram LM with decoding graph around 5 gigabytes (Gbs). The cltLSTM totally has 24-frame lookahead, which corresponds to 480ms duration. The training of both models exploit 3-stage training strategy: from CE to maximum mutual information (MMI) [37], and then followed by sequential teacher-student (T/S) learning [38]. The cltLSTM trained with such a strategy improves from the CE baseline by 16.2% relative WER reduction, and it also improves from its LSTM counterpart by 18.7% relative WER reduction. Hence, this cltLSTM model presents a very challenging streaming hybrid model to beat.

4.2. Surpassing hybrid models

Now, we report how the RNN-T models can be improved to exceed the accuracy of hybrid models. We denote all RNN-T models' encoders as MpN_FxL, where M is number of cells in LSTM, N is the projection layer size, F is the number of lookahead frames at each layer, and L is the number of layers. For simplicity, the prediction network always uses 2 layers of LSTM with the same structure as the encoder's LSTM without any lookahead. That is to say when the encoder's structure is MpN_FxL, the prediction network structure will be MpNx2. The decoding is beam search using 5 as the beam size.

We first examine the impact of initialization for RNN-T by using the encoder structure 1600p800_4x6 in Table 2. This model has 1600 LSTM memory cells and the output is projected to 800. The encoder has 6 layers and has context modeling with 4 frames lookahead at each layer. The CTC initialization slightly improves the RNN-T model with random initialization, while the CE initialization improves from the random initialization by 11.6% relative WER reduction. 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. Because CTC training doesn't need any alignment information while CE training needs, the result indicates learning alignment information for the encoder may help RNN-T training to focus more on reasonable forward-backward paths instead of all the paths.

Table 2: *WERs of initialization methods for RNN-T.*

models	Random	CTC	CE
1600p800_4x6	10.55	10.40	9.33

Table 3: *Comparison of RNN-T models.*

encoder network	WER	parameter number	encoder lookahead
1280p640x6	11.25	62 M	0 ms
1280p640_4x6	9.81	62 M	720 ms
1600p800_4x6	9.33	94 M	720 ms
2048p640_4x6	9.27	87 M	720 ms
2048p640_4x8	9.28	119 M	960 ms
2560p800_4x6	8.88	147 M	720 ms
2560p800_2x6	9.05	147 M	360 ms

In Table 3, we compare all RNN-T models with different setups in terms of WER, parameter number, and the encoder lookahead. The encoders of all models are initialized with CE training. The first model, 1280p640x6, uses standard layer-normalized LSTM with projection layers as in most literature [8, 13]. This model has 6 layers and the LSTM at each layer has 1280 memory cells with the output projected to 640 dimensions. It has 62 M parameters and 0 ms encoder lookahead. It obtained 11.25% WER on the 1.8 M word test sets, about 2.1% relative WER reduction from the T/S trained LSTM hybrid model in Table 1 which also has 0-frame encoder lookahead.

Next, by looking ahead 4 future frames at each layer with context modeling (Eqs. (2) and (3)), the model 1280p640_4x6 significantly reduced the WER from 11.25% to 9.81%, about 12.8% relative WER reduction. The model size is the same as 1280p640x6, but has 720 ms (6x4x30 ms) encoder lookahead.

Next, we increased the model size to 94 M with model 1600p800_4x6, and got further WER reduction to 9.33%. Another model, 2048p640_4x6, with 87 M parameters obtained slightly better WER as 9.27%. Then we increased the model to 8 layers as 2048p640_4x8. Surprisingly we didn't get any gain although the model size and encoder lookahead are both increased. From this set of experiments, it seems that in our context modeling setup, it is better to enlarge memory cell sizes of LSTMs instead of going too deep.

Encouraged by our observation, we further increased the memory cell of LSTMs to 2560, and the projection dimension to 800. The model, 2560p800_4x6, obtained 8.88% WER, which is 4.9% relatively better than the T/S trained hybrid model cltLSTM model in Table 1 which has 480 ms lookahead. Finally, we reduced the lookahead at every layer from 4 to 2 frames, generating model 2560p800_2x6 which has 360 ms (6x2x30 ms) encoder lookahead. Such model obtained 9.05% WER, which has 3.1% relative WER reduction from the T/S trained hybrid model cltLSTM model in Table 1.

In Figure 1, we look at the gap (in frames) between ground truth word alignment obtained by force alignment with a hybrid model and the word alignment generated by greedy decoding from three RNN-T models in Table 3. They are 1280p640x6, 2560p800_2x6, and 2560p800_4x6 with 0, 360 ms and 720 ms encoder lookahead, respectively. As shown in Figure 1, the 1280p640x6 model with zero lookahead in the encoder network, plotted in the right most curve, has larger delay than

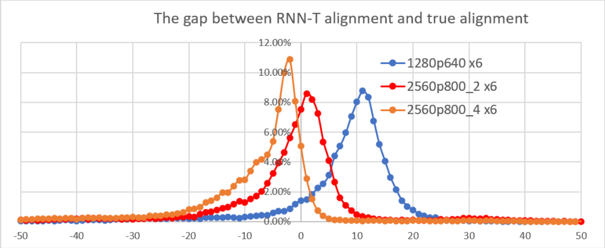

Figure 1: *The gap (in frames) between ground truth word alignment and the word alignment from 1280p640x6, 2560p800_2x6, and 2560p800_4x6 RNN-T models.*

the ground truth alignment. The average delay is about 11 input frames, which corresponds to 330 ms. In contrast, the 2560p800_2x6 model with 360 ms lookahead is plotted in the center curve and has less alignment discrepancy, with average 1 input frame delay. This is because its encoder has total 12 frames lookahead, which provides more information to RNN-T so that it makes decision much earlier than the zero-lookahead model. The average latency of this 2560p800_2x6 model is (12+1)*30 ms = 390 ms. Finally, the 2560p800_4x6 model which has totally 24 frames lookahead is plotted in the left most curve and has even -2 frames latency, and the average latency of this model is (-2+24)*30 ms = 660ms. The 2560p800_2x6 RNN-T model has clear advantages, surpassing the very well trained cltLSTM model with smaller WER and latency.

4.3. Customization

In this section, we evaluate RNN-T customization in a new domain using text-only data with 1280p640x6, 2560p800_4x6 and 1600p800_4x6 RNN-T models which have the highest, lowest, intermediate WERs respectively in Table 3 when evaluated with those related test sets. Reported in Table 4, 1600p800_4x6 instead of 2560p800_4x6 has the best WER in this new domain, indicating the good performance of an E2E model extremely well trained with even 65 K hours data may not generalize to an unseen test set. On the other hand, both 1600p800_4x6 and 2560p800_4x6 still significantly outperformed 1280p640x6, which is consistent with the results in Table 3.

The text data in this new domain contains 1.5 M sentences. Based on this text, about 1.5 K hours of audio was synthesized from randomly selected 300 speakers in TTS training data. We did data augmentation by adding noise and room impulse response as in [39] to the original TTS audio. The final audio is about 3000 hours. The test data is collected in this new domain.

Then, we used the TTS audio to adapt RNN-T models by updating all parameters of RNN-T models. We obtained significant degradation which is not a surprise because the RNN-T encoder was updated to fit those 300 TTS speakers, resulting in bad generalization to speakers in this new domain. Clearly, the domain-specific text-only data should benefit the LM related component in RNN-T. Therefore, we updated only prediction and joint networks in RNN-T models. We obtained 16.04%, 13.69%, and 13.88% WERs for 1280p640x6, 1600p800_4x6, and 2560p800_4x6 models respectively, representing 7.9%, 7.2%, and 9.8% relative WER reduction respectively. Although the largest relative WER reduction was obtained with the 2560p800_4x6 model, it still didn't outperform the corresponding 1600p800_4x6 model. Therefore, in the following experiments, we mainly investigate the effectiveness of customization methods using 1280p640x6 and 1600p800_4x6 models, which have the highest and lowest WERs respectively in the new domain.

Table 4: *Comparison of customization methods.*

	1280p640	1600p800	2560p800
	x6	_4x6	_4x6
baseline	17.41	14.75	15.39
TTS only			
update all	22.97	19.72	19.63
update Pre+Joint	16.03	13.69	13.88
speech + TTS	16.31	13.91	-
Spelling correction			
w/o RoBERT	16.78	14.31	-
w/ RoBERT	16.03	13.85	-
LM rescoring	16.73	14.51	-

We tried to mix 20 K hours original speech data together with the TTS audio, and then update the prediction and joint networks of RNN-T models. The results are shown as (speech+TTS) in Table 4. It doesn't improve the adaption with TTS only audio. This means that if we have enough text-only data, and we don't need to regularize the adaptation with original speech training data.

The proposed spelling correction transformer model has 6 layers in the encoder and decoder. The attention layer has 8 heads with a hidden dimension size of 512 and the hidden size in the feed forward layers is set to 2048. The hidden dimension of LocalRNN is 256 with a local window of size 6. We applied this spelling correction model on top of RNN-T models, and obtained 3.6% and 3.0% relative WER reduction from baseline 1280p640x6 and 1600p800_4x6 models, respectively. With pretrained RoBERTa, we obtained further improvement, with 7.9% and 6.1% relative WER reduction from baseline 1280p640x6 and 1600p800_4x6 models, respectively.

At last, we also built an LSTM-LM with the new-domain text. With 1 hidden layer and 512 hidden units, the LSTM-LM predicts the posteriors of 4k word pieces at the output layer. Each word piece is encoded as a 512-dim vector before feeding into the LM. Rescoring RNN-T with LSTM-LM gave 3.9% and 1.6% relative WER reduction for baseline 1280p640x6 and 1600p800_4x6 models, respectively.

5. Conclusions

In this paper, we elaborated our efforts of developing high-quality RNN-T models and evaluated with 65 K hours Microsoft training data. The CE initialization of RNN-T encoder significantly reduced WER by 11.6% relatively and the model with future context improved from the zero-lookahead model by 12.8% relatively. Thanks to all these methods, an RNN-T model with 6-layer encoder using 2-frame lookahead at each layer surpasses the best hybrid model trained with delicate 3-stage optimization and advanced modeling technology by 3.1% relative WER reduction. This RNN-T model also has 120 ms less encoder lookahead latency than the best hybrid model.

We further investigated how to leverage text-only data to adapt RNN-T models to a new domain. Adapting RNN-T models' prediction and joint networks using the TTS audio generated from the domain text was shown to be more effective than either spelling correction or LM rescoring, which needs to introduce additional networks during runtime while adaption doesn't need.

6. References

[1] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *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*. IEEE, 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] 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.

[7] 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.

[8] 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.

[9] J. Li, Y. Wu, Y. Gaur, C. Wang, R. Zhao, and S. Liu, "On the comparison of popular end-to-end models for large scale speech recognition," in *Proc. Interspeech*, 2020.

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

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

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

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

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

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

[16] 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.

[17] 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.

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

[19] C. Gulcehre, O. Firat, K. Xu, K. Cho, L. Barrault, H.-C. Lin, F. Bougares, H. Schwenk, and Y. Bengio, "On using monolingual corpora in neural machine translation," *arXiv preprint arXiv:1503.03535*, 2015.

[20] K. C. Sim, F. Beaufays, A. Benard *et al.*, "Personalization of end-to-end speech recognition on mobile devices for named entities," in *Proc. ASRU*, 2019.

[21] J. Guo, T. N. Sainath, and R. J. Weiss, "A spelling correction model for end-to-end speech recognition," in *Proc. ICASSP*, 2019, pp. 5651–5655.

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

[23] M. Schuster and K. Nakajima, "Japanese and Korean voice search," in *Proc. ICASSP*. IEEE, 2012, pp. 5149–5152.

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

[25] Y. Gaur, J. Li, Z. Meng, and Y. Gong, "Acoustic-to-phrase models for speech recognition," *Proc. Interspeech*, pp. 2240–2244, 2019.

[26] M. Ghodsi, X. Liu, J. Apfel, R. Cabrera, and E. Weinstein, "RNN-transducer with stateless prediction network," in *Proc. ICASSP*, 2020, pp. 7049–7053.

[27] 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.

[28] M. Sundermeyer, R. Schlüter, and H. Ney, "Lstm neural networks for language modeling," in *Thirteenth annual conference of the international speech communication association*, 2012.

[29] Y. Deng, L. He, and F. K. Soong, "Modeling multi-speaker latent space to improve neural tts: Quick enrolling new speaker and enhancing premium voice," *ArXiv*, vol. abs/1812.05253, 2018.

[30] A. v. d. Oord, Y. Li, I. Babuschkin, K. Simonyan, O. Vinyals, K. Kavukcuoglu, G. v. d. Driessche, E. Lockhart, L. C. Cobo, F. Stimberg *et al.*, "Parallel wavenet: Fast high-fidelity speech synthesis," *arXiv preprint arXiv:1711.10433*, 2017.

[31] O. Hrinchuk, M. Popova, and B. Ginsburg, "Correction of automatic speech recognition with transformer sequence-to-sequence model," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 7074–7078.

[32] 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.

[33] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[34] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.

[35] Z. Wang, Y. Ma, Z. Liu, and J. Tang, "R-transformer: Recurrent neural network enhanced transformer," *arXiv preprint arXiv:1907.05572*, 2019.

[36] 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.

[37] P. C. Woodland and D. Povey, "Large scale discriminative training of hidden Markov models for speech recognition," *Computer Speech and Language*, vol. 16, no. 1, pp. 25–47, 2002.

[38] J. H. Wong and M. J. Gales, "Sequence student-teacher training of deep neural networks," in *Proc. Interspeech*, 2016.

[39] J. Li, R. Zhao, Z. Chen *et al.*, "Developing far-field speaker system via teacher-student learning," in *Proc. ICASSP*, 2018.