TRANSDUCER-BASED STREAMING DELIBERATION FOR CASCADED ENCODERS

Ke Hu, Tara N. Sainath, Arun Narayanan, Ruoming Pang, Trevor Strohman

Google LLC, USA

{huk,tsainath,arunnt,rpang,strohman}@google.com

ABSTRACT

Previous research on applying deliberation networks to automatic speech recognition has achieved excellent results. The attention decoder based deliberation model often works as a rescorer to improve first-pass recognition results, and requires the full first-pass hypothesis for second-pass deliberation. In this work, we propose a transducer-based streaming deliberation model. The joint network of a transducer decoder often receives inputs from the encoder and the prediction network. We propose to use attention to the first-pass text hypothesis as the third input to the joint network. The proposed transducer based deliberation model naturally streams, making it more desirable for on-device applications. We also show that the model improves rare word recognition compared to cascaded encoders, with relative WER reductions ranging from 3.6% to 10.4% for a variety of test sets. Our model does not use any additional text data for training.

1. INTRODUCTION

Research in RNN transducers (RNN-T) has made significant progress in recent years [1, 2, 3, 4, 5, 6, 7, 8]. Different layer structures such as long short-term memory (LSTM) [9], transformer self-attention layers [10], and conformer [11] have been proposed to improve encoder context modeling as well as computational efficiency. Novel model structures and training techniques have also been investigated [12, 7, 8, 13, 14]. To improve language modeling, [12] subtracts an internal language model (LM) from an RNN-T and uses a conventional LM for decoding. [7] cascades a syllable-to-character RNN-T to a regular one to leverage text-only data in language modeling. Other research has tried to incorporate attention mechanism in RNN-T, e.g., [13] uses prediction network output to compute attention on encoder outputs as a joint network input. Recently, a cascaded encoder [2] is proposed to use right-context conformer to attend to future audio to improve recognition quality. The non-causal characteristics of the cascaded encoder leads to significantly better contextual modeling by trading off on latency. To further achieve latency requirements, FastEmit [15] and end-to-end (E2E) endpointing [16] improve decoding speed significantly. Together with neural LM rescoring, the transducer model becomes a competitive candidate for on-device applications [1].

E2E models often find rare word recognition to be challenging. One solution is to leverage rich text-only training data to work with the E2E models. For example, rescoring methods rely on rich text-only data to train an external LM to for hypothesis re-ranking or re-decoding [1, 17]. Bidirectional textual context is incorporated by using BERT [18, 19, 20]. Shallow fusion [21, 22, 23] combines E2E model scores with an external LM in beam search steps. As more text data has been seen by the model, these methods tend to improve rare word recognition. Another approach to tackle rare word recognition is to utilize text metadata available during the application, and attend to the text encoding for contextual biasing to recognize rare words [24, 25]. However, the use of metadata limits its use for general rare word recognition.

Deliberation models show promising improvements on rare word recognition [26, 27]. Compared to LSTM or transformer rescoring, deliberation models [26, 27] excel at its correction abilities and an attention mechanism that looks at a full audio context. Recent research has been focusing on incorporating text-only data in model training [28]. However, so far the deliberation model works as a rescorer, making it less desirable for streaming applications.

In this work, we propose a streaming deliberation model by using a transducer decoder in the second pass. Similar to an RNN-T decoder [29, 30], the deliberation transducer decoder consists of a embedding prediction network [31] and a joint network. Further, we add attention as the third input to the joint layer. The attention is computed using cascaded encoder outputs [2] as queries to summarizes the first-pass hypothesis. This serves as an LM context for correcting the first-pass errors. Similar to [2], our model streams with delay. We show that the proposed model achieves significant improvement for rare word recognition of Google Voice Search and several rare word test sets compared to cascaded encoders [2]. The relative WER reduction ranges from 3.6% to 10.4%. We did not find improvement on the general Voice Search sets probably because the cascaded encoder already incorporates future context. Our model differs from previous deliberation [26, 27, 32] in that 1) we use a transducer decoder instead of attention decoder, and this naturally adds streaming capabilities to deliberation, and 2) the deliberation decoder uses cascaded encoder outputs [2] for deliberation. Compared to the aforementioned LM rescoring or shallow fusion, our model does not need any text-only data in training.

2. TRANSDUCER-BASED DELIBERATION WITH CASCADED ENCODERS

The proposed transducer based deliberation is illustrated in Fig. 1. Similar to [26, 27, 32], the deliberation model includes a conformer transducer (Conf-T) which produces first-pass decoding results $\boldsymbol{y}$. Note the first-pass Conf-T consists of only causal layers and uses FastEmit [15] to achieve latency requirements. The text output $\boldsymbol{y}$ will be encoded by a text encoder and used to compute attention. In the second pass, the causal encoder is cascaded with a non-causal encoder for better contextual modeling [2]. The non-causal encoder outputs, attention from first-pass, and second-pass prediction network are fed to a separate joint layer for deliberation decoding. We further elaborate the system in following sections.

2.1. Cascaded Encoders

Our cascaded encoder is similar to [2], which consists of right-context conformer layers to look into future audio. Different from

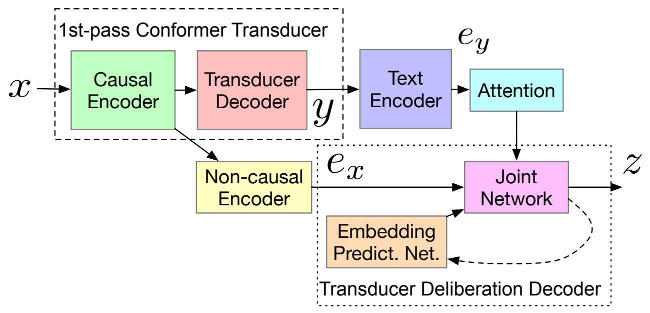

Fig. 1. Diagram of an transducer-based deliberation model for cascaded encoders. The first pass conformer transducer (dashed box) generates first-pass hypotheses, which are encoded and used to compute attention between cascaded encoder outputs. The attention is fed into the joint layer of the second-pass transducer decoder (dotted box), together with cascaded encoder and embedding prediction network outputs. This is a streaming deliberation model due to the nature of a transducer decoder.

[2], our cascaded encoder is connected to a separate transducer decoder dedicated to deliberation (dotted box in Fig. 1). The cascaded non-causal encoder usually has a much smaller size than the causal encoder. An embedding prediction network is used for both first-pass and second-pass for model efficiency without significantly degrading quality [31].

2.2. Streaming Deliberation

A distinctive structure of a deliberation network is that it uses both audio and first-pass hypotheses for decoding [32]. To implement streaming deliberation, we first model latencies in text encoding (Sect. 2.2.1), and then compute encoded partial first-pass hypothesis at each streaming step for deliberation (Sect. 2.2.2).

2.2.1. *Text Encoder*

For streaming purposes, we use right-context conformer layers [11, 2] rather than bidirectional layers [27, 26] in text encoding. By encoding only non-blank tokens, the right-context conformer provides contextual modeling from future. To model the latency introduced by the text encoder, we first denote the ith non-blank token in $\boldsymbol{y}$ as y_i, and the corresponding time frame t'_i. If we use a conformer encoder with a total right-context of R tokens, the right-most non-blank token we need for encoding y_i is thus y_{i+R}. This means that the right-most time frame we need to encode y_i is :

$$r_i = min(t'_{i+R}, T') \tag{1}$$

where T' is the maximum time frame of $\boldsymbol{y}$. In other words, r_i denotes the latest time frame needed for encoding ith token in $\boldsymbol{y}$. Note that we use causal convolutions in conformer layers, and there is no convolution subsampling before the text encoder.

2.2.2. *Partial Hypothesis*

In this section, we model the streaming capability of the transducer-based deliberation to attend to the first-pass hypotheses. Note that the cascaded encoder is non-causal (similar to [2]) and has a latency equal to its right context. Therefore it allows us to look at a partially streamed first-pass hypothesis for deliberation. To simulate streaming, we first encode the full hypothesis sequence ($\boldsymbol{y}$ in Fig. 1) and then compute a partial sequence of the encoding ($\boldsymbol{e}_y$ in Fig. 1) by looking ahead A frames from a current frame. We denote the encoded partial hypothesis up to time frame t as

$$\boldsymbol{e_y}(t) = \{e_{y,k} | \text{ where } r_k < t + A \text{ and } k < L\} \tag{2}$$

where $e_{y,k}$ denotes the kth encoded token in $\boldsymbol{e_y}$, and r_k is the latest time frame required for encoding the kth token (see Eq. 1). t is the time frame of cascaded encoder output. L is the total number of tokens in $\boldsymbol{y}$. There are a couple of notes about Eq. (2). First, the partial encoded hypothesis is computed for every second-pass frame t, i.e., in a streaming fashion. Second, $r_k < t + A$ ensures that only first-pass hypothesis tokens within the lookahead A are used. The attention thus does not use any tokens beyond specified lookahead. We pad the rest of the encoding sequence to the length L by zeros. An alternative streaming implementation would be to mask $\boldsymbol{y}$ first and then encode, but this is expensive to do for every t. We will further discuss the choice of A in Sect. 4.5.

2.2.3. *Attention and Joint Layer*

At every second-pass frame t, we compute attention a_t between the partial encoded hypothesis $\boldsymbol{e_y}(t)$ (as key and value) and a frame of cascaded encoder output $\boldsymbol{e_x}(t)$ (as query). This summarizes first-pass hypothesis by querying using the current time frame. The attention context vector, a_t, is first merged with the encoded audio $\boldsymbol{e_x}(t)$, and the output c_t is combined with the prediction network output l_u to be fed into the joint layer:

$$c_t = \text{Merge}(a_t, \boldsymbol{e_x}(t)) \tag{3}$$
$$h_{t,u} = \tanh(W_{ch}c_t + W_{lh}l_u + b_h) \tag{4}$$

where $h_{t,u}$ denotes the joint layer output at time frame t and token step u, and l_u is the prediction network output at step u. $h_{t,u}$ will be fed to a softmax layer [33]. We have also tried using the prediction network output as the query and found no improvement (see Sect. 4.1 for results). We use a similar merger layer as in [27] and tried sum, concatenation, and gated average. We have also tried to directly combine the attention, encoded audio, and prediction network output using a merger layer and did not find improvement either.

3. EXPERIMENTAL DETAILS

We perform our experiments using large-scale data [34] based on a state-of-the-art Conf-T with cascaded encoders [29].

3.1. Modeling Details

3.1.1. *First-pass Conformer Transducer*

We use a domain-id conformer transducer (Conf-T) [29] as the first-pass model. For feature extraction, we divide a speech waveform using 32-ms hanning windows at a rate of 10 ms, and then compute 128-D log-Mel features. Each log-Mel feature is stacked with three previous frames to form a 512-D vector, and then downsampled to a 30-ms frame rate. A 16-D one-hot vector is used to represent a domain id and appended to the log-Mel feature. Our conformer encoder consists of 17 causal conformer layers [11]. Each conformer layer has a model dimension of 512, and we use 8-head self-attention and a kernel size of 15 in convolution. The encoder output is later on projected to 640 dimension as inputs for the joint layer. We ensure the causality of this encoder by using causal convolution and left-context attention. A time-reduction layer is further added after the fourth conformer layer to increase frame rate to 60 ms for efficiency. We use an embedding prediction network [31] consisting of two separate embedding layers for the previous two tokens, respectively. Their embeddings are then concatenated and projected to 640 dimension to a joint network with a single projection layer of 640 units. The first-pass Conf-T model is trained to predict 4,096 lowercase wordpieces [35].

3.1.2. Deliberation with Cascaded Encoders

On the non-causal path, we use four 512-D conformer layers with a total of 2.88-s right context cascaded to the causal encoder. The non-causal encoder has around 25M parameters. The transducer-based deliberation has a separate prediction network which has the same structure as the first-pass one. The second-pass transducer decoder has around 8M parameters. We use sampling to estimate first-pass hypothesis as in [32] for efficiency, where a token is sampled at every frame according to the softmax output probabilities. We have also tried beam search and found similar deliberation recognition results. The deliberation text encoder is a 2-layer 640-D conformer encoder with a two-token right-context. The text encoder has around 19M parameters. We use an 8-headed attention layer to compute a context vector between text encoder outputs and cascaded encoder outputs. The attention lookahead is chosen to be 2.88 seconds to match the cascaded encoder latency. The trasducer deliberation decoder is trained by the RNN-T loss [36] to predict the same 4,096 wordpieces as the first pass. The whole model is jointly updated during training.

Our models are trained in Tensorflow [37] using the Lingvo framework [38] on 8×8 Tensor Processing Units (TPU) slices with a global batch size of 4,096. We use the transformer learning schedule [10] and Adam optimizer with synchronized stochastic gradient descent in training, and maintain an exponential moving average [39] of the trained model parameters for evaluation.

3.2. Datasets

Our training dataset is described in [34]. The English utterances are sampled from multiple domains such as general Google traffic, far-field environments, telephony conversations, and YouTube. The set also contains accented speech from multiple English locales [40]. These utterances are anonymized and hand-transcribed and have a total amount of ~400k hours. We augment the clean training utterances by artificially corrupting them by using a room simulator, varying degrees of noise, and reverberation such that the signal-to-noise ratio (SNR) is between 0dB and 30dB [41]. We also use mixed-bandwidth utterances at 8kHz or 16 kHz for training [42]. SpecAug [43] and random state sampling [34] are used as regularizations.

Our test sets include general Google Voice Search (GVS) test set, which contains ~14K anonymized and hand-transcribed utterances sampled from general Google Voice Search traffic, and a Chrome Voice Search (CVS) test set with ~1K utterances. The CVS set represents search traffic from Chrome OS and potentially contains more challenging rare proper nouns. To specifically measure rare word recognition performance, we use three TTS test sets: *Apps*, *Songs*, and *Contacts*, with around 16K, 15K and 15K utterances, respectively. The sets consist of voice commands querying proper nouns such as app, song and person names that are often rare in training data [44]. For example, a typical *Song* query is `play rihanna music` and a contact request is `call John Snow`. The utterances are synthesized using a Parallel WaveNet TTS model [45], and noise is artificially added to the synthetic data [41].

4. RESULTS

Below we first show impacts of different architecture variants and parameters to the deliberation model, and compare deliberation to related E2E models.

4.1. Attention Variants

We tried three different ways to merge encoded audio and attention in Eq. (3): concatenation, sum, and gated average. In the concatenation case, we first concatenate and then use a projection layer project the dimension back to the original encoded audio. We use a lookahead A equal to the cascaded encoder right-context, i.e., 48 frames and R is chosen to be 2 empirically. In Table 1, we find that sum works the best: 5.4% for GVS, and significantly better than other merging methods for other test sets. We thus choose sum for for further experiments.

Model	GVS	CVS	APP	Song	Contacts	Avg.
Sum	**5.4**	**22.2**	**9.7**	**10.3**	**24.7**	**14.5**
Concat	5.5	23.0	10.5	11.3	27.5	15.6
Gated	5.6	23.4	13.4	12.2	27.5	16.4

Table 1. WERs (%) by computing attention using encoder outputs as queries.

We also tried to compute attention a_t (in Sect. 2.2.3) by using prediction network output as query, instead of encoded audio $\boldsymbol{e_x}(t)$. The attention is then merged with l_u similarly to Eq. (3). As shown in Table 2, the WERs are generally worse than using audio as the query (Table 1). This is probably because the encoded audio is from the cascaded encoder and contains future context.

Model	GVS	CVS	APP	Song	Contacts	Avg.
Sum	**5.4**	**22.7**	**11.7**	**11**	**26.4**	**15.4**
Concat	5.6	22.4	12.8	12.1	27.1	16.0
Gated	5.7	22.4	11.9	11.2	27.6	15.8

Table 2. WERs (%) by computing attention using prediction network outputs as queries.

4.2. Text Encoder Right-Context and Lookahead

We experiment with different right contexts in text encoder, i.e., R in Eq. (1), and lookahead at first-pass hypothesis, i.e., A in Eq. (2). A bigger R corresponds to a longer right-context in text encoding, and A decides how much ahead we look into the first-pass hypothesis for text encoding. From Table 4, we see that WER improves as we increase either R and A. The improvement is most significant when R is increased from 24 to 48 frames (i.e., 1.44s to 2.88s). Further increasing R and A gives slight improvement.

4.3. Joint Training

When training the deliberation model, one can either only update the deliberation decoder or train the first and second passes jointly. We find that joint training is uniformly better than only training the second-pass deliberation decoder (Table 5). We also evaluated the first-pass Conf-T and did not observe regression: GVS WER is 6.7%. We did not observe any regression on other datasets either.

Model		WER (%)					
		GVS	CVS	App	Song	Contacts	Avg.
B1	Conf-T	6.8	24.2	17.2	14.8	38.8	20.4
B2	Cascaded encoders	5.4	23.8	10.6	11.5	27.3	15.7
E1	Dual decoder	5.3	23.8	12.8	11.9	27.1	16.1
E2	Cascaded deliberation	**5.4**	**22.2**	**9.7**	**10.3**	**24.7**	**14.5**
	rel. reduct. v.s. B2	*0%*	*-6.7%*	*-8.5%*	*-10.4%*	*-9.5%*	*-7.6%*

Table 3. WER (%) comparison between different E2E models. The last row shows the relative WER reductions by comparing deliberation (E2) to cascaded encoders (B2).

(R, A)	GVS	CVS	APP	Song	Contacts	Avg.
(2, 0)	5.5	23.0	11.7	10.8	26.9	15.6
(2, 24)	5.5	22.4	11.3	11.3	25.3	15.2
(2, 48)	**5.4**	**22.2**	**9.7**	**10.3**	**24.7**	**14.5**
(6, 65)	5.4	22.6	9.2	9.7	23.9	14.2

Table 4. Comparing WERs (%) by using different right contexts in conformer text encoder and lookahead in first-pass hypotheses. We use a right-context of 2 and lookahead of 48 frames in final comparison considering both quality and latency requirements.

Model	GVS	CVS	APP	Song	Contacts	Avg.
Joint	**5.4**	**22.2**	**9.7**	**10.3**	**24.7**	**14.5**
2nd Pass	5.5	29.9	11.4	10.7	25.2	16.5

Table 5. WER (%) comparison between joint training and only training the second pass.

4.4. WER Comparison

In Table 3, we compare the proposed deliberation model (E2) to three different models: A Conf-T (B1) which is used as the first-pass model for deliberation, a Conf-T with cascaded encoder (B2) [29, 2], and a dual-decoder model (E1). The dual decoder differs from the cascaded encoder (B2) in that it has a separate decoder for the non-causal path, while the cascaded encoder share the same one with the causal path. Deliberation adds an additional input of attention to the second-pass transducer, compared to the dual-decoder.

From Table 3, we observe that the cascaded encoder (B2), dual-decoder (E1), and deliberation (E2) all perform significantly and uniformly better than the Conf-T (B1) on all test sets. This is because the three models all have lookahead into future which gives them strong contextual information. Comparing B1 with deliberation (E2), the WER improvement ranges from 8%-44% relative.

Second, we compare deliberation (E2) to cascaded encoders (B2). Their major difference is that deliberation attends to first-pass hypotheses in second-pass transduction. We see that the improvement centers around rare word test sets such as CVS, App, Song and Contacts, from 6.7% to 10.4% relative in WER. These test sets generally have higher WERs than GVS potentially because of the rare proper nouns. The lack of improvement on GVS is probably because utterances are sampled from general Google traffic and an WER of 5.4% is already very competitive. Cascaded encoders achieve this by looking into future audio using the non-causal encoder. We further measure a Rare WER for on GVS for B2 and E2 (Table 6). The Rare WER is computed by first removing the top 500 English words from both the hypotheses and ground-truth transcripts, and then compute error rates between the two similar to a regular WER. There is a 3.6% relative WER improvement for deliberation and this indicates that deliberation still improves rare word recognition for GVS.

Model	B2	E2
Rare WER (%)	8.3	**8.0**

Table 6. Rare word WER (%) comparison between Conf-T with cascaded encoders (B2) and deliberation (E2).

4.5. Latency Comparison

An inherent latency of the proposed deliberation model comes from the lookahead, A, defined in Sect. 2.2.2. A reasonable choice of A is equal to the right-context of the cascaded encoder. A can be chosen to be greater than the cascaded encoder's right-context. In that case, further delay is introduced in second-pass, but with longer first-pass sequence to deliberate. On the contrary, if A is less than the right-context, a shorter first-pass hypothesis is used and the latency bottleneck is the cascaded encoder. Choosing A equal to the cascaded encoder right context (2.88s) should not introduce significant latency. This is because the cascaded encoder and deliberation can compute in parallel. To compare their computation, we compute the floating-point operations (FLOPs) needed the for a 3.27s GVS utterance (i.e., 90th percentile) using a similar method as Eq. (3) in in [26]. The computation needed for a cascaded encoder is 1327 FLOPs, which is significantly greater than a sampling-based first-pass decoding and text encoding (383 FLOPs). This means deliberation should not impose more latency if computing is in parallel.

4.6. Decoding Examples

We show some decoding examples between cascaded encoders and deliberation in Table 7. We see that the deliberation wins are mainly on rare proper nouns, and losses are due to spelling errors and insertions, which may be corrected by using an endpointer [16].

Model	Cascaded Encoder	Deliberation
Wins	okay edwards express open	okay adwords express open
	check google accents	check google adsense
Losses	open up the wear os phone	open up the wearos phone
	open up stress meditation	open up stress meditation okay

Table 7. Decoding examples of cascaded encoders and deliberation. Deliberation wins are in green and losses in red.

5. CONCLUSION

We proposed a streaming deliberation model by using a transducer decoder in the second pass. Compared to cascaded encoders, the model improves the WER of rare word test sets by 6.7% to 10.4%. The improvement on Voice Search centered on rare words: 3.6% by Rare WER. The model introduces an additional input to the joint layer of a transducer decoder that attends to encoded first-pass hypotheses. The attention potentially extracts contextual information for the second-pass to correct recognition errors in the first-pass. Our model does not need additional text-only data for training.

6. REFERENCES

[1] T. N. Sainath, Y. He, A. Narayanan, R. Botros, R. Pang, D. Rybach, C. Allauzen, E. Variani, J. Qin, Q.-N. Le-The, S.-Y. Chang, B. Li, A. Gulati, C.-C. Yu, Jiahui Chiu, D. Caseiro, W. Li, Q. Liang, P. Rondo, et al., "An efficient streaming non-recurrent on-device end-to-end model with improvements to rare-word modeling," *Interspeech*, 2021.

[2] A. Narayanan, T. N. Sainath, R. Pang, J. Yu, C.-C. Chiu, R. Prabhavalkar, E. Variani, and T. Strohman, "Cascaded encoders for unifying streaming and non-streaming asr," in *ICASSP*. IEEE, 2021, pp. 5629–5633.

[3] X. Chen, Y. Wu, Z. Wang, S. Liu, and J. Li, "Developing real-time streaming transformer transducer for speech recognition on large-scale dataset," in *ICASSP*. IEEE, 2021, pp. 5904–5908.

[4] J. Li, R. Zhao, Z. Meng, Y. Liu, W. Wei, S. Parthasarathy, V. Mazalov, Z. Wang, L. He, S. Zhao, et al., "Developing RNN-T models surpassing high-performance hybrid models with customization capability," *arXiv preprint arXiv:2007.15188*, 2020.

[5] 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 *ICASSP*. IEEE, 2020, pp. 7829–7833.

[6] 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.

[7] X. Wang, Z. Yao, X. Shi, and L. Xie, "Cascade RNN-transducer: Syllable based streaming on-device Mandarin speech recognition with a syllable-to-character converter," in *SLT*. IEEE, 2021, pp. 15–21.

[8] G. Saon, Z. Tüske, D. Bolanos, and B. Kingsbury, "Advancing RNN transducer technology for speech recognition," in *ICASSP*. IEEE, 2021, pp. 5654–5658.

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

[10] 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.

[11] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, et al., "Conformer: Convolution-augmented transformer for speech recognition," *arXiv preprint arXiv:2005.08100*, 2020.

[12] E. Variani, D. Rybach, C. Allauzen, and M. Riley, "Hybrid autoregressive transducer (hat)," in *ICASSP*. IEEE, 2020, pp. 6139–6143.

[13] B. Wang, Y. Yin, and H. Lin, "Attention-based transducer for online speech recognition," *arXiv preprint arXiv:2005.08497*, 2020.

[14] W. Huang, W. Hu, Y. T. Yeung, and X. Chen, "Conv-transformer transducer: Low latency, low frame rate, streamable end-to-end speech recognition," *arXiv preprint arXiv:2008.05750*, 2020.

[15] J. Yu, C.-C. Chiu, B. Li, S.-y. Chang, T. N. Sainath, Y. He, A. Narayanan, W. Han, A. Gulati, Y. Wu, et al., "FastEmit: Low-latency streaming asr with sequence-level emission regularization," in *ICASSP*. IEEE, 2021, pp. 6004–6008.

[16] B. Li, S.-y. Chang, T. N. Sainath, R. Pang, Y. He, T. Strohman, and Y. Wu, "Towards fast and accurate streaming end-to-end ASR," in *ICASSP 2020*. IEEE, 2020, pp. 6069–6073.

[17] C.-C. Chiu, T. N. Sainath, Y. Wu, R. Prabhavalkar, P. Nguyen, Z. Chen, A. Kannan, R. J. Weiss, K. Rao, E. Gonina, et al., "State-of-the-art speech recognition with sequence-to-sequence models," in *ICASSP*. IEEE, 2018, pp. 4774–4778.

[18] J. Shin, Y. Lee, and K. Jung, "Effective sentence scoring method using BERT for speech recognition," in *ACML*. PMLR, 2019, pp. 1081–1093.

[19] D. Fohr and I. Illina, "BERT-based semantic model for rescoring n-best speech recognition list," in *INTERSPEECH*, 2021.

[20] O. Hrinchuk, M. Popova, and B. Ginsburg, "Correction of automatic speech recognition with transformer sequence-to-sequence model," in *ICASSP*. IEEE, 2020, pp. 7074–7078.

[21] A. Kannan, Y. Wu, P. Nguyen, T. N. Sainath, Z. Chen, and R. Prabhavalkar, "An analysis of incorporating an external language model into a sequence-to-sequence model," in *Proc. ICASSP*. IEEE, 2018, pp. 5824–5828.

[22] A. Zeyer, K. Irie, R. Schlüter, and H. Ney, "Improved training of end-to-end attention models for speech recognition," *arXiv preprint arXiv:1805.03294*, 2018.

[23] D. Le, G. Keren, J. Chan, J. Mahadeokar, C. Fuegen, and M. L. Seltzer, "Deep shallow fusion for RNN-T personalization," in *SLT*. IEEE, 2021, pp. 251–257.

[24] M. Jain, G. Keren, J. Mahadeokar, G. Zweig, F. Metze, and Y. Saraf, "Contextual RNN-T for open domain ASR," *arXiv preprint arXiv:2006.03411*, 2020.

[25] G. Pundak, T. N. Sainath, R. Prabhavalkar, A. Kannan, and D. Zhao, "Deep context: end-to-end contextual speech recognition," in *SLT*. IEEE, 2018, pp. 418–425.

[26] K. Hu, T. N. Sainath, R. Pang, and R. Prabhavalkar, "Deliberation model based two-pass end-to-end speech recognition," in *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 7799–7803.

[27] K. Hu, R. Pang, T. N. Sainath, and T. Strohman, "Transformer based deliberation for two-pass speech recognition," in *SLT*. IEEE, 2021, pp. 68–74.

[28] S. Mavandadi, T. N. Sainath, K. Hu, and Z. Wu, "A deliberation-based joint acoustic and text decoder," in *Interspeech*, 2021.

[29] B. Li, A. Gulati, J. Yu, T. N. Sainath, C.-C. Chiu, A. Narayanan, S.-Y. Chang, R. Pang, Y. He, J. Qin, et al., "A better and faster end-to-end model for streaming ASR," in *ICASSP*. IEEE, 2021, pp. 5634–5638.

[30] A. Graves, "Sequence transduction with recurrent neural networks," *arXiv preprint arXiv:1211.3711*, 2012.

[31] R. Botros, T. N. Sainath, R. David, E. Guzman, W. Li, and Y. He, "Tied & reduced RNN-T decoder," *arXiv preprint arXiv:2109.07513*, 2021.

[32] Y. Xia, F. Tian, L. Wu, J. Lin, T. Qin, N. Yu, and T.-Y. Liu, "Deliberation networks: Sequence generation beyond one-pass decoding," in *Advances in Neural Information Processing Systems*, 2017, pp. 1784–1794.

[33] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *ICASSP*. IEEE, 2013, pp. 6645–6649.

[34] A. Narayanan, R. Prabhavalkar, C.-C. Chiu, D. Rybach, T. N. Sainath, and T. Strohman, "Recognizing long-form speech using streaming end-to-end models," in *ASRU*. IEEE, 2019, pp. 920–927.

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

[36] T. Bagby, K. Rao, and K. C. Sim, "Efficient implementation of recurrent neural network transducer in tensorflow," in *SLT*. IEEE, 2018, pp. 506–512.

[37] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al., "TensorFlow: A system for large-scale machine learning," in *12th USENIX Symposium on OSDI*, 2016, pp. 265–283.

[38] J. Shen, P. Nguyen, Y. Wu, Z. Chen, M. X. Chen, Y. Jia, A. Kannan, T. Sainath, Y. Cao, C.-C. Chiu, et al., "Lingvo: A modular and scalable framework for sequence-to-sequence modeling," *arXiv preprint arXiv:1902.08295*, 2019.

[39] B. T. Polyak and A. B. Juditsky, "Acceleration of stochastic approximation by averaging," *SIAM journal on control and optimization*, vol. 30, no. 4, pp. 838–855, 1992.

[40] T. N. Sainath, Y. He, B. Li, A. Narayanan, R. Pang, A. Bruguier, S.-y. Chang, W. Li, R. Alvarez, Z. Chen, et al., "A streaming on-device end-to-end model surpassing server-side conventional model quality and latency," in *ICASSP*. IEEE, 2020, pp. 6059–6063.

[41] C. Kim, A. Misra, K. Chin, T. Hughes, A. Narayanan, T. N. Sainath, and M. Bacchiani, "Generation of large-scale simulated utterances in virtual rooms to train deep-neural networks for far-field speech recognition in Google home," in *Proc. Interspeech*, 2017, pp. 379–383.

[42] D. Yu, M. L. Seltzer, J. Li, J.-T. Huang, and F. Seide, "Feature learning in deep neural networks-studies on speech recognition tasks," *arXiv preprint arXiv:1301.3605*, 2013.

[43] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *arXiv preprint arXiv:1904.08779*, 2019.

[44] D. Zhao, T. N. Sainath, D. Rybach, P. Rondon, D. Bhatia, B. Li, and R. Pang, "Shallow-fusion end-to-end contextual biasing.," in *Interspeech*, 2019, pp. 1418–1422.

[45] 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.