INTERSPEECH 2023
20-24 August 2023, Dublin, Ireland

Time-synchronous one-pass Beam Search for Parallel Online and Offline Transducers with Dynamic Block Training

Yui Sudo[1], *Muhammad Shakeel*[1], *Yifan Peng*[2], *Shinji Watanabe*[2]

[1]Honda Research Institute Japan Co., Ltd., Saitama, Japan
[2]Language Technologies Institute, Carnegie Mellon University, Pittsburgh, PA, USA
{yui.sudo, shakeel.muhammad}@jp.honda-ri.com, yifanpen@andrew.cmu.edu, shinjiw@ieee.org

Abstract

End-to-end automatic speech recognition (ASR) has become an increasingly popular area of research, with two main models being online and offline ASR. Online models aim to provide real-time transcription with minimal latency, whereas offline models wait until the end of the speech utterance before generating a transcription. In this work, we explore three techniques to maximize the performance of each model by 1) proposing a joint parallel online and offline architecture for transducers; 2) introducing dynamic block (DB) training, which allows flexible block size selection and improves the robustness for the offline mode; and, 3) proposing a novel time-synchronous one-pass beam search using the online and offline decoders to further improve the performance of the offline mode. Experimental results show that the proposed method consistently improves the character/word error rates on the CSJ and LibriSpeech datasets.

Index Terms: speech recognition, transducer, beam search, conformer, blockwise

1. Introduction

End-to-end automatic speech recognition (ASR) has garnered significant attention as a practical system, and various models such as connectionist temporal classification (CTC), recurrent neural network transducers (RNN-T), attention mechanism, and CTC/attention have been actively studied [1–5]. Although these models have shown significant performance improvements, minimizing latency is also an important issue for the wider practical application of ASR systems. To address this issue, many studies have focused on developing end-to-end ASR models that can perform streaming processing [6–10].

The simplest approach is the unidirectional long short term memory (LSTM) [6]. Causal transformer/conformer [7] enables streaming processing, but restricts context, resulting in performance degradation compared to the full context model. Another approach is to limit the window length of self-attention in transformers and conformers using blockwise processing. This method has been used in systems based on hidden Markov models, [11, 12], RNN-T [8, 13], attention [9], and CTC/attention [10,14–16]. Although these methods outperform causal-based methods at the sacrifice of mildly increasing the latency, it remains difficult for these methods to outperform full-context models.

Several attempts have been made to improve the performance of the online and offline model by combining them. One approach is to jointly train the decoders for online and offline processing with multitask learning and use two-pass rescoring to improve the performance of the offline mode [17–19]. The performance of the two-pass rescoring is highly dependent on the first pass, because it cannot be corrected in the second pass unless it includes the correct hypothesis in the first pass.

The second approach uses a cascaded encoder structure, in which the online and offline encoders are connected in a cascaded manner [20, 21]. The full-context encoder in this method corrects the hidden state vector output by the causal encoder in the offline mode. This cascaded encoder structure has also been extended to chunk-based encoders with two different chunk sizes [22, 23]. It has been reported that this structure provides significant performance gains for the offline mode, but only marginal gains for the online mode [20]. In addition, the chunk-based cascaded encoder structure only allows predetermined chunk size combinations. ASR systems typically have different accuracy and latency requirements for different scenarios, which increases model training overhead.

The third approach improves the online performance via knowledge distillation (KD) [24–26]. In KD-based methods, the offline model is used as the teacher model to improve the performance of the online mode (student model). Although KD is effective in bridging the performance gap between online and offline modes, it may also transfer errors from the teacher model to the student model. In addition, it is usually difficult to improve the performance of the teacher model with KD-based methods because the distillation loss between the teacher and student models is backpropagated only to the student model. If we can take better advantage of both online and offline modes, we can further improve the performance of both modes.

This paper proposes a novel joint online/offline transducer model that uses a blockwise and full-context encoders in parallel to exploit the advantage of both online and offline modes (Figure 1b). Furthermore, the online and offline models are jointly optimized by dynamically selecting the block size during training - we refer to this as dynamic block (DB) training [27, 28]. Unlike cascaded encoder-based methods [20, 21], the proposed method improves the online mode and allows flexible block size selection during inference through DB training. The DB training also improves the offline mode, unlike the KD-based methods [24–26], as it increases the variability of the encoder output during training. In addition, the one-pass beam search algorithm, which incorporates both online and offline decoders further enhances the complementarity of each decoder in the offline mode. Our contributions are as follows:

- We demonstrate that the proposed joint model improves the performance of both online and offline modes.
- We introduce the DB training to allow flexible block size selection and improves the robustness of the offline mode.
- We propose a novel one-pass beam search to further improve the performance of the offline mode.

10.21437/Interspeech.2023-1333

2. Preliminary

This section describes blockwise and full-context conformer-transducer [15, 29, 30], which are used in the proposed method.

2.1. Full-context conformer encoder

The conformer [29, 30] encoder comprises two convolutional layers for downsampling, a linear projection layer, and a positional encoding layer, followed by N conformer blocks. The convolutional layers take an audio feature sequence, $\boldsymbol{X}$, and subsample it. The conformer blocks then transform this subsampled feature sequence to an T-length hidden states $\boldsymbol{H}_{\text{off}} = [\boldsymbol{h}_1, ..., \boldsymbol{h}_T]$ as:

$$\boldsymbol{H}_{\text{off}} = \text{FullEnc}(\boldsymbol{X}). \tag{1}$$

Each conformer block has two feed forward layers, a multi-headed self-attention layer, a convolution layer, and a layer-normalization layer, with residual connections.

2.2. Blockwise conformer encoder

The encoder can be computed blockwise for streaming scenarios as described in [15]. Let L_{block} and L_{hop} represent the block size and hop length, respectively. The b-th block of the input audio feature seuence $\boldsymbol{X}_b$ is defined as,

$$\boldsymbol{X}_b = (\boldsymbol{X}_t | t = (b-1)L_{\text{hop}}+1, ..., (b-1)L_{\text{hop}}+L_{\text{block}}+1) \tag{2}$$

The corresponding hidden state for the b-th block are encoded for each block containing L_{block}-length hidden states. This process is performed sequentially and finally yields a T-length hidden state described as follows,

$$\boldsymbol{H}_{\text{on}} = \text{BlockEnc}(\boldsymbol{X}). \tag{3}$$

2.3. Transducer decoder

The transducer decoder consists of a prediction and a joint network. The prediction network generates a high-level representation g_s by conditioning on the previous nonblank token sequence $\boldsymbol{g}_{s-1}$, where s represents a nonblank token index described as follows:

$$g_s = \text{PredNet}(\boldsymbol{g}_{s-1}). \tag{4}$$

The joint network is a feed-forward network that combines $\boldsymbol{h}_t$ and g_s in Eq. (4) described as follows:

$$\boldsymbol{z}_{t,s} = \text{JointNet}(\boldsymbol{h}_t, g_s). \tag{5}$$

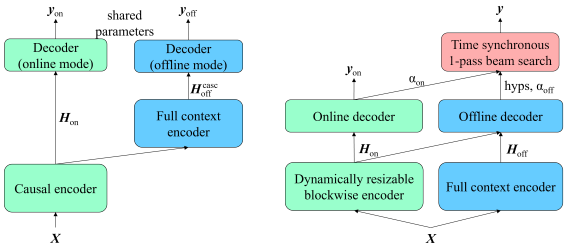
The transducer model marginalizes the potential alignments $\boldsymbol{z}$ that output $\boldsymbol{y}$ as follows:

$$P(\boldsymbol{y} \mid \boldsymbol{H}) = \sum_{\boldsymbol{z} \in \mathcal{Z}(\boldsymbol{y})} P(\boldsymbol{z}|\boldsymbol{H}) = \sum_{\boldsymbol{z} \in \mathcal{Z}(\boldsymbol{y})} \left[\prod_{i=1}^{T+S} P\left(\boldsymbol{z}_i \mid \boldsymbol{h}_{t_i}, g_{s_i}\right) \right], \tag{6}$$

where S denotes the total length of the complete token sequence, and i represents a position in $(T+S)$-length alignment path specified by t_i-th decoder state and s_i-th token, respectively. For both online and offline transducers, the model parameters are optimized by minimizing the negative log-likelihood as follows:

$$L_{\text{on}} = -\log P_{\text{on}}(\boldsymbol{y}_{\text{on}} \mid \boldsymbol{H}_{\text{on}}), \tag{7}$$

$$L_{\text{off}} = -\log P_{\text{off}}(\boldsymbol{y}_{\text{off}} \mid \boldsymbol{H}_{\text{off}}). \tag{8}$$



(a) *Cascaded encoder [20].* (b) *Proposed model.*

Figure 1: *Comparison of the proposed model with the cascaded encoder: The proposed model is jointly trained using dynamic block training and decoded using one-pass beam search.*

3. Proposed method

Figure 1b illustrates the architecture of the proposed method, which includes blockwise and full-context encoders capable of handling both online and offline modes. The audio feature sequence $\boldsymbol{X}$ is fed in parallel to the blockwise and full-context encoder. In the online mode, the blockwise encoder produces a hidden state vector, $\boldsymbol{H}_{\text{on}}$, which is fed to the online decoder, as in the conventional online RNN-T which corresponds to the green part in Figure 1b. In the offline mode, the hidden state vectors $\boldsymbol{H}_{\text{on}}$ and $\boldsymbol{H}_{\text{off}}$ in Eq. (1), (3) are vertically stacked, unlike the cascaded encoder (Figure 1a) as follows:

$$\boldsymbol{H}_{\text{off}}^{\text{casc}} = \text{FullEnc}(\boldsymbol{H}_{\text{on}}), \tag{9}$$

$$\boldsymbol{H}_{\text{off}}^{\text{para}} = \text{concat}(\boldsymbol{H}_{\text{on}}, \boldsymbol{H}_{\text{off}}). \tag{10}$$

The proposed structure avoids the error accumulation that often occurs in cascade structures. In addition, it is designed with the concept that the blockwise and full context encoders can better extract local and global features, respectively. The concatenation of the hidden state vectors from both encoders, $\boldsymbol{H}_{\text{on}}$ and $\boldsymbol{H}_{\text{off}}$, enables offline mode using both local and global features. Multitask learning of online and offline outputs can improve the robustness of the blockwise encoder as described in Section 3.1. A one-pass beam search is then used to further improve offline mode performance, as described in Section 3.2. Note that separate transducer decoders are used in this study to avoid online and offline modes interfering with each other.

3.1. Joint training with dynamic block selection

The proposed model is optimized by multitask learning using the weighted sum of losses as shown in Eqs. (7), (8) and described below:

$$L = \lambda L_{\text{on}} + (1-\lambda) L_{\text{off}}, \tag{11}$$

where λ represents the training weights. Similar to [27, 28], we used the DB training by dynamically changing L_{block} in Eq. (2). The proposed DB training randomly selects a block size from $[L_{\text{min}}$ - $L_{\text{max}}]$ for each batch during training. This technique improves the robustness of the offline mode by increasing the variability of the output of the blockwise encoder during training. Moreover, it also allows flexible block size selection during inference without compromising performance. In other words, the block size can be flexibly adjusted according to accuracy and latency requirements.

3.2. Joint decoding using one-pass beam search

We also propose a novel one-pass beam search, which combines online and offline transducer decoders to improve offline mode

Algorithm 1 Proposed one-pass beam search

1: $hyps$ = {<blank>:1.0}
2: **for** $t = 1$ to T **do**
3: $A = hyps$; ext_hyps = {}
4: **while** ext_hyps contains less than k_{pre} elements more probable than the most probable in A **do**
5: l = most probable in A; remove l from A
6: **for** $\boldsymbol{z}_t \in$ top-k_pre($P_{\text{off}}(\boldsymbol{z}_t)$) **do**
7: Add l and score to ext_hyps (if $\boldsymbol{z}_t$ is ϕ else Add $\boldsymbol{l} \oplus \boldsymbol{z}_t$ to A)
8: **end for**
9: **end while**
10: **for** $\boldsymbol{l} \in ext\_hyps$ **do**
11: $\alpha_{\text{off}} = ext\_hyps[\boldsymbol{l}]$
12: $\alpha_{\text{on}} = \text{OnScore}(\boldsymbol{l}, \boldsymbol{h}_{\text{on}}, P_{\text{on}}(\boldsymbol{z}_{t-1}))$
13: $ext\_hyps[\boldsymbol{l}] = \mu\alpha_{\text{on}} + (1-\mu)\alpha_{\text{off}}$
14: **end for**
15: $hyps$ = top-k(ext_hyps, k_{beam})
16: **end for**
17: **return** $hyps$

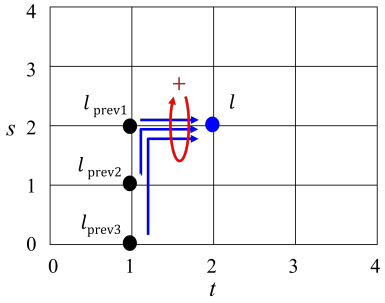


Figure 2: *RNN-T scoring: For the hypotheses generated by the offline transducer decoder at t, the probability of generating the same hypothesis from the previous hypotheses is computed.*

performance, as described in Algorithm 1. This method differs from other one-pass beam search methods proposed in [31–34] in that it scores the hypotheses using two RNN-T decoders. Moreover, unlike the label-synchronous one-pass beam search proposed in [34], which computes the probability of all possible alignment paths in CTC prefix scoring, the proposed method avoids this inefficient scoring using time synchrony.

The proposed method uses the offline transducer decoder as the primary decoder, which generates the initial hypotheses (lines 4-9 in Algorithm 1). The online transducer decoder is used to score the generated hypotheses to search more probable hypotheses described as OnScore(·) in line 12 of Algorithm 1. Specifically, the probability of generating ext_hyps from the previous hypotheses with the probabilities, $P_{\text{on}}(\boldsymbol{z}_{t-1})$, held at time $t-1$ are computed for the hypotheses generated by the offline transducer decoder at time t. Because the transducer decoder may emit multiple tokens for each time frame, all possible paths must be added together (Figure 2). Joint score, α_{joint}, is then calculated using a decoding weight μ described as,

$$ext\_hyps[\boldsymbol{l}] = \alpha_{\text{joint}} = \mu\alpha_{\text{on}} + (1-\mu)\alpha_{\text{off}}. \quad (12)$$

The joint score, α_{joint}, is used to retain the top k_{beam} hypotheses, $hyps$, for the next time frame (line 15 in Algorithm 1), where k_{beam} represents the beam size.

Table 1: *Effect of the proposed method on the CSJ (CER). DBT represents the DB training.*

Mode	Method	eval1	eval2	eval3	ave
Online	Separated	7.44	5.46	12.70	8.53
	Cascaded encoder [20]	8.11	5.62	13.66	9.13
	Parallel encoder (ours)	**7.24**	5.53	12.40	8.39
	+ DBT (ours)	7.50	**5.45**	**12.20**	**8.38**
Offline	Separated	5.78	4.15	9.94	6.62
	Cascaded encoder [20]	5.72	4.07	9.81	6.53
	Parallel encoder (ours)	5.72	4.11	10.33	6.72
	+ 2-pass rescoring	5.64	4.13	10.25	6.67
	+ 1-pass (ours)	5.71	4.03	10.15	6.63
	+ DBT (ours)	5.65	4.06	9.87	6.53
	+ DBT + 1-pass (ours)	**5.55**	**3.89**	**9.65**	**6.37**

4. Experiments

4.1. Experimental setup

The input features consisted of 80-dimensional Mel-scale filterbank features with a window size of 512 samples and a hop length of 160 samples, sampled at 16 kHz, followed by SpecAugment [35]. Both full-context and blockwise encoders consisted of two convolutional layers with stride two and a 256-dimensional linear projection layer followed by 12 conformer layers with 1024 linear units. During training, the block size, L_{block} in Eq. (2), was randomly selected from [5 - 50] using the DB training as described in Section 3.1. For both transducer decoders, a single LSTM layer with a hidden size of 256 and a linear layer of 320 joint sizes was used for prediction and joint networks. The proposed joint model was trained for 50 epochs with the Adam optimizer at a learning rate of 0.0015 and a training weight, λ in Eq. (11), of 0.5. The decoder weight for the online decoder, μ in Eq. (12), was 0.3. The proposed method was tested using two datasets: Corpus of Spontaneous Japanese (CSJ) [36], and LibriSpeech 960 h [37]. For the CSJ corpus, we used 299 h of academic presentation speech data. The character/word error rate (CER/WER) were calculated for CSJ and LibriSpeech, respectively. The emission delay (ED) as defined in [38] is calculated to evaluate the latency of the online mode using a GPU (NVIDIA RTX3090). The ESPnet [39] toolkit was used for the evaluation.

4.2. Main results

Table 1 presents the experimental results on CSJ. The proposed method was compared to other approaches such as separately trained online/offline transducers and a cascaded encoder. For a fair comparison we implemented the cascaded encoder as proposed in [20] and used same number of encoder layers as proposed in our architecture.

The proposed method, which utilized three proposed techniques – the parallel encoder structure, DB training, and one-pass beam search, demonstrated the best performance in both online and offline modes. Furthermore, the proposed method also outperformed the cascaded encoder in both modes. In particular, the cascaded encoder did not show any improvement in the online mode, whereas the proposed method achieved performance improvement in both modes. In addition, the one-pass beam search, which tightly combines the online and offline modes, resulted in greater performance improvement than the two-pass rescoring.

4.3. Detail analysis of the DB training for online mode

Figure 3 shows the relationship between block size and CER during decoding on the CSJ (average of eval1-3). The blue line represents the proposed model with the DB training, while the

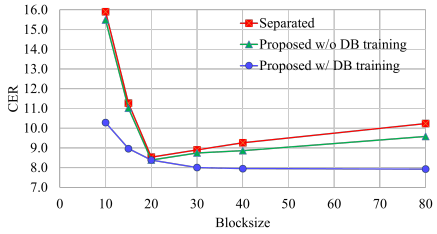


Figure 3: *Effect of the DB training for online mode.*

Table 2: *Comparison of emission delay for different block sizes.*

Block size	Block length (ms)	Separated CER	Separated ED (ms)	Proposed w/ DB CER	Proposed w/ DB ED (ms)
10	400	15.90	160	**10.29**	190
20	800	8.53	300	**8.38**	290
40	1600	9.27	470	**7.95**	460
80	3200	10.23	720	**7.93**	710

red and green lines represent the separate online model and the proposed model trained with a block size of 20, respectively. Although the proposed model without the DB training showed slightly better CER than the separate model, its performance deteriorated significantly for block sizes other than the predetermined block size of 20. In contrast, the proposed model with DB training outperformed the baseline for all block sizes, with a tradeoff between block size and CER.

Table 2 shows the ED measured using a GPU (NVIDIA RTX3090) in both the proposed model with DB training and the separate model. The proposed method significantly outperformed the separated model for all block sizes, while maintaining comparable ED at almost all block sizes, with a slight increase observed at a block size of 10.

4.4. Effect of the DB training for offline mode

Figure 4 shows the relationship between block size and CER for the offline mode on the CSJ (average of eval1-3). The red line represents a separate offline model, and the green and blue lines represent the proposed model using the proposed one-pass beam search with and without DB training, respectively. The proposed model trained with a block size of 20 slightly outperformed the separate model when the block size was 20. However, offline mode performance decreased for block sizes other than 20. The proposed DB training achieved equal or better CER than the separate model for all block sizes. Notably, the proposed DB training resulted in enhanced robustness of the offline mode, outperforming the baseline when the block size was greater than 10.

4.5. Effect of the one-pass beam search

We examined the effect of the decoder weight, μ, in Eq. (12) with the proposed one-pass beam search on the CSJ eval1. Figure 5 shows the relationship between the decoder weight and CER. Compared to the case with no decoder weight, (μ = 0), the CER improved as μ increased, with the smallest CER at μ = 0.3. This improvement suggests that the decoder weight can play a significant role in enhancing the performance of the one-pass beam search. The results demonstrate that the proposed method can enhance performance stably over a broad range of μ values ranging from 0 to 0.8. We also observed a similar trend with the Librispeech 960 h, which also showed the best CER when the decoder weights were 0.3.

4.6. Evaluation on LibriSpeech 960 h

Table 3 presents the evaluation of the proposed method on Librispeech 960 h. The results demonstrate that the proposed

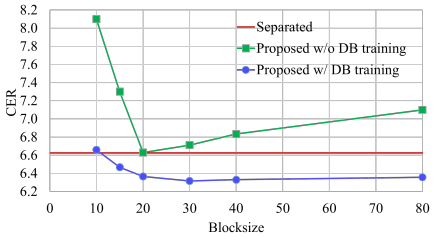


Figure 4: *Effect of the DB training for offline mode.*

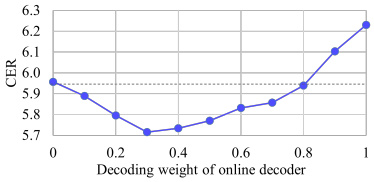


Figure 5: *Effect of the decoding weight.*

Table 3: *Evaluation (WER) on Librispeech 960 dev-clean (dc), dev-other (do), test-clean (tc), and test-other (to).*

Mode	Method	dc	do	tc	to
Online	Separated	3.64	9.91	3.80	9.67
	Cascaded encoder [20]	3.54	9.63	3.79	9.31
	Parallel encoder (ours)	**3.49**	**9.20**	**3.73**	**9.28**
Offline	Separated	2.37	5.85	2.66	5.82
	Cascaded encoder [20]	2.33	5.85	2.61	5.78
	Parallel encoder (ours)	2.31	5.72	2.49	5.73
	+ 2-pass rescoring	2.26	5.68	2.49	5.73
	+ 1-pass beam search (ours)	**2.19**	**5.54**	**2.35**	**5.57**

Ref: at	hayes in those	days must have been	quite an exciting	experience
A1:	that he is in those	days miss have been	quite in the suning	experience
A2:	saint hays in **thuther's**	days mess **I've** been **it**	quite in these suning	experience
B1:	aunt hazel and tho the	days must have been	quite and a shining	spirits
B2: at	hays in thursday's	days must have been	quite an his signing	experience
B3:	that hayes in those	days must have been	quite an his signing	experience

Figure 6: *Typical decoding results. A1/A2, B1/B2, represent the online/offline mode of the separate and proposed models, respectively, and B3 represents the one-pass beam search.*

method surpassed the baseline in both online and offline modes, which is consistent with the results of the CSJ evaluation. Moreover, the one-pass beam search algorithm further improved the performance in the offline mode.

Figure 6 shows typical decoding results for the separate models (A1, 2) and the proposed method (B1-3). A1 and B1 correspond to online mode, A2 and B2 represent offline mode, and B3 represents one-pass beam search results. The incorrect decoding results are highlighted in red, and results that were worse in the offline mode than in the online mode are further bolded. The separated offline model could have more errors than the separated online model, whereas the proposed method benefit from the both modes, allowing to effectively correct the errors in the offline mode.

5. Conclusion

This paper presented a joint online/offline transducer that integrates blockwise and full-context encoders. The joint training method improved the performance of both the online and offline modes. The proposed DB training technique enhanced the robustness of the offline mode while allowing for flexible block size selection for the online mode. Furthermore, the proposed one-pass beam search utilizing the online/offline transducer decoders further improved the performance of the offline mode.

6. References

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

[2] A. Graves, "Sequence transduction with recurrent neural networks," in *Proc. ICML*, 2012.

[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] S. Kim, T. Hori, and S. Watanabe, "Joint ctc-attention based end-to-end speech recognition using multi-task learning," in *Proc. ICASSP*, 2017, pp. 4835–4839.

[5] S. Karita, N. Chen, T. Hayashi, T. Hori, H. Inaguma, Z. Jiang, M. Someki, N. E. Y. Soplin, R. Yamamoto, X. Wang *et al.*, "A comparative study on Transformer vs RNN in speech applications," in *Proc. ASRU*, 2019, pp. 449–456.

[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, pp. 193–199.

[7] 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 *Proc. ICASSP*, 2021.

[8] L. Lu, C. Liu, J. Li, and Y. Gong, "Exploring transformers for large-scale speech recognition," *Proc. Interspeech*, pp. 5041–5045, 2020.

[9] R. Fan, P. Zhou, W. Chen, J. Jia, and G. Liu, "An online attention-based model for speech recognition," *Proc. Interspeech*, pp. 4390–4394, 2019.

[10] N. Moritz, T. Hori, and J. Le, "Streaming automatic speech recognition with the transformer model," in *Proc. ICASSP*, 2020, pp. 6074–6078.

[11] D. Povey, H. Hadian, P. Ghahremani, K. Li, and S. Khudanpur, "A time-restricted self-attention layer for ASR," in *Proc. ICASSP*, 2018, pp. 5874–5878.

[12] Y. Shi, Y. Wang, C. Wu, C.-F. Yeh, J. Chan, F. Zhang, D. Le, and M. Seltzer, "Emformer: Efficient memory transformer based acoustic model for low latency streaming speech recognition," in *Proc. ICASSP*, 2021.

[13] Y. Shi, V. Nagaraja, C. Wu, J. Mahadeokar, D. Le, R. Prabhavalkar, A. Xiao, C.-F. Yeh, J. Chan, C. Fuegen *et al.*, "Dynamic encoder transducer: A flexible solution for trading off accuracy for latency," in *Proc. Interspeech*, 2021, pp. 2042–2046.

[14] M. Li, C. Zorilă, and R. Doddipatla, "Head-synchronous decoding for transformer-based streaming asr," in *Proc. ICASSP*, 2021.

[15] E. Tsunoo, Y. Kashiwagi, T. Kumakura, and S. Watanabe, "Transformer ASR with contextual block processing," in *Proc. ASRU*, 2019, pp. 427–433.

[16] E. Tsunoo, Y. Kashiwagi, and S. Watanabe, "Streaming transformer ASR with blockwise synchronous beam search," in *Proc. SLT*, 2021, pp. 22–29.

[17] T. N. Sainath, R. Pang, D. Rybach, Y. He, R. Prabhavalkar, W. Li, M. Visontai, Q. Liang, T. Strohman, Y. Wu *et al.*, "Two-pass end-to-end speech recognition," in *Proc. Interspeech*, 2019, pp. 2713–2777.

[18] K. Hu, R. Prabhavalkar, R. Pang, and T. Sainath, "Deliberation model based two-pass end-to-end speech recognition," in *Proc. ICASSP*, 2020, pp. 7799–7803.

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

[20] 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 *Proc. ICASSP*, 2020, pp. 5629–5633.

[21] W. Wang, K. Hu, and T. N. Sainath, "Deliberation of streaming rnn-transducer by non-autoregressive decoding," in *Proc. ICASSP*, 2022, pp. 7452–7456.

[22] J. Mahadeokar, Y. Shi, K. Li, D. Le, J. Zhu, V. Chandra, O. Kalinli, and M. L. Seltzer, "Streaming parallel transducer beam search with fast-slow cascaded encoders," in *Proc. Interspeech*, 2022, pp. 2083–2087.

[23] K. Li, J. Mahadeokar, J. Guo, Y. Shi, G. Keren, O. Kalinli, M. L. Seltzer, and D. Le, "Improving fast-slow encoder based transducer with streaming deliberation," *arXiv preprint arXiv:2212.07650*, 2022.

[24] J. Yu, W. Han, A. Gulati, C.-C. Chiu, B. Li, T. N. Sainath, Y. Wu, and R. Pang, "Dual-mode ASR: Unify and improve streaming ASR with full-context modeling," in *Proc. ICLR*, 2021.

[25] N. Moritz, T. Hori, and J. L. Roux, "Dual causal/non-causal self-attention for streaming end-to-end speech recognition," in *Proc. Interspeech*, 2021, pp. 1822–1826.

[26] F. Weninger, M. Gaudesi, M. A. Haidar, N. Ferri, J. Andrés-Ferrer, and P. Zhan, "Conformer with dual-mode chunked attention for joint online and offline asr," in *Proc. Interspeech*, 2022, pp. 2053–2057.

[27] Z. Yao, D. Wu, X. Wang, B. Zhang, F. Yu, C. Yang, Z. Peng, X. Chen, L. Xie, and X. Lei, "Wenet: Production oriented streaming and non-streaming end-to-end speech recognition toolkit," in *Proc. Interspeech*, 2021, pp. 4045–4058.

[28] S. Horiguchi, S. Watanabe, P. García, Y. Takashima, and Y. Kawaguchi, "Online neural diarization of unlimited numbers of speakers using global and local attractors," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 31, pp. 706–720, 2023.

[29] P. Guo, F. Boyer, X. Chang, T. Hayashi, Y. Higuchi, H. Inaguma, N. Kamo, C. Li, D. Garcia-Romero, J. Shi *et al.*, "Recent developments on espnet toolkit boosted by conformer," in *Proc. ICASSP*, 2021, pp. 5874–5878.

[30] A. Gulati, C. Chiu, J. Qin, J. Yu, N. Parmar, R. Pang, S. Wang, W. Han, Y. Wu, Y. Zhang, and Z. Zhang, "Conformer: Convolution-augmented transformer for speech recognition," in *Proc. Interspeech*, 2020, pp. 5036–5040.

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

[32] B. Yan, S. Dalmia, Y. Higuchi, G. Neubig, F. Metze, A. W. Black, and S. Watanabe, "Ctc alignments improve autoregressive translation," *arXiv preprint arXiv:2210.05200*, 2022.

[33] Y. Sudo, M. Shakeel, B. Yan, J. Shi, and S. Watanabe, "4D ASR: Joint modeling of CTC, attention, transducer, and mask-predict decoders," *arXiv preprint arXiv:2212.10818*, 2022.

[34] S. Watanabe, T. Hori, S. Kim, J. R. Hershey, and T. Hayashi, "Hybrid ctc/attention architecture for end-to-end speech recognition," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 8, pp. 1240–1253, 2017.

[35] D. S. Park, W. Chan, Y. Zhang, C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," in *Proc. Interspeech*, 2019, pp. 2613–2617.

[36] K. Maekawa, "Corpus of spontaneous Japanese: Its design and evaluation," in *ISCA & IEEE Workshop on SSPR*, 2003.

[37] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *Proc. ICASSP*, 2015, pp. 5206–5210.

[38] J. Mahadeokar, Y. Shangguan, D. Le, G. Keren, H. Su, T. Le, C. feng Yeh, C. Fuegen, and M. L. Seltzer, "Alignment restricted streaming recurrent neural network transducer," in *Proc. SLT*, 2020, pp. 52–59.

[39] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. Enrique Yalta Soplin, J. Heymann, M. Wiesner, N. Chen, A. Renduchintala, and T. Ochiai, "ESPnet: End-to-end speech processing toolkit," in *Proc. Interspeech*, 2018, pp. 2207–2211.