Interspeech 2024
1-5 September 2024, Kos, Greece

Learning from Back Chunks: Acquiring More Future Knowledge for Streaming ASR Models via Self Distillation

Yuting Yang, Guodong Ma, Yuke Li, Binbin Du, Haoqi Zhu, Liang Ruan*

NetEase Yidun AI Lab, Hangzhou, China

{yangyuting04, maguodong, liyuke, dubinbin, zhuhaoqi, ruanliang}@corp.netease.com

Abstract

The performance of streaming automatic speech recognition (ASR) is often inferior to that of non-streaming speech recognition due to the absence of complete contextual information. However, we cannot optimize the model by merely accessing more future frames, as this would lead to considerable latency. In this paper, we propose **F**uture-**a**ware **T**ransformer (**FaT**) that models long-distance future contextual dependencies by transferring information from later chunks to former chunks through look-ahead windows. Specifically, the chunk-based context is used to encode audio sequence features. On this basis, the look-ahead window provides more context information for each chunk and acts as a bridge to progressively transfer long-distance future information from later chunks to earlier ones via a future-aware distillation mechanism. Experiments on AISHELL-1 and AISHELL-2 demonstrate that the proposed method achieves superior accuracy and better streaming latency than several strong baselines.

Index Terms: streaming ASR, future-aware Transformer, knowledge distillation

1. Introduction

Streaming speech recognition is of importance to bridge human-computer interaction, whose goal is to transcribe audio data into text in real-time as quickly and accurately as possible. It analyzes and transcribes spoken audio in real-time as it is being spoken, which poses a challenge to state-of-the-art end-to-end (E2E) ASR systems like Transformer [1, 2]. Specifically, the self-attention mechanism attends to all the position pairs of a sequence to summarize global context information, which is not feasible in real-time scenarios where the input is continuously streaming. There are several popular ASR systems adopting the Transformer backbone, e.g., Connectionist Temporal Classification (CTC) based models [3, 4], Transducer networks [5, 6], and some attention-based encoder-decoder models [2, 7]. Controlling the receptive field of the encoder module is essential for making these models suitable for streaming ASR tasks.

To address this challenge, researchers have proposed various methods to make the Transformer encoder suitable for streaming tasks. For instance, the left attention mechanism [8] masks future context to control time cost. Look-ahead methods [9, 10] set a right window equally for each frame to incorporate future context information. However, the receptive field of each frame grows linearly with the number of layers, which results in a significant latency. In addition, chunk-based methods [11–14] split the sequence into several chunk blocks and process sequentially. Typically, there are overlaps between the chunks in which each chunk is comprised of three components:

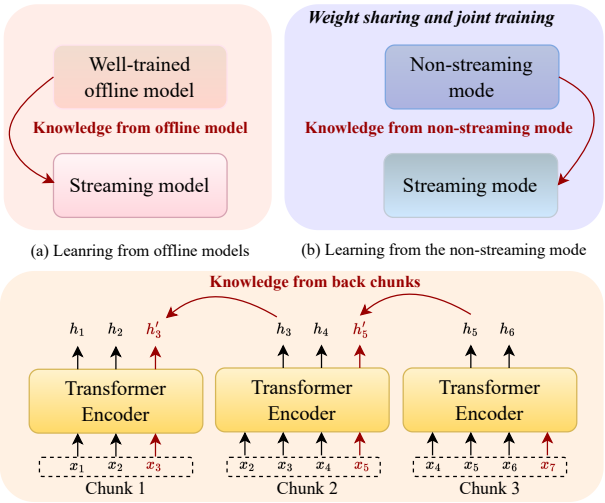

Figure 1: *A figure of different knowledge distillation methods for streaming ASR. ↗ indicates the transfer of knowledge from teachers to students. (a) knowledge distillation from a well-trained offline model. (b) knowledge distillation from the non-streaming mode. (c) A brief illustration of our **F**uture-**a**ware **T**ransformer (**FaT**) method, which can be figuratively interpreted as the act of gradually **fat**tening up each chunk by feeding it with future knowledge.*

the current part, the past part, and the future part. Specifically, the current part serves as the output position of the chunk, while the other two parts provide context for its calculation without producing any output. Emformer [15] proposes an efficient memory transformer to enhance the long-range history context for streaming ASR tasks. However, it does not address the issue of long-range future information modeling.

There has been a growing interest in utilizing knowledge distillation [16] to improve the performance of streaming ASR, which encourages student networks to mimic the behavior of teacher networks, thereby transferring the knowledge from the teacher network to the student network. Researchers have proposed various knowledge distillation methods to improve the performance of streaming ASR systems. These methods can be broadly divided into two categories: (1) Knowledge distillation from the offline model [17–20]. As depicted in Figure 1(a), it employs a well-trained offline model as the teacher model. However, this approach necessitates the additional training of an offline model capable of serving as an effective teacher. (2) Knowledge distillation from the non-streaming mode [8, 21]. As illustrated in Figure 1(b), it leverages the knowledge from non-streaming mode into streaming mode, which build upon the

* Corresponding Author.

10.21437/Interspeech.2024-1527

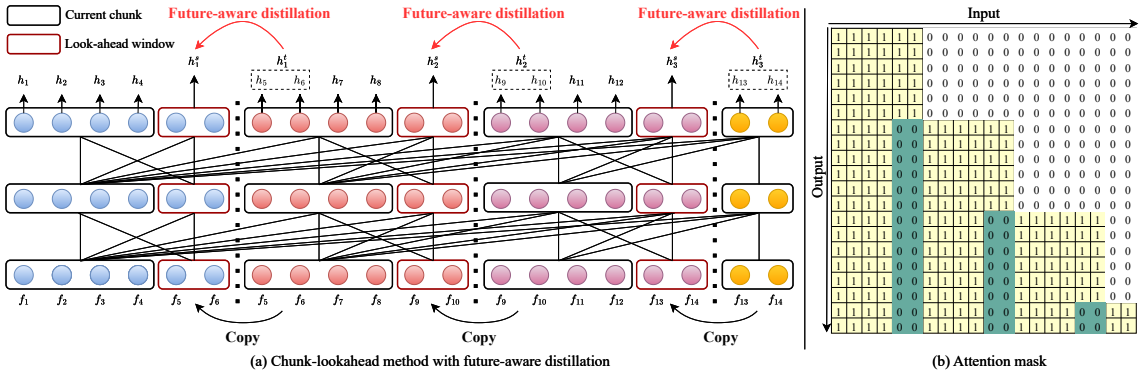

Figure 2: *A schematic diagram of the proposed method.* ***Left****: The proposed* ***FaT*** *method with efficient chunk-lookahead training and future-aware distillation. In this work, we adopt two CNN layers before the Transformer encoder to subsample the sequence, and f denotes the frames after downsampling. The current chunk size is 4, the look-ahead (right context) size is 2, and the full left context is used for each chunk.* ***Right****: An illustration of attention mask M used for self-attention layer. If $M(i, j)$ is 1, then the j-th input will be used for computation in the i-th frame.*

weight sharing and joint training mechanisms.

To compensate for the performance degradation caused by the lack of future information, in this work, we propose a novel self-distillation framework named **F**uture-**a**ware **T**ransformer (**FaT**). FaT is designed to capture long-distance future context dependencies for streaming ASR by transferring the knowledge from back chunks (teacher) to previous chunks (student). As illustrated in Figure 1(c), FaT adopts the chunk-based method and sets a look-ahead window for each chunk to increase the amount of available right context, which we named as **chunk-lookahead** in this work. Based on the chunk-lookahead method, FaT extract knowledge from the upcoming portion of the sequence (teacher) and transfer it to look-ahead windows (student), motivating the student to replicate the behavior of the teacher who attends to more future information of the sequence. Therefore, our FaT gains awareness of long-distance future knowledge through a series of look-ahead windows that propagate information in a cascading manner.

We conducted extensive experiments on AISHELL-1 and AISHELL-2 corpora to verify the effectiveness of our method. FaT achieves obvious accuracy improvements and slightly better streaming latency over the baseline models. Specifically, FaT achieves CER of 5.2%/5.9% on the AISHELL-1 dev/test sets and 6.4%/6.4% on the AISHELL-2 dev/test sets with a latency of 640ms, without using external language models (LM).

2. Background

Transformer is a popular neural network (NN) architecture for sequence modeling tasks, such as natural language processing and speech recognition. It utilizes self-attention mechanisms to capture the dependencies between all positions in a given sequence. It adopts the encoder-decoder structure, both of which are composed of multiple identical layers. Given a speech sequence, the encoder block extracts acoustic features represented as h. Then, the decoder module predicts the next word y_i based on the given acoustic representations h and previous words $y_{1:i-1}$. Specifically, an encoder layer of the Transformer comprises a self-attention module followed by a feed-forward module. The decoder layer is composed of three modules stacked together, i.e., a self-attention module, a source-target multi-head attention module, and a feed-forward module.

The key for Transformer encoder suitable for streaming tasks is to control the receptive field of the self-attention module, which is usually achieved by a predefined mask matrix. Formally, the masked self-attention mechanism (detailed in [1]) can be defined as

$$\text{Attention}(Q, K, V) = \text{Softmax}(\frac{mask(QK^T)}{\sqrt{d_k}})V, \quad (1)$$

where $\{Q, K, V\}$ are query, key and value vectors transformed from the input of the encoder layer.

3. Proposed Method

We use the feature sequence $\mathbf{x} = \{x_1, x_2, ..., x_T\}$ and the text sequence $\mathbf{y} = \{y_1, y_2, ..., y_n\}$ to optimize the network parameters during the training process. Figure 2 briefly shows the proposed **FaT** model. Specifically, our method adopts Transformer backbone, integrating an efficient chunk-lookahead training method and a future-aware distillation mechanism.

3.1. Efficient Training with Chunk-lookahead

Inspired by Emformer [15], we employ a *copy* mechanism to enable efficient parallel processing of the input sequence $\mathbf{x}$ for the chunk-lookahead method, without the need to physically divide it into multiple chunks and process them sequentially. As illustrated in Figure 2, we utilize a reconstruction of the input sequence and define a corresponding mask matrix used in Eq.1.

As depicted in Figure 2(a), we extract a few adjacent frames from the following chunk as the look-ahead frames, these frames are subsequently appended at the end of the current chunk to provide additional context, thereby augmenting the context information of the current chunk. Additionally, we incorporate an attention mask to determine the range of the input sequence for computation in each Transformer layer. The mask design in this study is illustrated in Figure 2(b). We assign a value of 0 to the corresponding position in the mask matrix to ignore the look-ahead windows of other chunks during attention calculation, which we highlight with a green background color.

3.2. Future-aware Distillation for Streaming ASR

In this work, we utilize the look-ahead window to incorporate more contextual information and to facilitate the propagation of future information from back chunks to front chunks of the sequence. The latter is achieved by the proposed future-aware knowledge distillation mechanism.

As shown in Figure 2(a), we consider the look-ahead frames as the student and the same frames in the next chunk as the

teacher, encouraging the student to reproduce the behaviour of the teacher who attends to more contextual information of the sequence. For example, the student h_1^s that attends to contextual information from f_1 to f_6, is encouraged to learn from the teacher h_1^t that attends to contextual information from f_1 to f_{10}. Similarly, the frames in h_1^t perceive the context information beyond the current chunk through the look-ahead window h_2^s. As shown by the red arrow in Figure 2(a), the look-ahead window builds a cascading bridge for information to pass from the back chunks to the front chunks.

We gather the encoder module's outputs from the look-ahead window as h^s, and the corresponding frames from the next chunk as h^t. The student can acquire knowledge from the teacher by minimizing the distillation objective. In this work, both the prediction-layer's probabilities distillation and the feature distillation are considered.

Prediction-Layer Distillation. We transfer knowledge from the teacher to the student by enforcing their predictions to be close, which is measured by KL-divergence:

$$\mathcal{L}_{\text{distill}} = \text{KLD}(\text{z}^{\text{t}}, \text{z}^{\text{s}}), \tag{2}$$

where z^s and z^t are defined as

$$\begin{aligned} \text{z}^s &= \text{Softmax}(\text{Linear}_{\text{d}_\text{m}\rightarrow|\text{V}|}(\text{h}^{\text{s}})), \\ \text{z}^t &= \text{Softmax}(\text{Linear}_{\text{d}_\text{m}\rightarrow|\text{V}|}(\text{h}^{\text{t}})), \end{aligned} \tag{3}$$

where d_m and $|V|$ are the dimensions of the attention layer and vocabulary, respectively.

Feature Distillation. We also investigate the feature based distillation to encourage that the feature knowledge can be transferred from the teacher to the student. Specifically, the learning objective is defined as

$$\mathcal{L}_{\text{distill}} = \text{MSE}(\text{h}^{\text{s}}, \text{h}^{\text{t}}), \tag{4}$$

where $\text{MSE}(\cdot)$ indicates the mean square estimation.

For the streaming ASR with future-aware distillation, the total loss function is a combination of the ASR loss $\mathcal{L}_{\text{ASR}}$ and the distillation loss $\mathcal{L}_{\text{distill}}$. Formally,

$$\mathcal{L} = \mathcal{L}_{\text{ASR}} + \mathcal{L}_{\text{distill}}. \tag{5}$$

4. Experiments

We conduct experiments on two ASR datasets, AISHELL-1 [22] and AISHELL-2 [23]. AISHELL-1 contains over 170 hours of Mandarin speech data from 400 speakers. For AISHELL-2, we utilize all the training data (1000 hours) for training, dev_ios for validation, and test_ios sets for evaluation. We apply speed perturbation [24] and SpecAugment [25] to the training data. For all experiments, the input speech features are 80-dimensional filterbank (FBank) features computed on a 25ms window with a 10ms shift. The vocabulary includes 4232 characters for AISHELL-1, and 5207 characters for AISHELL-2, respectively.

4.1. Experimental Setup

We conduct experiments using Wenet [26]. We adopt the hybrid CTC/Attention [27] architecture, and the ASR loss in Eq.5 is the combined CTC and attention losses. Specifically, we use two CNN-based downsampling layers to conduct four times downsampling. We utilize a 12-layer encoder and a 6-layer decoder for all experiments. For AISHELL-1, the dimension of the attention layer is 256 with 4 heads, and the dimension of the feed-forward layer is 2048. For AISHELL-2, except for the utilization of 512 attention layer dimension with 8 heads, the rest is the same as AISHELL-1. We utilize the Adam optimizer [28] with 25000 warmup steps, and we basically follow the same training settings as in the recipe. We train the models for 240 epochs on the AISHELL-1 dataset and 130 epochs on the AISHELL-2 dataset, respectively.

Table 1: *The settings of chunk (c) and look-ahead (la) for different models, and the latency of different chunk settings during inference.*

Models	Training		Inference
	c	la	Decoding windows
Baseline	$\{16, 12, 8\}$	$\{0\}$	$\{640ms, 480ms, 320ms\}$
Baseline_la	$\{16, 12, 8\}$	$\{4\}$	$\{800ms, 640ms, 480ms\}$
FaT	$\{16, 12, 8\}$	$\{4\}$	$\{800ms, 640ms, 480ms\}$

We adopt the dynamic chunk training [29] approach for all of our experiments. The specified chunk and look-ahead sizes are sampled from the corresponding set as shown in Table 1. Additionally, Table 1 shows the latency of different models under the chunk settings of $\{16, 12, 8\}$, respectively. For decoding, we show the first-pass results with CTC prefix beam search [30] and the second-pass results with attention rescore [29], respectively. The 4-gram LM used in Table 3 is trained with training transcriptions.

Measuring Emission Latency. Motivated by [31–33], we measure First Token emission Delay (FTD), Last Token emission Delay (LTD), and Average Token emission Delay (ATD). FTD is defined as the difference of two timestamps: (1) when the first token is emitted, (2) when the user speak the first token. Similarly, LTD is defined as the emission latency for the last token and ATD is defined as the average transmission delay of all tokens. We report both median (P50) and 90th percentile (P90) of all utterances in AISHELL-1 test set.

4.2. Main Results

Table 2 shows the CERs (Character Error Rates) on AISHELL-1 and AISHELL-2 datasets. From the table, our **FaT** model achieves CER of 6.1%/6.8% with the CTC prefix beam search decoding and 5.2%/5.9% with the attention rescore decoding on the AISHELL-1 dev/test sets, under the inference setting ($chunk\ size = 12, look\text{-}ahead\ size = 4$) with a latency of 640ms. FaT outperforms both the *Baseline* model and the *Baseline_la* model with the same latency. Moreover, our method obviously outperforms (median relative improvement of 4%) the $Baseline\_la$ model under the same experimental settings for fair comparison, without increasing parameters and computational complexity.

The experimental results from the table show that our model is also effective on the AISHELL-2 corpus, which contains more training data than AISHELL-1. Specifically, FaT achieves the lowest CER (i.e., 6.4%/6.4% on dev/test sets with a latency of 640ms) among all other baseline models. Additionally, all experimental results from Table 2 show that both the feature distillation and the prediction-layer distillation make improvements toward streaming ASR performance. Relatively, feature distillation achieves better performance in more scenarios. Furthermore, consistent improvements in all experiments show that the future-aware distillation mechanism makes the model benefit from the knowledge from the "future", which alleviates the performance degradation caused by the absence of future information in streaming ASR.

Table 3 shows the comparisons with recently published streaming systems on the AISHELL-2 test set. Specifically,

Table 2: *CER on the AISHELL-1 and AISHELL-2 datasets, respectively, with different inference settings of* ***chunk*** *(c) and* ***look-ahead*** *(la).* ***FaT-fea*** *and* ***FaT-pred*** *denote feature distillation and prediction-layer distillation, respectively. The results are obtained w/o LM.*

Models	la	AISEHLL-1 $c=16$ dev / test	AISEHLL-1 $c=12$ dev / test	AISEHLL-1 $c=8$ dev / test	AISHELL-2 $c=16$ dev / test	AISHELL-2 $c=12$ dev / test	AISHELL-2 $c=8$ dev / test
♠ The first-pass decoding with CTC prefix beam search							
Baseline	0	6.8% / 7.4%	7.0% / 7.7%	7.4% / 8.2%	8.0% / 8.1%	8.1% / 8.2%	8.6% / 8.6%
Baseline_la	4	6.4% / 7.0%	6.4% / 7.1%	6.7% / 7.4%	7.7% / 7.7%	8.0% / 7.9%	8.1% / 8.0%
FaT-pred (ours)	4	**6.1%** / 6.8%	**6.1%** / 6.9%	**6.3% / 7.1%**	7.7% / **7.4%**	7.8% / **7.5%**	7.9% / **7.6%**
FaT-fea (ours)	4	**6.1% / 6.7%**	**6.1% / 6.8%**	**6.3% / 7.1%**	**7.3% / 7.4%**	**7.5%** / 7.6%	**7.6%** / 7.7%
♠ The second-pass decoding with attention rescore							
Baseline	0	5.7% / 6.1%	5.8% / 6.4%	6.0% / 6.6%	6.8% / 6.7%	6.9% / 6.8%	7.2% / 7.2%
Baseline_la	4	5.4% / 6.0%	5.4% / 6.1%	5.6% / 6.3%	6.7% / 6.6%	6.8% / 6.7%	6.9% / 6.8%
FaT-pred (ours)	4	5.3% / 5.9%	5.3% / **5.9%**	**5.4% / 6.1%**	6.5% / **6.4%**	6.6% / 6.5%	6.7% / 6.6%
FaT-fea (ours)	4	**5.2% / 5.8%**	**5.2% / 5.9%**	**5.4% / 6.1%**	**6.3% / 6.4%**	**6.4% / 6.4%**	**6.5% / 6.5%**

Table 3: *Compare with published works on AISHELL-2 test set.* $\star$ *denotes an estimated value based on the literature's settings.*

Methods	Latency	Test
CIF [34]	2560ms*	6.04%
Universal ASR [35]	-	6.15%
U2++ Transformer [36]	640ms	6.70%
U2 Transformer [29]	640ms	7.31%
U2++ Conformer [36]	640ms	**5.78%**
Future-aware Transformer (ours)	640ms	6.4%
+ 4-gram LM		**5.9%**

our method achieves a CER of 5.9% on the AISHELL-2 test set with a 4-gram LM for shallow fusion. The results show that FaT's performance is better than other Transformer-based models on the AISHELL-2 test set. When compared to the SOTA model U2++ Conformer [36], which adopts stronger backbones and additional right-to-left decoder rescoring, our method achieves comparable results(ours 5.9% vs 5.78%) to it, also demonstrating FaT's effectiveness in the field.

4.3. Analyses

4.3.1. Ablation Study

We study the impact of different self distillation methods on the AISHELL-1 dataset, which includes learning from non-streaming mode as well as from back chunks (our proposed method). When utilizing the non-streaming mode as the teacher, the model is trained with weight sharing and joint training [8] mechanisms, and the experimental setting is the same as in Section 4.1. The results are shown in Table 4, both the use of non-streaming mode as the teacher and back chunks as the teacher (our FaT) improve the accuracy of the streaming model. However, our proposed method demonstrates superior performance in achieving this objective, which suggests the effectiveness of the FaT model.

4.3.2. Emission Latency

We compare the emission latency of FaT and two baseline models ($Baseline$, $Baseline\_la$) on the AISHELL-1 test set, with the inference setting of chunk size 16. From Table 5, we can see that $Baseline\_la$ achieves low latency compared with $baseline$. This is because look-ahead windows introduce additional future information for each chunk, and improved contextual capture results in faster emission. It's important to note that ASR models that capture more extensive contexts may generate the token even before it is spoken, resulting in negative latency. Moreover, FaT has better streaming emission latency than the $Baseline\_la$ model, demonstrating FaT benefits from richer contextual information introduced by our future-aware distillation mechanism. FaT has higher accuracy and slightly better streaming latency clearly demonstrates the effectiveness of our method.

Table 4: *Ablation study of different self distillation methods on the AISHELL-1 test set. 1st pass and 2st pass stand for CTC prefix beam search and attention rescoring, respectively.*

Teacher	Decoding mode	Chunk size 16	12	8
Non-streaming mode	1st pass	6.8%	6.9%	7.4%
	2st pass	5.9%	6.0%	6.3%
Back chunks (Our FaT)	1st pass	6.7%	6.8%	7.1%
	2st pass	**5.8%**	**5.9%**	**6.1%**

Table 5: *Emission latency on the AISHELL-1 test set.*

Models	FTD (ms) P50	FTD (ms) P90	LTD (ms) P50	LTD (ms) P90	ATD (ms) P50	ATD (ms) P90
Baseline	50	110	-50	40	9	29
Baseline_la	0	50	-100	-50	-40	-23
FaT (ours)	-10	40	-100	-50	-45	-28

5. Conclusions

This paper proposes a Future-aware Transformer (FaT) model for streaming ASR tasks. The goal is to address the challenge of modeling long-distance future context dependencies in real-time scenarios where traditional self-attention mechanisms are not feasible. The FaT model adopts a chunk-lookahead approach to control the range of the self-attention mechanism and introduces a copy mechanism to enable efficient parallel processing. Additionally, a future-aware knowledge distillation approach is proposed to transfer knowledge from the upcoming portion of the sequence to the look-ahead windows, allowing the model to replicate the behavior of attending to more contextual information. Thus, the FaT model gains awareness of long-distance future knowledge through cascading look-ahead windows. Experiments demonstrate that the proposed FaT achieves remarkable results on both AISHELL-1 and AISHELL-2 test sets. We believe that extending our method to more network architecture (like Conformer [37]) and multi-level distillation knowledge fusion will be interesting future work.

6. References

[1] A. Vaswani, N. Shazeer, N. Parmar *et al.*, "Attention is all you need," in *NIPS*, 2017.

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

[3] A. Graves, S. Fernández, F. Gomez *et al.*, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 369–376.

[4] Y. Yang, Y. Li, and B. Du, "Improving ctc-based asr models with gated interlayer collaboration," in *ICASSP*. IEEE, 2023, pp. 1–5.

[5] E. Battenberg, J. Chen, R. Child *et al.*, "Exploring neural transducers for end-to-end speech recognition," in *2017 ASRU*. IEEE, 2017, pp. 206–213.

[6] C.-F. Yeh, J. Mahadeokar, K. Kalgaonkar *et al.*, "Transformer-transducer: End-to-end speech recognition with self-attention," *arXiv preprint arXiv:1910.12977*, 2019.

[7] Y. Higuchi, S. Watanabe, N. Chen *et al.*, "Mask CTC: Non-Autoregressive End-to-End ASR with CTC and Mask Predict," in *Proc. Interspeech 2020*, 2020, pp. 3655–3659.

[8] J. Yu, W. Han, A. Gulati *et al.*, "Dual-mode asr: Unify and improve streaming asr with full-context modeling," in *ICLR*, 2021.

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

[10] Q. Zhang, H. Lu, H. Sak *et al.*, "Transformer transducer: A streamable speech recognition model with transformer encoders and rnn-t loss," in *ICASSP*. IEEE, 2020, pp. 7829–7833.

[11] H. Miao, G. Cheng, C. Gao *et al.*, "Transformer-based online ctc/attention end-to-end speech recognition architecture," in *ICASSP*. IEEE, 2020, pp. 6084–6088.

[12] L. Dong, F. Wang, and B. Xu, "Self-attention aligner: A latency-control end-to-end model for asr using self-attention network and chunk-hopping," in *ICASSP*. IEEE, 2019, pp. 5656–5660.

[13] E. Tsunoo, Y. Kashiwagi, T. Kumakura *et al.*, "Towards online end-to-end transformer automatic speech recognition," *arXiv preprint arXiv:1910.11871*, 2019.

[14] Z. Dai, Z. Yang, Y. Yang *et al.*, "Transformer-xl: Attentive language models beyond a fixed-length context," in *ACL*, 2019, pp. 2978–2988.

[15] Y. Shi, Y. Wang, C. Wu *et al.*, "Emformer: Efficient memory transformer based acoustic model for low latency streaming speech recognition," in *ICASSP*. IEEE, 2021, pp. 6783–6787.

[16] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[17] G. Kurata and G. Saon, "Knowledge distillation from offline to streaming rnn transducer for end-to-end speech recognition." in *Proc. Interspeech 2020*, 2020, pp. 2117–2121.

[18] S. Panchapagesan, D. S. Park, C.-C. Chiu, Y. Shangguan, Q. Liang, and A. Gruenstein, "Efficient knowledge distillation for rnn-transducer models," in *ICASSP*. IEEE, 2021, pp. 5639–5643.

[19] T. Doutre, W. Han, M. Ma, Z. Lu, C.-C. Chiu, R. Pang, A. Narayanan, A. Misra, Y. Zhang, and L. Cao, "Improving streaming automatic speech recognition with non-streaming model distillation on unsupervised data," in *ICASSP*. IEEE, 2021, pp. 6558–6562.

[20] K. Shim, J. Lee, S. Chang, and K. Hwang, "Knowledge distillation from non-streaming to streaming asr encoder using auxiliary non-streaming layer," *arXiv preprint arXiv:2308.16415*, 2023.

[21] C. Liang, X.-L. Zhang, B. Zhang, D. Wu, S. Li, X. Song, Z. Peng, and F. Pan, "Fast-u2++: Fast and accurate end-to-end speech recognition in joint ctc/attention frames," in *ICASSP*. IEEE, 2023, pp. 1–5.

[22] H. Bu, J. Du, X. Na *et al.*, "Aishell-1: An open-source mandarin speech corpus and a speech recognition baseline," in *2017 20th conference of the oriental chapter of the international coordinating committee on speech databases and speech I/O systems and assessment (O-COCOSDA)*. IEEE, 2017, pp. 1–5.

[23] J. Du, X. Na, X. Liu, and H. Bu, "Aishell-2: Transforming mandarin asr research into industrial scale," *arXiv preprint arXiv:1808.10583*, 2018.

[24] T. Ko, V. Peddinti, D. Povey *et al.*, "Audio augmentation for speech recognition," in *Proc. Interspeech 2015*, 2015, pp. 3586–3589.

[25] D. S. Park, W. Chan, Y. Zhang *et al.*, "Specaugment: A simple data augmentation method for automatic speech recognition," in *Proc. Interspeech 2020*, 2019.

[26] Z. Yao, D. Wu, X. Wang *et al.*, "Wenet: Production oriented streaming and non-streaming end-to-end speech recognition toolkit," in *Proc. Interspeech*. Brno, Czech Republic: IEEE, 2021.

[27] S. Watanabe, T. Hori, S. Kim *et al.*, "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.

[28] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2015.

[29] B. Zhang, D. Wu, Z. Yao *et al.*, "Unified streaming and non-streaming two-pass end-to-end model for speech recognition," *arXiv preprint arXiv:2012.05481*, 2020.

[30] A. Y. Hannun, A. L. Maas, D. Jurafsky *et al.*, "First-pass large vocabulary continuous speech recognition using bi-directional recurrent dnns," *arXiv preprint arXiv:1408.2873*, 2014.

[31] S.-Y. Chang, B. Li, D. Rybach, Y. He, W. Li, T. N. Sainath, and T. Strohman, "Low latency speech recognition using end-to-end prefetching." in *Proc. Interspeech 2020*, 2020, pp. 1962–1966.

[32] Y. Shangguan, R. Prabhavalkar, H. Su, J. Mahadeokar, Y. Shi, J. Zhou, C. Wu, D. Le, O. Kalinli, C. Fuegen *et al.*, "Dissecting user-perceived latency of on-device e2e speech recognition," *arXiv preprint arXiv:2104.02207*, 2021.

[33] X. Song, D. Wu, Z. Wu *et al.*, "Trimtail: Low-latency streaming asr with simple but effective spectrogram-level length penalty," in *ICASSP*. IEEE, 2023, pp. 1–5.

[34] L. Dong and B. Xu, "Cif: Continuous integrate-and-fire for end-to-end speech recognition," in *ICASSP*. IEEE, 2020, pp. 6079–6083.

[35] Z. Gao, S. Zhang, M. Lei *et al.*, "Universal asr: Unifying streaming and non-streaming asr using a single encoder-decoder model," *arXiv preprint arXiv:2010.14099*, 2020.

[36] D. Wu, B. Zhang, C. Yang *et al.*, "U2++: Unified two-pass bidirectional end-to-end model for speech recognition," *arXiv preprint arXiv:2106.05642*, 2021.

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