SCALE: SEMANTIC CHUNKING AND LABEL-DELAY ENGINE FOR STREAMING SPEECH-LLM

Akshat Jaiswal, Debmalya Chakrabarty, Ritwik Kotra, Harish Arsikere, Nikhil Bhave, Sambuddha Bhattacharya, Sri Garimella

Amazon AGI, Bangalore
{akshatkj, debmac, ksritwik, arsikere, nikbha, bsambudd, srigar}@amazon.com

ABSTRACT

Streaming automatic speech recognition (ASR) systems based on Large Language Models (LLMs) face a fundamental trade-off between accuracy and latency. Existing approaches typically employ fixed-size chunking to maintain low latency, which often compromises recognition accuracy. We propose **SCALE**, a streaming ASR framework that addresses this challenge through three key techniques: (a) dynamic chunk boundary prediction leveraging semantic information, replacing rigid fixed-size chunking, (b) intra-chunk bidirectional attention mechanism for efficient acoustic context modeling, and (c) label delay training strategy enabling stable word predictions with smaller chunks. Our experiments show consistent improvements in streaming ASR performance, with up to 25% relative reduction in Word Error Rate (WER) compared to existing LLM based streaming ASR systems reported in literature.

Index Terms— Semantic-aware chunking, dynamic lookahead, label-delay mechanism, SpeechLLM

1. INTRODUCTION

Large language models (LLMs) trained on massive text corpora have demonstrated remarkable generalization across diverse language tasks. By conditioning decoder-only LLMs on audio encoder embeddings, "SpeechLLMs" achieve competitive ASR performance within a unified generative framework [1, 2, 3, 4]. However, these models exhibit fundamental limitations in long-form and streaming scenarios: premature end-of-sequence predictions due to weak length extrapolation [5, 6], quadratically scaling inference costs [7, 8], and high user-perceived latency due to non-streaming architectures [9, 10, 11].

Recent streaming approaches, including SpeechLLM-XL [8], CUSIDE [12], and WeNet [13], implement fixed-size chunking and controlled lookahead mechanisms [14]. While these methods achieve controllable latency and reduced computation costs, their rigid time-based segmentation introduces systematic weaknesses: linguistic boundary misalignment, unstable edge predictions in causal encoders [15, 16], and premature word-end commitments [17, 18]. These limitations necessitate streaming frameworks that better align with natural speech structure.

In this work, we propose **SCALE**, the *Semantic Chunking and Label-delay Engine*, an efficient dynamic lookahead framework for streaming ASR that addresses the existing limitations of current chunk based methodologies. First, we introduce semantic-aware chunking that aligns segmentation with natural speech boundaries rather than fixed time intervals [19, 20]. Second, we develop a Universal Acoustic Encoder (UAE) with variable attention masking that enables flexible intra-chunk lookahead while maintaining streaming constraints [21, 22]. Finally, we propose a label-delay mechanism that defers word-end commitments, allowing the incorporation of limited future context to improve recognition accuracy without compromising real-time operation [23, 24]. While prior works have explored individual aspects like chunk-based attention [13, 25] or controlled lookahead [14], SCALE provides a unified framework that holistically addresses the challenges of streaming ASR through intelligent segmentation, adaptive encoding, and robust label assignment.

In addition, we propose a hardware-agnostic latency formulation that decomposes word-level delay into chunking and compute components, enabling reproducible estimation and systematic tuning of streaming LLM systems. To the best of our knowledge, this is the first work to jointly analyze encoder-side lookahead and decoder-side deferred supervision under this unified formulation. This combined treatment provides clear insights into how these factors interact, allowing us to identify optimal operating points for streaming Speech LLMs. Section 2 describes the proposed methodology, detailing our unified framework of adaptive chunking, lookahead-aware encoding, and label delay. Section 3 presents experimental results and ablation studies, analyzing accuracy-latency trade-offs. Section 4 concludes with key findings and future directions.

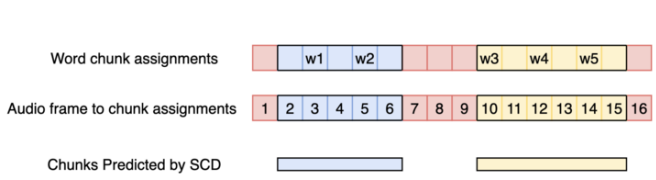
2. METHODOLOGY

Our framework extends SpeechLLM-XL's streaming design with three key components: (i) a semantic-aware chunk detector (SCD) that identifies linguistically stable boundaries, (ii) a Universal Acoustic Encoder (UAE) with intra-chunk lookahead for improved speech representations, and (iii) a label-delay mechanism that enables future context incorporation while maintaining bounded latency. Additionally, we propose a hardware-independent latency formulation to enable systematic optimization of streaming configurations.

2.1. Streaming-aware Speech LLMs

A streaming speech–LLM [8] processes audio incrementally, emitting tokens as input arrives rather than after the full utterance is observed. We divide the input sequence $\mathbf{x}_{1:T} = (x_1, \ldots, x_T)$ into consecutive chunks. Let t_k denote the end-frame index of the k-th chunk, so that $\{t_k\}_{k=1}^{m}$ represents the sequence of chunk boundaries. Each chunk k has length $L_k = t_k - t_{k-1}$ frames (with $t_0 = 0$), and corresponds to the subsequence, $\mathbf{x}^{(k)} = \mathbf{x}_{t_{k-1}+1:t_k}$. Each chunk is encoded with optional lookahead of δ frames:

$$\mathbf{h}^{(k)} = \mathrm{Enc}\big(\mathbf{x}_{t_{k-1}+1:t_k+\delta}\big). \quad (1)$$



(a) Semantic-Aware Chunking with full intra-chunk lookahead

(b) Semantic-Aware Chunking with mini-chunks and cross-boundary label delay

Fig. 1. (a) Semantic-aware chunking (SCD) with full intra-chunk lookahead: frames are grouped into semantically aligned segments (e.g., frames 1–16), with words w_1, w_2 in the first segment and w_3–w_5 in the second. Frames 1, 7–9, and 16 denote non-speech regions excluded from SCD chunks. (b) Buffer-capped mini-chunks with label delay τ: words crossing mini-chunk boundaries are deferred to subsequent chunks to avoid premature word finalization.

The decoder-only LLM then generates a block of tokens $\mathbf{y}^{(k)} = (y_1^{(k)}, \ldots, y_{n_k}^{(k)})$, conditioned on the current embeddings $\mathbf{h}^{(k)}$ and a truncated history $\mathcal{C}_{\leq k-1}^{(W)}$ consisting of the most recent W tokens from prior chunks:

$$p(\mathbf{y}^{(k)} \mid \mathbf{h}^{(k)}, \mathcal{C}_{\leq k-1}^{(W)}) = \prod_{i=1}^{n_k} p\Big(y_i^{(k)} \mid \mathbf{y}_{<i}^{(k)}, \mathbf{h}^{(k)}, \mathcal{C}_{\leq k-1}^{(W)}\Big). \quad (2)$$

Restricting the decoder context to W tokens allows keeping the memory usage in check. This chunked, limited-context design forms the basis of streaming LLM decoding.

2.2. Semantic-Aware Dual Mode Universal Acoustic Encoder

We initialize our Universal Acoustic Encoder (UAE) from a large self-supervised model (BEST-RQ [26]) and adapt it with a supervised RNN-T ASR objective. During this training, we apply *variable attention masking* [21] so the encoder learns to operate under different context granularities, ranging from fully causal to fully non-causal. This yields a single encoder that produces representations robust to both local acoustics and longer-range semantics, and can flexibly run in causal, lookahead, or fully non causal modes. From this UAE, we derive our semantic chunk detector (SCD) and bounded lookahead encoder.

Semantic-Aware Chunk Detection (SCD). We operate the UAE in *causal mode* and take the hidden representation from its final block at each frame. Two lightweight heads are attached: a Voice-Activity-Detection (VAD) head for short-term silence/onset and an End of Utterance (EoU) head for longer prosodic or semantic pauses. Both heads are trained with a binary cross-entropy (BCE) [27] objective using targets obtained from a forced aligner, while the UAE itself remains frozen. Their outputs are combined into a boundary score [19]

$$s_t = \alpha\, p_t^{\text{eou}} + (1-\alpha)\, p_t^{\text{vad}}, \quad (3)$$

where α is tuned on a development set. A new chunk is emitted whenever $s_t \geq \theta$, with θ selected to balance false positives and boundary recall. This ensures that boundaries align with stable acoustic and semantic cues rather than arbitrary time windows. In practice, this fusion helps the model fire only at semantically meaningful pauses instead of reacting to every momentary silence. Since SCD is implemented via lightweight auxiliary heads operating on frozen encoder representations, it introduces minimal additional compute or memory overhead.

Bounded Lookahead Encoder. Our speech encoder employs a hybrid attention mechanism inspired by DuRep [21]. For a given speech segment [u,v] and a past context window Δ, all frames maintain uniform attention boundaries: left context to $u - \Delta$ and right context to v. This design ensures adherence to streaming constraints while enabling full bidirectional attention within the current chunk and limited past context.

The architecture offers several advantages: it eliminates fixed future context waiting times (static lookaheads), enables efficient inference with a fixed-size Δ KV cache, and allows single-pass forward computation without causal masks. Notably, this design seamlessly accommodates causal (single-frame processing), chunked (segment-wise processing), and non-streaming (full-sequence processing) without architectural modifications, simplifying deployment across various inference scenarios while maintaining computational efficiency.

2.3. Label Delay for Sub-chunking

Semantic chunking can inherently produce longer segments, see Fig. 1(a), to manage latency while preserving streaming behavior we cap segment length at N frames and implement mini-chunks:

$$\mathbf{x}_{u:v}^{(k)} \subseteq \mathbf{x}^{(k)}, \quad (v-u+1) \leq N. \quad (4)$$

To address word splitting at chunk boundaries, we implement label delay during training. This mechanism shifts word start times forward by a fixed offset τ (320ms), effectively pushing boundary-straddling words into subsequent mini-chunks. The offset τ is optimized to balance responsiveness and prediction stability.

However, naive shifting of boundaries could encourage the model to perpetually defer word predictions. To prevent this behavior, we introduce a "⟨DND⟩" (do-not-delay) marker at each final mini-chunk, teaching the model to immediately flush pending tokens while maintaining delayed predictions at other boundaries.

During inference, chunks are processed immediately upon arrival, with the model having learned this balanced approach of delayed prediction and prompted flushing. As shown in Fig. 1(b), this strategy ensures robust boundary handling while maintaining causal, low-latency runtime behavior. The complete inference orchestration is detailed in Algorithm 1.

2.4. Latency Formulation for Streaming Constraints

Traditional streaming ASR latency metrics often combine both model-intrinsic delays and hardware-specific factors (e.g., TTFT, RTF) [9], motivating the need for more reproducible estimation frameworks. We propose a simple, analytical word-level metric that depends only on alignment and model characteristics, making it hardware-agnostic.

Consider an utterance $\mathbf{x}_{1:T}$ with transcript $\mathcal{W} = \{w_i\}_{i=1}^{n}$ and

Algorithm 1 Streaming ASR Inference using SCD and Label Delay

```
procedure PROCESSAUDIO(audio_stream, scd_model, speech_encoder, llm_decoder, max_buffer_size)
    outputs ← []
    frame_buffer ← []
    while frame ← next(audio_stream) do
        state ← scd_model(frame)
        if state = "outside_semantic_chunk" then
            encoded_frames ← speech_encoder([frame])
            llm_decoder(encoded_frames)                ▷ [Optional]
            continue
        frame_buffer.append(frame)
        if state = "end_of_semantic_chunk" or size(frame_buffer) = max_buffer_size then
            encoded_frames ← speech_encoder(frame_buffer)
            llm_decoder(encoded_frames)
            if state = "end_of_semantic_chunk" then
                llm_decoder([⟨DND⟩])
            outputs.append(generate(llm_decoder))
            frame_buffer.clear()
    return outputs
```

chunk boundaries $\{t_k\}_{k=1}^{m}$, where t_k denotes the end frame of the k-th chunk. The latency of a word w_i decomposes into chunking delay and compute delay such that $L(w_i) = L_{\text{chunk}}(w_i) + L_{\text{compute}}(w_i)$ **Chunking latency** is the delay between the ground-truth end of a word $T_{\text{end}}(w_i)$ and the next available chunk boundary:

$$L_{\text{chunk}}(w_i) = \min\{t_k \mid t_k \geq T_{\text{end}}(w_i)\} - T_{\text{end}}(w_i). \quad (5)$$

Compute latency captures encoder time-to-first-token (TTFT) and autoregressive decoding cost. Let $Q(w_j)$ be the number of tokens assigned to word w_j under the LLM tokenizer, and let TPOT denote the average time per output token. Then

$$L_{\text{compute}}(w_i) = L_{\text{encoding}} + \text{TPOT} \times \sum_{j \leq i} Q(w_j), \quad (6)$$

where L_{encoding} corresponds to TTFT measured once per chunk. This cumulative form reflects that all earlier tokens must be decoded before w_i.This formulation ties architectural choices such as SCD, mini-chunking, and label delay directly to latency, while remaining reproducible and independent of hardware.

3. EXPERIMENTS AND RESULTS

3.1. Model Architecture and Training

In this section, we detail the architecture and training methodology of our proposed SCALE framework. Firstly, input audio is converted to 128-dim log-Mel features (25 ms window/10 ms hop), stacked to a 40 ms frame rate. The Universal Acoustic Encoder (UAE; Sec. 2.2) is a 2B-parameter, 20-layer Conformer (d_{model}=512, d_{ff}=2048) with causal convolutions, pretrained with an RNN–T objective under *variable attention masking* (causal/chunked/bidirectional) [21]. For SCD, two lightweight heads (VAD/EoU) on UAE features yield boundary scores; heads use BCE with Montreal Forced Aligner (MFA) alignments [28]. The training data and framework for UAE is same as [21]. UAE embeddings are projected into a 2B-parameter multilingual in-house trained LLM using a MLP-based adapter. The UAE is kept frozen throughout, while the model training proceeds in two stages: (1) speech-text pre-training of the LLM and adapter using cross-entropy based next-token prediction loss [29], followed by (2) supervised fine-tuning on streaming and non-streaming ASR tasks. To ensure consistent comparison across ablation studies, all models use a fixed context length of 2048 tokens.

Our training methodology integrates dynamic chunk sizing and variable lookahead encoding. Audio inputs are segmented into chunks using SCD and mini-chunks are sampled from $\sim \mathcal{N}(2\text{s}, \sigma^2)$, with corresponding transcripts extracted via word alignments. Concurrently, audio frames are encoded using either no lookahead or a 520ms lookahead, alternating between these configurations. This approach optimizes the model's adaptability to diverse streaming conditions while maintaining efficiency. At inference, chunk size is bounded by either SCD-only, a 4 s cap, or a 2 s cap. Candidate model(s) with label delay use $\tau = 320$ ms and employ a ⟨DND⟩ token during both training and inference to flush deferred word finalizations (Algorithm 1), without introducing additional runtime buffering. The choice of τ was guided by a sweep over candidate values, where $\tau \approx 320$ ms yielded the best stability–latency trade-off.

We evaluate both **streaming** (SCD+mini-chunks) and **non-streaming** (N=∞) configurations. All models in our ablation studies were trained for 100k steps on 200k hours of multilingual data including various externally available datasets (VoxPopuli, MLS, CommonVoice, PeopleSpeech, FLEURS, LibriVox).

Buffer Size (s)	Chunk Latency (ms)			Compute Latency (ms)		
	mean	p50	p90	mean	p50	p90
5.20	2260	2050	4448	175	160	320
4.00	1829	1702	3493	148	140	260
2.00	988	967	1799	96	100	160
1.20	608	607	1094	74	60	120
0.80	410	406	732	60	60	100

Table 1. Hardware-agnostic latency decomposition into chunking and compute components under different buffer sizes

3.2. Results and Discussion

Latency budgets: Using our hardware-agnostic framework (Section 2.4) on the MLS English test set, we analyzed word-level latencies, with MFA alignments, assuming a conservative TPOT of 20ms/token [30]. Chunking latency dominates the overall delay as reported in Table 1, with 5.2s buffers resulting in 2.2s mean chunking latency versus only 175ms compute latency. Reducing buffer size to 2s brings mean latency below 1s with compute under 100ms. While smaller buffers (0.8s) further reduce latency, they risk accuracy degradation through aggressive segmentation.

For latency-accuracy trade-off, we focus on 1-4s buffer regime and evaluate across two parameters: encoder lookahead (affects compute latency) and chunking policy (determines chunking latency). We compare three configurations: fixed-length chunking baselines, SCD-guided chunking, and SCD-guided with 4s/2s sub-chunks, with varying encoder lookaheads, with and without label delay.

Word Error Rate Analysis: We evaluate SCALE on Multilingual LibriSpeech (MLS) [31] across five languages (en, de, es, fr, it). Table 2 reports WER under different chunking strategies, encoder lookahead settings, and with/without label delay (LD). For clarity, we summarize the findings along three key dimensions.

1. Impact of Encoder Lookahead: Future acoustic context significantly influences ASR performance across all chunking strategies.

Decoder Chunking Strategy	Buffer Size(s)	Encoder Lookahead(s)	Label Delay(s)	Language					Avg
				en	de	es	fr	it	
Fixed	∞	∞	-	4.8	4.8	3.2	5.3	8.6	5.3
		0.52	-	5.9	6.4	3.9	6.4	10.6	6.6
		0.04	-	6.2	6.9	4.1	7.3	10.9	7.0
	4.00	**4.00**	-	5.6	5.8	3.7	6.2	10.3	6.3
		0.04	-	7.9	7.6	4.8	7.9	14.3	8.5
	2.00	**2.00**	-	6.2	6.3	4.2	6.9	11.0	6.9
		0.04	-	8.3	9.1	6.4	10.0	14.7	9.7
SCD	**SCD**	**SCD**	-	5.5	5.4	3.5	5.9	9.7	6.0
		0.04	-	7.4	7.3	4.4	7.2	11.5	7.6
	4.00	**4.00**	-	6.0	5.9	4.0	6.4	10.4	6.6
		0.04	-	7.4	7.3	4.4	7.2	11.5	7.6
	2.00	**2.00**	-	6.6	6.6	4.6	7.4	11.3	7.3
			0.32	5.9	6.2	4.9	6.3	10.5	6.7
		0.04	-	8.1	8.4	5.8	9.2	15.2	9.3
			0.32	7.0	10.0	6.0	7.8	14.6	9.0

Table 2. Comparative analysis of ASR performance (WER) across different chunking strategies. For simplicity, we define Lookahead as the number of acoustic frames available to the encoder, including the current frame. Thus, for a 2.0 s chunk (40 ms frame rate), full encoder lookahead is reported as 2.0 s, even though the exact receptive field is 1.96 s.

Full lookahead configurations achieve strong accuracy (WER 5.3% offline, 6.0% with SCD-guided chunking), while zero lookahead leads to notable degradation (WER: 7.0% fixed, 7.6% SCD-guided). This performance gap demonstrates that future acoustic context serves as a crucial stabilizer, helping the encoder resolve coarticulation and prosodic cues near chunk boundaries.

2. Impact of Semantic Chunk Detection: Semantic chunking achieves near non-streaming performance (6.0% vs 5.3% WER), demonstrating effective boundary detection. However, latency constraints necessitate sub-chunking of longer segments, increasing WER to 6.6% at 4s buffer. Interestingly, fixed chunking (6.3%) outperforms semantically-guided sub-chunking (6.6%) at matched buffer sizes. To understand this counterintuitive behavior, consider a speech segment: "The committee's deliberation of the amendments [pause] recommended implementation." While SCD naturally segments this into say (4.8s, 1.2s), a 4s buffer forces splits into (4s, 0.8s, 1.2s), breaking mid-phrase at "amendments". This fragmentation of semantic units proves more detrimental than fixed-interval segmentation, as it disrupts the model's ability to maintain contextual coherence across related speech segments, thus motivating our label delay approach.

3. Impact of SCD + Label Delay (LD): Label delay effectively mitigates performance degradation at sub-chunk boundaries by offsetting word-finalization. At (2s lookahead, 2s buffer), LD improves WER from 7.3% to 6.7% (8% relative gain). Similar improvements are observed in zero-lookahead configurations, demonstrating LD's ability to compensate for both missing context and artificial segmentation. Using a $\tau = 320$ supervision delay with ⟨DND⟩ token, SCD+LD outperforms fixed chunking, particularly for English and Italian (average relative gain of $\sim 5\%$). While improvements are modest at relaxed settings (4s buffer), they become significant with tighter constraints (2s buffer), enabling sub-second latency without compromising accuracy.

Comparative performance: Operating under tight streaming constraints (1.2s lookahead, 1.2s buffer), SCALE achieves 2.2/5.0% WER on LibriSpeech test-clean/other, **outperforming previous streaming LLM systems**. Compared to SpeechLLM-XL (0.24/1.28s buffer, 2.7/6.7% WER), SCALE demonstrates relative improvements of 18% (clean) and 25% (other). It also surpasses non-LLM streaming baselines, showing 26% and 35% relative gains over Transducer (3.0/7.7% WER). While SCALE uses slightly larger encoder lookahead than SpeechLLM-XL, both maintain sub-1.3s latency. Our SCD-guided chunking with label delay optimization achieves state-of-the-art accuracy among reported streaming LLMs. While acknowledging differences in model sizes and training configurations across these systems, these results demonstrate the effectiveness of our proposed optimizations even with strong baseline models.

Model	Lookahead (s)	Chunk (s)	test clean	test other
LAS	—	—	2.8	6.8
CTC	0.24	1.28	3.6	8.8
Transducer	0.24	1.28	3.0	7.7
SpeechLLM-XL	0.24	1.28	2.7	6.7
Chunked AED	0.30	1.20	—	6.7
ReaLLM	0.96	1.92	3.0	7.4
CTC-prompt-LM	0.64	0.64	3.2	7.9
SCALE (ours)	**1.20**	**1.20**	**2.2**	**5.0**

Table 3. Benchmark comparison: WER results of SCALE and prior state-of-the-art streaming systems on LibriSpeech. Prior rows reproduced from SpeechLLM-XL [8]

4. CONCLUSION

We presented SCALE, a latency-aware streaming ASR framework that combines semantic-aware chunking, bounded intra-chunk lookahead, and label-delay mechanism. Together with an analytical latency formulation, SCALE provides a principled approach to balancing recognition accuracy with responsiveness in streaming SpeechLLMs. Results demonstrate that semantic chunking reduces boundary fragmentation, while label delay maintains accuracy under small buffer sizes, enabling sub-second latencies. Future directions include adaptive chunking policies informed by interaction cues, end-to-end latency-aware objectives, robustness under domain shift, and broader multilingual benchmarks.

5. REFERENCES

[1] Yuan Gong, Alexander H. Liu, Hongyin Luo, Leonid Karlinsky, and James Glass, "Joint audio and speech understanding," in *2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2023, pp. 1–8.

[2] Changli Tang, Wenyi Yu, Guangzhi Sun, Xianzhao Chen, Tian Tan, Wei Li, Lu Lu, Zejun Ma, and Chao Zhang, "Salmonn: Towards generic hearing abilities for large language models," *arXiv preprint arXiv:2310.13289*, 2023.

[3] Paul K Rubenstein, Chulayuth Asawaroengchai, Duc Dung Nguyen, et al., "Audiopalm: A large language model that can speak and listen," *arXiv preprint arXiv:2306.12925*, 2023.

[4] Yunfei Chu, Jin Xu, Qian Yang, Haojie Wei, Xipin Wei, Zhifang Guo, Yichong Leng, Yuanjun Lv, Jinzheng He, Junyang Lin, Chang Zhou, and Jingren Zhou, "Qwen2-audio technical report," *arXiv preprint arXiv:2407.10759*, 2024.

[5] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever, "Robust speech recognition via large-scale weak supervision," *arXiv preprint arXiv:2212.04356*, 2022.

[6] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, et al., "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023.

[7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[8] Junteng Jia, Gil Keren, Wei Zhou, Egor Lakomkin, et al., "Efficient streaming llm for speech recognition," in *ICASSP 2025 - 2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2025, pp. 1–5.

[9] Matt Shannon, Gabor Simko, Shuo-Yiin Chang, and Carolina Parada, "Improved end-of-query detection for streaming speech recognition," in *Interspeech 2017*, 2017, pp. 1909–1913.

[10] Cristian Garbacea, Ananya Rao, Lucas Rencker, Alexei Baevski, and Wei-Ning Hsu, "Real-time whisper: Streaming asr with better latency and less compute," *arXiv preprint arXiv:2311.12407*, 2023.

[11] Yong Zhao, Nan Li, Tianzi Zhang, Hui Wang, Cheng Li, Jindong Han, Jiaqi Zhu, and Binghuai Liu, "Speechllm: A conversational speech language model," *arXiv preprint arXiv:2308.06873*, 2023.

[12] Yushi Wu and Shinji Watanabe, "Cuside: Chunk-based unified speech interpretation and detection model," *arXiv preprint arXiv:2309.03883*, 2023.

[13] Binbin Zhang, Di Wu, Zhuoyuan Yao, et al., "Wenet: Production oriented streaming and non-streaming end-to-end speech recognition toolkit," *arXiv preprint arXiv:2102.01547*, 2021.

[14] Yiming Yang, Daniel Peng, Yiming Wang, and William Yang Wang, "Future-context attention for streaming speech recognition," *arXiv preprint arXiv:2401.03089*, 2024.

[15] Qian Zhang, Han Han, Jianfeng Li, and Rong Zhang, "Streaming transformer-based acoustic models using self-attention with augmented memory," *Proc. Interspeech 2020*, pp. 2132–2136, 2020.

[16] Aditya Tripathi, Junteng Han, and Alice Coucke, "Understanding and improving encoder behavior at chunk boundaries in streaming asr," *Proc. Interspeech 2022*, pp. 3668–3672, 2022.

[17] Ke Li, Zijian Shi, Tara N Sainath, Pang Ruoming, and Yu Wu, "Adaptive endpointing for online speech recognition," *Proc. Interspeech 2020*, pp. 956–960, 2020.

[18] Yunfei Wu, Binbin Zhang, Hui Yang, Chao Yuan, Zhendong Yan, and Yanjun Li, "Fastconformer: Fast streaming transformer acoustic model using block processing," *Proc. Interspeech 2022*, pp. 3673–3677, 2022.

[19] Ruiqing Zheng, Xiangyu Chen, Mingbo Guo, Yanli Ma, Mengge Liu, Minghan Zhang, Boxing Chen, and Chuang Huang, "Dynamic sentence boundary detection for real-time speech translation," *Proc. Interspeech 2021*, pp. 2247–2251, 2021.

[20] Elizabeth Salesky and Shinji Watanabe, "Streaming transformer asr with dynamic chunk size," *Proc. ICASSP 2021*, pp. 5939–5943, 2021.

[21] Prabash Reddy Male, Swayambhu Nath Ray, Harish Arsikere, et al., "Durep: Dual-mode speech representation learning via asr-aware distillation," *arXiv preprint arXiv:2505.19774*, 2025.

[22] Yongqiang Shi, Yu Zhang, Mingxing Cai, and Jinyu Liu, "Emformer: Efficient memory transformer for streaming asr," *Proc. Interspeech 2021*, pp. 3637–3641, 2021.

[23] Niko Moritz, Takaaki Hori, and Jonathan Le, "Streaming automatic speech recognition with the transformer model," *Proc. ICASSP 2020*, pp. 6074–6078, 2020.

[24] Aditya Tripathi, Junteng Han, et al., "Streaming end-to-end asr with joint ctc-attention," *Proc. Interspeech 2022*, pp. 3663–3667, 2022.

[25] Qian Zhang, Hao Lu, Hasim Sak, Anshuman Tripathi, Erik McDermott, Stephen Koo, and Shankar Kumar, "Transformer transducer: A streamable speech recognition model with transformer encoders and rnn-t loss," *Proc. ICASSP 2020*, pp. 7829–7833, 2020.

[26] Chung-Cheng Chiu, James Qin, Yu Zhang, Jiahui Yu, and Yonghui Wu, "Self-supervised learning with random-projection quantizer for speech recognition," in *International Conference on Machine Learning*. PMLR, 2022, pp. 3915–3924.

[27] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, "Deep learning," *MIT press*, 2016.

[28] Michael McAuliffe, Michaela Socolof, Sarah Mihuc, Michael Wagner, and Morgan Sonderegger, "Montreal forced aligner: Trainable text-speech alignment using kaldi," in *Interspeech 2017*, 2017, pp. 498–502.

[29] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, et al., "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[30] Artificial Analysis, "Llm speed benchmarks," `https://artificialanalysis.ai/models#speed`, 2025, Accessed: [September 17, 2025].

[31] Vineel Pratap, Qiantong Xu, Anuroop Sriram, Gabriel Synnaeve, and Ronan Collobert, "Mls: A large-scale multilingual dataset for speech research," 2020.