TC-BiMamba: Efficient Dynamic-Chunk Bidirectional Mamba for Unified Streaming and Offline ASR

Anonymous ACL submission

Abstract

Unified streaming and offline ASR aims to deliver full-context accuracy and low-latency incremental decoding with a single model. Transformer/Conformer systems are effective, but long-form streaming inference still suffers from growing attention masks, historical caches, and memory states. Latency-Controlled BiMamba (LC-BiMamba) offers attention-free bidirectional modeling, yet its backward branch is confined to fixed chunks, tying the model to one latency. Extending it to dynamic chunk training expands mini-batches and slows training. We propose TC-BiMamba, whose Trans-Chunk formulation casts local-future backward modeling as sequence-level dynamic-chunk training over complete sequences. TC-BiMamba supports multiple chunk-wise latency settings, keeps the encoder/CTC path memory-stable on long-form inputs, and improves training scalability. On AISHELL-1, AISHELL-2, and LibriSpeech, it outperforms U2++ and remains competitive with externally reported LC-BiMamba results using fewer parameters, while achieving 1.3$\times$ training speedup and 50% GPU memory reduction.

1 Introduction

Unified streaming and offline automatic speech recognition (ASR) aims to serve two deployment regimes with a single model: offline recognition can exploit full utterance context for accuracy, whereas streaming recognition must emit partial hypotheses under tight latency, memory, and compute budgets. End-to-end ASR has advanced rapidly with recurrent neural network transducers (Graves, 2012), transformers (Vaswani et al., 2017), and conformers (Gulati et al., 2020). Production-oriented toolkits such as WeNet operationalize unified streaming and non-streaming ASR, while dual-mode ASR and U2/U2++ instantiate this paradigm through full-context modeling, dynamic chunk training, CTC prefix beam search, and two-pass rescoring (Yu et al., 2021b; Yao et al., 2021; Zhang et al., 2022; Wu et al., 2021). Yet most unified ASR systems still rely on attention-based encoders or decoders. In long-form and industrial streaming scenarios, attention masks, left-context caches, and memory states introduce a latency–memory trade-off that becomes increasingly costly as the retained history grows (Vaswani et al., 2017; Shi et al., 2021).

Mamba and selective state space models (SSMs) offer an attention-free alternative for long-sequence modeling. Building on structured SSMs for efficient long-range sequence modeling, Mamba summarizes history with recurrent state transitions rather than explicit pairwise token interactions and uses hardware-aware parallel scan algorithms to achieve linear sequence complexity (Gu et al., 2022; Gu and Dao, 2023; Dao and Gu, 2024). This makes Mamba a natural candidate for long-utterance and streaming ASR. However, ASR is not merely a causal sequence modeling problem: acoustic disambiguation, boundary detection, and language-level prediction often benefit from local future context. A unidirectional Mamba is streaming-friendly but only observes the past, whereas a fully bidirectional Mamba can use future context but violates streaming latency constraints. The central question is therefore not simply whether Mamba can replace attention in ASR, but whether bidirectional Mamba can be made dynamically latency-controllable without the training overhead of chunk splitting.

Latency-Controlled BiMamba (LC-BiMamba) is the closest prior work to this setting. It constrains the backward Mamba within fixed chunks, allowing the model to exploit local future context while respecting a fixed streaming latency budget (Moriya et al., 2025). Despite this important step, LC-BiMamba leaves two challenges unresolved for unified ASR. First, fixed-chunk decoding is difficult to reconcile with deployments that

require different latency settings across devices, networks, and applications. Second, extending its chunk-wise backward processing to dynamic chunk training requires physically splitting sequences into many short chunks. To make the shape change explicit, let an encoder input be $X \in \mathbb{R}^{B\times T\times D}$, where B, T, and D denote the batch size, sequence length, and hidden dimension. With chunk size $C = 2$, for example, $X \in \mathbb{R}^{50\times 100\times 256}$ contains $N_c = \lceil T/C \rceil = 50$ chunks per utterance, so physical chunk splitting feeds the backward module a tensor of shape $\mathbb{R}^{(50\cdot 50)\times 2\times 256} = \mathbb{R}^{2500\times 2\times 256}$. This expands the number of sequences processed by the backward branch by a factor of N_c, degrading training throughput and memory efficiency.

We propose TC-BiMamba, an efficient dynamic-chunk bidirectional Mamba architecture for unified streaming and offline ASR. The key idea is not the reversal operation itself, but a sequence-level training formulation: LC-style local future modeling is expressed as a chunk-aware full-sequence ordering, so Mamba can train with continuous state propagation instead of treating each chunk as an independent short sequence. The forward Mamba scans the original temporal order, while the backward Mamba scans a reordered sequence obtained by reversing frames within each sampled chunk and then restores the outputs to the original temporal order. As illustrated in Fig. 1, this Trans-Chunk formulation preserves the sequence-level tensor shape, avoids mini-batch inflation, and keeps training closer to offline BiMamba while supporting flexible chunk-wise latency settings.

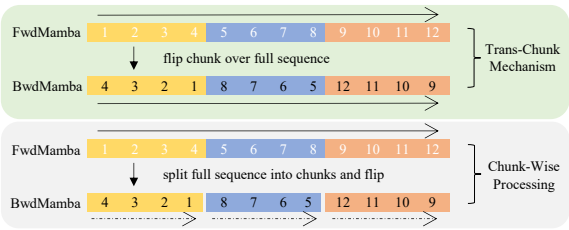

Figure 1: Illustration of the proposed Trans-Chunk mechanism and conventional chunk-wise processing with chunk size $C = 4$. Trans-Chunk formulates backward training as a full-length Mamba scan with chunk-aware ordering, whereas chunk-wise processing reshapes the backward direction into many mini-chunks, increasing training time and GPU memory consumption.

Our contributions are summarized as follows:

- **A unified dynamic-chunk architecture.** We introduce TC-BiMamba with a BiMamba–convolution encoder and Hybrid decoder, using Mamba as a latency-controllable bidirectional sequence model rather than a drop-in attention substitute, and supporting offline recognition and multiple streaming latency configurations within a single model.

- **A Trans-Chunk training formulation.** We reformulate LC-style local future modeling as sequence-level Mamba training, avoiding chunk-level mini-batch inflation and achieving 1.3× training speedup with about 50% lower GPU memory consumption.

- **Strong ASR performance at smaller scale.** Experiments on AISHELL-1, AISHELL-2, and LibriSpeech show that TC-BiMamba outperforms U2++ at comparable scale and remains competitive with externally reported LC-BiMamba results using substantially fewer parameters.

2 Related Work

Unified streaming and offline ASR. Unified ASR seeks to support full-context recognition and low-latency incremental decoding with shared parameters. Early streaming end-to-end systems commonly used RNN-T, monotonic chunk-wise attention, or low-latency transducer objectives to enforce causality and encourage timely emissions (Graves, 2012; Chiu and Raffel, 2018; Yu et al., 2021a). With the success of Transformer and Conformer encoders (Vaswani et al., 2017; Gulati et al., 2020), subsequent work unified streaming and non-streaming modes through attention masks, limited left context, memory caches, and dynamic chunk training. Using the notation introduced above, full self-attention computes a $T \times T$ interaction map and has $O(T^2)$ context-mixing cost; streaming variants reduce the visible right context but typically retain a left-context cache or memory state of length L, incurring an approximate per-layer cache cost of $O(LD)$. Dual-mode ASR, WeNet, and U2/U2++ use full-context modeling, dynamic chunk training, and two-pass rescoring to run one model under offline and streaming chunk configurations (Yu et al., 2021b; Yao et al., 2021; Zhang et al., 2022; Wu et al., 2021). U2++ extends U2 with a bidirectional decoder: CTC prefix beam search first generates an n-best list under a selected chunk size, and left-to-right/right-to-left attention rescoring reranks candidates, making it a natural

unified ASR baseline. Emformer further introduces explicit memory banks for efficient low-latency streaming recognition (Shi et al., 2021). These approaches validate dynamic chunking, but their contextual modeling modules remain attention- or cache-based, leaving long-form streaming inference sensitive to memory overhead.

Mamba-based speech modeling. State space models, and Mamba in particular, provide a different long-sequence modeling primitive (Gu et al., 2022; Gu and Dao, 2023; Dao and Gu, 2024). In simplified form, Mamba replaces pairwise attention with selective recurrent state updates,

$$\begin{aligned} h_t &= \bar{A}_t h_{t-1} + \bar{B}_t x_t, \\ y_t &= \bar{C}_t h_t, \end{aligned} \tag{1}$$

where x_t, h_t, and y_t denote the frame-level input, recurrent state, and output at time t, and $\bar{A}_t$, $\bar{B}_t$, and $\bar{C}_t$ are input-dependent transition, input, and readout parameters. Thus, a causal Mamba output can summarize historical inputs $x_{\leq t}$ with linear sequence complexity (Gu and Dao, 2023; Dao and Gu, 2024). Recent studies have explored Mamba for speech separation, ASR, ConMamba encoders, streaming ASR, and real-time speech interaction systems (Jiang et al., 2024; Hou et al., 2024; Fang and Li, 2025; Lu et al., 2025). These results suggest that Mamba-based architectures can reduce the computational and storage burden of attention for speech sequences. However, a causal Mamba output y_t depends only on $x_{\leq t}$, while ASR often benefits from a bounded right-context window $x_{t:t+r}$, where r denotes the number of future frames allowed by the latency budget. This makes latency-controlled bidirectionality more central to ASR than a direct replacement of attention with unidirectional Mamba.

Latency-controlled bidirectional Mamba. Bidirectional Mamba introduces separate forward and backward branches to recover future context for speech modeling (Zhang et al., 2025). LC-BiMamba adapts this idea to streaming ASR by restricting the backward branch to fixed chunks: frames are reversed within each chunk to expose local future context and then restored to the original order, forming a strong attention-free baseline with a fixed latency budget (Moriya et al., 2025). Its limitation appears when dynamic chunk training is implemented through physical chunk splitting. For an encoder input $X \in \mathbb{R}^{B\times T\times D}$ and chunk size C, stacking all $N_c = \lceil T/C \rceil$ chunks changes the backward input shape from $B \times T \times D$ to $(BN_c) \times C \times D$. Small chunks therefore inflate the effective mini-batch size, increasing memory consumption and training time. TC-BiMamba instead keeps the full-sequence shape and applies chunk-aware reordering to the backward sequence, preserving local future constraints while retaining state propagation across chunk boundaries.

3 Methods

As shown in Fig. 2, TC-BiMamba consists of a BiMamba–convolution encoder and a hybrid decoder. The method is designed to address the two limitations highlighted in Sections 1 and 2: fixed-latency backward modeling and the batch inflation caused by physical chunk splitting. Unless otherwise specified, let $X \in \mathbb{R}^{B\times T\times D}$ denote the subsampled encoder sequence entering a TC-BiMamba block, where B, T, and D are the batch size, sequence length, and hidden dimension.

3.1 TC-BiMamba Encoder

3.1.1 BiMamba

We use Mamba as the core sequence modeling module. For an input frame $x_t \in \mathbb{R}^D$ and recurrent state $h_t \in \mathbb{R}^{d_s}$, one selective SSM branch can be written as

$$\begin{aligned} h_t &= \bar{A}_t h_{t-1} + \bar{B}_t x_t, \\ y_t &= \bar{C}_t h_t + s \odot x_t, \end{aligned} \tag{2}$$

where d_s is the state dimension, $y_t \in \mathbb{R}^D$ is the output frame, and $s \in \mathbb{R}^D$ is the skip parameter. The discretized parameters are input-dependent: $\bar{A}_t = \exp(\Delta_t A)$ and $\bar{B}_t = (\Delta_t A)^{-1}(\exp(\Delta_t A) - I)\Delta_t B_t$, with Δ_t, B_t, and $\bar{C}_t$ generated from x_t. Thus, each branch maps a length-T sequence to another length-T sequence with linear temporal complexity.

To recover future evidence needed by ASR while keeping the attention-free sequence modeling advantage, we use independent forward and backward Mamba branches. Once their outputs are restored to the original temporal order, we denote them as $\overrightarrow{X}$ and $\overleftarrow{X}$, respectively. Instead of concatenating the two branches and adding another projection layer, we use dimension-wise gated addition:

$$\begin{aligned} \beta &= \sigma(\theta_\beta), \quad \theta_\beta \in \mathbb{R}^{1\times 1\times D}, \\ \mathrm{BiMamba}(X) &= \beta \odot \overrightarrow{X} + (1-\beta) \odot \overleftarrow{X}, \end{aligned} \tag{3}$$

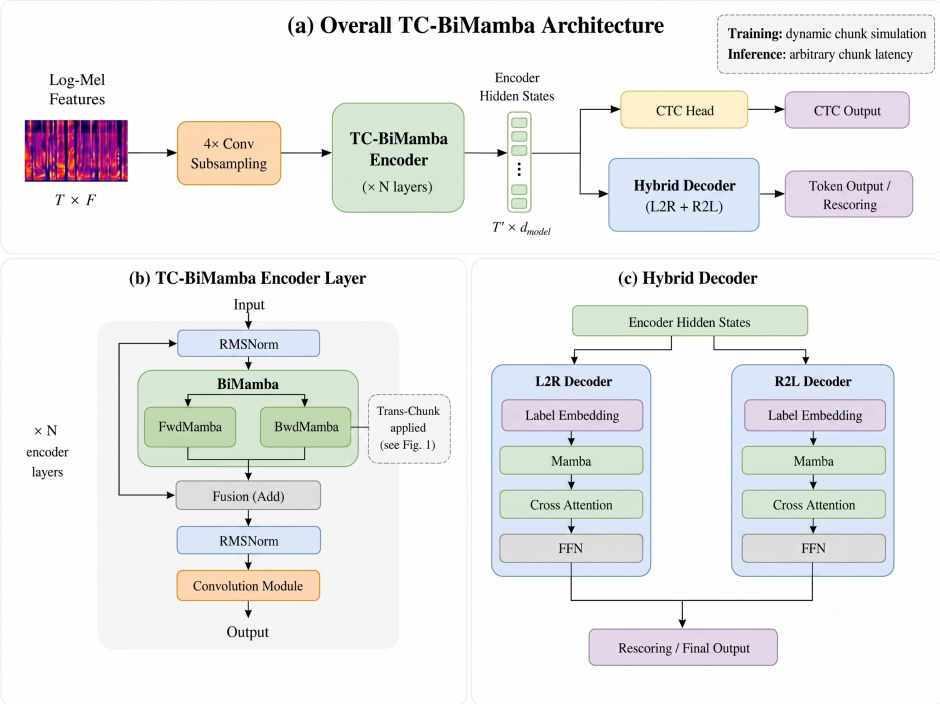

Figure 2: Overall architecture of TC-BiMamba: (a) unified CTC/rescoring pipeline, (b) TC-BiMamba encoder layer with Trans-Chunk backward modeling, and (c) Hybrid L2R/R2L Mamba decoder.

where $\sigma(\cdot)$ is the sigmoid function, $\odot$ denotes element-wise multiplication, and β is broadcast over the batch and time axes. This fusion keeps the parameter cost small while allowing each channel to adaptively balance past and bounded future context. A convolution module is then applied after BiMamba to strengthen local acoustic pattern modeling, complementing Mamba's long-range recurrent state with short-range spectral-temporal cues.

3.1.2 Trans-Chunk Mechanism

Dynamic chunking is the mechanism that allows one model to operate under multiple latency budgets. Attention-based encoders implement it with masks, whereas a direct LC-BiMamba extension would physically split X into BN_c short sequences and feed them to the backward branch. As discussed above, this preserves the latency constraint but inflates the effective mini-batch size and resets the sequence modeling problem at every chunk. Trans-Chunk is therefore introduced as a training formulation for Mamba-based local future modeling, not merely as a tensor permutation: it keeps the backward computation chunk-aware without changing the batch dimension or discarding cross-chunk state propagation.

For a sampled chunk size C with $2 \leq C \leq T$, let $N_c = \lceil T/C \rceil$ and $T_C = N_c C$. We first pad X along the temporal axis to obtain $\widehat{X} = \mathrm{Pad}_C(X) \in \mathbb{R}^{B \times T_C \times D}$. For a temporal index $i = kC + j$, where $0 \leq k < N_c$ and $0 \leq j < C$, define the within-chunk reversal permutation

$$\begin{aligned} \pi_C(kC + j) &= kC + (C - 1 - j), \\ \widetilde{X}_{:,i,:} &= \widehat{X}_{:,\pi_C(i),:}. \end{aligned} \tag{4}$$

Equivalently, $\widetilde{X} = \mathcal{R}_C(\widehat{X})$, where $\mathcal{R}_C$ reverses frames inside each chunk but keeps the chunk order unchanged. The forward and backward branches are then computed as

$$\begin{aligned} \overrightarrow{X} &= \mathrm{FwdMamba}_{\theta_f}(X), \\ \widetilde{Z} &= \mathrm{BwdMamba}_{\theta_b}(\widetilde{X}), \\ \overleftarrow{X} &= \mathrm{Crop}_T\Big(\mathcal{R}_C(\widetilde{Z})\Big), \end{aligned} \tag{5}$$

where θ_f and θ_b are independent branch parameters, and $\mathrm{Crop}_T(\cdot)$ removes padded frames and restores length T. Since $\mathcal{R}_C$ is self-inverse, applying it after BwdMamba returns the backward output to the original temporal order before gated fusion. Because chunks remain in chronological order, the recurrent state is not reset at chunk boundaries; the backward branch therefore combines past information propagated through the Mamba state with bounded future evidence introduced by the reversed order inside the current chunk.

This formulation directly addresses the training-overhead issue raised in Section 2: the backward

branch processes tensors of shape $B \times T_C \times D$ rather than $(BN_c) \times C \times D$. More importantly, it makes LC-style local future modeling compatible with Mamba's sequence-level state evolution during dynamic chunk training. It also preserves the latency-control interpretation: smaller C restricts the backward branch to a smaller right-context window, while $C = T$ reduces to the offline full-utterance backward scan. We use $C \geq 2$ because $C = 1$ provides no future context and degenerates to causal modeling.

Causality and latency. For an original frame $t = kC + j$ with $0 \leq j < C$, Trans-Chunk exposes the backward branch only to future frames in the same chunk, $\mathcal{F}_C(t) = \{kC+j+1, \ldots, \min((k+1)C-1, T-1)\}$. Hence the additional right context is at most $C - 1 - j$ subsampled frames and never exceeds $C - 1$. Cross-chunk recurrent state does not leak future information because chunks are scanned in chronological order: the state entering chunk k depends only on chunks before k, while the within-chunk reversal supplies bounded local future evidence. With the 4× encoder subsampling and 10 ms frame shift used in our setup, one chunk unit corresponds to about 40 ms, so C directly controls streaming latency and $C = T$ recovers offline full context.

3.2 Hybrid Decoder

The hybrid decoder follows the bidirectional decoder spirit of U2++ while replacing masked self-attention with Mamba. Let $H_{\text{enc}} \in \mathbb{R}^{B \times T_e \times D}$ be the encoder output and let $E_d \in \mathbb{R}^{B \times U \times D}$ be the token embedding sequence for direction $d \in \{\text{L2R}, \text{R2L}\}$, where U is the label length. A decoder block can be summarized as

$$\begin{aligned} M_d &= \text{Mamba}_d(E_d), \\ H_d^{\text{dec}} &= \text{CrossAttn}_d(M_d, H_{\text{enc}}), \end{aligned} \tag{6}$$

with normalization, residual connections, and feed-forward layers following the standard decoder design. Mamba models dependencies within the label sequence of a fixed direction, while cross-attention is retained to align decoder states with encoder frames of different length. This separation lets the decoder benefit from linear-complexity sequence modeling without removing the encoder–decoder interaction needed for ASR.

3.3 Training & Inference

During training, TC-BiMamba samples one chunk size per mini-batch. Let $l_{\max}$ be the maximum subsampled encoder length in the batch and $x \sim \mathcal{U}(0, 1)$:

$$\begin{aligned} C &= \begin{cases} l_{\max}, & x > 0.5, \\ \ell, & x \leq 0.5, \end{cases} \\ \ell &\sim \mathcal{U}_{\mathbb{Z}}(2, \min(25, l_{\max} - 1)) \,. \end{aligned} \tag{7}$$

where $\mathcal{U}_{\mathbb{Z}}$ is a discrete uniform distribution. The first case gives full-context offline training, while the second samples streaming settings with at least two frames of local future context.

We train TC-BiMamba with a joint CTC and sequence-to-sequence criterion. For a target label sequence, the loss is

$$\begin{aligned} L_{\text{E2E}} &= (1-\alpha)L_{\text{L2R}} + \alpha L_{\text{R2L}}, \\ L &= \lambda L_{\text{CTC}} + (1-\lambda)L_{\text{E2E}}, \end{aligned} \tag{8}$$

where L_{CTC} is the CTC loss, L_{L2R} and L_{R2L} are the directional decoder losses, $\lambda \in [0, 1]$ controls the CTC–decoder trade-off, and $\alpha \in [0, 1]$ controls the R2L decoder contribution.

During inference, CTC prefix beam search first produces an n-best list $\mathcal{Y}_n$. For each candidate $Y \in \mathcal{Y}_n$ conditioned on X, we compute

$$\begin{aligned} S_{\text{dec}} &= (1-\alpha)S_{\text{L2R}} + \alpha S_{\text{R2L}}, \\ S &= \lambda S_{\text{CTC}} + (1-\lambda)S_{\text{dec}}. \end{aligned} \tag{9}$$

Here S_{CTC}, S_{L2R}, and S_{R2L} are log-probability scores from the CTC, L2R decoder, and R2L decoder for the same candidate. We output the candidate in $\mathcal{Y}_n$ with the highest S. The same symbols λ and α denote the corresponding interpolation weights used at decoding time, with their values specified in the experimental setup.

4 Experimental Setup

This section describes the datasets, implementation, model configurations, training recipe, and decoding protocol used throughout the evaluation.

Experiments are conducted on AISHELL-1 (Bu et al., 2017), AISHELL-2 (Du et al., 2018), and LibriSpeech (Panayotov et al., 2015), which contain approximately 178, 1,000, and 960 hours of training data, respectively. AISHELL-1 and AISHELL-2 use 4,232 and 5,257 Chinese characters as vocabularies. For LibriSpeech, we use a unigram byte pair encoding (BPE) (Sennrich et al., 2016) vocabulary with 5,002 tokens.

We implement TC-BiMamba with the WeNet toolkit (Zhang et al., 2022) and use U2++ as

Table 1: CERs (%) on AISHELL-1 and AISHELL-2. Res denotes attention rescoring. This table evaluates architectural choices and cross-dataset generalization.

Test set	Model	Encoder	Decoder	Fusion	Offline		Chunk16		Params (M)
					Res	CTC	Res	CTC	
AISHELL-1	U2++	Conformer	BiTransformer	-	5.31	6.12	5.75	6.78	46.11
	TC-BiMamba(S)	BiMamba	BiTransformer	Addition	5.47	6.04	6.00	6.73	46.07
		BiMamba	Hybrid	Addition	**5.25**	**5.92**	**5.70**	**6.60**	47.03
		- CNNs	Hybrid	Addition	5.30	5.96	5.82	6.81	46.94
		BiMamba	Hybrid	Concatenation	5.40	6.40	5.95	7.17	47.26
	ConMamba	ConMamba	Hybrid	Addition	5.38	6.33	5.79	6.94	47.13
	UniMamba	Mamba	Hybrid	-	6.28	7.43	6.28	7.43	45.36
AISHELL-2	U2++	Conformer	BiTransformer	-	6.41	7.08	6.74	7.72	47.36
	TC-BiMamba(S)	BiMamba	Hybrid	Concatenation	6.15	**6.86**	6.70	7.59	48.51
		BiMamba	Hybrid	Addition	**6.00**	6.88	**6.43**	**7.53**	48.28

the main unified ASR baseline. We extract 80-dimensional filter-bank features using a 25 ms window and a 10 ms frame shift. The input features are subsampled by a factor of 4 using two convolution layers with kernel size 3 and stride 2. SpecAugment (Park et al., 2019), speed perturbation, and SpecSub (Wu et al., 2021) are applied during training.

The Conformer encoder baseline contains 12 blocks with 4-head self-attention, a 2048-dimensional feed-forward network, a convolution kernel size of 8, and model dimension 256. To keep parameter counts comparable, the Mamba expansion factor, convolution kernel size, and state size are set to 2, 4, and 64 for ConMamba, and 4, 4, and 64 for TC-BiMamba(S). The number of blocks is fixed to 9 for ConMamba and 17 for TC-BiMamba(S). For LibriSpeech, TC-BiMamba(L) increases the model dimension to 320 and the convolution kernel size to 15. To validate the role of future-context modeling, UniMamba removes the BwdMamba branch from TC-BiMamba(S) and increases the block number to 32. The BiTransformer decoder follows U2++; the Hybrid decoder uses Mamba expansion factor 2, convolution kernel size 4, and state size 64.

For the controlled Trans-Chunk comparison, TC-BiMamba(L)-CS denotes the Chunk-Splitting variant: it keeps the TC-BiMamba(L) configuration and parameter count but replaces sequence-level Trans-Chunk training with explicit physical chunk splitting in the backward branch. This baseline isolates whether Trans-Chunk improves scalability and accuracy beyond a direct dynamic-chunk extension of LC-style processing.

All models are trained with Adam (Kingma and Ba, 2015), 25,000 warmup steps, peak learning rate 10^{-3}, and dynamic batching up to 15,000 frames on NVIDIA RTX 3090 GPUs. We train AISHELL-1 and AISHELL-2 models for 100 epochs and average the 10 checkpoints with the lowest validation loss. LibriSpeech models are trained for 120 epochs and averaged over the last 30 checkpoints. During training, α and λ are set to 0.3; during decoding, both are set to 0.5. We report CER on AISHELL and WER on LibriSpeech using CTC prefix beam search and attention rescoring with beam size 10.

5 Evaluation

We evaluate TC-BiMamba along three axes: recognition accuracy, latency flexibility, and scalability. The following sections analyze architectural ablations, dynamic chunk sizes, comparison with LC-BiMamba, long-form inference, and Trans-Chunk training efficiency.

5.1 Accuracy and Architectural Effectiveness

We first evaluate whether the proposed architecture improves unified ASR and whether each component is justified. As shown in Table 1, TC-BiMamba(S) with the Hybrid decoder and additive fusion achieves the best overall AISHELL-1 performance under both offline and Chunk16 decoding, while using a parameter budget comparable to U2++. Compared with U2++, it reduces AISHELL-1 rescoring CER from 5.31 to 5.25 in offline mode and from 5.75 to 5.70 in Chunk16 mode; the corresponding CTC results also improve from 6.12 to 5.92 and from 6.78 to 6.60. These gains are modest but consistent across decoding modes, showing that the proposed encoder does not sacrifice either offline accuracy or streaming accuracy.

Table 2: LibriSpeech rescoring WERs (%) under different chunk sizes. Chunk 2–Chunk 32 denote streaming chunk sizes; CS denotes chunk splitting, and unsupported settings are marked by –.

Test set	Model	Streaming chunk size					Offline	Params (M)
		Chunk 2	Chunk 4	Chunk 8	Chunk 16	Chunk 32		
test-clean	U2++	4.57	4.29	4.04	3.80	3.66	3.32	47.05
	TC-BiMamba(S)	4.64	4.19	3.89	3.66	3.44	3.12	47.97
	TC-BiMamba(L)-CS	4.43	4.02	3.77	3.60	3.36	3.07	70.76
	TC-BiMamba(L)	**4.33**	**3.91**	**3.69**	**3.51**	**3.24**	**2.97**	70.76
	LC-BiMamba	–	–	–	3.70	–	3.00	118.70
test-other	U2++	12.36	11.84	11.06	10.38	9.97	8.67	47.05
	TC-BiMamba(S)	12.27	11.39	10.57	9.78	9.22	8.19	47.97
	TC-BiMamba(L)-CS	11.82	11.09	10.22	9.44	8.94	7.92	70.76
	TC-BiMamba(L)	**11.52**	**10.52**	**9.83**	9.44	**8.75**	7.72	70.76
	LC-BiMamba	–	–	–	**8.90**	–	**7.10**	118.70

The ablations further support the design choices. Removing the backward branch in UniMamba causes a clear degradation, confirming that ASR benefits from local future evidence even in streaming-oriented models. TC-BiMamba(S) also outperforms ConMamba, suggesting that explicit bidirectional Mamba modeling is more effective than inserting Mamba into a Conformer-style encoder.

The ConMamba result further shows that TC-BiMamba is not merely a drop-in replacement of self-attention with Mamba in a Conformer-style block. Our ConMamba baseline follows a convolution-augmented Mamba encoder (Hou et al., 2024) under the same dynamic-chunk recipe and comparable parameter budget, but remains weaker than TC-BiMamba(S). We hypothesize that Conformer's macaron FFNs become partly redundant with Mamba's input-dependent projections, gating, and selective state updates (Gu and Dao, 2023). TC-BiMamba instead uses a BiMamba–CNN encoder, combining latency-controlled bidirectional Mamba modeling with local convolutional acoustic modeling.

Removing CNNs, replacing the Hybrid decoder with a BiTransformer decoder, or switching to concatenation fusion does not improve AISHELL-1 performance, so the final configuration keeps additive fusion, local convolution, and the Hybrid decoder. On AISHELL-2, TC-BiMamba(S) reduces rescoring CER from 6.41 to 6.00 offline and from 6.74 to 6.43 under Chunk16, indicating that the architecture generalizes beyond the smaller AISHELL-1 benchmark. The concatenation variant gives the best offline CTC result on AISHELL-2, but additive fusion remains stronger in rescoring and streaming settings, which is the main target of unified ASR.

5.2 Flexible-Latency Streaming Recognition

We next test whether one TC-BiMamba model can operate under multiple latency budgets. Table 2 reports LibriSpeech rescoring WERs from Chunk2 to Chunk32 (40 ms per chunk unit; Chunk16 $\approx$640 ms) and offline decoding. Larger chunks improve U2++ and TC-BiMamba by exposing more right context. TC-BiMamba(S) improves over U2++ in most settings, and TC-BiMamba(L) further improves all clean-set results and gives the best TC-BiMamba results on test-other, while LC-BiMamba remains stronger at its reported fixed-latency point. Since LC-BiMamba code and checkpoints are unavailable, those numbers are an external reference rather than a fully controlled reproduction. Overall, TC-BiMamba gains from additional right context and capacity without being tied to a single operating point.

5.3 Comparison with Fixed-Latency BiMamba

Table 2 also compares TC-BiMamba with LC-BiMamba, the strongest fixed-latency bidirectional Mamba baseline. Unsupported dynamic-chunk entries are marked by –. Because the official implementation is unavailable, this is an external comparison based on published results, not a fully controlled same-codebase comparison. Despite using about 40% fewer parameters, TC-BiMamba(L) is competitive and is better on test-clean in both offline and Chunk16 modes, while LC-BiMamba remains stronger on test-other. We therefore position TC-BiMamba as a flexible and parameter-efficient alternative, not a uniformly superior replacement.

The controlled comparison between TC-BiMamba(L) and TC-BiMamba(L)-CS isolates Trans-Chunk: with the same parameter count, TC-BiMamba(L) is no worse in all settings and improves most of them. This supports the

Table 3: Peak GPU memory (GB) on long-form inputs with increasing durations.

Input length	U2++ baseline		TC-BiMamba(S)	
	CTC path	Rescore	CTC path	Rescore
10s	**0.41**	**0.45**	0.65	0.89
20s	**0.44**	**0.51**	0.65	0.89
40s	**0.60**	**0.71**	0.65	0.89
80s	1.47	1.91	**0.65**	**1.16**
160s	4.66	6.49	**0.66**	**2.19**
320s	8.08	8.35	**0.81**	**7.39**
10240s	OOM	OOM	**10.51**	OOM

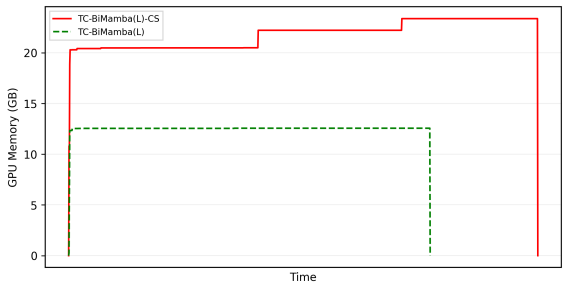

Figure 3: Training overhead comparison between TC-BiMamba(L) and TC-BiMamba(L)-CS on LibriSpeech.

hypothesis in Section 2 that avoiding physical chunk splitting reduces overhead while preserving sequence-level state evolution.

5.4 Long-Form Inference Scalability

To evaluate scalability beyond utterance-level benchmarks, Table 3 reports memory results on long-form inputs. U2++ is more efficient up to 40s, but grows rapidly at longer durations and runs out of memory on the longest input. TC-BiMamba(S) keeps CTC memory almost unchanged from 10s to 160s and remains feasible for 10240s, so its advantage becomes clearer as input length increases. Rescoring memory still grows for very long inputs, indicating that the main scalability gain comes from the Mamba encoder and CTC path.

On approximately 60-second concatenated LibriSpeech inputs, the WER results in Table 4 show that TC-BiMamba(S) improves offline rescoring WER from 7.20 to 5.67 on Clean-60s and from 23.35 to 15.33 on Other-60s. Under Chunk16, it remains stronger on Other-60s, while U2++ is better on Clean-60s, suggesting that clean-set streaming performance can still be sensitive to concatenation boundaries and limited right context.

These results suggest a practical deployment pattern rather than a claim that every component scales equally. For long recordings, the encoder and CTC branch can serve as the primary streaming path, while attention rescoring is better applied to endpointed segments or selected hypotheses. The weaker Chunk16 result on Clean-60s also indicates that concatenation boundaries remain challenging under fixed right-context windows.

Table 4: WERs (%) on 60-second concatenated LibriSpeech inputs.

Test set	Model	Offline		Chunk16	
		Res	CTC	Res	CTC
Clean-60s	U2++	7.20	7.35	**7.63**	**7.76**
	TC-BiMamba(S)	**5.67**	**5.87**	8.52	8.83
Other-60s	U2++	23.35	23.54	24.45	24.58
	TC-BiMamba(S)	**15.33**	**15.48**	**21.53**	**21.78**

5.5 Training Efficiency of Trans-Chunk

Finally, we use TC-BiMamba(L)-CS as a controlled counterfactual for Trans-Chunk: it keeps the architecture and parameter count fixed but explicitly splits backward sequences into chunks, whereas TC-BiMamba(L) keeps the proposed sequence-level formulation. As shown in Fig. 3, this gives a 1.3× training speedup and reduces GPU memory by approximately 50%, because the backward branch no longer expands the mini-batch by the number of chunks, leaving more GPU budget for longer utterances or richer chunk-size sampling. Table 2 confirms that this scalability gain does not sacrifice accuracy.

6 Conclusion

We presented TC-BiMamba, a dynamic-chunk bidirectional Mamba framework for unified streaming and offline ASR. Its Trans-Chunk formulation recasts LC-style local future modeling as sequence-level Mamba training, supporting multiple latency settings without mini-batch inflation. Experiments show that TC-BiMamba improves over U2++ at comparable scale, remains competitive with externally reported LC-BiMamba results using fewer parameters, keeps long-form CTC memory stable, and achieves 1.3× training speedup with about 50% lower GPU memory than explicit chunk splitting.

In deployment, chunk size therefore becomes an inference-time latency knob: the same model can favor responsiveness with small chunks, accuracy with larger chunks or offline decoding, and scalable encoder/CTC processing for long recordings.

Limitations

This work focuses on controlled evidence for the proposed Trans-Chunk formulation. Although AISHELL-1, AISHELL-2, LibriSpeech, and constructed long-form LibriSpeech inputs cover standard Mandarin and English ASR settings, broader multilingual and production-scale evaluations remain future work. Moreover, the main long-form efficiency gain comes from the Mamba encoder and CTC path; attention-based rescoring is still better applied to endpointed segments or selected hypotheses rather than entire unsegmented recordings.

Ethics Statement

This work focuses on ASR model architecture and efficiency, and does not introduce new data collection procedures or user-facing deployment mechanisms. Its potential risks are those generally associated with ASR systems, such as inaccurate transcripts in sensitive settings or deployment on speech data without appropriate consent.

References

Hui Bu, Jiayu Du, Xingyu Na, Bengu Wu, and Hao Zheng. 2017. AISHELL-1: An open-source mandarin speech corpus and a speech recognition baseline. In *Proceedings of the Conference of the Oriental Chapter of the International Coordinating Committee on Speech Databases and Speech I/O Systems and Assessment (O-COCOSDA 2017)*, pages 1–5.

Chung-Cheng Chiu and Colin Raffel. 2018. Monotonic chunkwise attention. In *Proceedings of the International Conference on Learning Representations (ICLR 2018)*.

Tri Dao and Albert Gu. 2024. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In *Proceedings of the International Conference on Machine Learning (ICML 2024)*, volume 235 of *Proceedings of Machine Learning Research*, pages 10041–10071.

Jiayu Du, Xingyu Na, Xuechen Liu, and Hui Bu. 2018. AISHELL-2: Transforming mandarin ASR research into industrial scale. *arXiv preprint arXiv:1808.10583*.

Ying Fang and Xiaofei Li. 2025. Mamba for streaming ASR combined with unimodal aggregation. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2025)*, pages 1–5.

Alex Graves. 2012. Sequence transduction with recurrent neural networks. *arXiv preprint arXiv:1211.3711*.

Albert Gu and Tri Dao. 2023. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*.

Albert Gu, Karan Goel, and Christopher Ré. 2022. Efficiently modeling long sequences with structured state spaces. In *Proceedings of the International Conference on Learning Representations (ICLR 2022)*.

Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. 2020. Conformer: Convolution-augmented transformer for speech recognition. In *Proceedings of the Annual Conference of the International Speech Communication Association (Interspeech 2020)*, pages 5036–5040.

Haoxiang Hou, Xun Gong, and Yanmin Qian. 2024. ConMamba: A convolution-augmented Mamba encoder model for efficient end-to-end ASR systems. In *Proceedings of the IEEE International Symposium on Chinese Spoken Language Processing (ISCSLP 2024)*, pages 711–715.

Xilin Jiang, Yinghao Aaron Li, Adrian Nicolas Florea, Cong Han, and Nima Mesgarani. 2024. Speech slytherin: Examining the performance and efficiency of Mamba for speech separation, recognition, and synthesis. *arXiv preprint arXiv:2407.09732*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations (ICLR 2015)*.

Xiangyu Lu, Wang Xu, Haoyu Wang, Hongyun Zhou, Haiyan Zhao, Conghui Zhu, Tiejun Zhao, and Muyun Yang. 2025. DuplexMamba: Enhancing real-time speech conversations with duplex and streaming capabilities. In *Proceedings of the CCF International Conference on Natural Language Processing and Chinese Computing (NLPCC 2025)*, pages 62–74.

Takafumi Moriya, Masato Mimura, Kiyoaki Matsui, Hiroshi Sato, and Kohei Matsuura. 2025. Attention-free dual-mode ASR with latency-controlled selective state spaces. In *Proceedings of the Annual Conference of the International Speech Communication Association (Interspeech 2025)*, pages 3588–3592.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. LibriSpeech: An ASR corpus based on public domain audio books. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2015)*, pages 5206–5210.

Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D. Cubuk, and Quoc V. Le. 2019. SpecAugment: A simple data augmentation method for automatic speech recognition. In *Proceedings of the Annual Conference of the International Speech Communication Association (Interspeech 2019)*, pages 2613–2617.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016), Volume 1: Long Papers*, pages 1715–1725.

Yangyang Shi, Yongqiang Wang, Chunyang Wu, Ching-Feng Yeh, Julian Chan, Frank Zhang, Duc Le, and Mike Seltzer. 2021. Emformer: Efficient memory transformer based acoustic model for low-latency streaming speech recognition. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2021)*, pages 6783–6787.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, pages 5998–6008.

Di Wu, Binbin Zhang, Chao Yang, Zhendong Peng, Wenjing Xia, Xiaoyu Chen, and Xin Lei. 2021. U2++: Unified two-pass bidirectional end-to-end model for speech recognition. *arXiv preprint arXiv:2106.05642*.

Zhuoyuan Yao, Di Wu, Xiong Wang, Binbin Zhang, Fan Yu, Chao Yang, Zhendong Peng, Xiaoyu Chen, Lei Xie, and Xin Lei. 2021. WeNet: Production oriented streaming and non-streaming end-to-end speech recognition toolkit. In *Proceedings of the Annual Conference of the International Speech Communication Association (Interspeech 2021)*, pages 4054–4058.

Jiahui Yu, Chung-Cheng Chiu, Bo Li, Shuo-Yiin Chang, Tara N. Sainath, Yanzhang He, Arun Narayanan, Wei Han, Anmol Gulati, Yonghui Wu, and Ruoming Pang. 2021a. FastEmit: Low-latency streaming ASR with sequence-level emission regularization. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2021)*, pages 6004–6008.

Jiahui Yu, Wei Han, Anmol Gulati, Chung-Cheng Chiu, Bo Li, Tara N. Sainath, Yonghui Wu, and Ruoming Pang. 2021b. Universal ASR: Unify and improve streaming ASR with full-context modeling. In *Proceedings of the International Conference on Learning Representations (ICLR 2021)*.

Binbin Zhang, Di Wu, Zhendong Peng, Xingchen Song, Zhuoyuan Yao, Hang Lv, Lei Xie, Chao Yang, Fuping Pan, and Jianwei Niu. 2022. WeNet 2.0: More productive end-to-end speech recognition toolkit. In *Proceedings of the Annual Conference of the International Speech Communication Association (Interspeech 2022)*, pages 1661–1665.

Xiangyu Zhang, Qiquan Zhang, Hexin Liu, Tianyi Xiao, Xinyuan Qian, Beena Ahmed, Eliathamby Ambikairajah, Haizhou Li, and Julien Epps. 2025. Mamba in speech: Towards an alternative to self-attention. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 33:1933–1948.

A AI Usage

Generative AI tools were used to assist with language polishing and the refinement of figure presentation. The authors reviewed, edited, and approved all AI-assisted outputs. All scientific ideas, technical content, experiments, results, analyses, citations, and final decisions remain the responsibility of the authors. No generative AI tool was used as an author or as a source of scientific claims.