TRAINING HYBRID BLOCK DIFFUSION LANGUAGE MODELS WITH PARTIAL BIDIRECTIONALITY

Anonymous authors
Paper under double-blind review

ABSTRACT

High-throughput long-context generation is one of the central challenges for large language models. Generation is typically memory-bandwidth-bound rather than compute-bound: each decoding step must stream the accumulated key/value (KV) cache from memory, so bandwidth demand grows with context length while only one token is emitted. Two parallel approaches have therefore emerged: reducing memory access with efficient attention variants and linear-time mixers such as Mamba, or increasing parallel computation by generating blocks of tokens at once. However, technical challenges arise when combining these two ideas. Earlier hybrid diffusion models such as DiffuMamba use bidirectional Mamba mixing, including a reverse-direction scan relative to causal generation. This reverse scan needs to scan the entire sequence, so its states are not prefix-only and cannot be precisely reused as a cache even when diffusion is performed block by block. We propose a BDLM Mamba–attention hybrid that addresses this challenge by restricting the reverse Mamba scan to the active denoising block, which enables exact caching across blocks. In an 87M-parameter DCLM sweep, BDLM Mamba-H achieves the best C4-en validation perplexity compared to BDLM attention and full-sequence baselines. At 350M parameters, it remains competitive with BDLM attention. In long-context inference benchmarks with 700M random-initialized models, BDLM Mamba-H reaches 19.7x the throughput of full-sequence DiffuMamba-H at 65K tokens and 3.7x the throughput of BDLM attention at 262K, showing that Mamba hybrids are a potential long-context diffusion architecture.

1 INTRODUCTION

Serving language models at long context remains largely a memory-bandwidth problem. In autoregressive decoding, every generated token reads the accumulated key/value (KV) cache, so bytes moved grow with context length while only one token is emitted. Two lines of work address this pressure from different directions. Hybrid autoregressive models reduce cache traffic by replacing many attention layers with linear-time recurrent or state-space mixers such as Mamba (Gu & Dao, 2024; Dao & Gu, 2024; Qin et al., 2024; Kimi Team et al., 2025; Li et al., 2025; Ling Team et al., 2025). Masked diffusion language models instead generate many tokens in parallel by iterative denoising (Austin et al., 2021; Hoogeboom et al., 2021; Sahoo et al., 2024; Shi et al., 2024; Nie et al., 2025). Block diffusion language models (BDLMs) make diffusion cacheable by generating blocks left to right: completed prefix blocks become fixed conditioning, and the current block is denoised in parallel (Arriola et al., 2025).

DiffuMamba combines diffusion with Mamba and shows strong long-context inference throughput for full-sequence denoisers (Singh et al., 2026). Its fully bidirectional Mamba scan, however, includes a reverse-direction pass over the whole denoising window. Those reverse states depend on the active tokens and are therefore not reusable prefix states. We study a native BDLM Mamba–attention hybrid, BDLM Mamba-H, that restricts the reverse-direction Mamba scan to the active denoising block. Completed prefix blocks are stored as attention KV states for attention layers and as forward Mamba boundary states for Mamba layers. The all-block objective constructs these same prefix caches during training and lets downstream block losses train them directly, while the active block keeps local bidirectional mixing.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

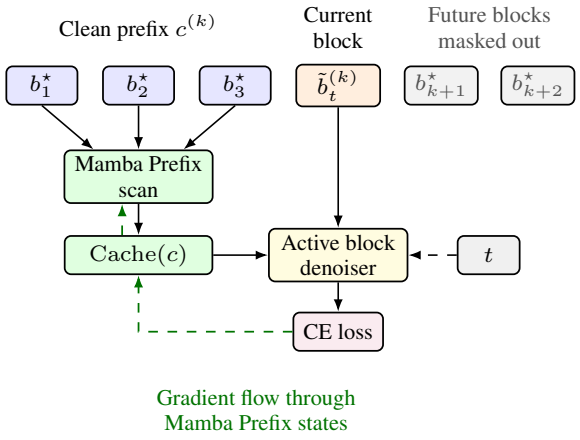


Figure 1: BDLM Mamba-H training. The all-block objective applies this active-block computation at all block positions in parallel. Solid arrows show forward computation, dashed green arrows show gradient flow, and dashed black arrows mark optional timestep input. The denoiser receives a clean prefix cache, the noisy current block, and optional timestep conditioning while future blocks remain masked out. For Mamba hybrid layers, the cached prefix is forward-only and the reverse scan is local to the active block, so active-block losses train the same prefix cache object used by the sampler.

In this work we explore whether cacheable BDLM Mamba hybrids can retain competitive validation quality while improving long-context inference throughput. We also describe timestep-conditioning factorization that applies to only the current active block for BDLM Mamba-H, so timestep modulation can be applied without making clean-prefix caches depend on the reverse diffusion step (Appendix D). Our contributions are as follows.

- **Cache-aligned training.** We formulate a BDLM all-block objective whose cached objects match inference-time block caches: attention KV states for attention layers and Mamba prefix states for recurrent layers. For the BDLM Mamba-H architecture, downstream block losses directly train the prefix scan that constructs the recurrent state reused at inference.
- **Controlled validation and scale-up.** We train full-sequence attention, full-sequence DiffuMamba-H, BDLM attention, and BDLM Mamba-H at 87M parameters on DCLM with tuned hyperparameters. BDLM Mamba-H reaches the best 87M C4-en validation perplexity, 61.6 compared with 76.5 for BDLM attention and 83.7 and 87.6 for the full-sequence baselines. We scale BDLM attention and BDLM Mamba-H to 350M parameters with a Complete(d)P-style learning-rate transfer rule (Mlodozeniec et al., 2025); BDLM Mamba-H remains comparable to BDLM attention in perplexity at this scale.
- **Long-context throughput gains.** We evaluate 350M training throughput at 8192-token context on 8x A100-80GB and 700M random-initialized inference throughput on 1x A100-80GB with batch size 1. BDLM Mamba-H overtakes BDLM attention (and full-sequence DiffuMamba-H in throughput) as generation length grows, reaching 19.7x full-sequence DiffuMamba-H throughput at 65K and 3.7x BDLM attention throughput at 262K.

2 Related Work

Diffusion language models. Discrete diffusion language models corrupt text with masking or discrete noise and train denoisers to reconstruct clean tokens (Austin et al., 2021; Hoogeboom et al., 2021; Sahoo et al., 2024; Nie et al., 2025). BDLMs add an autoregressive block factorization so each completed block becomes clean prefix context for later blocks (Arriola et al., 2025). This gives block diffusion a native cache boundary, unlike full-sequence MDLMs whose hidden states are tied to the current noisy sequence. Training-free cache methods retrofit reuse onto pretrained diffusion language models through approximate KV reuse or adaptive caching (Wu et al., 2026; Ma et al., 2025; Nguyen-Tri et al., 2025; Liu et al., 2025; Wang et al., 2025). Our focus is instead to train a block-diffusion architecture whose cache is part of both the objective and the sampler.

Mamba hybrids for long-context generation. Mamba replaces attention over token pairs with a selective state-space scan whose transition, input, and output maps are functions of the current token (Gu & Dao, 2024). In the notation used below, a Mamba layer carries a recurrent boundary state s_i^ℓ after position or block i in layer ℓ; this state includes the convolutional state and the selective-SSM state needed to continue the forward scan without reprocessing the prefix. Mamba-2 recasts this family through structured state-space duality, yielding more efficient scan kernels and a closer relationship to attention (Dao & Gu, 2024). Recent autoregressive systems increasingly use hybrid stacks that preserve some full-attention layers while replacing many layers with recurrent, linear-attention, or state-space modules: examples include Jamba's Transformer–Mamba mixture-of-experts design, MiniMax-01's Lightning Attention and softmax-attention mixture, Qwen3-Next's Gated DeltaNet/full-attention hybrid, and Kimi Linear's Kimi Delta Attention/MLA hybrid (Lieber et al., 2024; Li et al., 2025; Qwen Team, 2025; Kimi Team et al., 2025). These autoregressive models motivate hybridization as a practical long-context scaling direction; our work studies the analogous question for block diffusion, where cache reuse must be aligned with the denoising objective rather than with a left-to-right next-token likelihood. DiffuMamba-H interleaves attention and bidirectional Mamba layers for masked diffusion denoising and demonstrates that Mamba-backed denoisers improve long-context inference throughput (Singh et al., 2026).

3 Background

Mamba is a selective state-space sequence mixer that replaces attention over tokens with a recurrent scan whose state can be continued across sequence segments (Gu & Dao, 2024). Mamba-2 improves this family with a structured state-space duality and more efficient kernels (Dao & Gu, 2024). In diffusion language models, DiffuMamba-H uses a sparse hybrid schedule that interleaves attention and bidirectional Mamba layers, showing that full-sequence diffusion denoisers can benefit from linear-time mixers (Singh et al., 2026). The present work uses the same broad hybrid motivation, but modifies the scan directionality so BDLM generation can reuse a prefix cache natively at training and inference time. At layer ℓ, we write a simplified selective state-space update as

$$s_i^\ell = A_i^\ell s_{i-1}^\ell + B_i^\ell x_i^\ell, \qquad y_i^\ell = C_i^\ell s_i^\ell + D^\ell x_i^\ell, \tag{1}$$

where $A_i^\ell, B_i^\ell, C_i^\ell$ are token-dependent projections and s_i^ℓ is the recurrent boundary state after token or block i. A prefix cache is the collection of these forward Mamba states, together with attention KV tensors for attention layers, at a block boundary.

Masked discrete diffusion corrupts clean text by replacing tokens with masks and trains a denoiser to recover the original tokens (Austin et al., 2021; Sahoo et al., 2024). If $\tilde{x}_t$ is the corrupted sequence at timestep t, a full-sequence denoiser optimizes

$$\mathcal{L}_{\text{full}}(\theta) = \mathbb{E}_{x,t,M_t}\left[\frac{1}{|M_t|}\sum_{i\in M_t} \text{CE}(f_\theta(\tilde{x}_t, t)_i, x_i)\right]. \tag{2}$$

This objective gives each masked token bidirectional context over the denoising window. It also makes the entire window part of the timestep-conditioned computation. During generation, the hidden states are tied to the current noisy sequence, so a stable prefix has no reusable cache boundary and must be reprocessed at every reverse step.

BDLMs introduce that boundary by partitioning a sequence into K blocks $b^{\star(1)}, \ldots, b^{\star(K)}$. Following the BDLM formulation (Arriola et al., 2025), the likelihood is autoregressive over blocks,

$$\log p_\theta(x) = \sum_{k=1}^{K} \log p_\theta\left(b^{\star(k)} \,\middle|\, b^{\star(<k)}\right), \tag{3}$$

and each block conditional is represented by a discrete diffusion process over only the current block (Arriola et al., 2025). Applying the diffusion NELBO to each term gives the block-summed objective

$$-\log p_\theta(x) \le \mathcal{L}_{\text{BD}}(x;\theta) := \sum_{k=1}^{K} \mathcal{L}\left(b^{\star(k)}, b^{\star(<k)}; \theta\right). \tag{4}$$

For the masked continuous-time parameterization used in BDLM-style training, this can be written as a sum of weighted current-block denoising losses,

$$\mathcal{L}_{\mathrm{BD}}(x;\theta) = \sum_{k=1}^{K} \mathbb{E}_{t,q}\left[w(t)\ \mathrm{CE}\Big(f_\theta(b_t^{(k)}, b^{\star(<k)}, t), b^{\star(k)}\Big)\right], \qquad w(t) = \frac{-\alpha_t'}{1-\alpha_t}, \tag{5}$$

where $b_t^{(k)} \sim q_t(\cdot \mid b^{\star(k)})$ is the noised current block, $b^{\star(<k)}$ is a clean prefix, and future blocks are excluded. The all-block training objective evaluates this sum over block boundaries within the same training window, so every block is trained as a current denoising target conditioned on its clean prefix.

This factorization enables caching for block diffusion. For an attention BDLM, the denoiser for block k can be written with the same cache interface used by autoregressive Transformers:

$$z^{(k)}, K_k, V_k \leftarrow f_\theta^{(k)}\Big(b_t^{(k)}, K_{1:k-1}, V_{1:k-1}\Big) \equiv f_\theta^{(k)}\Big(b_t^{(k)}, b^{\star(<k)}\Big), \tag{6}$$

where $z^{(k)}$ are clean-token logits for the current block and $K_{1:k-1}, V_{1:k-1}$ are key/value tensors computed from completed prefix blocks. During sampling, the model denoises all tokens within the current block in parallel while reading this prefix cache; after the block is finalized, a clean forward pass over the new block appends K_k, V_k for subsequent blocks. Thus cache reuse follows from the block conditional structure itself: completed blocks are fixed inputs to later block denoisers, while reverse diffusion updates are confined to the active block.

NELBO perplexity. For diffusion language models, exact $\log p_\theta(x)$ is generally intractable because likelihood marginalizes over reverse denoising trajectories. We therefore evaluate a negative evidence lower bound (NELBO). For a validation corpus with token counts n_j and sequence-level NELBO estimates $\widehat{\mathcal{L}}_{\mathrm{NELBO}}(x_j)$, the reported token-weighted perplexity is

$$\mathrm{PPL}_{\mathrm{NELBO}} = \exp\left(\frac{\sum_j \widehat{\mathcal{L}}_{\mathrm{NELBO}}(x_j)}{\sum_j n_j}\right). \tag{7}$$

This is an upper-bound-derived perplexity metric rather than an exact autoregressive likelihood. It lets full-sequence MDLMs and BDLMs be compared under the same sequence-token-weighted diffusion validation protocol.

Complete(d)P learning-rate transfer. For scale-up, we transfer learning rates from 87M to 350M using a Complete(d)P-style rule (Mlodozeniec et al., 2025). The motivation is that a larger run changes width, depth, batch size, and duration simultaneously. We use the hidden-weight transfer form

$$\eta_{\mathrm{large}} = \eta_{\mathrm{small}}\, r_{\mathrm{width}}^{a_w} r_{\mathrm{depth}}^{a_d} r_{\mathrm{batch}}^{a_b} r_{\mathrm{duration}}^{a_T}, \tag{8}$$

where each r is the large-to-small ratio for that axis and the exponents are the Complete(d)P transfer exponents for hidden weights. This gives scale factors of 0.31 for BDLM attention and 0.33 for BDLM Mamba-H in our 87M to 350M transfer; the resulting learning rates are reported in Table 2.

4 Block Diffusion Language Model

We use $c^{(k)} = b^{\star(<k)}$ for the clean prefix before block k, $\tilde{b}_t^{(k)}$ for the noised active block at reverse timestep t, and f_θ for the block denoiser. For Mamba layer ℓ, s_k^ℓ denotes the reusable forward recurrent boundary state produced by scanning the clean prefix. The symbols $\rightarrow$ and $\leftarrow$ denote forward and reverse Mamba scan directions, while $y_{k,t}^{\rightarrow,\ell}$ and $y_{k,t}^{\leftarrow,\ell}$ are the corresponding active-block hidden states.

Let $\mathcal{C}^\ell(c)$ denote the layer-ℓ prefix cache. Prefix-cache reuse requires this cache to be a function of the clean prefix alone, so $\mathcal{C}^\ell(c)$ must be independent of t.

Naively reusing a diffusion denoiser can violate this requirement because diffusion Transformers often inject timestep information through adaptive normalization applied to every token (Peebles & Xie, 2023). If the same modulation touches prefix tokens, the cache becomes timestep-specific and

must be bespoke for every reverse step. We therefore factor each layer into a timestep-free prefix operator and a timestep-conditioned active-block continuation:

$$\mathcal{C}^\ell(c) = P^\ell(c), \qquad h_{b,t}^{\ell+1} = B^\ell(h_{b,t}^\ell; \mathcal{C}^\ell(c), e_t). \tag{9}$$

Appendix E gives the corresponding AdaLN factorization.

Mamba directionality adds a second constraint. A fully bidirectional Mamba denoiser over $[c; \tilde{b}_t]$ combines a forward scan with a reverse scan over the reversed prefix-plus-block sequence. The reverse prefix states then depend on the active block, so they are not prefix-only values and cannot be reused as a BDLM prefix cache. Our BDLM Mamba hybrid instead runs the reverse-direction Mamba only within the current denoising block, rather than over the full sequence as in DiffuMamba (we denote this as a *block partial-reverse* architecture). BDLM Mamba caches only the forward boundary states,

$$s_k^\ell = F_{\theta,\rightarrow}^\ell(c^{(k)}), \tag{10}$$

and continues them into the active block:

$$y_{k,t}^{\rightarrow,\ell} = F_{\theta,\rightarrow}^\ell(\tilde{b}_t^{(k)}; s_k^\ell), \tag{11}$$

$$y_{k,t}^{\leftarrow,\ell} = \mathrm{rev}\left(F_{\theta,\leftarrow}^\ell(\mathrm{rev}(\tilde{b}_t^{(k)}))\right), \tag{12}$$

$$y_{k,t}^\ell = y_{k,t}^{\rightarrow,\ell} + y_{k,t}^{\leftarrow,\ell}. \tag{13}$$

The active block still receives both left-to-right and right-to-left mixing, but only within the current block. The reusable prefix cache remains valid because the reverse scan is local to the active block. This serves as a native BDLM analogue of generation with block-caching: generated prefix blocks are summarized by forward recurrent state, while denoising inside the current block keeps local bidirectionality.

The prefix-cache objective trains the same forward cached representations which will be used at inference:

$$\mathcal{L}_{\mathrm{prefix}}(\theta) = \frac{1}{K}\sum_k \mathbb{E}_t \ell_\theta(b^{\star(k)} \mid s_k^{1:L}, \tilde{b}_t^{(k)}, t). \tag{14}$$

For attention layers, the corresponding cache object is the usual prefix key/value tensor. For Mamba layers, the cache object is the recurrent boundary state, including convolution and selective-SSM state needed to continue the scan into the active block.

4.1 Block Diffusion Mamba Hybrid Training

Consider a sequence split into blocks b_1, b_2, b_3, b_4, and let b_4 be the active diffusion block. The model first scans the clean prefix to produce the Mamba prefix cache

$$C_4 = F_\theta(b_1, b_2, b_3), \tag{15}$$

where C_4 contains the convolution and state-space states at the boundary before b_4. The active block is then denoised from that same cache object:

$$\ell_4 = \mathrm{CE}\Big(D_\theta(C_4, \tilde{b}_4, t), b_4\Big). \tag{16}$$

Across all-block training, the same construction is evaluated for every block boundary in parallel. Backpropagation follows the prefix-cache construction path:

$$\ell_4 \rightarrow C_4 \rightarrow F_\theta(b_1, b_2, b_3) \rightarrow \theta. \tag{17}$$

The cache still contains the same prefix-only Mamba states that are reused at inference. Downstream block losses shape the representations stored in those states, aligning the training computation with the sampler's cache interface: the state consumed by later blocks is produced by the same prefix scan that is used during native BDLM generation.

5 EXPERIMENTS

5.1 EXPERIMENT SETUP

We evaluate training and inference throughput as well as validation performance of our BDLM Mamba hybrid against full-sequence attention, full-sequence DiffuMamba-H, and BDLM attention baselines. The 87M sweep trains all four architectures on the following learning rates: $\{5\times10^{-4}, 10^{-3}, 2\times10^{-3}, 4\times10^{-3}, 8\times10^{-3}\}$. All configurations use 5 billion tokens sampled from DCLM (Li et al., 2024), 8192-token training sequences, 16 diffusion steps, and no-timestep modulation; BDLM configurations use block size 256. The full-sequence DiffuMamba-H model uses a fully bidirectional Mamba hybrid denoiser over the denoising window, whereas BDLM Mamba-H only runs the reverse Mamba on the current active denoising block to leverage prefix-caching construction as described in Section 4.1.

Likelihood validation uses the same BDLM/MDLM-compatible diffusion-NELBO evaluator for all models at 8192-token context, with one Monte Carlo timestep/mask corruption sample per validation sequence. We report C4-en and Paloma-C4 NELBO perplexity and bits per byte (BPB). NELBO perplexity is computed as the exponential of the token-weighted average NELBO estimate, and BPB divides the same token-weighted negative log-likelihood in bits by the number of UTF-8 source-document bytes. We report MCQA separately as pseudo-likelihood answer-choice accuracy over fixed candidate answers. Masked-token accuracy from the diffusion-NELBO evaluator is included in the full 87M sweep in Table 5.

For MCQA, we evaluate HellaSwag, PIQA, ARC-Easy, ARC-Challenge, BoolQ, and WinoGrande (Zellers et al., 2019; Bisk et al., 2020; Clark et al., 2018; 2019; Sakaguchi et al., 2021) and report macro-average accuracy across tasks. For each example, we score each provided answer choice under the model's pseudo-likelihood scoring rule and select the highest-scoring choice. We use fixed-choice evaluation over evaluations that rely on free-form generation and answer extraction, making it suitable for comparing base models with no post training . The suite includes binary-choice tasks (PIQA, BoolQ, and WinoGrande) and multi-choice tasks with fixed or variable answer sets (HellaSwag, ARC-Easy, and ARC-Challenge).

Appendix D compares no-timestep and timestep-conditioned objectives. We use the no-timestep objective in the main experiments for two reasons. First, it matches the original BDLM experimental convention (Arriola et al., 2025). Second, it keeps clean-prefix caches timestep-invariant: if timestep modulation is applied to prefix tokens, the cached prefix states depend on the reverse diffusion step and cannot be reused across denoising steps. Even when a timestep is held fixed, a technically cache-aligned BDLM implementation should avoid learned timestep embeddings in the prefix-cache construction, because completed prefix states are properties of clean prefix text rather than of a particular reverse step. The AdaLN factorization in Appendix E describes how BDLM Mamba-H can make use of prefix caching; timestep modulation is applied only to the active denoising block, while the clean-prefix scan remains timestep-free. Empirically, the fixed learning rate comparison in Appendix D shows that timestep conditioning improves BDLM attention, but worsens BDLM Mamba-H and both full-sequence baselines on NELBO perplexity. We therefore use the no-timestep objective for the main model-selection and scale-up experiments.

For the 350M scale-up experiments, we transfer learning rates from the selected 87M configurations using a Complete(d)P-style hidden-weight rule (Mlodozeniec et al., 2025). We use this rule as a principled way to choose scale-up candidates because the 350M runs change width, depth, global batch size, and training horizon simultaneously. Recent diffusion-language-model scaling studies that find learning rate and batch size transfer rules remain useful for DLMs (Ni et al., 2025), and with work using CompleteP for stable learning-rate transfer across width and depth in discrete DLM scaling (von Rütte et al., 2026). The resulting learning rates are shown in Table 2. We select the best 87M models by C4-en heldout perplexity: 4×10^{-3} for BDLM attention and 8×10^{-3} for BDLM Mamba-H. Both 350M models use the same warmup length, global batch, sequence length, data stream, and no-timestep objective. Both models train for 33,668 optimizer steps (17.65B tokens), giving the token-to-parameter ratios reported in Table 2; they differ in architecture and Complete(d)P-transferred learning rate.

5.2 Language Modeling Quality

We select one checkpoint per architecture by the best C4-en $\mathrm{PPL}_{\mathrm{NELBO}}$ in the 87M-class learning-rate sweep. Under this selection rule, BDLM Mamba-H gives the lowest likelihood-bound validation perplexity: 61.6, compared with 76.5 for BDLM attention, 83.7 for full-sequence DiffuMamba-H, and 87.6 for full-sequence attention; the full 20-row sweep is present in Appendix Table 5. MCQA provides a complementary fixed-choice preference evaluation separate to the NELBO-perplexity ranking.

At 350M scale, the two selected BDLM models remain comparable with BDLM attention obtaining 37.0 C4-en $\mathrm{PPL}_{\mathrm{NELBO}}$ and 36.1 Paloma-C4 $\mathrm{PPL}_{\mathrm{NELBO}}$, while BDLM Mamba-H obtains 38.2 and 36.8, respectively. The corresponding MCQA accuracies are 41.3 for BDLM attention and 43.3 for BDLM Mamba-H. Thus we show BDLM Mamba-H remains in the same validation-quality range while enabling native long-context cached inference.

Table 1: Best 87M-class DCLM (Li et al., 2024) no-timestep validation configurations, selected by C4-en NELBO perplexity. PPL denotes the BDLM/MDLM-compatible diffusion-NELBO validation path with one timestep sample at 8192-token context. All models are trained on 5.00B DCLM tokens. BPB divides token-weighted negative log-likelihood in bits by the number of UTF-8 bytes in the source documents. MCQA is pseudo-likelihood accuracy over fixed answer choices. The full learning-rate sweep is reported in Table 5.

Model	LR	C4 PPL	C4 BPB	Paloma C4 PPL	Paloma BPB	MCQA
Partially Reverse BDLM Mamba-H	8×10^{-3}	**61.59**	**1.30**	**58.48**	**1.28**	35.58
BDLM attention	4×10^{-3}	76.46	1.37	70.81	1.34	36.31
Full-sequence DiffuMamba-H	4×10^{-3}	83.73	1.39	80.42	1.38	37.32
Full-sequence attention	8×10^{-3}	87.64	1.41	83.25	1.39	35.03

Table 2: 350M BDLM scale-up configuration. Learning rates are transferred from the best 87M C4-en validation settings using a Complete(d)P-style hidden-weight transfer rule (Mlodozeniec et al., 2025). Both models use no timestep conditioning, DCLM, the BDLM (Arriola et al., 2025) objective, 8192-token training sequences, global batch 64, 2000 warmup steps, the same optimizer-step horizon, and 17.65B DCLM training tokens.

Model	Params	87M LR	Transfer factor	350M LR	Steps	Token:param
BDLM attention	353.03M	4×10^{-3}	0.31	1.23×10^{-3}	33,668	50.00
Partially Reverse BDLM Mamba-H	350.97M	8×10^{-3}	0.33	2.65×10^{-3}	33,668	50.29

Table 3: 350M BDLM no-timestep validation results. C4 and Paloma C4 are evaluated at 8192-token context with 256 examples per dataset. BPB divides token-weighted negative log-likelihood in bits by the number of UTF-8 bytes in the source documents. MCQA is pseudo-likelihood accuracy over fixed answer choices.

Model	LR	C4 PPL	C4 BPB	Paloma C4 PPL	Paloma BPB	MCQA
BDLM attention	1.23×10^{-3}	**36.97**	**1.14**	**36.14**	**1.13**	41.28
Partially Reverse BDLM Mamba-H	2.65×10^{-3}	38.24	1.15	36.81	1.13	**43.29**

5.3 Training and Inference Throughput

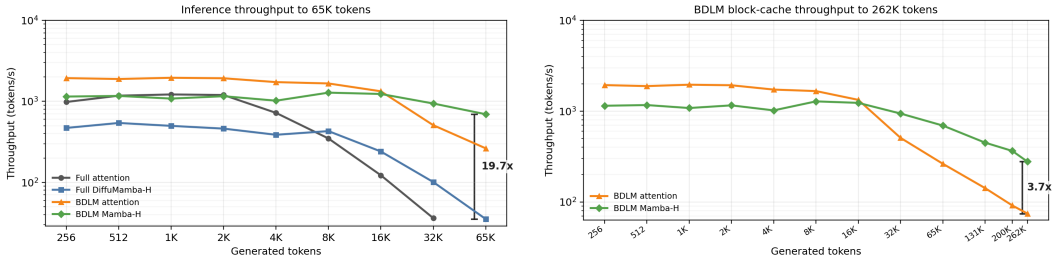
Training throughput is measured at 350M scale on 8x A100-80GB with synthetic 8192-token batches. Inference throughput is measured with 700M random-initialized models on 1x A100-80GB with BF16 and batch size 1. Both benchmarks use 5 warmup iterations and average over 15 measured iterations. The context generation lengths follow the DiffuMamba long-context protocol: full-sequence denoisers are measured through the 65K-token regime, while native BDLM models are extended along the block-cached ladder up to 262k generated tokens.

At 350M scale, all four architectures use the target global batch on 8x A100-80GB and fit into memory without activation checkpointing. The full-sequence models are significantly faster in 8192-token training throughput, reflecting the additional cost of evaluating the all-block BDLM objective during training. Table 4 shows that BDLM Mamba-H achieves 166,849 training tokens/s versus 151,976 for BDLM attention, a 9.8% improvement. BDLM Mamba-H also uses less peak reserved memory. At inference time, the native BDLM models retain substantially higher throughput than full-sequence denoisers as generation length grows. BDLM attention is faster at short lengths in our implementation, while the measured crossover to BDLM Mamba-H occurs between 16K and 32K generated tokens (Figure 2; Table 6).

Table 4: 350M 8x A100-80GB training throughput at 8192-token context. All models use synthetic batches, global batch 64, BF16, ZeRO-2, tiled linear cross-entropy, and FlashAttention kernels when available.

Model	GPUs	Global batch	tok/s
Full-sequence attention	8x A100	64	229,957
Full-sequence DiffuMamba-H	8x A100	64	223,825
BDLM attention	8x A100	64	151,976
Partially Reverse BDLM Mamba-H	8x A100	64	166,849

Figure 2(a) shows that, at 65K, BDLM Mamba-H is 19.7x faster than full-sequence DiffuMamba-H at the same length. Figure 2(b) extends the comparison to longer generation lengths, where BDLM Mamba-H is 3.7x faster than BDLM attention at 262K. This crossover is qualitatively consistent with DiffuMamba's finding that Mamba-backed block generation increasingly separates from attention-backed block generation as sequence length grows (Singh et al., 2026). Table 6 provides the underlying tokens/s measurements; Appendix B details the measurement protocol and length-by-length speedup ratios.



(a) Throughput to 65K tokens. (b) BDLM throughput on the extended ladder.

Figure 2: 700M random-initialized single-A100 inference throughput. Full-sequence baselines use repeated full-sequence denoising with CUDA graphs and are reported. Native BDLM models use block size 256, BF16, cacheable prefix states, and optimized cached denoising; the extended ladder includes completed BDLM attention and BDLM Mamba-H points through 262K generated tokens.

6 Limitations

While the block objective improves long-context generation, we find that at 350M parameters on 8x A100-80GB, BDLM attention trains 33.9% slower than the full-sequence attention baseline, and BDLM Mamba-H trains 25.5% slower than the full-sequence DiffuMamba-H baseline. We leave it to future work to investigate optimal scaling laws for hyperparameter transfer for dLLMs and hybrid variants thereof. Complete(d)P provided a parameterization-based method of transferring learning rates across width, depth, batch, and duration (Mlodozeniec et al., 2025), but did not explicitly derive these laws for dLLMs. Masked and block diffusion objectives have different timestep sampling, loss weighting, and caching constraints, so the transferred 350M learning rates are strong candidates rather than a proof of optimality.

7 CONCLUSION

We presented a training recipe for block diffusion Mamba hybrids that trains the same prefix cache used during native BDLM generation. The method keeps the prefix scan timestep-free to optionally support valid AdaLN timestep conditioning for BDLM Mamba, confines reverse-direction Mamba mixing to the active block rather than applying a full-window reverse pass, and trains the prefix cache through downstream block losses.

Our trained models preserve validation quality while improving the long-context systems regime. At 87M parameters, BDLM Mamba-H reaches the best C4-en PPL in the DCLM sweep, and at 350M parameters its PPL remains in the same range as BDLM attention. In inference, the native BDLM models extend to lengths that full-sequence denoisers do not practically cover, and BDLM Mamba-H increasingly separates at long context, reaching 19.7x the full-sequence DiffuMamba-H counterpart at 65K tokens and 3.7x BDLM attention throughput at 262K. We show that training block partial-reverse BDLM Mamba hybrids yields a practical long-context diffusion architecture worth further exploring.

AI USE STATEMENT

We used generative AI tools to assist with developing the training pipeline, including dataset streaming and the implementation of parallelization strategies for block diffusion hybrid models. AI tools also assisted with manuscript preparation, LaTeX formatting, and cross-reference checks. All AI-generated or AI-assisted code was reviewed and tested by the authors before incorporation into the reported experiments, and AI-assisted text was reviewed and edited for accuracy and clarity. The authors take full responsibility for the final manuscript, implementation, and reported results.

ETHICS STATEMENT

This work studies language-model architectures and computational efficiency. The reported likelihood and multiple-choice evaluations do not establish safety, fairness, privacy protection, or suitability for deployment. Models trained on web text can reproduce harmful biases and sensitive information; improved generation efficiency can also lower the cost of misuse. Any release or deployment requires appropriate attention to data and model licenses, privacy, and application-specific safety evaluation.

REPRODUCIBILITY STATEMENT

Section 4.1 describes the prefix-cache training construction. Appendix C reports architecture and hyperparameter settings, Appendix A contains the full 87M validation sweep, and Appendix D reports the conditioning comparison. Appendix B describes the inference measurement protocol and provides raw throughput values. The inference throughput results use random-initialized 700M models and are separate from the trained 87M and 350M quality evaluations.

REFERENCES

Marianne Arriola et al. Block diffusion: Interpolating between autoregressive and diffusion language models. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2503.09573.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*, 2021.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about physical commonsense in natural language. In *AAAI Conference on Artificial Intelligence*, 2020.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Conference of the North American Chapter of the Association for Computational Linguistics*, 2019.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018. URL https://arxiv.org/abs/1803.05457.

Tri Dao and Albert Gu. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In *International Conference on Machine Learning*, 2024. URL https://arxiv.org/abs/2405.21060.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. In *Advances in Neural Information Processing Systems*, 2022.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=tEYskw1VY2.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. In *Advances in Neural Information Processing Systems*, 2021.

Kimi Team, Yu Zhang, Zongyu Lin, Xingcheng Yao, Jiaxi Hu, Fanqing Meng, Chengyin Liu, Xin Men, Songlin Yang, Zhiyuan Li, Wentao Li, Enzhe Lu, Weizhou Liu, Yanru Chen, Weixin Xu, Longhui Yu, Yejie Wang, Yu Fan, Longguang Zhong, Enming Yuan, Dehao Zhang, Yizhi Zhang, T. Y. Liu, Haiming Wang, Shengjun Fang, Weiran He, Shaowei Liu, Yiwei Li, Jianlin Su, Jiezhong Qiu, Bo Pang, Junjie Yan, Zhejun Jiang, Weixiao Huang, Bohong Yin, Jiacheng You, Chu Wei, Zhengtao Wang, Chao Hong, Yutian Chen, Guanduo Chen, Yucheng Wang, Huabin Zheng, Feng Wang, Yibo Liu, Mengnan Dong, Zheng Zhang, Siyuan Pan, Wenhao Wu, Yuhao Wu, Longyu Guan, Jiawen Tao, Guohong Fu, Xinran Xu, Yuzhi Wang, Guokun Lai, Yuxin Wu, Xinyu Zhou, Zhilin Yang, and Yulun Du. Kimi linear: An expressive, efficient attention architecture. *Technical report*, 2025. URL https://github.com/MoonshotAI/Kimi-Linear.

Aonian Li et al. Minimax-01: Scaling foundation models with lightning attention. *Technical report*, 2025. URL https://github.com/MiniMax-AI/MiniMax-01.

Jeffrey Li, Alex Fang, Georgios Smyrnis, Maor Ivgi, Matt Jordan, Samir Yitzhak Gadre, Hritik Bansal, Etash Guha, Sedrick Keh, Kushal Arora, Saurabh Garg, Rui Xin, Niklas Muennighoff, Reinhard Heckel, Jean Mercat, Mayee Chen, Suchin Gururangan, Mitchell Wortsman, Alon Albalak, Yonatan Bitton, Marianna Nezhurina, Amro Abbas, Cheng-Yu Hsieh, Dhruba Ghosh, Josh Gardner, Maciej Kilian, Hanlin Zhang, Rulin Shao, Sarah Pratt, Sunny Sanyal, Gabriel Ilharco, Giannis Daras, Kalyani Marathe, Aaron Gokaslan, Jieyu Zhang, Khyathi Chandu, Thao Nguyen, Igor Vasiljevic, Sham Kakade, Shuran Song, Sujay Sanghavi, Fartash Faghri, Sewoong Oh, Luke Zettlemoyer, Kyle Lo, Alaaeldin El-Nouby, Hadi Pouransari, Alexander Toshev, Stephanie Wang, Dirk Groeneveld, Luca Soldaini, Pang Wei Koh, Jenia Jitsev, Thomas Kollar, Alexandros G. Dimakis, Yair Carmon, Achal Dave, Ludwig Schmidt, and Vaishaal Shankar. DataComp-LM: In search of the next generation of training sets for language models. In *Advances in Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. URL https://openreview.net/forum?id=CNWdWn47IE.

Opher Lieber et al. Jamba: A hybrid transformer-mamba language model. *OpenReview*, 2024. URL https://openreview.net/forum?id=JFPaD7lpBD.

Ling Team, Bin Han, Caizhi Tang, Chen Liang, Donghao Zhang, Fan Yuan, Feng Zhu, Jie Gao, Jingyu Hu, Longfei Li, Meng Li, Mingyang Zhang, Peijie Jiang, Peng Jiao, Qian Zhao, Qingyuan Yang, Wenbo Shen, Xinxing Yang, Yalin Zhang, Yankun Ren, Yao Zhao, Yibo Cao, Yixuan Sun, Yue Zhang, Yuchen Fang, Zibin Lin, Zixuan Cheng, and Jun Zhou. Every attention matters: An efficient hybrid architecture for long-context reasoning. *arXiv preprint arXiv:2510.19338*, 2025.

Zhiyuan Liu, Yicun Yang, Yaojie Zhang, Junjie Chen, Chang Zou, Qingyuan Wei, Shaobo Wang, and Linfeng Zhang. dLLM-Cache: Accelerating diffusion large language models with adaptive caching. *arXiv preprint arXiv:2506.06295*, 2025. URL https://arxiv.org/abs/2506.06295.

Xinyin Ma, Runpeng Yu, Gongfan Fang, and Xinchao Wang. dKV-Cache: The cache for diffusion language models. *arXiv preprint arXiv:2505.15781*, 2025. URL https://arxiv.org/abs/2505.15781.

Bruno Mlodozeniec, Pierre Ablin, Louis Béthune, Dan Busbridge, Michal Klein, Jason Ramapuram, and Marco Cuturi. Completed hyperparameter transfer across modules, width, depth, batch and duration. *arXiv preprint arXiv:2512.22382*, 2025. doi: 10.48550/arXiv.2512.22382. URL https://arxiv.org/abs/2512.22382.

Quan Nguyen-Tri, Mukul Ranjan, and Zhiqiang Shen. Attention is all you need for KV cache in diffusion LLMs. *arXiv preprint*, 2025.

Jinjie Ni, Qian Liu, Chao Du, Longxu Dou, Hang Yan, Zili Wang, Tianyu Pang, and Michael Qizhe Shieh. Training optimal large diffusion language models. *arXiv preprint arXiv:2510.03280*, 2025. URL https://arxiv.org/abs/2510.03280.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. In *Advances in Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=KnqiC0znVF.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *International Conference on Computer Vision*, 2023. URL https://arxiv.org/abs/2212.09748.

Zhen Qin, Weigao Sun, Dong Li, Xuyang Shen, Weixuan Sun, and Yiran Zhong. Various lengths, constant speed: Efficient language modeling with lightning attention. In *Proceedings of the 41st International Conference on Machine Learning*, 2024. URL https://proceedings.mlr.press/v235/qin24c.html.

Qwen Team. Qwen3-next: Towards ultimate training efficiency. Qwen technical blog, 2025. URL https://qwenlm.github.io/blog/qwen3-next/.

Subham Sekhar Sahoo, Marianne Arriola, Aaron Gokaslan, Justin T. Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. In *Advances in Neural Information Processing Systems*, 2024.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial winograd schema challenge at scale. In *AAAI Conference on Artificial Intelligence*, 2021.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis K. Titsias. Simplified and generalized masked diffusion for discrete data. In *Advances in Neural Information Processing Systems*, volume 37, pp. 103131–103167, 2024. doi: 10.52202/079017-3277. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/bad233b9849f019aead5e5cc60cef70f-Abstract-Conference.html.

Vaibhav Singh, Oleksiy Ostapenko, Pierre-André Noel, Eugene Belilovsky, and Torsten Scholak. Diffumamba: High-throughput diffusion LMs with mamba backbone. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2511.15927.

Dimitri von Rütte, Janis Fluri, Omead Pooladzandi, Bernhard Schölkopf, Thomas Hofmann, and Antonio Orvieto. Scaling behavior of discrete diffusion language models. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=GDYaNzxt9T.

Xu Wang, Chenkai Xu, Yijie Jin, Jiachun Jin, Hao Zhang, and Zhijie Deng. Diffusion LLMs can do faster-than-AR inference via discrete diffusion forcing. *arXiv preprint arXiv:2508.09192*, 2025. URL https://arxiv.org/abs/2508.09192.

Chengyue Wu, Hao Zhang, Shuchen Xue, Zhijian Liu, Shizhe Diao, Ligeng Zhu, Ping Luo, Song Han, and Enze Xie. Fast-dLLM: Training-free acceleration of diffusion LLM by enabling KV cache and parallel decoding. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=3Z3Is6hnOT.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Annual Meeting of the Association for Computational Linguistics*, 2019.

A Full 87M Validation Sweep

Table 5: Full 87M DCLM no-timestep validation sweep. All checkpoints are evaluated at step 19,074 with the same 8192-token diffusion-NELBO validation path used in Table 1.

Architecture	LR	C4 PPL	C4 BPB	C4 acc.	Paloma PPL	Paloma BPB	Paloma acc.	MCQA
Full attention	5×10^{-4}	111.59	1.48	10.94	106.39	1.47	11.32	33.76
Full attention	10^{-3}	101.48	1.46	11.47	95.39	1.43	11.78	37.31
Full attention	2×10^{-3}	98.78	1.45	11.45	94.12	1.43	11.70	36.79
Full attention	4×10^{-3}	89.02	1.41	11.98	83.80	1.39	12.41	33.95
Full attention	8×10^{-3}	87.64	1.41	12.04	83.25	1.39	12.43	35.03
Full-sequence DiffuMamba-H	5×10^{-4}	104.39	1.46	11.27	98.98	1.44	11.55	36.04
Full-sequence DiffuMamba-H	10^{-3}	95.52	1.44	11.50	92.33	1.42	11.82	35.52
Full-sequence DiffuMamba-H	2×10^{-3}	93.97	1.43	11.53	90.31	1.42	11.95	35.67
Full-sequence DiffuMamba-H	4×10^{-3}	83.73	1.39	12.08	80.42	1.38	12.50	37.32
Full-sequence DiffuMamba-H	8×10^{-3}	85.50	1.40	12.00	81.46	1.38	12.41	34.49
BDLM attention	5×10^{-4}	91.35	1.42	10.60	84.46	1.40	10.72	41.04
BDLM attention	10^{-3}	83.97	1.40	10.95	76.86	1.37	11.10	40.08
BDLM attention	2×10^{-3}	77.10	1.37	11.24	71.05	1.34	11.39	41.50
BDLM attention	4×10^{-3}	76.46	1.37	11.29	70.81	1.34	11.43	36.31
BDLM attention	8×10^{-3}	80.43	1.38	11.09	73.54	1.35	11.24	36.56
Partially Reverse BDLM Mamba-H	5×10^{-4}	81.94	1.39	10.80	74.46	1.36	11.02	34.76
Partially Reverse BDLM Mamba-H	10^{-3}	72.97	1.35	11.17	67.06	1.32	11.41	36.48
Partially Reverse BDLM Mamba-H	2×10^{-3}	67.02	1.32	11.60	62.69	1.30	11.66	34.21
Partially Reverse BDLM Mamba-H	4×10^{-3}	62.08	1.30	11.90	58.42	1.28	12.00	35.82
Partially Reverse BDLM Mamba-H	8×10^{-3}	61.59	1.30	11.92	58.48	1.28	11.99	35.58

B Inference Throughput Protocol and Raw Results

All inference models use the same block size, denoising-step ratio, precision, and measurement policy. Full-sequence baselines use fixed-shape CUDA graphs for repeated full-window denoising and FlashAttention kernels where available (Dao et al., 2022). BDLM attention uses preallocated mutable prefix KV caches, tiled key-cache append, BDLM-style FlashAttention for prefix-cache and active-block attention, graph-captured block denoising steps, and tiled LM-head evaluation. BDLM Mamba-H uses the block partial-reverse Mamba construction: native Mamba kernels, chunked prefix prefill, read-only recurrent cache assumptions, compiled fixed-shape Mamba block continuations with CUDA graphs enabled, graph-captured block denoising steps, and the same chunked LM-head path.

Table 6 gives the raw inference throughput numbers used in Figure 2. In our implementation, BDLM attention is faster at short lengths, where the KV cache is still small and softmax attention remains efficient. BDLM Mamba-H overtakes BDLM attention at longer lengths: 1.9x at 32K, 2.6x at 65K, 3.2x at 131K, 4.0x at 200K, and 3.7x at 262K relative to BDLM attention. These absolute ratios are specific to the 700M random-initialized A100 benchmark used here.

Table 6: Raw 700M random-initialized inference throughput in tokens/s. Values are measured on 1x A100-80GB with BF16 and batch size 1. Full-sequence baselines use CUDA graphs for fixed-shape denoising. Native BDLM models use block size 256 and optimized cached denoising with graph-captured block steps where supported.

Model	256	512	1K	2K	4K	8K	16K	32K	65K	131K	200K	262K
Full-sequence attention	1,035	915	1,421	1,062	761	351	122	36	–	–	–	–
Full-sequence DiffuMamba-H	470	541	498	461	387	429	240	101	35	–	–	–
BDLM attention	1,935	1,887	1,955	1,927	1,728	1,663	1,331	508	263	142	91	74
Partially Reverse BDLM Mamba-H	1,146	1,167	1,082	1,158	1,019	1,281	1,234	941	694	450	364	278

C Model Architecture Details

Tables 7 and 8 give the model geometry used for the reported 87M sweep and 350M scale-up configurations. All configurations use an MLP ratio of 4, timestep embedding dimension 256, Mamba convolution width 4, and Mamba expansion factor 2. DiffuMamba-H hybrid configurations use attention at layers $0, 6, 12, \ldots$ and Mamba-2 mixers in the intervening layers. The BDLM Mamba-H architecture keeps that sparse-attention hybrid schedule, but uses the block partial-reverse construction described in Section 4.1 rather than a fully bidirectional full-sequence denoiser.

Table 7: 87M architecture details. Parameter counts are total trainable parameters for the released model definitions. Hybrid models use attention at layers 0 and 6.

Model	Width	Layers	Heads	Head dim.	Mamba state/chunk	Params
Full-sequence attention	448	12	8	56	–	89.17M
Full-sequence DiffuMamba-H	448	12	8	56	64 / 128	96.98M
BDLM attention	448	12	8	56	–	89.17M
BDLM Mamba-H	448	12	8	56	64 / 128	96.98M

Table 8: 350M architecture details. Parameter counts are total trainable parameters for the released model definitions. Hybrid models use attention at layers 0, 6, and 12; the BDLM Mamba-H architecture uses a width-adjusted 350M-class geometry to match the BDLM attention scale.

Model	Width	Layers	Heads	Head dim.	Mamba state/chunk	Params
Full-sequence attention	896	18	14	64	–	353.03M
Full-sequence DiffuMamba-H	896	18	14	64	128 / 128	399.53M
BDLM attention	896	18	14	64	–	353.03M
BDLM Mamba-H	832	18	16	52	128 / 128	350.97M

Table 9: Key training and inference hyperparameters used across the reported DCLM experiments unless stated otherwise. Learning-rate values differ across the 87M sweep and 350M scale-up configurations as shown in Tables 1 and 2.

Category	Setting
Backbones	Attention, DiffuMamba-H, BDLM attention, BDLM Mamba-H
Diffusion type	Absorbing-state masked diffusion
Training objective	BDLM/MDLM continuous-time diffusion-NELBO (Arriola et al., 2025)
Parameterization	Substitution denoising target
Noise schedule	Log-linear; $t \sim \mathrm{Uniform}(10^{-3}, 1)$
Timestep conditioning	Disabled for main no-timestep sweep and 350M scale-up
Dataset	DCLM baseline, streaming document packing
Tokenizer	GPT-2 BPE, vocabulary size 50,257 plus mask token
Training sequence length	8192 tokens
RoPE context capacity	262,144 tokens, $\theta = 10{,}000$
BDLM block length	256 tokens
Diffusion steps / factor p	16
Precision	BF16
Distributed training	DeepSpeed ZeRO-2
Activation checkpointing	Disabled
Optimizer	AdamW
Adam betas / epsilon	$(0.9, 0.95)$ / 10^{-8}
Weight decay	0.10
Gradient clipping	1.00
LR scheduler	Linear warmup, then cosine decay to 10^{-6}
Warmup steps	2000
EMA / antithetic sampling	Neither used
87M global batch / steps	32 / 19,074
87M LR sweep	5×10^{-4}, 10^{-3}, 2×10^{-3}, 4×10^{-3}, 8×10^{-3}
350M global batch / steps	64 / 33,668
350M max learning rates	1.23×10^{-3} for BDLM attention; 2.65×10^{-3} for BDLM Mamba-H
Inference batch size	1
Inference precision	BF16
Inference block length	256 tokens for native BDLM models
Inference denoising factors	$p \in \{8, 16\}$ in sweeps; reported settings use $p = 16$

D No-Timestep Versus Timestep Conditioning

Table 10 compares no-timestep and original timestep conditioning at the 87M fixed-learning-rate slice used during recipe validation. The relevant question for this paper is whether BDLM Mamba-H can be trained with a timestep-free prefix path without losing the validation signal needed for model selection.

Table 10: 87M DCLM conditioning comparison at LR 4×10^{-3} and 8192-token validation context. PPL is the BDLM/MDLM-compatible diffusion-NELBO perplexity. No-timestep values are copied from the main 87M sweep; timestep-conditioned models use the DDiT-normalization path and the same one-sample BDLM/MDLM-compatible validation protocol.

Model	Objective	No-timestep C4 PPL	Timestep C4 PPL	No-timestep MCQA	Timestep MCQA
Full attention	MDLM	89.02	92.22	33.95	41.92
Full DiffuMamba-H	MDLM	83.73	89.58	37.32	42.51
BDLM attention	BDLM	76.46	64.13	36.31	44.85
BDLM Mamba-H	BDLM	62.08	65.55	35.82	40.70

We find timestep conditioning improves BDLM attention at this learning rate, slightly worsens BDLM Mamba-H, and worsens both full-sequence baselines. We therefore use the five-learning-rate no-timestep sweep for main model selection, matching the original BDLM experimental convention (Arriola et al., 2025) and preserving a timestep-invariant prefix path. Timestep conditioning

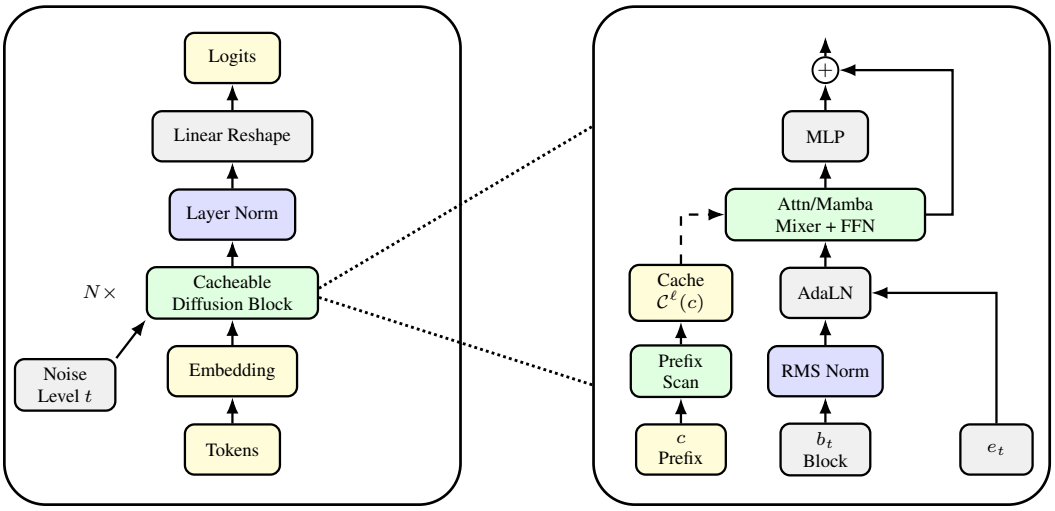


Figure 3: Cacheable AdaLN factorization for block diffusion. The clean prefix path is evaluated once without timestep modulation, producing a timestep-invariant cache $\mathcal{C}^\ell(c)$ containing attention keys/values or recurrent sequence state. At each reverse step, AdaLN(e_t) modulates only the active-block continuation; the top $+$ denotes the residual add after the MLP update. This keeps prefix reuse aligned with the BDLM factorization, rather than a post-hoc approximation to a full-sequence denoiser.

remains architecturally supported through the active-block-only AdaLN factorization in Appendix E, which keeps clean-prefix caches independent of the reverse timestep.

E AdaLN Factorization

The architectural requirement for reusable prefix states is that they do not depend on the reverse timestep. Standard diffusion Transformers often inject timestep conditioning into every layer through adaptive normalization. If that modulation touches prefix tokens, then the layer cache becomes

$$\mathcal{C}^\ell(c,t) = F^\ell_{\text{prefix}}(c, e_t), \tag{18}$$

which changes at every reverse step and cannot be reused across the active block's denoising trajectory. Adaptive layer normalization commonly maps a timestep embedding e_t into shift, scale, and gate parameters and applies them before mixer and feed-forward updates:

$$(\delta^\ell_1, \gamma^\ell_1, g^\ell_1, \delta^\ell_2, \gamma^\ell_2, g^\ell_2) = A^\ell(e_t), \tag{19}$$

$$u^\ell_i(t) = (1+\gamma^\ell_1) \odot \mathrm{LN}(h^\ell_i) + \delta^\ell_1. \tag{20}$$

For a full-sequence denoiser this modulation can touch every token. For a BDLM prefix cache, applying it to prefix tokens would make cached states timestep-dependent and force a different cache for every reverse step. The cacheable layer therefore factors as

$$\mathcal{C}^\ell(c) = F^\ell_{\text{prefix}}(c), \qquad h^{\ell+1}_{b,t} = F^\ell_{\text{active}}\left(h^\ell_{b,t}, \mathcal{C}^\ell(c), e_t\right). \tag{21}$$

The clean prefix path is evaluated once without timestep modulation; AdaLN and timestep gating are applied only to the active-block continuation. This remains the cleaner cache semantics even if an implementation uses a fixed timestep for all prefix tokens: the cached state should summarize completed text, not a learned timestep embedding attached to that text.

This factorization is the reason the same prefix cache can be reused across reverse diffusion steps. It is independent of whether the cached state contains attention keys and values, Mamba recurrent states, or another layer-specific summary.