

Signals from the Future: Future-aware Semi-Autoregressive Decoding for Diffusion Language Models

Yangryeol Park Jeonhwi Kang Kunhui Lee
Hyeonju Kim Jinyoung Choi* Seung-Hoon Na*
Ulsan National Institute of Science and Technology (UNIST)
{yangreal,jin0.choi,nash}@unist.ac.kr

Abstract

Standard semi-autoregressive decoding for full-attention masked diffusion language models (MDLMs) computes predictions for future blocks but discards them until those blocks become active. When introduced as soft-future context, these predictions reveal a stability contrast: hypotheses later revised during decoding exhibit larger distributional shifts than those eventually retained. We introduce **FUSA** (**F**uture-aware **S**emi-Autoregressive Decoding), a training-free decoder that exploits this contrast by alternating the next block between masked and soft representations and committing active-block tokens that remain stable across the two views, without additional model forward calls. On LLaDA-8B-Instruct, FUSA improves scores across reasoning and code-generation benchmarks while delivering 2.10–3.94× wall-clock speedups over Top-1 decoding, with gains of up to 5.1 points. Additional results show that FUSA transfers to other MDLMs and remains effective under KV-cached inference and in long-form instruction following. Our code is available at <https://github.com/yangryeol/FUSA>.

1 Introduction

Masked diffusion language models (MDLMs) generate text by iteratively denoising a masked sequence (Shi et al., 2024; Sahoo et al., 2024; Nie et al., 2025). Due to the computational inefficiency and scheduling difficulty of fully parallel denoising over an entire output sequence, modern MDLM inference commonly adopts semi-autoregressive decoding, where the output sequence is divided into blocks, processed from left to right, and denoised in parallel within the active block (Sahoo et al., 2024; Nie et al., 2025; Arriola et al., 2025).

Standard semi-autoregressive decoding methods, such as LLaDA (Nie et al., 2025), employ full-

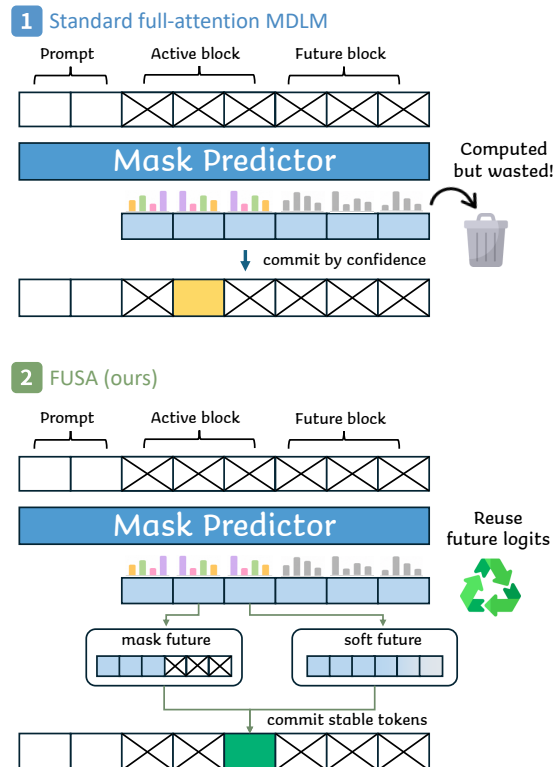


Figure 1: Standard blockwise decoding masks future blocks and discards their predicted distributions. FUSA reuses these predictions as soft future views, accepting active-block tokens whose predictions remain stable across future contexts.

attention MDLMs trained with bidirectional attention while applying blockwise decoding only at inference time. Figure 1 briefly illustrates that these methods discard future-position logits and commit only active-block tokens.

This raises a natural question: **if future-block distributions are already available, can they help determine which active-block tokens should be unmasked?** Since the active block and future blocks eventually form a coherent sequence, the model’s own future predictions may provide a useful stability signal. When introduced as *soft-future*

*Corresponding authors

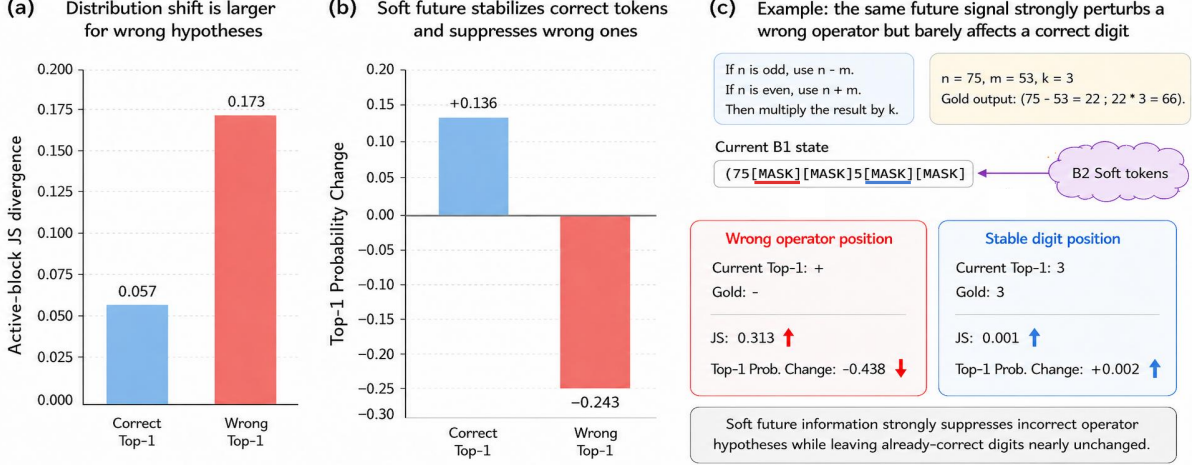


Figure 2: Soft-future contrast in the arithmetic perturbation probe. Wrong Top-1 hypotheses are more sensitive to soft future evidence, while correct Top-1 hypotheses are comparatively stable and often reinforced. The token-level example illustrates how soft future context perturbs an incorrect prediction but leaves an already-correct one nearly unchanged. We summarize this diagnostic in Section 5.3 and provide the full protocol and additional details in Appendix B.1.

context, these predictions allow active-block hypotheses to be tested for their sensitivity to plausible future evidence. Figure 2 illustrates this idea in a controlled arithmetic probe, where Wrong Top-1 hypotheses are more sensitive to soft-future context than Correct Top-1 hypotheses. Section 5.3 complements this controlled probe with a trajectory-level diagnostic across reasoning and code-generation tasks.

Motivated by these observations, we propose **FUSA (Future-aware Semi-Autoregressive Decoding)**. As illustrated in Figure 3, FUSA progressively unmask active-block tokens while alternating the next block between masked and soft representations. In a **soft-future step**, future positions carry temporary soft future tokens constructed from the previous step’s future-block distributions, whereas in the **mask-future step**, they are represented by [MASK] embeddings. During each step transition, FUSA evaluates token stability by comparing the predictive distributions obtained at the current step with those cached from the preceding step. Active-block tokens are then progressively unmasked using this future-view stability signal together with confidence.

Our contributions are summarized as follows:

- 1) We identify a future-conditioned stability signal in full-attention MDLM decoding, supported by a controlled probe and cross-task trajectory diagnostics.
- 2) We propose **FUSA**, a training-free future-aware

semi-autoregressive decoder that uses this signal to guide active-block token commitment without additional model forward calls.

- 3) We validate FUSA across multiple full-attention MDLMs and generation settings, demonstrating substantial decoding speedups alongside strong performance across reasoning, code generation, and long-form instruction following.

2 Related Work

2.1 Masked Diffusion Language Models

Masked diffusion language models extend masked token prediction into iterative generative denoising over discrete tokens: BERT introduced bidirectional masked language modeling (Devlin et al., 2019), D3PM formalized discrete diffusion with absorbing mask states (Austin et al., 2021a), and SEDD and MDLM developed effective discrete-token diffusion objectives and samplers (Lou et al., 2024; Sahoo et al., 2024), building on earlier text diffusion systems such as Diffusion-LM and Dif-fuSeq (Li et al., 2022; Gong et al., 2023). Recent large-scale systems show that this paradigm can scale to multi-billion-parameter language models (Nie et al., 2025; Ye et al., 2025). Their appeal is bidirectional refinement: unlike autoregressive models, they can predict and revise multiple masked positions in parallel, but this creates a decoding problem of how to group, select, commit, and reuse token predictions across denoising steps.

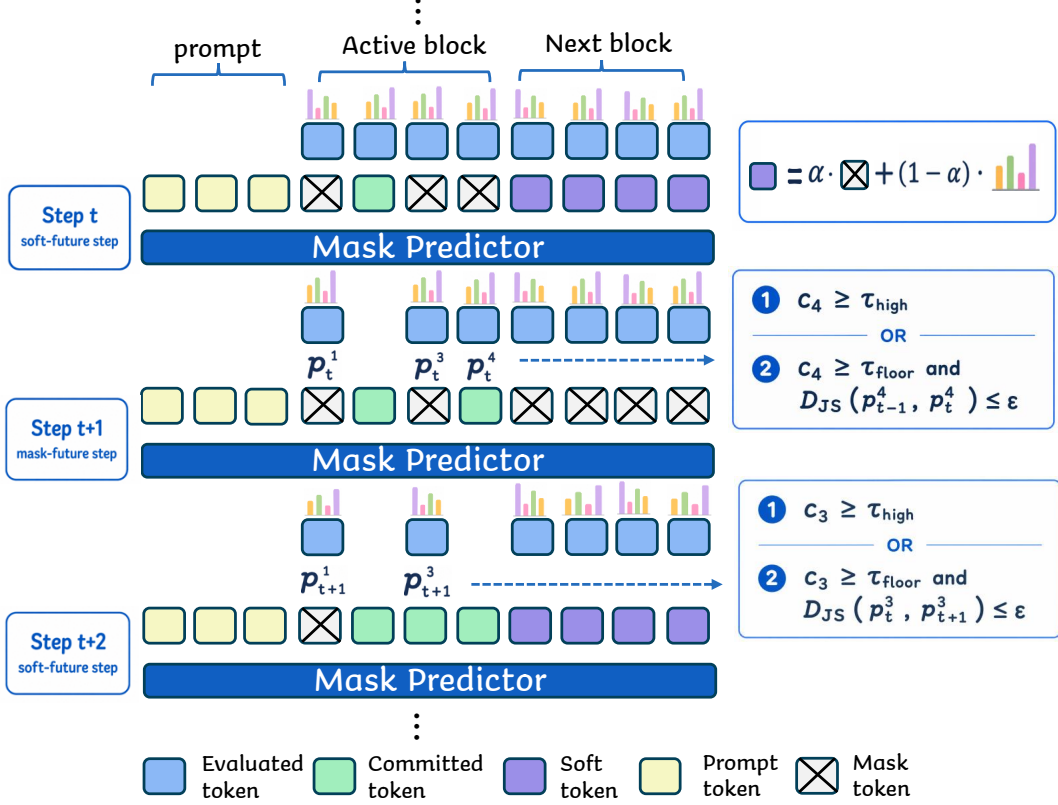


Figure 3: Overview of FUSA: each denoising step uses one model forward. For an active-block position, the decoder takes the top prediction y_t^i and its confidence $c_t^i = p_t^i(y_t^i)$. A position is unmasked when either (1) its confidence exceeds the standard threshold, or (2) its prediction remains stable across future views while satisfying a lower confidence floor. Next-block logits from a mask-future step become temporary soft future tokens for the following soft-future step.

2.2 Semi-Autoregressive and Block-Structured Decoding

A common efficient inference strategy is semi-autoregressive sampling, which divides the output into fixed-size blocks, processes blocks left to right, and denoises tokens within the active block in parallel; LLaDA applies this blockwise procedure while keeping previously generated blocks fixed (Nie et al., 2025). In standard full-attention MDLMs, positions after the active block are still forwarded as [MASK] tokens and receive predictive distributions, but conventional decoding commits only active-block tokens, discards predictions for later blocks, and resets those positions to [MASK] until their block becomes active. Thus, future-block predictions are computed but not retained. By contrast, block-structured diffusion and diffusion-autoregressive models often remove future blocks from the current computation, hide them behind block-causal attention, or process them only when active; recent examples include Block Diffusion, SDAR, Fast-dLLM v2, LLaDA2.0, and LLaDA2.1

(Arriola et al., 2025; Cheng et al., 2026; Wu et al., 2026a; Bie et al., 2025, 2026).

2.3 Beyond Confidence-Based Diffusion Decoding

Because MDLM inference speed depends on how many tokens can be safely committed at each step, confidence-based decoding prioritizes high-probability predictions for early commitment, and Fast-dLLM similarly uses confidence-aware parallel decoding to improve throughput (Wu et al., 2026b). However, local confidence can be myopic: a token that appears confident under the current masked context can still change under later denoising updates or additional uncommitted context. Recent methods therefore add signals beyond local probability, including KLASS-style temporal stability, lookahead predictions, decoding-path uncertainty, position-aware calibration, and adaptive sequence lengths (Kim et al., 2025; Lee et al., 2026; Chen et al., 2025b; Huang et al., 2025; Li et al., 2026). Related work on soft, residual, or

latent token states further shows that intermediate token distributions can carry useful information beyond the ordinary [MASK] state (Hu et al., 2026; Zhu et al., 2025; Hersche et al., 2026; Zhong et al., 2026; Xia et al., 2026; He et al., 2026).

FUSA also moves beyond local confidence by testing the stability of current-block predictions with next-block distributions that standard semi-autoregressive decoding would otherwise reset to [MASK].

3 Preliminaries

3.1 Masked Diffusion Language Models

A masked diffusion language model generates a discrete sequence by reversing an absorbing masking process. Let $\mathbf{x}_0 = (x_0^1, \dots, x_0^L)$ denote a clean sequence of length L , and let $m = [\text{MASK}]$ denote the mask token. At noise level γ_t , each token is independently corrupted as

$$x_t^i \sim (1 - \gamma_t)\delta_{x_0^i} + \gamma_t\delta_m, \quad i = 1, \dots, L. \quad (1)$$

This absorbing-mask corruption process is widely used in discrete and masked diffusion language models (Austin et al., 2021a; Shi et al., 2024; Sahoo et al., 2024; Nie et al., 2025).

Given a partially masked sequence $\mathbf{x}_t$, the denoising model predicts a vocabulary distribution at each position. Let $h_\theta^i(\mathbf{x}_t) \in \mathbb{R}^{|\mathcal{V}|}$ denote the logits at position i . We define

$$p_\theta^i(v | \mathbf{x}_t) = [\text{softmax}(h_\theta^i(\mathbf{x}_t))]_v, \quad v \in \mathcal{V}. \quad (2)$$

A common training objective is masked cross-entropy over corrupted positions, optionally weighted by the noise schedule:

$$\mathcal{L} = \mathbb{E} \left[w(t) \sum_i \mathbf{1}\{x_t^i = m\} (-\log p_\theta^i(x_0^i | \mathbf{x}_t)) \right], \quad (3)$$

where the expectation is over $(\mathbf{x}_0, t, \mathbf{x}_t)$, the sum ranges over positions $i = 1, \dots, L$, and $w(t)$ is a schedule-dependent weight. Prior work derives this objective from the diffusion variational objective; equivalently, it serves as an upper bound on the negative log-likelihood of the induced model distribution (Shi et al., 2024; Sahoo et al., 2024; Ou et al., 2025; Nie et al., 2025).

3.2 Iterative Decoding

Generation starts from masked output positions. At decoding step t , let $\mathbf{x}_t$ be the current sequence and

$\mathcal{M}_t$ the set of remaining masked token positions. For each $i \in \mathcal{M}_t$, let $p_t^i = p_\theta^i(\cdot | \mathbf{x}_t)$ be the model’s predictive distribution, with argmax prediction y_t^i and confidence c_t^i . A confidence-based rule commits token positions whose confidence exceeds a threshold τ :

$$\mathcal{U}_t = \{i \in \mathcal{M}_t : c_t^i \geq \tau\}. \quad (4)$$

Positions in $\mathcal{U}_t$ are filled with y_t^i , while all other masked token positions remain masked. Committed tokens are kept fixed.

3.3 Semi-Autoregressive Sampling

In semi-autoregressive sampling, output token positions are partitioned into contiguous blocks $\mathcal{B}_1, \dots, \mathcal{B}_M$, where each block $\mathcal{B}_b$ is a set of token positions. The blocks are processed from left to right. At decoding step t , token commitment is restricted to the current active block:

$$\mathcal{U}_t \subseteq \mathcal{M}_t \cap \mathcal{B}_{b(t)}, \quad (5)$$

where $b(t)$ denotes the active block index. We also write $\mathcal{B}_{\leq b(t)} = \bigcup_{r=1}^{b(t)} \mathcal{B}_r$ for the positions in completed and active blocks, and $\mathcal{B}_{b(t)+1}$ for the immediately next block when $b(t) < M$.

In our full-attention MDLM setting, masked positions beyond the active block are still forwarded and receive predictive distributions. Conventional semi-autoregressive sampling does not retain these future-block predictions: after each step, future positions are reset to the mask token m until their block becomes active.

4 Method

4.1 Design Principle: Stability Across Future Views

The key idea of FUSA is to test active-block predictions under two views of the next block. FUSA alternates this future view between ordinary mask tokens and soft future tokens built from the model’s own next-block distributions, and commits active-block tokens whose predictions remain stable across the two views.

Architectural prerequisites. FUSA requires that (i) future masked positions remain in the same forward pass, (ii) the active block can attend to these positions, and (iii) the resulting future distributions can be re-injected as temporary soft embeddings. These conditions hold for full-attention MDLMs. In strict block-causal architectures, the same signal

is unavailable without additional forward computation or architectural modification.

4.2 Alternating Mask-future and Soft-future Steps

Each denoising step performs one model forward pass and unmaskes selected token positions in the active block. Future positions are not committed during this process; instead, only the immediately next block $\mathcal{B}_{b(t)+1}$ is presented to the model either as ordinary mask embeddings or as soft future tokens when $b(t) < M$. Later blocks remain ordinary mask tokens. Masked-future and soft-future steps use the same decoding rule for active-block unmasking and differ only in how this next block is represented, as described below:

Mask-future step. The next block is represented using mask embeddings, and the resulting next-block logits are immediately converted into soft future embeddings, which are cached for the following soft-future step. Here $\mathbf{x}_t$ is the hard decoding state, containing committed tokens and masks. For each forward pass, FUSA forms a temporary vector input $\tilde{\mathbf{x}}_t$ from $\mathbf{x}_t$. Note that this input view is not committed to the state. In the masked-future step, $\tilde{\mathbf{x}}_t$ is defined as:

$$\tilde{\mathbf{x}}_t^i = \begin{cases} \mathbf{E}_{x_t^i}, & \text{if } i \in \mathcal{B}_{\leq b(t)}, \\ \mathbf{E}_m, & \text{otherwise,} \end{cases} \quad (6)$$

where $\mathbf{E}_{x_t^i}$ and $\mathbf{E}_m$ denote the embedding of the token x_t^i and the [MASK] token, respectively.

Soft-future step. The next block is represented using soft future tokens, while all future positions remain uncommitted. Formally,

$$\tilde{\mathbf{x}}_t^i = \begin{cases} \mathbf{E}_{x_t^i}, & \text{if } i \in \mathcal{B}_{\leq b(t)}, \\ \mathbf{g}_t^i, & \text{if } b(t) < M \text{ and } i \in \mathcal{B}_{b(t)+1}, \\ \mathbf{E}_m, & \text{otherwise,} \end{cases} \quad (7)$$

where $\mathbf{g}_t^i$ denotes the soft token embedding at next-block position i at the t -th step, described in Section 4.3.

Unmasking Rule for Commitment. For both mask-future and soft-future steps, FUSA performs active-block unmasking by reading the predictive distributions at masked active-block positions and computing their argmax predictions $y_t^i = \arg \max_v p_t^i(v)$ and confidences $c_t^i = p_t^i(y_t^i)$, following the commitment rule.

To incorporate future-conditioned stability signals, FUSA extends the original commitment set $\mathcal{U}_t$ in Eq. 4 by introducing an additional stability-based unmasking set $\mathcal{U}_{JS,t}$. The resulting enlarged commitment set is defined as:

$$\tilde{\mathcal{U}}_t = \mathcal{U}_t \cup \mathcal{U}_{JS,t}. \quad (8)$$

where $\mathcal{U}_{JS,t}$ denotes the set of token positions selected based on future-view stability, which is described in Section 4.4.

4.3 Representing Soft Future Tokens

Soft future tokens provide next-block belief to the active block without committing future positions. They avoid two forms of information loss: mask tokens hide predictive signals, while argmax tokens discard the remaining probability mass.

At decoding step t , when $b(t) < M$, FUSA reduces tail noise by applying top- p filtering ($p = 0.85$) to each next-block distribution p_t^j , where $j \in \mathcal{B}_{b(t)+1}$, and renormalizes the retained probabilities. Let $\text{Top}_p(j; t) \subseteq \mathcal{V}$ denote the set of tokens retained by top- p filtering, and let $\tilde{p}_t^j$ denote the renormalized distribution: $\tilde{p}_t^j(v) = p_t^j(v) / \sum_{u \in \text{Top}_p(j; t)} p_t^j(u)$. FUSA then constructs the soft future representation as:

$$\begin{aligned} \mathbf{z}_{t+1}^j &= \sum_{v \in \text{Top}_p(j; t)} \tilde{p}_t^j(v) \mathbf{E}_v, \\ \mathbf{g}_{t+1}^j &= \alpha \mathbf{E}_m + (1 - \alpha) \mathbf{z}_{t+1}^j. \end{aligned} \quad (9)$$

Mixing with $\mathbf{E}_m$ keeps $\mathbf{g}_t^j$ close to the model's original masked-denoising input distribution, thereby reducing distributional drift from the pretrained inference regime. The coefficient α controls the strength of the soft future signal: $\alpha = 1$ recovers the mask-future view, while smaller values expose more of the soft future representation $\mathbf{z}_t^j$.

4.4 JS-based Unmasking Criterion

The commitment decision compares two predictive distributions for the same masked position across consecutive steps. Since neither distribution serves as a fixed reference, FUSA employs the Jensen–Shannon (JS) divergence as a symmetric and bounded stability score. Letting $M = \frac{1}{2}(P + Q)$, it is defined as:

$$D_{JS}(P, Q) = \frac{1}{2} D_{KL}(P \| M) + \frac{1}{2} D_{KL}(Q \| M), \quad (10)$$

where the divergences are computed using natural logarithms.

FUSA first commits y_t^i at any active-block position with $c_t^i \geq \tau_{\text{high}}$. Following recent confidence-gated diffusion decoders, we use $\tau_{\text{high}} = 0.95$ as a high-confidence fast path (Cao et al., 2026; Liang et al., 2026). The remaining active-block positions are then routed to the JS-stability gate.

At decoding step t , FUSA computes $D_{\text{JS}}(p_{t-1}^i, p_t^i)$, where p_{t-1}^i and p_t^i denote the predictive distributions at position i in two consecutive steps. A remaining masked position is additionally unmasked when

$$\mathcal{U}_{\text{JS},t} = \{i \in \mathcal{M}_t \cap \mathcal{B}_{b(t)} : c_t^i \geq \tau_{\text{floor}} \wedge D_{\text{JS}}(p_{t-1}^i, p_t^i) \leq \epsilon\}. \quad (11)$$

If neither rule selects any active-block position, FUSA commits the argmax prediction at the highest-confidence masked position in the active block to guarantee progress.

5 Experiments

5.1 Models, Benchmarks, and Setup

We evaluate FUSA on LLaDA-8B-Instruct (Nie et al., 2025) and LLaDA-1.5 (Zhu et al., 2026) across GSM8K (Cobbe et al., 2021), MATH500 (Hendrycks et al., 2021; Lightman et al., 2024), HumanEval (Chen et al., 2021), and MBPP (Austin et al., 2021b). All decoding methods use the same semi-autoregressive setting, prompts, graders, and generation configuration unless otherwise stated. All experiments are conducted on a single NVIDIA RTX PRO 6000 GPU. Detailed hyperparameter choices, baseline settings, and benchmark prompts are provided in Appendix G.1, Appendix G.2, and Appendix G.3.

5.2 Baselines

All baselines use the same blockwise semi-autoregressive setting and differ in active-block token selection or refinement strategy. We compare against Uniform, Entropy, Top-1, Top-2, Confidence, EB-Sampler (Ben-Hamu et al., 2025), KLASS (Kim et al., 2025), and LRD (Zhu et al., 2025). Confidence denotes threshold decoding that commits active-block tokens with $c_i \geq 0.9$. LRD denotes a soft latent refinement baseline that adds refinement-only forwards before block decoding. EB-Sampler, KLASS, and LRD are reported under their benchmark-specific or recommended settings.

5.3 Soft-Future Stability Diagnostics

Figure 2 suggests that soft future perturbations provide a useful stability signal. On 500 arithmetic examples and 11,745 active-block positions, wrong Top-1 hypotheses show larger JS shifts and tend to lose probability mass under the soft future probe, whereas correct Top-1 hypotheses are more stable and are often reinforced. This supports using soft-future consistency as an active-token selection signal; Appendix B.1 gives the full diagnostic setup and metrics.

Cross-task trajectory diagnostic. We analyze 400 confidence-decoding trajectories, with 100 samples each from GSM8K, MATH500, HumanEval, and MBPP. Because these tasks may admit multiple valid sequences, we label a current Top-1 hypothesis as *Retained* if it matches the token eventually committed at that position and as *Revised* otherwise; these labels measure trajectory stability rather than correctness. The soft-future view is used only as a diagnostic and does not alter decoding. Table 3 reports the gap between the JS divergences of Revised and Retained hypotheses. The gap is positive on all four benchmarks, with every 95% confidence interval excluding zero. This complementary analysis provides trajectory-level evidence that soft-future sensitivity is associated with whether a hypothesis is eventually retained or revised across reasoning and code-generation tasks. Token-category and generation-stage breakdowns are provided in Appendix B.2.

5.4 Main Results

Tables 1 and 2 report the main results. On LLaDA-8B-Instruct, FUSA obtains the best score on all four benchmarks while remaining in the fastest group of methods. It matches or exceeds the strongest HumanEval score, improves MBPP by 4.67 points over EB-Sampler and 0.78 points over LRD, improves GSM8K by 1.29 points over confidence decoding, and gives the best MATH500 accuracy. At the same time, it uses substantially fewer denoising steps than EB-Sampler, KLASS, and LRD on every benchmark, showing that the soft-future stability test improves accuracy without reverting to slow conservative decoding. On this model, compared with simple confidence decoding, FUSA spends a modest number of additional steps but converts those steps into higher task scores on all four benchmarks.

On LLaDA-1.5, we use the same hyperparameter

Table 1: Main results on LLaDA-8B-Instruct. Each benchmark reports task score and average denoising steps; each step corresponds to one model forward call. Numbers in parentheses denote the number of in-context examples. Uniform averages over three seeds; all other methods are deterministic.

Method	GSM8K(0)		MATH500(0)		HumanEval(0)		MBPP(3)	
	Acc	Steps ↓	Acc	Steps ↓	Pass@1	Steps ↓	Pass@1	Steps ↓
Uniform	66.67	256.00	23.73	256.00	15.65	256.00	26.20	256.00
Entropy	74.60	256.00	31.20	256.00	35.98	256.00	33.46	256.00
Top-1	73.92	256.00	33.00	256.00	35.98	256.00	37.74	256.00
Top-2	73.54	128.00	32.00	128.00	25.61	128.00	29.57	128.00
Confidence	74.60	70.66	33.60	97.55	34.76	76.43	35.80	58.56
EB-Sampler	74.53	130.73	33.60	160.38	37.20	139.92	38.13	104.27
KLASS	72.86	96.46	33.60	127.84	37.20	125.12	37.35	86.51
LRD	73.54	264.14	31.00	264.10	32.93	264.13	42.02	264.11
FUSA	75.89	81.18	34.40	102.69	37.20	85.37	42.80	62.90

Table 2: Main results on LLaDA-1.5. Each benchmark reports task score and average denoising steps; each step corresponds to one model forward call. Numbers in parentheses denote the number of in-context examples. Uniform averages over three seeds; all other methods are deterministic.

Method	GSM8K(0)		MATH500(0)		HumanEval(0)		MBPP(3)	
	Acc	Steps ↓	Acc	Steps ↓	Pass@1	Steps ↓	Pass@1	Steps ↓
Uniform	68.41	256.00	24.93	256.00	14.23	256.00	24.12	256.00
Entropy	76.72	256.00	32.00	256.00	35.98	256.00	37.74	256.00
Top-1	76.42	256.00	33.40	256.00	38.41	256.00	42.41	256.00
Top-2	75.74	128.00	32.60	128.00	29.88	128.00	21.40	128.00
Confidence	76.35	68.73	31.60	94.57	35.98	75.24	41.63	68.22
EB-Sampler	77.03	125.65	32.20	152.71	37.80	131.38	42.80	114.58
KLASS	74.07	95.24	28.40	127.90	37.20	125.98	40.08	104.67
LRD	76.27	264.16	35.00	264.15	37.80	264.11	42.02	264.14
FUSA	76.88	71.76	34.80	100.52	35.98	84.12	42.80	72.30

Table 3: Cross-task trajectory diagnostic (100 samples per benchmark). The gap is REVISED JS minus RETAINED JS; brackets give pointwise 95% paired sample-bootstrap CIs. Gaps use unrounded values.

Benchmark	RETAINED	REVISED	JS gap [95% CI]
GSM8K	.1734	.2540	+.0806 [.0739, .0874]
MATH500	.1812	.2852	+.1039 [.0925, .1152]
HumanEval	.1770	.3133	+.1363 [.1265, .1462]
MBPP	.1617	.2483	+.0866 [.0703, .1025]

setting transferred from the LLaDA experiments, without a separate validation-set search. Even under this direct transfer, FUSA remains highly competitive: it ties the best MBPP pass@1, is within 0.2 points of the best MATH500 accuracy, and is close to the best GSM8K result while using far fewer steps than EB-Sampler and LRD. LRD is competitive on MBPP and LLaDA-1.5 MATH500, but its soft latent refinement phase increases average steps to about 264. Compared with KLASS, FUSA is faster on all four benchmarks and improves accu-

racy on MBPP, GSM8K, and MATH500. Relative to confidence decoding, FUSA improves MBPP, GSM8K, and MATH500, while tying HumanEval pass@1 with more denoising steps. Although Top-1 is strongest on HumanEval for LLaDA-1.5, it requires the full 256 denoising steps, whereas FUSA attains competitive pass@1 with roughly one third of the steps. Additional transfer results on CoDA-1.7B-Instruct (Chen et al., 2025a) are provided in Appendix F.

5.5 Computational Overhead and KV-Cache Compatibility

Table 4 reports FUSA’s measured wall-clock speedup and score gain over Top-1. Computationally, FUSA adds no extra model forward calls; its extra storage is dominated by $O(BV)$ cached active-block distributions and $O(Fd)$ soft-future embeddings, plus transient $O(FV)$ buffers during nucleus construction. Appendix A provides full latency, throughput, and memory results; on GSM8K,

Table 4: Wall-clock summary on LLaDA-8B-Instruct. Parentheses report the FUSA score gain over Top-1, and speedup is computed from the displayed latency values relative to Top-1 and rounded to two decimal places. Full runtime diagnostics are in Table 8.

Benchmark	FUSA Score	Speedup vs. Top-1
HumanEval	37.20 (+1.22)	2.82×
MBPP	42.80 (+5.06)	3.94×
GSM8K	75.89 (+1.97)	2.79×
MATH500	34.40 (+1.40)	2.10×

Table 5: HumanEval controls for the soft-future block averaged over three seeds.

Variant	Pass@1	Steps
FUSA	37.20 ± 0.00	85.37 ± 0.00
Shuffled soft	34.76 ± 0.61	84.31 ± 0.91
Random soft	33.94 ± 2.46	84.36 ± 0.46
Argmax tokens	25.00 ± 0.00	69.09 ± 0.00

FUSA increases peak memory by only 252 MiB over Top-1.

KV-cache compatibility. We adapt Fast-dLLM’s DualCache (Wu et al., 2026b) to FUSA by re-computing the active and immediately following blocks while reusing cached KV states elsewhere. On GSM8K with 5-shot prompting, this increases throughput from 41.06 to 96.35 TPS (2.35×), with accuracy changing from 78.39% to 77.56%. Full results are provided in Appendix A.2.

5.6 Future-Block Representation Ablation

Table 5 reports a focused HumanEval control study for the soft-future block representation. All variants use the same LLaDA HumanEval setup; they differ only in what the model sees in the next block while decoding the current block.

FUSA uses model-predicted next-block soft distributions at their original positions. The controls disrupt position alignment, replace the future view with random soft embeddings, or substitute argmax future tokens; Appendix C.1 gives their detailed definitions. FUSA gives the best pass@1 among these controls with nearly unchanged step counts, whereas the argmax-token variant reduces steps but sharply lowers pass@1 to 25.00. Overall, these controls justify using soft future tokens: the useful signal comes from a position-aligned, uncertainty-preserving future view, rather than from replacing it with unrelated soft noise or substituting argmax tokens for the soft future view.

Additional controls in Appendices C.2 and C.3 rule out temporal stability and future-view alter-

Table 6: WritingBench results on the fixed 100-prompt subset using LLaDA-8B-Instruct with a maximum generation length of 1,024 tokens. Speedup is measured relative to Top-1.

Method	Critic ↑	Avg. tokens	Steps ↓	Latency ↓	Speedup
Top-1	4.678	973.42	1024.00	50.70s	1.00×
FUSA	4.718	973.33	651.51	31.33s	1.62×

nation alone as explanations for the gains, while Appendix D.1 characterizes sensitivity to the soft-mask mixing coefficient.

5.7 Long-Form Instruction-Following Generation

We evaluate FUSA on a fixed, domain-balanced subset of 100 English WritingBench prompts (Wu et al., 2025) using LLaDA-8B-Instruct with a maximum generation length of 1,024 tokens. Top-1 and FUSA use identical prompts and generation settings, and writing quality is evaluated using the benchmark’s fine-tuned critic model. As shown in Table 6, FUSA achieves a critic score comparable to that of Top-1 (4.718 vs. 4.678) at nearly identical output lengths, while reducing average forward evaluations from 1,024 to 651.51 and latency from 50.70 s to 31.33 s (1.62× speedup). Subset construction details are provided in Appendix G.4.

6 Conclusion

We introduced FUSA, a training-free decoding method for full-attention MDLMs. FUSA addresses a simple inefficiency in standard block-wise decoding: future-block logits are already computed while denoising the active block, but they are normally discarded. FUSA reuses these logits to form temporary soft-future views and unmask active-block tokens whose predictions remain consistent across alternating mask-future and soft-future states.

This soft-future consistency signal helps FUSA select stable active-block tokens with fewer denoising steps. The empirical results show that FUSA delivers consistent benefits across diverse models and generation settings, while the diagnostics show that soft-future stability provides information beyond confidence and temporal stability alone.

More broadly, our results suggest that full-attention MDLMs are useful not only because they support parallel refinement, but also because they can test present commitments against plausible future evidence already available in the model’s own

predictions. Future work may explore richer future representations and adaptive schedules that decide when and how much future information to expose.

7 Limitations

FUSA is specifically an inference-time decoder for full-attention MDLMs satisfying the architectural prerequisites described in Section 4.1. It therefore does not directly apply to strict block-causal architectures that remove future blocks or hide them from active-block attention. We also do not directly compare FUSA with separately trained block-diffusion systems. Our LLaDA-8B-Instruct long-form evaluation is limited to a fixed subset of 100 English WritingBench prompts and relies on the benchmark’s fine-tuned critic model rather than direct human evaluation.

FUSA also introduces several decoding hyperparameters. Although the selected settings transferred unchanged to LLaDA-1.5 and, for code generation, to CoDA-1.7B-Instruct, substantially different model or task settings may still require a new hyperparameter search using validation data. In addition, wider future perturbation windows are not always beneficial, suggesting that the usefulness of soft future evidence depends on how local and reliable the model’s next-block belief is.

Acknowledgments

This work was partly supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. RS-2025-25441313, Professional AI Talent Development Program for Multimodal AI Agents, Contribution: 50%) and Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. RS-2020-II201336, Artificial Intelligence Graduate School Program (UNIST)).

References

Marianne Arriola, Aaron Gokaslan, Justin T Chiu, Zhihan Yang, Zhixuan Qi, Jiaqi Han, Subham Sekhar Sahoo, and Volodymyr Kuleshov. 2025. [Block Diffusion: Interpolating between autoregressive and diffusion language models](#). In *The Thirteenth International Conference on Learning Representations*.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. 2021a. [Structured denoising diffusion models in discrete state-spaces](#).

In *Advances in Neural Information Processing Systems*.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. 2021b. [Program synthesis with large language models](#). *Preprint*, arXiv:2108.07732.

Heli Ben-Hamu, Itai Gat, Daniel Severo, Niklas Nolte, and Brian Karrer. 2025. [Accelerated sampling from masked diffusion models via entropy bounded unmasking](#). In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Tiwei Bie, Maosong Cao, Xiang Cao, Bingsen Chen, Fuyuan Chen, Kun Chen, Lun Du, Daozhuo Feng, Haibo Feng, Mingliang Gong, Zhuocheng Gong, Yanmei Gu, Jian Guan, Kaiyuan Guan, Hongliang He, Zenan Huang, Juyong Jiang, Zhonghui Jiang, Zhenzhong Lan, and 31 others. 2026. [LLaDA2.1: Speeding up text diffusion via token editing](#). *Preprint*, arXiv:2602.08676.

Tiwei Bie, Maosong Cao, Kun Chen, Lun Du, Mingliang Gong, Zhuochen Gong, Yanmei Gu, Jiaqi Hu, Zenan Huang, Zhenzhong Lan, Chengxi Li, Chongxuan Li, Jianguo Li, Zehuan Li, Huabin Liu, Lin Liu, Guoshan Lu, Xiaocheng Lu, Yuxin Ma, and 12 others. 2025. [LLaDA2.0: Scaling up diffusion language models to 100b](#). *Preprint*, arXiv:2512.15745.

Mingyu Cao, Alvaro Correia, Christos Louizos, Shiwei Liu, and Lu Yin. 2026. [Search or accelerate: Confidence-switched position beam search for diffusion language models](#). In *Forty-third International Conference on Machine Learning*.

Haolin Chen, Shiyu Wang, Can Qin, Bo Pang, Zuxin Liu, Jieli Qiu, Jianguo Zhang, Yingbo Zhou, Zeyuan Chen, Ran Xu, Shelby Heinecke, Silvio Savarese, Caiming Xiong, Huan Wang, and Weiran Yao. 2025a. [Coda: Coding lm via diffusion adaptation](#). *Preprint*, arXiv:2510.03270.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, and 39 others. 2021. [Evaluating large language models trained on code](#). *Preprint*, arXiv:2107.03374.

Ziyu Chen, Xinbei Jiang, Peng Sun, and Tao Lin. 2025b. [Optimizing decoding paths in masked diffusion models by quantifying uncertainty](#). *Preprint*, arXiv:2512.21336.

Shuang Cheng, Yihan Bian, Dawei Liu, Yuhua Jiang, Yihao Liu, Linfeng Zhang, Qian Yao, Zhongbo Tian, Wenhai Wang, Qipeng Guo, Kai Chen, Biqing Qi, and Bowen Zhou. 2026. [SDAR: A synergistic diffusion-AutoRegression paradigm for scalable sequence generation](#). In *Findings of the Association for Computational Linguistics: ACL 2026*, pages

- 22058–22075, San Diego, California, United States. Association for Computational Linguistics.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. [Training verifiers to solve math word problems](#). *Preprint*, arXiv:2110.14168.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. [BERT: Pre-training of deep bidirectional transformers for language understanding](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- Aaron Gokaslan and Vanya Cohen. 2019. Open-webtext corpus. <http://Skylion007.github.io/OpenWebTextCorpus>.
- Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and Lingpeng Kong. 2023. [DiffuSeq: Sequence to sequence text generation with diffusion models](#). In *International Conference on Learning Representations*.
- Andre He, Sean Welleck, and Daniel Fried. 2026. [Reasoning with latent tokens in diffusion language models](#). *Preprint*, arXiv:2602.03769.
- Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. [Measuring mathematical problem solving with the MATH dataset](#). In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.
- Michael Hersche, Samuel Moor-Smith, Thomas Hofmann, and Abbas Rahimi. 2026. [Soft-masked diffusion language models](#). In *The Fourteenth International Conference on Learning Representations*.
- Yuezhou Hu, Harman Singh, Monishwaran Maheswaran, Haocheng Xi, Coleman Richard Charles Hooper, Jintao Zhang, Aditya Tomar, Michael W. Mahoney, Sewon Min, Mehrdad Farajtabar, Kurt Keutzer, Amir Gholami, and Chenfeng Xu. 2026. [Residual context diffusion language models](#). In *Forty-third International Conference on Machine Learning*.
- Pengcheng Huang, Shuhao Liu, Zhenghao Liu, Yukun Yan, Shuo Wang, Zulong Chen, and Tong Xiao. 2025. [Pc-sampler: Position-aware calibration of decoding bias in masked diffusion models](#). *Preprint*, arXiv:2508.13021v1.
- Seo Hyun Kim, Sunwoo Hong, Hojung Jung, Youngrok Park, and Se-Young Yun. 2025. [KLASS: KL-guided fast inference in masked diffusion models](#). In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.
- Sanghyun Lee, Seungryong Kim, Jongho Park, and Dongmin Park. 2026. [Lookahead unmasking elicits its reliable decoding in diffusion language models](#). In *Forty-third International Conference on Machine Learning*.
- Jinsong Li, Xiaoyi Dong, Yuhang Zang, Yuhang Cao, Jiaqi Wang, and Dahua Lin. 2026. [Beyond fixed: Training-free variable-length denoising for diffusion large language models](#). In *The Fourteenth International Conference on Learning Representations*.
- Xiang Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori B. Hashimoto. 2022. [Diffusion-LM improves controllable text generation](#). In *Advances in Neural Information Processing Systems*, volume 35.
- Yihao Liang, Ze Wang, Hao Chen, Ximeng Sun, Jialian Wu, Xiaodong Yu, Jiang Liu, Emad Barsoum, Zicheng Liu, and Niraj K. Jha. 2026. [CD4LM: Consistency distillation and adaptive decoding for diffusion language models](#). *Preprint*, arXiv:2601.02236.
- Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. [Let’s verify step by step](#). In *The Twelfth International Conference on Learning Representations*.
- Aaron Lou, Chenlin Meng, and Stefano Ermon. 2024. [Discrete diffusion modeling by estimating the ratios of the data distribution](#). In *Forty-first International Conference on Machine Learning*.
- Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. 2025. [Large language diffusion models](#). In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.
- Jingyang Ou, Shen Nie, Kaiwen Xue, Fengqi Zhu, Jiacheng Sun, Zhenguo Li, and Chongxuan Li. 2025. [Your absorbing discrete diffusion secretly models the conditional distributions of clean data](#). In *The Thirteenth International Conference on Learning Representations*.
- Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. 2021. [MAUVE: Measuring the gap between neural text and human text using divergence frontiers](#). In *Advances in Neural Information Processing Systems*, volume 34, pages 4816–4828.
- Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.
- Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Mariano Marroquin, Justin T Chiu, Alexander M Rush, and Volodymyr Kuleshov. 2024. [Simple and effective masked diffusion language models](#). In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Jiixin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis Titsias. 2024. [Simplified and generalized masked diffusion for discrete data](#). In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Guanghan Wang, Yair Schiff, Subham Sahoo, and Volodymyr Kuleshov. 2025. [Remasking discrete diffusion models with inference-time scaling](#). In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Chengyue Wu, Hao Zhang, Shuchen Xue, Shizhe Diao, Yonggan Fu, Zhijian Liu, Pavlo Molchanov, Ping Luo, Song Han, and Enze Xie. 2026a. [Fast-dLLM v2: Efficient block-diffusion LLM](#). In *The Fourteenth International Conference on Learning Representations*.

Chengyue Wu, Hao Zhang, Shuchen Xue, Zhijian Liu, Shizhe Diao, Ligeng Zhu, Ping Luo, Song Han, and Enze Xie. 2026b. [Fast-dLLM: Training-free acceleration of diffusion LLM by enabling KV cache and parallel decoding](#). In *The Fourteenth International Conference on Learning Representations*.

Yuning Wu, Jiahao Mei, Ming Yan, Chenliang Li, Shaopeng Lai, Yuran Ren, Zijia Wang, Ji Zhang, Mengyue Wu, Qin Jin, and Fei Huang. 2025. [Writingbench: A comprehensive benchmark for generative writing](#). In *Advances in Neural Information Processing Systems*, volume 38, Main Conference. Curran Associates, Inc.

Kejing Xia, Mingzhe Li, Lixuan Wei, Zhenbang Du, Xiangchi Yuan, Dachuan Shi, Qirui Jin, and Wenke Lee. 2026. [Metastate: Persistent working memory enhances reasoning in discrete diffusion language models](#). In *Third Conference on Language Modeling*.

Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. 2025. [Dream 7B: Diffusion large language models](#). *Preprint*, arXiv:2508.15487.

Lin hao Zhong, Linyu Wu, Bozhen Fang, Tianjian Feng, Chenchen Jing, Wen Wang, Jiaheng Zhang, Hao Chen, and Chunhua Shen. 2026. [Beyond hard masks: Progressive token evolution for diffusion language models](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6936–6959, San Diego, California, United States. Association for Computational Linguistics.

Fengqi Zhu, Rongzhen Wang, Shen Nie, Xiaolu Zhang, Chunwei Wu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. 2026. [LLaDA 1.5: Variance-reduced preference optimization for large language diffusion models](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11425–11460, San Diego, California, United States. Association for Computational Linguistics.

Qinglin Zhu, Yizhen Yao, Runcong Zhao, Yanzheng Xiang, Amrutha Saseendran, Chen Jin, Philip Teare,

Bin Liang, Yulan He, and Lin Gui. 2025. [Latent refinement decoding: Enhancing diffusion-based language models by refining belief states](#). *Preprint*, arXiv:2510.11052.

A Runtime Diagnostics

A.1 Uncached Runtime and Memory

Let B be the active block size, F the future probe block size, V the vocabulary size, and d the hidden dimension. The persistent extra memory used by FUSA comes from active-block reference distributions, which cost $O(BV)$, and the short-lived future embedding cache, which costs $O(Fd)$. Building the soft future embeddings also uses temporary next-block probability buffers for nucleus filtering and renormalization, which cost $O(FV)$ and are released once the soft embeddings are constructed. This computation uses the same forward pass as the current denoising step and adds no extra model forward call.

Consistent with this analysis, Table 7 shows only a modest peak-memory increase in the GSM8K profiling run.

Method	Peak Memory	Δ Peak Memory
Top-1	16,369 MiB	0
FUSA	16,621 MiB	+252 MiB

Table 7: Peak-memory profiling for Top-1 and FUSA on GSM8K. The additional memory comes from cached distributions and soft future embedding construction.

Table 8 reports per-sample wall-clock latency and throughput for all main decoding methods under the same hardware and inference backend. Confidence decoding is faster on some benchmarks, but it gives lower task scores; compared with Top-1, EB-Sampler, and KLASS, FUSA provides a stronger overall accuracy–latency trade-off.

A.2 KV-Cached Inference

To evaluate whether FUSA is compatible with KV reuse for full-attention MDLMs, we adapt Fast-dLLM’s DualCache (Wu et al., 2026b). The original DualCache recomputes the active block while reusing cached KV states outside it. Because FUSA additionally modifies the immediately following block through its soft-future representation, our adaptation recomputes both the active and next blocks and reuses cached KV states elsewhere. We evaluate both configurations on all 1,319 GSM8K examples with 5-shot prompting using identical

Benchmark	Method	Score (%) $\uparrow$	Latency/sample $\downarrow$	TPS $\uparrow$	Latency / FUSA
HumanEval	Top-1	35.98	7.90s	32.41	2.82 $\times$
	Confidence	34.76	2.36s	108.25	0.84 $\times$
	EB-Sampler	37.20	4.66s	54.99	1.66 $\times$
	KLASS	37.20	5.01s	51.10	1.79 $\times$
	LRD	32.93	11.95s	21.43	4.27 $\times$
	FUSA	37.20	2.80s	91.37	1.00 $\times$
MBPP	Top-1	37.74	15.62s	16.39	3.94 $\times$
	Confidence	35.80	3.57s	71.65	0.90 $\times$
	EB-Sampler	38.13	6.19s	41.36	1.56 $\times$
	KLASS	37.35	7.07s	36.19	1.79 $\times$
	LRD	42.02	23.87s	10.73	6.03 $\times$
	FUSA	42.80	3.96s	64.59	1.00 $\times$
GSM8K	Top-1	73.92	7.14s	35.87	2.79 $\times$
	Confidence	74.60	2.47s	103.79	0.96 $\times$
	EB-Sampler	74.53	3.62s	70.66	1.41 $\times$
	KLASS	72.86	3.27s	78.30	1.28 $\times$
	LRD	73.54	10.22s	25.06	3.99 $\times$
	FUSA	75.89	2.56s	99.87	1.00 $\times$
MATH500	Top-1	33.00	7.07s	36.23	2.10 $\times$
	Confidence	33.60	2.75s	93.18	0.82 $\times$
	EB-Sampler	33.60	4.64s	55.20	1.38 $\times$
	KLASS	33.60	4.58s	55.95	1.36 $\times$
	LRD	31.00	10.73s	23.87	3.18 $\times$
	FUSA	34.40	3.37s	75.94	1.00 $\times$

Table 8: Runtime diagnostics across decoding methods. Latency is measured in seconds per example, and TPS denotes generated tokens per second. Latency ratios are computed from the displayed latency values and rounded to two decimal places.

FUSA hyperparameters. Despite the small increase in denoising steps, KV reuse raises throughput to 2.35 $\times$ that of the uncached variant, as shown in Table 9.

Table 9: KV-cached inference for FUSA on GSM8K with 5-shot prompting. We use an adapted DualCache that recomputes the active and next blocks. Speedup is measured relative to FUSA without caching.

Method	Acc. $\uparrow$	Steps $\downarrow$	TPS $\uparrow$	Speedup $\uparrow$
FUSA	78.39	86.02	41.06	1.00 $\times$
+DualCache	77.56	90.16	96.35	2.35 $\times$

B Future-View Stability Diagnostic

B.1 Controlled Arithmetic Probe

We use a controlled arithmetic probe to test whether future-block predictions contain useful information for distinguishing correct and incorrect active-block hypotheses. This experiment is not intended as a benchmark of arithmetic accuracy; rather, it is a diagnostic intervention that measures how the active block reacts when the model’s own future belief is softly injected. Unlike the actual FUSA decoder, this diagnostic deliberately performs a same-state counterfactual re-run; it validates the signal but is not the runtime decoding schedule.

Setup. Each example consists of an arithmetic prompt and a canonical output segmented into three

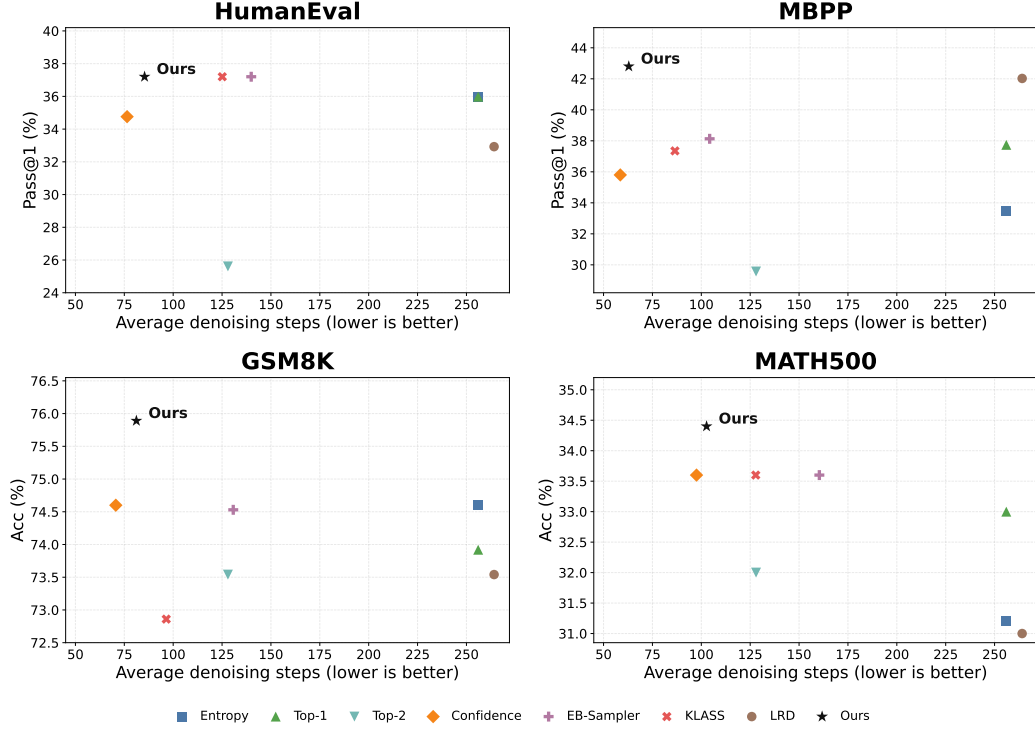


Figure 4: Score–step trade-off across decoding methods on the four main benchmarks. Each panel plots task score against the average number of denoising steps, where lower step counts indicate fewer model forward calls. The upper-left region is preferable because it combines higher task score with fewer denoising steps. Across benchmarks, FUSA lies in the high-score, low-step region compared with conservative baselines such as EB-Sampler and KLASS.

equal-size blocks:

$$\mathcal{B}_1 = y_{1:8}, \quad \mathcal{B}_2 = y_{9:16}, \quad \mathcal{B}_3 = y_{17:24}. \quad (12)$$

The block size is fixed to 8 tokens and the total generation window is 24 tokens. We analyze only the active block $\mathcal{B}_1$. The soft future probe is applied to $\mathcal{B}_2$, while $\mathcal{B}_3$ remains masked. We use LLaDA as the base masked diffusion language model.

For each diagnostic state, we first run the model with $\mathcal{B}_2$ and $\mathcal{B}_3$ fully masked. Suppose that the t -th step corresponds to a masked-future step, and the $(t+1)$ -th step corresponds to a soft-future step. **Note that with a slight abuse of notation, we denote the masked-future and soft-future evaluations by steps t and $t+1$, respectively, although both evaluations are performed on the same underlying decoding state.**

For an active-block position $i \in \mathcal{B}_1$, this gives the baseline distribution

$$p_t^i(\cdot) = p_\theta^i(\cdot \mid x, \mathcal{B}_1, \mathcal{B}_2 = [\text{MASK}]^{|\mathcal{B}_2|}, \mathcal{B}_3 = [\text{MASK}]^{|\mathcal{B}_3|}). \quad (13)$$

The top-1 token under this baseline distribution is used to define the hypothesis:

$$\hat{y}_i = \arg \max_v p_t^i(v). \quad (14)$$

We label the position as *Correct Top-1* if $\hat{y}_i$ matches the canonical token, and *Wrong Top-1* otherwise.

Future-view probe. The same baseline forward pass also produces logits for the future block $\mathcal{B}_2$. Instead of inserting argmax future tokens, we convert these logits into soft token embeddings. For each future position $j \in \mathcal{B}_2$, we form

$$\mathbf{z}_{t+1}^j = \sum_{v \in \mathcal{V}_j^p} \tilde{p}_t^j(v) \mathbf{E}_v, \quad (15)$$

where $\mathbf{E}_v$ is the token embedding, $\mathcal{V}_j^p$ is the top- p token set with $p = 0.85$, and $\tilde{p}_t^j$ is the renormalized probability distribution over this top- p set. This diagnostic-only notation corresponds to the soft future distribution and embedding denoted by $\tilde{p}_t^j$ and $\mathbf{g}_j$ in the main decoder description. In this diagnostic experiment, we use the pure soft setting, i.e., no mask interpolation. We then re-run the model with $\mathcal{B}_2$ replaced by these soft embeddings and obtain

$$p_{t+1}^i(\cdot) = p_\theta^i(\cdot \mid x, \mathcal{B}_1, \mathcal{B}_2 = (\mathbf{z}_{t+1}^j)_{j \in \mathcal{B}_2}, \mathcal{B}_3 = [\text{MASK}]^{|\mathcal{B}_3|}). \quad (16)$$

Metrics. We measure two quantities for active-block positions. First, we compute the Jensen–Shannon divergence between the baseline and soft-probed distributions:

$$\begin{aligned} \text{JS}_i &= \frac{1}{2} D_{\text{KL}}(p_t^i \| m^i) + \frac{1}{2} D_{\text{KL}}(p_{t+1}^i \| m^i), \\ m^i &= \frac{1}{2} (p_t^i + p_{t+1}^i). \end{aligned} \quad (17)$$

This measures how much the active-block distribution shifts under soft future evidence.

Second, we measure the probability change of the initial top-1 token:

$$\Delta p_{t+1}^{i, \text{top1}} = p_{t+1}^i(\hat{y}_i) - p_t^i(\hat{y}_i). \quad (18)$$

A positive value indicates that the initial top-1 hypothesis is reinforced by the soft future block, while a negative value indicates that the hypothesis loses probability mass after conditioning on the future soft evidence.

Result. We evaluate 500 arithmetic examples and analyze 11,745 active-block positions. Under the fully masked baseline, 11,013 positions are labeled as Correct Top-1 and 732 as Wrong Top-1. The aggregate pattern is summarized in Figure 2. Wrong Top-1 hypotheses exhibit substantially larger active-block JS divergence than Correct Top-1 hypotheses. Moreover, the direction of the probability shift is contrastive: Correct Top-1 hypotheses gain probability mass, whereas Wrong Top-1 hypotheses lose probability mass under the same soft future probe. This suggests that the model’s own next-block belief provides a useful stability signal for active-block decoding.

B.2 Broader Soft-Future Stability Diagnostics

We analyze the same 400 trajectories as in Section 5.3, using a confidence threshold of 0.9 for trajectory collection. Within each sample and stratum, we average JS divergence separately over REVISED and RETAINED hypotheses and compute their difference. Only samples containing at least one hypothesis of both types in the corresponding stratum are included. We report pointwise 95% confidence intervals from 10,000 paired sample-level bootstrap resamples (seed 20260711), with samples weighted equally.

Token categories. A decoded token surface is classified as *Numeric* if it contains a digit, as *Alphabetic* if it contains a letter but no digit, and as

Structural otherwise, including operators, punctuation, delimiters, whitespace, and newlines.

Generation stages. Because generation proceeds sequentially in 64-token blocks, we define *Early*, *Middle*, and *Late* as the first three blocks (positions 0–63, 64–127, and 128–191). The final block is excluded because it has no subsequent block for constructing the soft-future view.

Tables 10 and 11 show positive JS gaps in every token category and generation stage, with all pointwise 95% confidence intervals excluding zero. Thus, the separation between retained and subsequently revised hypotheses is not concentrated in a particular token category or generation stage.

C Additional Controls

C.1 Future-Block Control Definitions

Table 5 isolates the temporary next-block representation while keeping the HumanEval decoder configuration fixed. *FUSA* uses the model-predicted soft future token g_j at its original future position j . *Shuffled soft* first constructs the same model-predicted soft future tokens as FUSA, but applies a random permutation across positions in the next block before the forward pass; this preserves the multiset of soft tokens while breaking position alignment. *Random soft* replaces each future position with the average of 16 randomly sampled vocabulary embeddings, producing soft embedding noise without model-predicted next-block semantics. *Argmax tokens* replaces each future position by the argmax token from the model’s next-block distribution, turning the temporary future view into a discrete argmax-token view.

C.2 Temporal-Only Control

To isolate temporal stability from soft-future information, Temporal-only removes the soft-future step and compares consecutive masked-state distributions using the same thresholds as FUSA.

Table 12: Temporal-only control on GSM8K and MBPP. Scores are percentages, and steps are average denoising steps.

Method	GSM8K		MBPP	
	Acc.	Steps	Pass@1	Steps
Temporal-only	74.53	78.83	37.74	63.86
FUSA	75.89	81.18	42.80	62.90

FUSA improves over Temporal-only on both benchmarks, with a particularly large 5.06-point

Table 10: Token-category JS gaps with pointwise 95% paired sample-bootstrap confidence intervals.

Benchmark	Numeric gap [95% CI]	Alphabetic gap [95% CI]	Structural gap [95% CI]
GSM8K	+0.0796 (n=100) [.0665, .0927]	+0.0510 (n=100) [.0417, .0603]	+0.0948 (n=100) [.0873, .1021]
MATH500	+0.0658 (n=98) [.0433, .0888]	+0.0803 (n=100) [.0643, .0968]	+0.1112 (n=99) [.0980, .1242]
HumanEval	+0.1042 (n=56) [.0688, .1403]	+0.1193 (n=100) [.1056, .1330]	+0.1369 (n=100) [.1247, .1491]
MBPP	+0.0613 (n=64) [.0372, .0883]	+0.0605 (n=100) [.0361, .0856]	+0.1004 (n=100) [.0885, .1128]

Table 11: Generation-stage JS gaps for the first three 64-token blocks. Brackets give pointwise 95% paired sample-bootstrap confidence intervals.

Benchmark	Early gap [95% CI]	Middle gap [95% CI]	Late gap [95% CI]
GSM8K	+0.0636 (n=100) [.0512, .0758]	+0.0923 (n=100) [.0822, .1022]	+0.0642 (n=97) [.0539, .0746]
MATH500	+0.1257 (n=100) [.1061, .1449]	+0.1059 (n=99) [.0897, .1225]	+0.0717 (n=99) [.0603, .0830]
HumanEval	+0.1703 (n=100) [.1519, .1884]	+0.1257 (n=74) [.1010, .1510]	+0.1072 (n=96) [.0901, .1249]
MBPP	+0.0709 (n=100) [.0565, .0853]	+0.1224 (n=57) [.0986, .1468]	+0.0892 (n=49) [.0639, .1137]

gain on MBPP. This shows that the soft-future view provides useful signal beyond ordinary temporal convergence.

C.3 Threshold-Only Soft-Future Alternation Control

We compare FUSA with a threshold-only control: Confidence decoding with masked/soft-future alternation, threshold 0.90, fallback top-confidence commitment, and no JS/floor acceptance rule.

Table 13: Threshold-only control using the same α as FUSA. Scores are percentages; parentheses give average denoising steps.

Benchmark	α	Control	FUSA
GSM8K	0.65	75.82 (70.15)	75.89 (81.18)
MATH500	0.75	31.20 (97.66)	34.40 (102.69)
HumanEval	0.65	36.59 (74.90)	37.20 (85.37)
MBPP	0.65	37.35 (57.26)	42.80 (62.90)

Table 13 shows that alternation alone is insufficient. The control nearly matches FUSA on GSM8K, but trails on MATH500 and especially MBPP. Thus, the key benefit comes from using soft futures as a JS/floor stability test rather than merely alternating future views.

D Additional Ablations

D.1 Soft-Mask Mixing Ablation

Table 14 studies the soft-mask mixing coefficient α on GSM8K and MBPP while holding the other decoder settings fixed. Each benchmark reports task score and average denoising steps.

Table 14: Ablation of the soft-mask mixing coefficient α . Each benchmark reports task score and average denoising steps in separate columns.

α	GSM8K		MBPP	
	Acc	Steps	Pass@1	Steps
0.00	6.75	50.69	31.52	33.92
0.50	10.31	60.86	41.63	55.73
0.65	75.89	81.18	42.80	62.90
0.75	75.13	81.23	40.86	64.62
0.85	75.28	81.09	40.08	65.49
1.00	74.75	81.12	37.74	65.16

The results show that the soft future signal should be mixed rather than used at either extreme. A fully soft future view ($\alpha = 0$) sharply degrades performance, while $\alpha = 1$ removes the semantic future signal and reduces to a masked future view. The intermediate setting $\alpha = 0.65$ performs best on both benchmarks, supporting an uncertainty-preserving soft future representation.

D.2 GSM8K Hyperparameter Ablation

Table 15 reports a focused ablation of FUSA on GSM8K using LLaDA, with the high-confidence threshold and soft-mask mixing coefficient fixed. We vary only the robust confidence floor and JS threshold. The main setting is selected from validation splits rather than by maximizing this test-set sensitivity grid; therefore, bold marks the best observed test accuracy, and [†] marks the main setting used in Table 1.

Table 15: GSM8K ablation of FUSA for LLaDA with $\tau_{\text{high}} = 0.95$ and $\alpha = 0.65$ fixed. Each cell reports accuracy with average denoising steps, equivalently model forward calls, in parentheses. Accuracy is computed over 1,319 examples.

τ_{floor}	JS threshold ϵ		
	0.004	0.006	0.008
0.70	75.51% (77.67)	74.53% (75.38)	74.22% (73.93)
0.75	76.57% (79.11)	75.74% (77.61)	75.44% (76.22)
0.80	75.89% [†] (81.18)	75.59% (79.99)	75.66% (78.95)

[†] denotes the main FUSA setting selected from validation splits: $\tau_{\text{floor}} = 0.80$ and JS threshold $\epsilon = 0.004$.

D.3 MBPP Hyperparameter Ablation

Table 16 reports the corresponding MBPP ablation under the same fixed high-confidence threshold and soft-mask mixing coefficient, $\tau_{\text{high}} = 0.95$ and $\alpha = 0.65$. As in the GSM8K ablation, we vary only the robust confidence floor and the JS threshold. Each cell reports pass@1 with the average number of denoising steps, equivalently model forward calls, in parentheses.

Table 16: MBPP ablation of FUSA with $\tau_{\text{high}} = 0.95$ and $\alpha = 0.65$ fixed. Each cell reports pass@1 with average denoising steps, equivalently model forward calls, in parentheses.

τ_{floor}	JS threshold ϵ		
	0.004	0.006	0.008
0.70	41.63% (63.39)	40.47% (60.77)	40.47% (58.72)
0.75	41.63% (63.32)	42.41% (61.96)	40.47% (60.32)
0.80	42.41% (64.43)	42.80% (62.90)	42.41% (62.50)

FUSA corresponds to $\tau_{\text{floor}} = 0.80$ and JS threshold $\epsilon = 0.006$.

The MBPP trend is consistent with the selected benchmark setting in Table 20. A higher robust confidence floor generally improves pass@1, suggesting that code-generation tokens benefit from a more conservative acceptance floor. The best observed setting is $\tau_{\text{floor}} = 0.80$ and $\epsilon = 0.006$, which improves pass@1 while retaining a lower step count than the stricter $\epsilon = 0.004$ setting at the same confidence floor.

D.4 Future-Perturbation Window Ablation

FUSA uses $K = 1$ as its default local-probe design, perturbing only the immediately following block. To examine whether broader future perturbations provide a stronger stability signal, Table 17 compares this default with $K = 2$, which perturbs the next two future blocks, and $K = \infty$,

which perturbs the full masked suffix. All other benchmark-specific hyperparameters are kept fixed as in Table 20.

Table 17: Ablation of the future perturbation window. K denotes how many masked future blocks are softly perturbed when computing the stability signal; $K = 1$ is the default local-probe setting and $K = \infty$ denotes the full masked suffix. Each cell reports score with average denoising steps in parentheses.

Benchmark	$K = 1$ (main)	$K = 2$	$K = \infty$
GSM8K	75.89% (81.18)	75.06% (81.08)	74.30% (81.56)
MATH500	34.40% (102.69)	34.80% (101.87)	33.80% (103.37)
HumanEval	37.20% (85.37)	35.98% (85.08)	35.37% (85.11)
MBPP	42.80% (62.90)	40.86% (63.50)	40.47% (62.46)

Evaluation sizes are 1,319 for GSM8K, 500 for MATH500, 164 for HumanEval, and 257 for MBPP.

The $K = 2$ variant consistently outperforms $K = \infty$, indicating that perturbing the entire masked suffix can dilute the local stability signal. However, it does not replace the default $K = 1$ setting: $K = 1$ is strongest on GSM8K, HumanEval, and MBPP, while MATH500 is the only exception, with a small 0.40-point gain from $K = 2$.

A plausible explanation is that FUSA uses future information as a diagnostic probe rather than as additional context. For this probe to be informative, it should perturb unstable predictions more strongly than stable ones. The immediately following block provides the most reliable continuation of the active block, whereas more distant future blocks are predicted while additional intermediate blocks remain masked and are therefore less reliable. Perturbing a broader future window can consequently affect both stable and unstable hypotheses, reducing the discriminative power of the stability signal. This explanation is a mechanistic hypothesis rather than a formal theoretical characterization.

E OpenWebText Generation Experiments

E.1 Evaluation and Results

We evaluate unconditional long-form generation using an OpenWebText-pretrained MDLM checkpoint. Table 18 compares 1,000 generated samples of length 1,024 against held-out OpenWebText (Gokaslan and Cohen, 2019) references. MDLM, ReMDM-loop (Wang et al., 2025), and FUSA share the OpenWebText-pretrained MDLM checkpoint; SEDD is a contextual discrete-diffusion baseline, and AR is the same-codebase autoregressive checkpoint. MAUVE (Pillutla et al.,

2021) measures distributional similarity to references, Gen. PPL scores fluency using GPT-2 (Radford et al., 2019), and entropy is a token-distribution diversity statistic.

Table 18: Unconditional OpenWebText text quality. Higher MAUVE is better, lower Gen. PPL is better, and entropy is reported as a diversity statistic. Gen. PPL is computed with GPT-2 (Radford et al., 2019) as the evaluator. The controlled same-checkpoint comparison is MDLM versus FUSA; SEDD and AR are contextual baselines.

Method	MAUVE $\uparrow$	Gen. PPL $\downarrow$	Entropy
AR	0.783	12.72	7.076
SEDD	0.040	102.21	7.488
MDLM	0.143	48.92	7.220
ReMDM-loop	0.382	35.09	7.190
FUSA	0.405	37.65	7.157

AR serves as a strong reference baseline rather than a diffusion sampler. The controlled MDLM comparison is MDLM versus FUSA under the same checkpoint, where FUSA improves MAUVE from 0.143 to 0.405 and reduces Gen. PPL from 48.92 to 37.65. ReMDM-loop is also strong, with lower Gen. PPL but slightly lower MAUVE than FUSA. Overall, the OpenWebText experiment supports the claim that soft future decoding improves unconditional MDLM text quality.

E.2 Sampler and Checkpoint Details

The OpenWebText text-quality experiment uses sampler implementations and checkpoints from the ReMDM codebase (Wang et al., 2025).

ReMDM-loop is a separate non-monotonic re-masking baseline rather than a future-perturbation method. Unlike the standard MDLM sampler, which mostly moves monotonically from masks to tokens, ReMDM-loop allows already revealed tokens to be remasked during an intermediate time window. In our run, it uses generation length 1024, 512 steps, top- $p = 0.9$, $\eta = 0.02$, $t_{\text{on}} = 0.55$, $t_{\text{off}} = 0.05$, $\alpha_{\text{on}} = 0.9$, and 1,000 total samples.

The FUSA run uses soft future perturbations with $\tau_{\text{high}} = 0.95$, $\alpha = 0.65$, $\tau_{\text{floor}} = 0.80$, JS threshold 0.006, soft top- $p = 0.85$, generation length 1024, and 1024 denoising steps.

SEDD is a contextual discrete-diffusion baseline, not a same-checkpoint ablation of FUSA; the controlled decoder comparison remains MDLM versus FUSA.

The AR baseline is not an external GPT-style model. It is the OpenWebText-trained autoregres-

sive checkpoint provided by the same codebase, using a causal autoregressive configuration with generation length 1024, 1024 steps, and top- $p = 0.9$.

F Transfer Evaluation on CoDA-1.7B-Instruct

To examine whether FUSA transfers beyond the LLaDA model family, we evaluate CoDA-1.7B-Instruct (Chen et al., 2025a), a code-specialized full-attention diffusion model, on HumanEval and MBPP. We use the evaluation setup described in Appendix G and directly reuse the MBPP-selected hyperparameters in Table 20, without any CoDA-specific tuning.

Table 19: Transfer results on CoDA-1.7B-Instruct. Steps denote average model forward evaluations, TPS denotes generated tokens per second, and speedup is measured relative to Top-1.

Task	Method	Pass@1 $\uparrow$	Steps $\downarrow$	TPS $\uparrow$	Speedup $\uparrow$
HumanEval	Top-1	39.63%	256.00	57.05	1.00 $\times$
	FUSA	39.63%	40.66	281.54	4.94 $\times$
MBPP	Top-1	57.59%	256.00	59.43	1.00 $\times$
	FUSA	58.37%	58.86	193.27	3.25 $\times$

As shown in Table 19, FUSA matches the Top-1 Pass@1 on HumanEval and improves it by 0.78 points on MBPP, while using 6.30 $\times$ and 4.35 $\times$ fewer forward evaluations, respectively, and achieving 4.94 $\times$ and 3.25 $\times$ wall-clock speedups. These results indicate that the efficiency gains of FUSA transfer to CoDA without model-specific tuning.

G Evaluation Setup

G.1 FUSA Hyperparameter Settings

We select task-level FUSA configurations using development subsets sampled from the official training splits: 500 examples from GSM8K, 100 examples from MATH, and 100 examples from MBPP. The HumanEval configuration directly reuses the setting selected on MBPP without an additional search.

The high-confidence threshold τ_{high} is fixed to 0.95 across all benchmarks. The confidence floor τ_{floor} , soft-mask mixing coefficient α , and JS threshold ϵ are selected using the corresponding development subsets. Table 20 summarizes the resulting configurations.

These task-level configurations are subsequently transferred unchanged to LLaDA-1.5. The MBPP-selected configuration is also applied to CoDA-1.7B-Instruct without model-specific tuning. Sen-

sitivity analyses for α , τ_{floor} , and ϵ are provided in Appendices D.1, D.2, and D.3.

Table 20: Benchmark-specific hyperparameter settings for FUSA.

Benchmark	τ_{high}	α	τ_{floor}	JS ϵ
GSM8K	0.95	0.65	0.80	0.004
MATH500	0.95	0.75	0.70	0.006
HumanEval	0.95	0.65	0.80	0.006
MBPP	0.95	0.65	0.80	0.006

G.2 Baseline Sampler Settings

All baseline samplers are reported under their best benchmark-specific settings, while keeping the model, prompt format, maximum generation length, block size, and evaluation script fixed across methods. Table 21 summarizes the fixed sampler settings, and Table 22 lists the benchmark-specific KLASS thresholds.

Sampler	Setting	Value
EB-Sampler	Method	eb
	Max new tokens	256
	Block size	64
	γ	0.01
	Proxy / scope	confidence / block
	Temperature	0.0
KLASS	Generation length / steps	256 / 256
	Block length	64
	History length	2
	Unmask strategy	all
	Temperature	0.0
LRD	Method	1rd
	Max new tokens	256
	Block size	64
	Temperature	0.0
	Top- p	0.85
	Base / flex rate	0.95 / 0.05
	Cold start	6
	Diff / KL threshold	0.1 / 0.1

Table 21: Fixed baseline sampler settings.

Benchmark	Conf. thr.	KL thr.
GSM8K	0.6	0.015
MATH500	0.6	0.010
HumanEval	0.9	0.010
MBPP	0.7	0.010

Table 22: Benchmark-specific KLASS hyperparameters.

G.3 Benchmark Prompts

All decoding methods use the same benchmark-specific prompts and graders. Table 23 summarizes the prompt format used for each benchmark.

Benchmark	Prompt
GSM8K	Question: {question} Let’s think step by step. Answer:
MATH500	Problem: {problem} Let’s think step by step. Conclude with the final answer in <code>\boxed{...}</code> . Answer:
HumanEval	Original HumanEval function-completion prompt.
MBPP	Three-shot code-generation prompt with unit tests.

Table 23: Benchmark prompts used for all decoding methods. GSM8K and MATH500 use minimal zero-shot reasoning prompts, HumanEval uses the original function-completion prompt, and MBPP uses a three-shot code-generation prompt with unit tests.

G.4 WritingBench Evaluation Setup

We use a total context length of 4,096 tokens and a maximum generation length of 1,024 tokens, leaving a 3,072-token prompt budget. Starting from 555 English WritingBench prompts, we exclude seven whose chat-formatted inputs exceed this budget. From the remaining 548 prompts, we construct a fixed, domain-balanced subset of 100 prompts by sampling with seed 0 in a round-robin manner across the six high-level domains, yielding 16–17 prompts per domain. The same subset is used for Top-1 and FUSA.

H Examples of Generated Samples

We present qualitative examples from mathematical reasoning and code-generation benchmarks to illustrate how FUSA avoids premature commitment errors made by greedy token selection. Across these examples, Top-1 produces locally plausible continuations that miss a key constraint, whereas FUSA preserves the relevant condition and reaches the expected behavior.

MATH sample 134, Intermediate Algebra level 4

Question. Let $F_1 = (10, 2)$ and $F_2 = (-16, 2)$. The set of points P such that

$$|PF_1 - PF_2| = 24$$

forms a hyperbola:

$$\frac{(x-h)^2}{a^2} - \frac{(y-k)^2}{b^2} = 1.$$

Find $h + k + a + b$. The correct answer is 16.

Top-1 (wrong).

The center is the midpoint of the foci:
 $((-3), 2)$.

The distance between the foci is 26, so it sets $2a = 26$
and $a = 13$.
Then $c = 13$.

Since the hyperbola is oriented along the x-axis, $b = 0$.

$h + k + a + b = -3 + 2 + 13 + 0 = 12$.

Final answer: 12.

FUSA (correct).

The distance between the foci is $2c = 26$, so $c = 13$.

The center is:
 $((-3), 2)$.

The condition $|PF_1 - PF_2| = 24$ means $2a = 24$, so $a = 12$.

Using $c^2 = a^2 + b^2$:
 $b^2 = 13^2 - 12^2 = 169 - 144 = 25$,
so $b = 5$.

Therefore:
 $h + k + a + b = -3 + 2 + 12 + 5 = 16$.

Final answer: 16.

Observation. Top-1 confuses the focal distance $2c$ with the hyperbola difference constraint $2a$, setting $a = 13$ and collapsing to $b = 0$. FUSA correctly uses $2a = 24$, recovers $b = 5$, and obtains the right answer.

HumanEval sample 27

Task. `flip_case(string: str) -> str`: flip the case of alphabetic characters while preserving non-letter characters. The Top-1 baseline

handles lowercase and uppercase letters, but drops non-letter characters because it has no fallback branch. FUSA adds the missing else case and appends unchanged characters.

Top-1.

```
def flip_case(string: str) -> str:
    result = ""
    for char in string:
        if char.islower():
            result += char.upper()
        elif char.isupper():
            result += char.lower()
    return result
```

FUSA.

```
def flip_case(string: str) -> str:
    result = ""
    for char in string:
        if char.islower():
            result += char.upper()
        elif char.isupper():
            result += char.lower()
        else:
            result += char
    return result
```

MBPP sample 46 / task 99

Task. Convert a decimal number to a binary string with no leading zeros. Tests include `decimal_to_binary(8) == '1000'`, `decimal_to_binary(18) == '10010'`, and `decimal_to_binary(7) == '111'`. The Top-1 baseline uses floating-point division in the loop, while FUSA uses integer floor division and preserves the intended recurrence over bits.

Top-1.

```
def decimal_to_binary(n):
    if n == 0:
        return '0'
    binary_str = ''
    while n > 0:
        binary_str = str(n % 2) + binary_str
        n /= 2
    return binary_str
```

FUSA.

```
def decimal_to_binary(n):
    if n == 0:
        return '0'
    binary_str = ''
    while n > 0:
        binary_str = str(n % 2) + binary_str
        n = n // 2
    return binary_str
```