TARGETED REMASKING: TOKEN-TO-MASK REFINEMENT AS INFERENCE-TIME CORRECTION FOR DIFFUSION LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion language models (dLLMs) generate text through iterative denoising, filling multiple masked positions at each step. Positions filled in the same step are predicted without conditioning on one another's newly filled values and can therefore be mutually inconsistent; once retained, these inconsistencies become context for later predictions. We introduce *Token-to-Mask* (T2M), a training-free inference-time correction method that identifies low-confidence positions using the model's probability of the current token, remasks them, and reconstructs them in later denoising steps. On dLLMs equipped with correction mechanisms, a single T2M configuration transfers across tasks and models without retuning and broadly improves task metrics over each model's native correction mechanism.

1 INTRODUCTION

Diffusion language models (dLLMs) generate text by starting from a fully masked sequence and iteratively revealing tokens. Unlike autoregressive models, which generate tokens one by one in a fixed left-to-right order, dLLMs determine the reveal order using model confidence, such as token probabilities, and typically commit multiple positions in parallel at each step (Austin et al., 2021; Sahoo et al., 2024; Lou et al., 2024). This parallelism can produce the same-length response in far fewer sequential decoding steps. The LLaDA family (Nie et al., 2025; Bie et al., 2026) further shows that dLLMs can be competitive with autoregressive models of comparable scale. Even so, parallel commitment introduces a distinctive source of error propagation. Tokens revealed in the same step are predicted from the same context, but without conditioning on each other's newly revealed values, and may therefore be mutually inconsistent (Kang et al., 2026); once committed, such inconsistencies are preserved and treated as context for subsequent token predictions unless an explicit revision mechanism intervenes (Bie et al., 2026).

To mitigate this propagation, some models add a post-fill correction stage over already-visible tokens (Bie et al., 2026; Wang et al., 2025; Zhai et al., 2026; Huang et al., 2026; Kim et al., 2026), while others (e.g., LLaDA-8B (Nie et al., 2025) and Dream (Ye et al., 2025)) keep pure fill decoding with no such stage. When a correction stage is included, existing systems mainly choose between two *actions*. *Edit* (overwrite) replaces a visible token with another vocabulary token under the current context, as in LLaDA2.1's Token-to-Token (T2T) editing (Bie et al., 2026) and CORE's perturbed-context resampling (Zhai et al., 2026). *Remask* resets a visible token to `[M]` and lets later fill steps re-predict it. ReMDM remasks visible tokens at random with a time-dependent probability, without targeting particular errors (Wang et al., 2025). RemeDi (Huang et al., 2026) ranks every position in the current block by the model's confidence, keeps the most confident k, remasks the rest, and increases k over denoising steps.

Edit typically fuses detecting an error with immediately overwriting it. LLaDA2.1's T2T rule is a representative case: it overwrites a committed token only when the model's top prediction under the current context differs from it *and* that prediction's probability exceeds τ_{t2t}. This trigger produces the three failure modes in Fig. 1. In Fig. 1(a), the current word has only moderate confidence and remains the top prediction, so the edit never fires. In Fig. 1(b), a correct token is overwritten because a confident but incorrect top prediction clears τ_{t2t}. In Fig. 1(c), the model already prefers a different token, yet that candidate's probability stays below τ_{t2t}, so the edit again never fires. Across these

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.

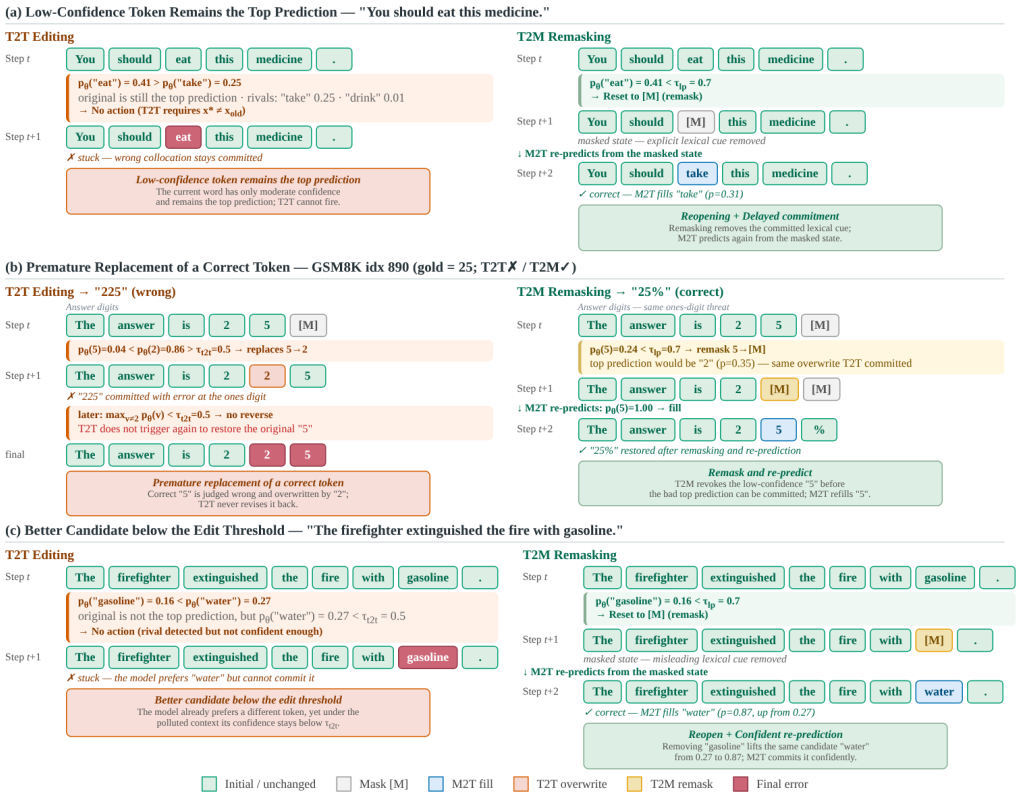


Figure 1: Three edit failure modes and T2M recovery on LLaDA2.1-mini: **(a)** low-confidence token remains the top prediction, **(b)** premature replacement of a correct token, **(c)** better candidate below the edit threshold. Rows proceed from step t to the final output. Panels (a) and (c) use hand-constructed sentences and directly read the model's token probabilities. Panel (b) visualizes token changes observed while generating the answer to a GSM8K example.

cases, remasking permits later reconstruction without first proposing a confident replacement. The traces therefore motivate our hypothesis that separating detection from immediate replacement is beneficial. The fused trigger is not the only issue; overwrite as an action has two further drawbacks. First, edit can create context pollution. Fig. 2 illustrates this possibility: concrete context tokens can sharply reshape the prediction at another masked position, whereas masking the clue removes that token-specific signal and leaves the prediction to the remaining context and model prior. Because edit must commit a concrete token rather than a masked placeholder, a wrong commitment remains an in-vocabulary, semantically loaded cue that can steer other positions toward continuations consistent with that cue, whereas a `[M]` removes that explicit lexical cue and returns the position to the masked state seen during denoising. Second, edit faces a training–inference noise mismatch. In some models, edit streams are trained to recover from *uniformly random* token perturbations injected into the input (Bie et al., 2026). Under this training regime, the model's own mistakes at inference time are usually semantically plausible and locally coherent with their neighbours (Fig. 1), so the editor is asked to handle a kind of structured error it was not trained on, an instance of the broader training–inference divide identified for masked dLLMs (He et al., 2025).

Remasking avoids forcing an immediate replacement, but existing remasking methods leave their own gaps. ReMDM remasks at random: wrong commitments need not be remasked, while correct ones may be remasked and then refilled incorrectly. RemeDi's keep-the-top-k rule is likewise not a per-position error test: a high-confidence commitment can still fall outside the cutoff and be re-masked, while a low-confidence one can remain inside top-k and escape; remasking is governed by cross-position rank rather than checking each position for error.

These gaps motivate a repair that keeps remask as the action but targets it using the same fill head that produces M2T predictions. We propose *Token-to-Mask* (T2M) refinement: a token flagged by

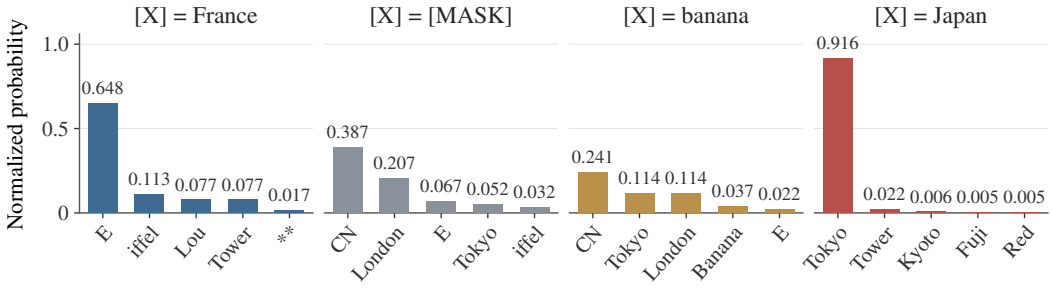


Figure 2: **Controlled context intervention on LLaDA2.1-mini.** For each of four context states at $[X]$ in "I went to $[X]$ and visited the `[M]` Tower," we plot the normalized top-5 distribution at the target mask.

a simple confidence test on the model's own logits (e.g., its probability falling below a threshold τ_{t2m}) is reset to `[M]` and re-predicted by the standard Mask-to-Token (M2T) fill rule like any other masked position. The procedure is purely inference-time: when the model already has a correction stage, it is a drop-in replacement for that stage. The paper makes the following contributions:

- We propose a training-free T2M rule guided by token probability: commitments whose probability falls below a threshold are reopened as `[M]`, decoupling detection from immediate replacement.
- We demonstrate task generality with one fixed configuration across 17 benchmarks spanning math, code, science, reasoning, and instruction following.
- We demonstrate transfer across models and correction mechanisms with one fixed configuration and no retuning: T2M replaces the native correction mechanism and improves performance on multiple models that already have a correction stage.
- By separating detector from action, we find that remasking outperforms replacement under every matched detector on a controlled dataset, and moreover that probability-guided remasking is a strong combination.

2 Related Work

Diffusion large language models. We briefly trace dLLMs from early diffusion models to current large-scale systems. Diffusion generative modeling first appeared for continuous data, especially images (Ho et al., 2020; Song et al., 2021). D3PM (Austin et al., 2021) then extended the framework to discrete tokens and singled out absorbing-state (mask) corruption as best suited to text. A series of works then progressively simplified this formulation, including SEDD (Lou et al., 2024), MDLM (Sahoo et al., 2024), and MD4 (Shi et al., 2024). RADD (Ou et al., 2025) further showed that the predictor depends only on the unmasked context and not on the diffusion time t, so a standard decoder-only Transformer (Vaswani et al., 2017) suffices. The framework has since been scaled up along two routes: LLaDA (Nie et al., 2025) trains an 8B dLLM from scratch, while Dream (Ye et al., 2025) and DiffuLLaMA (Gong et al., 2025) obtain comparable models by continued pretraining from autoregressive checkpoints; industrial systems such as Mercury (Inception Labs et al., 2025) push the same recipe to production scale. Beyond scaling the fill model, a growing line of work adds a post-fill correction stage to repair errors that arise when multiple positions are committed in parallel (Kang et al., 2026; Bie et al., 2026; Wang et al., 2025; Zhai et al., 2026; Huang et al., 2026; Kim et al., 2026). These strategies mainly take one of two actions, overwriting a committed token or reopening it as `[M]`, which we review next.

Edit-based correction strategies. Edit overwrites an already-visible token with another vocabulary token under the current context. LLaDA2.1 (Bie et al., 2026) runs Token-to-Token (T2T) editing after each fill step: every non-prompt visible position is re-scored, and the current token is replaced only when the model's top prediction differs from it and that prediction's probability clears a threshold τ_{t2t}. CORE (Zhai et al., 2026) instead selects which tokens to revise by measuring sensitivity to

masked-context perturbations and resamples replacements from those perturbed contexts; detection therefore relies on auxiliary probes rather than the logits on the actual generation trajectory, but the correction action remains overwrite.

Remask-based correction strategies. Remask resets an already-visible token to `[M]` and lets later fill steps re-predict it. ReMDM (Wang et al., 2025) applies a uniform remasking probability σ_t on each visible token. RemeDi (Huang et al., 2026) re-ranks every position in the current block at each step by a confidence score (the fill-head logit of the committed token if the position is visible, otherwise of the argmax fill candidate), keeps only the top K_n (with K_n rising over denoising steps toward the block length), and remasks every position outside that cutoff. PRISM (Kim et al., 2026) fine-tunes a separate binary head that scores each visible token for remasking. These methods avoid committing an immediate replacement, but either remask at random, fold remasking into a global unveiling order, or require additional training beyond the fill predictor. Our approach stays in this family while using a training-free, position-wise detector from the shared fill-head logits on models that already include a correction stage (Section 3).

3 METHOD

3.1 PRELIMINARIES

Mask-to-Token decoding. Masked dLLMs generate text through iterative Mask-to-Token (M2T) denoising (Nie et al., 2025; Ye et al., 2025; Sahoo et al., 2024). The positions to be generated are initialised as `[M]` tokens. At each denoising step, one forward pass produces a vocabulary distribution at every position that still holds `[M]`; the decoder then selects a subset of predictions according to its update rule and writes their token values into the corresponding positions. The filled tokens become context for subsequent predictions. The process repeats until no `[M]` remains.

Correction mechanisms. Parallel commitments are predicted from the same context without conditioning on one another and can therefore be mutually inconsistent (Kang et al., 2026). Some models therefore revise tokens after they have been filled. We take LLaDA2.1's Token-to-Token (T2T) editing as an example (Bie et al., 2026): after each M2T step, every non-mask, non-prompt position is re-evaluated and replaced when its argmax prediction exceeds a threshold:

$$\begin{aligned} x_i^* &= \arg\max_{v \in \mathcal{V}} p_\theta(v \mid \mathbf{z}_i), \quad p_i^* = \max_{v \in \mathcal{V}} p_\theta(v \mid \mathbf{z}_i), \\ x_i &\leftarrow \begin{cases} x_i^* & \text{if } p_i^* > \tau_{\text{t2t}} \text{ and } x_i^* \neq x_i^{\text{old}}, \\ x_i^{\text{old}} & \text{otherwise.} \end{cases} \end{aligned} \tag{1}$$

Here x_i is the token at position i with current value x_i^{old}, $\mathbf{z}_i$ is the sequence state used to predict i, $\mathcal{V}$ is the vocabulary, x_i^* and p_i^* are the argmax token and its probability under $p_\theta(\cdot \mid \mathbf{z}_i)$, and $\tau_{\text{t2t}} \in (0, 1)$ is the editing threshold.

3.2 TOKEN-TO-MASK REMASKING

Overview. T2M is a post-fill correction mechanism that replaces an existing correction rule, such as LLaDA2.1's T2T. The model and the M2T fill rule stay unchanged. At every non-mask, non-prompt position i, a detector judges whether the committed token remains trustworthy, and a flagged token is reset to `[M]`:

$$\begin{aligned} d_i &= \mathcal{D}_{\mathcal{S}}(i, \mathbf{z}_i, x_i^{\text{old}}, \tau_{\text{t2m}}), \quad d_i \in \{0, 1\}, \\ x_i &\leftarrow \begin{cases} \texttt{[M]} & \text{if } d_i = 1, \\ x_i^{\text{old}} & \text{otherwise.} \end{cases} \end{aligned} \tag{2}$$

where $d_i \in \{0, 1\}$ is the per-position remask indicator, $\mathcal{D}_{\mathcal{S}}$ is the detector under strategy $\mathcal{S}$, x_i^{old} is the current token at position i, and τ_{t2m} is the strategy-specific remask threshold. Algorithm 1 specifies the full T2M procedure. It runs after each M2T step. On a model with a correction stage, T2M replaces that stage (T2T editing on LLaDA2.1); on a model without one, the same rule is inserted after each fill step (Section 4.4).

Algorithm 1 Token-to-Mask (T2M) Remasking. Runs after each M2T step, in place of any existing correction stage.

Require: Current tokens $\mathbf{z} = (z_1, \ldots, z_B)$, positions $\mathcal{F}$ filled by the immediately preceding M2T step, model p_θ, mask id `[M]`, detector strategy $\mathcal{S}$, threshold τ_{t2m}, per-position remask counts $\{c_i\}_{i=1}^{B}$ (initialized to 0, persisted across M2T iterations), per-position budget $C_{\max}$, per-step ratio cap $\rho_{\max}$

$\mathcal{E} \leftarrow \{i : z_i \neq$ `[M]`$,\ i \notin$ prompt$,\ i \notin \mathcal{F}\}$ ▷ **editable positions**: previously filled, non-prompt
$\mathcal{R} \leftarrow \{i \in \mathcal{E} : \mathcal{D}_{\mathcal{S}}(i, \mathbf{z}_i, x_i^{\text{old}}, \tau_{\text{t2m}}) = 1$ and $c_i < C_{\max}\}$ ▷ **remask set**: flagged by the detector and under budget
if $|\mathcal{R}| > \rho_{\max} \cdot |\mathcal{E}|$ **then**
  $k \leftarrow \max(1, \lfloor \rho_{\max} \cdot |\mathcal{E}| \rfloor)$ ▷ ratio cap binds: keep top-k most suspicious
  $\mathcal{R} \leftarrow$ top-k positions in $\mathcal{R}$ ranked by $\mathcal{S}$
end if
for each $i \in \mathcal{R}$ **do**
  $z_i \leftarrow$ `[M]`; $\quad c_i \leftarrow c_i + 1$ ▷ reset flagged positions to `[M]`
end for
return updated $\mathbf{z}$

Error detection. We consider three instantiations of the detector $\mathcal{D}_{\mathcal{S}}$, each built around a different signal:

$$d_i^{\text{lp}} = \mathbf{1}\left\{p_\theta(x_i^{\text{old}} \mid \mathbf{z}_i) < \tau_{\text{lp}}\right\}, \tag{3}$$

$$d_i^{\text{tr}} = \mathbf{1}\left\{p_\theta(x_i^* \mid \mathbf{z}_i) > \tau_{\text{tr}} \text{ and } x_i^* \neq x_i^{\text{old}}\right\}, \quad x_i^* = \arg\max_{v \in \mathcal{V}} p_\theta(v \mid \mathbf{z}_i), \tag{4}$$

$$d_i^{\text{ld}} = \mathbf{1}\left\{p_\theta^{(t-1)}(x_i^{\text{old}} \mid \mathbf{z}_i^{(t-1)}) - p_\theta^{(t)}(x_i^{\text{old}} \mid \mathbf{z}_i^{(t)}) > \tau_{\text{ld}}\right\}. \tag{5}$$

- **LOWPROB** (Eq. 3). Remask whenever the model's probability for the currently committed token falls below threshold, with no dependence on any replacement candidate.
- **T2T-REMASK** (Eq. 4). Reuse T2T's trigger and swap its action for remasking, isolating the effect of "remask vs. replace" under a fixed detector.
- **LOGITDIFF** (Eq. 5). Track how the model's confidence in x_i^{old} evolves between consecutive iterations and fire on a probability drop. A falling confidence indicates the converging neighbourhood has stopped supporting the token; a rising one indicates it has been corroborated. The signal depends on the denoising trajectory rather than a single snapshot; LOGITDIFF abstains at the first iteration of a decoding window.

Safety caps. Remasking can in principle oscillate: a position is remasked, refilled with a similar token, remasked again. Two caps prevent this. A per-position budget $C_{\max}$ limits how many times a single position may be remasked, and a per-step ratio cap $\rho_{\max}$ limits the fraction of editable positions that can be remasked in one M2T iteration. When the ratio cap binds, the least-confident positions are selected first.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Models and decoding. LLaDA2.1-mini (Bie et al., 2026) (16B mixture-of-experts; Appendix A) is the backbone for the 17-benchmark suite, the mechanism comparisons, and all ablations; cross-model runs further include LLaDA2.1-flash (100B mixture-of-experts) (Bie et al., 2026), RemeDi-Instruct (9B) (Huang et al., 2026), Dream-7B (7B) (Ye et al., 2025), and LLaDA-8B-Base (8B) (Nie et al., 2025). Both LLaDA2.1 models are served by SGLang dLLM (Zheng et al., 2024) under official Q Mode defaults (fill τ_{m2t}=0.7, edit τ_{t2t}=0.5, block B=32, post-fill edit window $\leq$ 16, temperature 0). The correction practices under comparison are as follows. *Native baselines:* T2T editing on both LLaDA2.1 models; RemeDi-Instruct's released top-k remask under its released generator; pure M2T (no correction stage) on Dream-7B and LLaDA-8B-Base under their

released generators (Dream-7B: temperature 0.1, top-p=0.9, length/steps $256/256$; LLaDA-8B-Base: length/steps/block $256/256/256$). *Our method:* T2M remasking (Section 3) replaces the native correction action when one exists, and is inserted after each fill step when none does. *Additional baselines on LLaDA2.1-mini only:* ReMDM-style random remasking (Wang et al., 2025) ($\sigma \in \{0.02, 0.05, 0.10\}$, with higher-budget references reported separately) and CORE (Zhai et al., 2026) (gate 0.3, interval $8/2$; Appendix A), both ported onto the same Q Mode loop. Full per-model details are in Appendix A.

T2M configuration. Unless stated otherwise, T2M uses LowProb with τ_{lp}=0.7, $C_{\max}$=3, $\rho_{\max}$=0.5, selected on the GSM8K sweep of Section 4.6 and then held fixed. LLaDA2.1-flash reuses this setting without retuning; for the RemeDi-Instruct comparison we keep the same τ_{lp}=0.7 and replace its released top-k fill with threshold-M2T fill at 0.7; Dream-7B and LLaDA-8B-Base (no native correction stage) insert the same LowProb rule (τ_{lp}=0.7, $C_{\max}$=3, $\rho_{\max}$=0.5) after each fill step.

Benchmarks and protocols. The task-generality evaluation covers 17 benchmarks spanning math, code, science/knowledge, general reasoning, and instruction following under paired task-standard protocols (Ma et al., 2025; OpenCompass Contributors, 2023) (per-task split, prompt, and scorer in Appendix B). The generation budget is $16{,}384$ tokens for the LLaDA2.1-mini 17-benchmark suite (TriviaQA: 50 tokens). On GSM8K, Table 1 and Fig. 3 use $16{,}384$; the 53-configuration sweep (each run on all 1,319 examples), the LLaDA2.1-mini block of Table 3, the top-k ladder, and LLaDA2.1-flash use $4{,}096$ tokens. At $4{,}096$ tokens, the cap binds for $< 1\%$ of samples; all cells within a comparison share the same budget. The GSM8K cross-model cells use each model's released prompt, budget, and scorer. Every cell is the full split; paired comparisons share weights, prompts, budget, engine, and scorer.

Metrics. Task scores follow each benchmark's official metric (accuracy, exact match, or pass@1) under the corresponding scorer (Appendix B); we also report mean generated length. For compute we additionally report the **number of forward evaluations per generated token (NFE/token)**, measured per sample at batch size 1. The value 1.0 is one forward per generated token, the cost of autoregressive decoding without speculation. Both edit and remask reuse that fill forward; remask's extra cost is the later fill steps that close the reopened `[M]`s. Wall-clock latency is in Appendix G; reproduction notes in Appendix H.

4.2 Task Generality of T2M

With the configuration selected on GSM8K and applied unchanged to the other 16 benchmarks, T2M improves the task metric on 15 of 17 benchmarks (Table 1); the two non-improvements stay within 0.12 points of the baseline, while the largest increases are HumanEval+ ($+3.66$), GPQA ($+2.02$), and GSM8K ($+1.59$). Mean generated lengths stay close to the T2T lengths, so the metric changes are not from longer outputs. These improvements incur additional compute. Remasking reopens tokens as `[M]`, and later fill steps must close them: relative to T2T, T2M uses 2.0–$3.1\times$ as many forwards. On GSM8K, both the forward count and measured latency increase by $2.2\times$; full per-task latency results are in Appendix G. Ten of seventeen tasks remain below the AR reference of 1.0. The other seven are either short generations (TriviaQA and PIQA, roughly 10–10^2 tokens), where the same extra forwards are spread over few tokens, or tasks whose 2–$3\times$ multiplier raises NFE/token above 1.0 (GPQA, C-EVAL, IFEval, HellaSwag, OCNLI). If NFE/token exceeds 1.0, tightening the remask budget can recover a point below it. We illustrate this on C-EVAL ($C_{\max}$=1, $\rho_{\max}$=0.25: 77.04 at 0.86 NFE/token) and do not apply the same budget tightening to the remaining tasks.

4.3 Accuracy under Additional Compute

T2M uses 2.0–$3.1\times$ as many forward passes as T2T editing. Sweeping the fill threshold and the correction threshold for both methods shows that editing does not turn the extra forward passes into the accuracy of T2M (Fig. 3). On GSM8K, editing stays below the default T2M accuracy in both sweeps. On HumanEval+, MBPP+, C-EVAL, and PIQA, the best editing accuracy over the same sweeps also stays below the best T2M accuracy (Appendix D, Table S3).

Table 1: Task generality of T2M on LLaDA2.1-mini (paired task-standard protocols). † denotes the GSM8K selection benchmark; the same configuration is used without per-task retuning on the other 16 tasks. Metrics are accuracy, exact match (EM), or first-sample pass rate (pass@1), in percent. Len. is mean generated length; Len. and NFE/tok. are T2T / T2M (AR = 1.0). Row groups are math, code, science/knowledge, reasoning, and instruction following.

Benchmark	n	**T2T**	**T2M**	Δ	**Len.**	**NFE/tok.**
GSM8K†	1319	89.61	**91.21**	+1.59	296 / 292	0.26 / 0.58
OlympiadBench	674	58.16	**58.75**	+0.59	4440 / 4427	0.26 / 0.62
GSM-Plus	10552	81.37	**81.79**	+0.42	335 / 344	0.32 / 0.84
Omni-MATH	4428	40.31	**40.42**	+0.11	5452 / 5426	0.27 / 0.66
CMATH	1098	92.26	**92.35**	+0.09	301 / 302	0.32 / 0.77
HumanEval+	164	76.83	**80.49**	+3.66	353 / 335	0.14 / 0.29
MBPP+	378	68.78	**70.11**	+1.33	268 / 273	0.18 / 0.37
GPQA-Diamond	198	43.94	**45.96**	+2.02	4588 / 4629	0.41 / 1.12
C-EVAL	1346	75.63	**76.97**	+1.34	882 / 820	0.51 / 1.48
MMLU-Pro	12032	63.41	**63.56**	+0.15	1907 / 1921	0.37 / 0.97
TriviaQA	17944	53.18	**53.24**	+0.06	9 / 9	0.83 / 2.16
PIQA	1838	80.90	**82.64**	+1.74	95 / 105	0.76 / 2.04
OCNLI	2950	60.78	**61.32**	+0.54	138 / 138	0.58 / 1.69
DROP	9536	84.14	**84.58**	+0.44	232 / 223	0.33 / 0.73
BBH	6511	**81.92**	81.82	−0.10	665 / 620	0.42 / 0.96
HellaSwag	10042	**79.50**	79.38	−0.12	251 / 250	0.55 / 1.50
IFEval (prompt-strict)	541	83.36	**84.47**	+1.11	461 / 482	0.70 / 2.17

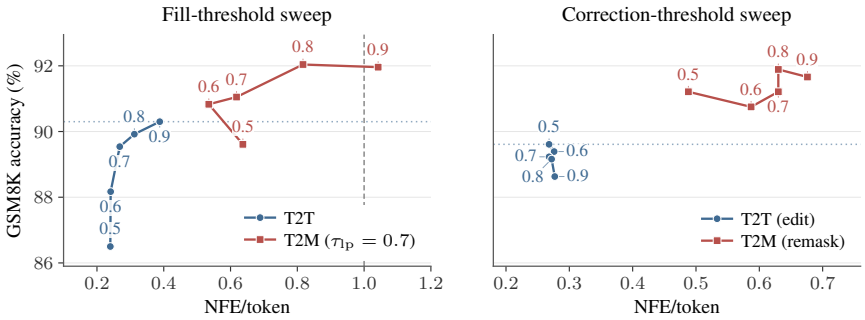


Figure 3: **Accuracy versus NFE/token on full GSM8K.** The left panel sweeps the fill threshold and the right panel sweeps the edit or remask threshold at fill 0.7; point labels give the swept value. The dashed line marks 1.0 NFE/token and the dotted line marks the highest T2T accuracy.

4.4 Cross-Model Transfer

T2M uses the configuration selected on LLaDA2.1-mini without retuning and is compared against each model's released generator (Table 2). *Targeted remasking shows positive transfer on models that already have a correction stage.* On the six reported LLaDA2.1-flash tasks, T2M improves performance, with the largest gain on GSM8K (+1.44). Dream-7B and LLaDA-8B-Base have no correction training; the gain is absent or not apparent.

4.5 Detector and Action Comparison

Correction methods bind a detector to an action. Table 3 compares these rules on full GSM8K, pairing each detector with both actions. With the detector held fixed, remasking is more accurate than replacement for every detector, and in both CORE rows it also uses fewer forward passes. The com-

Table 2: Cross-model study. Each baseline uses that model's released generator; T2M replaces native post-fill correction when present and is inserted after fill otherwise, with the remaining released settings fixed (Section 4.1).

Model	**Correction training**	**Benchmark**	**Baseline**	**T2M**	Δ
LLaDA2.1-flash 100B	T2T editing stream	GSM8K	92.87	**94.31**	+1.44
		MBPP+	76.46	**77.78**	+1.32
		IFEval	84.29	**85.40**	+1.11
		C-EVAL	85.36	**86.03**	+0.67
		PIQA	90.48	**91.02**	+0.54
		CMATH	94.08	**94.35**	+0.27
RemeDi-Instruct	top-k remask	GSM8K	86.28	**87.04**	+0.76
Dream-7B	none	GSM8K	**79.53**	79.23	−0.30
LLaDA-8B-Base	none	GSM8K	73.31	**73.39**	+0.08

bination we adopt, LOWPROB detection with remasking, is more accurate than the native correction rule of each method in the comparison: T2T editing, CORE, and RemeDi's released schedule. Its most accurate setting in the table also exceeds the best ReMDM-style random-remasking run, at a fraction of that run's forward count.

Table 3: Detector $\times$ action on full GSM8K with a 4,096-token generation budget. Each cell reports accuracy @ NFE/token (AR = 1.0). Upper block: LLaDA2.1-mini; bottom: RemeDi-Instruct under its released loop (Appendix A). Unless a row notes otherwise, the remask budget is $C_{\max}$=3, $\rho_{\max}$=0.5.

Detector \ action	**Replace**	**Remask**
None (pure M2T decoding, base)	86.58 @ 0.283	
Random, σ=0.02, C=3 (ReMDM-style)	—	87.11 @ 0.43
Random, σ=0.05, C=3	—	89.23 @ 0.98
Random, σ=0.10, C=3	—	89.76 @ 1.35
Random, σ=0.10, C=8	—	91.28 @ 2.88
T2T trigger	89.54 @ 0.262 (official)	89.76 @ 0.267 (T2T-REMASK)
CORE sensitivity, interval 8	87.11 @ 0.496 (CORE)	88.86 @ 0.457
CORE sensitivity, interval 2	85.22 @ 0.642	89.39 @ 0.523
LOWPROB	89.39 @ 0.258	**91.13 @ 0.584** (T2M, default)
LOWPROB, τ=0.9, C=3	—	**91.66 @ 0.672**
On RemeDi-Instruct's released loop (Appendix A):		
RemeDi released confidence score	67.02 @ 1.08	86.28 @ 1.08 (released top-K)
LOWPROB	39.73 @ 0.42	**87.04 @ 0.52** (T2M)

4.6 ABLATION STUDIES

Fig. 4 holds the remasking action fixed and compares detectors. Every LOWPROB setting lies above T2T editing, so the gain is not an artifact of one operating point. LOGITDIFF reads the denoising trajectory rather than a single-token probability, and its accuracy does not exceed LOWPROB. We use LOWPROB with τ_{lp}=0.7, $C_{\max}$=3, and $\rho_{\max}$=0.5 (the star in Fig. 4) in all cross-task and cross-model experiments. Among the 20 LOWPROB settings in Fig. 4, this setting is within 0.53 points of the most accurate one (τ_{lp}=0.9, 91.66) and requires fewer forward passes (0.584 vs. 0.672

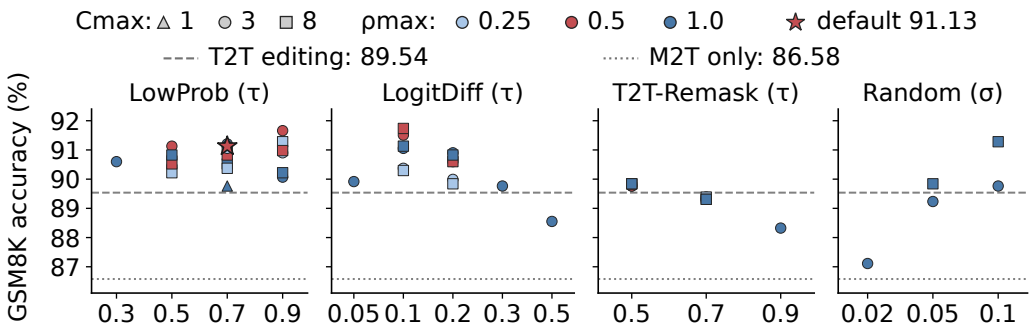


Figure 4: **Detectors under remasking on full GSM8K** (4,096-token generation budget).

NFE/token; Table 3). How often a remask changes the token, and how many answers that corrects or breaks, are in Appendix F.

5 Conclusion

We introduced Token-to-Mask (T2M), a training-free post-fill correction method for diffusion language models. Rather than immediately replacing suspicious tokens, T2M identifies low-confidence positions from the probabilities of their current tokens, remasks them, and lets subsequent standard M2T steps reconstruct them. Without task- or model-specific retuning, a single T2M configuration yields consistent gains across diverse tasks and multiple models equipped with post-fill correction, demonstrating cross-task and cross-model generality. Decomposing correction into detector and action shows that, with the detector held fixed, remasking outperforms replacement; detector ablations further show that identifying which tokens to revise is equally important, and that current-token probability provides a simple and effective per-position signal. Overall, by selectively reopening low-confidence positions and deferring reconstruction, T2M mitigates several failure modes of direct editing and converts additional inference compute into performance more effectively than editing; on most tasks, these gains retain the sequential-step advantage of dLLMs over autoregressive token-by-token decoding.

Limitations. T2M relies on the reconstruction capability induced by correction training; on fill-only models without such training, remasking yields little benefit. T2M also requires additional denoising steps to refill remasked positions, resulting in 2.0–3.1$\times$ the NFE of T2T and therefore higher inference cost.

AI use statement

We used generative AI tools to assist with manuscript polishing, code review and debugging, and the organization and management of experimental results. The authors take responsibility for the final manuscript, code, and reported results.

Ethics statement

This work studies inference-time correction for existing diffusion language models using established benchmarks. Improvements in benchmark performance should not be interpreted as guarantees of factual correctness, fairness, or safety. The proposed method does not explicitly address biases or harmful outputs that may be inherited from the underlying models. Remasking also incurs additional inference computation; we report computational cost and latency to make this trade-off explicit.

Reproducibility statement

The algorithmic description, parameter settings, and evaluation protocols provided in this paper offer sufficient detail to independently implement the proposed method and reproduce the experimental setup. The proposed decoding procedure is described in Section 3 and Algorithm 1. Section 4.1

presents the experimental setup, with model details and hyperparameter sweeps documented in Appendix A and Appendix C. Benchmark splits, prompting protocols, and evaluation metrics are summarized in Appendix B. Additional information on compute metrics and measurement settings is provided in Appendix G, while Appendix H discusses baseline reproduction.

REFERENCES

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*, volume 34, pp. 17981–17993. Curran Associates, Inc., 2021.

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, Chengxi Li, Jianguo Li, Zehuan Li, Huabin Liu, Lin Liu, Guoshan Lu, Yuan Lu, Yuxin Ma, Xingyu Mou, Zhenxuan Pan, Kaida Qiu, Yuji Ren, Jianfeng Tan, Yiding Tian, Zian Wang, Lanning Wei, Tao Wu, Yipeng Xing, Wentao Ye, Liangyu Zha, Tianze Zhang, Xiaolu Zhang, Junbo Zhao, Da Zheng, Hao Zhong, Wanli Zhong, Jun Zhou, Junlin Zhou, Liwang Zhu, Muzhi Zhu, and Yihong Zhuang. LLaDA2.1: Speeding up text diffusion via token editing. *arXiv preprint arXiv:2602.08676*, 2026.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 7432–7439, 2020. doi: 10.1609/aaai.v34i05.6239.

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, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

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. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. 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)*, pp. 2368–2378. Association for Computational Linguistics, 2019. doi: 10.18653/v1/N19-1246.

Bofei Gao, Feifan Song, Zhe Yang, Zefan Cai, Yibo Miao, Qingxiu Dong, Lei Li, Chenghao Ma, Liang Chen, Runxin Xu, Zhengyang Tang, Benyou Wang, Daoguang Zan, Shanghaoran Quan, Ge Zhang, Lei Sha, Yichang Zhang, Xuancheng Ren, Tianyu Liu, and Baobao Chang. Omni-MATH: A universal olympiad level mathematic benchmark for large language models. In *International Conference on Learning Representations*, 2025.

Shansan Gong, Shivam Agarwal, Yizhe Zhang, Jiacheng Ye, Lin Zheng, Mukai Li, Chenxin An, Peilin Zhao, Wei Bi, Jiawei Han, Hao Peng, and Lingpeng Kong. Scaling diffusion language models via adaptation from autoregressive models. In *International Conference on Learning Representations*, 2025.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. OlympiadBench: A challenging benchmark for promoting AGI with olympiad-level bilingual

multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3828–3850. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.211.

Haoyu He, Katrin Renz, Yong Cao, and Andreas Geiger. MDPO: Overcoming the training-inference divide of masked diffusion language models. *arXiv preprint arXiv:2508.13148*, 2025.

Jonathan Ho, Ajay N. Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851. Curran Associates, Inc., 2020.

Hai Hu, Kyle Richardson, Liang Xu, Lu Li, Sandra Kübler, and Lawrence S. Moss. OCNLI: Original Chinese natural language inference. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 3512–3526. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.findings-emnlp.314.

Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv, Yikai Zhang, Jiayi Lei, Yao Fu, Maosong Sun, and Junxian He. C-Eval: A multi-level multi-discipline Chinese evaluation suite for foundation models. In *Advances in Neural Information Processing Systems*, volume 36, pp. 62991–63010. Curran Associates, Inc., 2023. doi: 10.52202/075280-2749.

Zemin Huang, Yuhang Wang, Zhiyang Chen, and Guo-Jun Qi. Don't settle too early: Self-reflective remasking for diffusion language models. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=BsZeTuB5fD.

Inception Labs, Samar Khanna, Siddhant Kharbanda, Shufan Li, Harshit Varma, Eric Wang, Sawyer Birnbaum, Ziyang Luo, Yanis Miraoui, Akash Palrecha, Stefano Ermon, Aditya Grover, and Volodymyr Kuleshov. Mercury: Ultra-fast language models based on diffusion. *arXiv preprint arXiv:2506.17298*, 2025.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1601–1611. Association for Computational Linguistics, 2017. doi: 10.18653/v1/P17-1147.

Wonjun Kang, Kevin Galim, Seunghyuk Oh, Minjae Lee, Yuchen Zeng, Shuibai Zhang, Coleman Richard Charles Hooper, Yuezhou Hu, Hyung Il Koo, Nam Ik Cho, and Kangwook Lee. ParallelBench: Understanding the trade-offs of parallel decoding in diffusion LLMs. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=OsZr5T7Cd0.

Jaeyeon Kim, Seunggeun Kim, Taekyun Lee, David Z. Pan, Hyeji Kim, Sham M. Kakade, and Sitan Chen. Fine-tuning masked diffusion for provable self-correction. In *International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=2LRaNghfoM.

Qintong Li, Leyang Cui, Xueliang Zhao, Lingpeng Kong, and Wei Bi. GSM-Plus: A comprehensive benchmark for evaluating the robustness of LLMs as mathematical problem solvers. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2961–2984. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.163.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by ChatGPT really correct? rigorous evaluation of large language models for code generation. In *Advances in Neural Information Processing Systems*, volume 36, pp. 21558–21572. Curran Associates, Inc., 2023. doi: 10.52202/075280-0943.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32819–32848. PMLR, 2024.

Yuxin Ma, Lun Du, Lanning Wei, Kun Chen, Qian Xu, Kangyu Wang, Guofeng Feng, Guoshan Lu, Lin Liu, Xiaojing Qi, Xinyuan Zhang, Zhen Tao, Haibo Feng, Ziyun Jiang, Ying Xu, Zenan Huang, Yihong Zhuang, Haokai Xu, Jiaqi Hu, Zhenzhong Lan, Junbo Zhao, Jianguo Li, and Da Zheng. dInfer: An efficient inference framework for diffusion language models. *arXiv preprint arXiv:2510.08666*, 2025.

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*, volume 38, pp. 50608–50646. Curran Associates, Inc., 2025. doi: 10.52202/085713-1689.

OpenCompass Contributors. OpenCompass: A universal evaluation platform for foundation models, 2023. URL https://github.com/open-compass/opencompass.

Jingyang Ou, Shen Nie, Kaiwen Xue, Fengqi Zhu, Jiacheng Sun, Zhenguo Li, and Chongxuan Li. Your absorbing discrete diffusion secretly models the conditional distributions of clean data. In *International Conference on Learning Representations*, 2025.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level Google-proof Q&A benchmark. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98.

Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T. Chiu, Alexander M. Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. In *Advances in Neural Information Processing Systems*, volume 37, pp. 130136–130184. Curran Associates, Inc., 2024. doi: 10.52202/079017-4135.

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. Curran Associates, Inc., 2024. doi: 10.52202/079017-3277.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. Challenging BIG-Bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 13003–13051. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.findings-acl.824.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Guanghan Wang, Yair Schiff, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Remasking discrete diffusion models with inference-time scaling. In *Advances in Neural Information Processing Systems*, volume 38, pp. 147282–147339. Curran Associates, Inc., 2025. doi: 10.52202/085713-4927.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. MMLU-Pro: A more robust and challenging multi-task language understanding benchmark. In *Advances in Neural Information Processing Systems*, volume 37, pp. 95266–95290. Curran Associates, Inc., 2024. doi: 10.52202/079017-3018.

Tianwen Wei, Jian Luan, Wei Liu, Shuang Dong, and Bin Wang. CMATH: Can your language model pass Chinese elementary school math test? *arXiv preprint arXiv:2306.16636*, 2023.

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

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800. Association for Computational Linguistics, 2019. doi: 10.18653/v1/P19-1472.

Kevin Zhai, Sabbir Mollah, Zhenyi Wang, and Mubarak Shah. CORE: Context-robust remasking for diffusion language models. *arXiv preprint arXiv:2602.04096*, 2026.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E. Gonzalez, Clark Barrett, and Ying Sheng. SGLang: Efficient execution of structured language model programs. In *Advances in Neural Information Processing Systems*, volume 37, pp. 62557–62583. Curran Associates, Inc., 2024. doi: 10.52202/079017-2000.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

A Model Details

LLaDA2.1-mini (Bie et al., 2026) is a 20-layer mixture-of-experts model with 16B non-embedding parameters: 256 routed experts with 8 active per token plus one shared expert, hidden size 2048, a 32k-token context window, and a 157k vocabulary. LLaDA2.1-flash uses 4-way tensor parallelism. Beyond the decoding settings in Section 4.1: Dream-7B remasks by entropy; LLaDA-8B-Base is 4-shot chain-of-thought under the OpenCompass base configuration; RemeDi-Instruct uses its released boxed-answer setting, with generation length 1024 and block length 32. On that procedure, the released schedule modifies $\sim$384 positions per sample in 274 forward passes, LowProb remasking modifies $\sim$112 in 134, and argmax overwrite modifies $\sim$348 in 270 with RemeDi's confidence score and $\sim$90 in 96 with LowProb.

CORE. The CORE rows use its published detector and action on LLaDA2.1-mini (Zhai et al., 2026). Its released candidate count of 32 would mask our entire 32-token block, so we use 4, and it refills within the same step rather than leaving the position masked.

B Benchmark Details

Table S1 lists all benchmarks used in our evaluation with dataset details. The math suite comprises GSM8K (Cobbe et al., 2021), GSM-Plus (Li et al., 2024), CMATH (Wei et al., 2023), OlympiadBench (He et al., 2024), and Omni-MATH (Gao et al., 2025). Code evaluation uses HumanEval and MBPP under EvalPlus (Chen et al., 2021; Liu et al., 2023). Knowledge evaluation covers GPQA (Rein et al., 2024), C-EVAL (Huang et al., 2023), MMLU-Pro (Wang et al., 2024), and TriviaQA (Joshi et al., 2017). The remaining suite includes BBH (Suzgun et al., 2023), DROP (Dua et al., 2019), HellaSwag (Zellers et al., 2019), PIQA (Bisk et al., 2020), OCNLI (Hu et al., 2020), and IFEval (Zhou et al., 2023). IFEval uses the official prompt-strict checker. When optional arguments are absent, the checker's included instruction constructors sample random defaults, so we reset the random number generator to seed 0 before scoring each run separately. Both runs are then scored deterministically under the same seed.

C Hyperparameter Sweep Details

Fig. 4 contains 53 configurations, each on the full GSM8K test set under the Q Mode defaults of Section 4.1. The loop stops when every mask is filled and no remask fires, rather than after a fixed number of steps. Three remask hyperparameters vary:

1. The *remask threshold* τ_{t2m} controls how often each strategy triggers remasking. Whether a higher threshold increases or decreases the number of remasks depends on the strategy (Table S2).
2. The *per-position budget* $C_{\max} \in \{1, 3, 8\}$ limits how often any one position can be remasked.
3. The *ratio cap* $\rho_{\max} \in \{0.25, 0.50, 1.0\}$ caps the fraction of editable positions remasked in a single step; $\rho_{\max} = 1.0$ corresponds to no cap.

ReMDM-style random remasking uses $\sigma \in \{0.02, 0.05, 0.10\}$ under the same caps. Table 3 adds one run that keeps the default LowProb detector and restores replacement.

D Accuracy under Additional Compute

Figs. S1 and S2 repeat the sweeps of Section 4.3 on HumanEval+, MBPP+, C-EVAL, and PIQA, with GSM8K included for reference. T2M uses τ_{lp}=0.7; the τ_{lp}=0.9 setting is not included. Table S3 gives the highest accuracy of each method over the ten settings.

Table S1: Benchmark details. Generation budget $16{,}384$ except TriviaQA (50). Row groups are math, code, science/knowledge, reasoning, and instruction following.

Benchmark	Split	Samples	Prompt	Metric / Scorer
GSM8K	test	1,319	4-shot (dInfer template)	EM (numeric)
OlympiadBench	maths	674	0-shot, boxed answer	EM (numeric)
GSM-Plus	test	10,552	4-shot, unsolvable exemplar	EM (numeric)
Omni-MATH	test	4,428	0-shot, boxed answer	EM (numeric)
CMATH	test	1,098	0-shot, boxed answer	EM (numeric)
HumanEval+	test	164	EvalPlus instruction prompt	pass@1 (EvalPlus)
MBPP+	test	378	EvalPlus instruction prompt	pass@1 (EvalPlus)
GPQA-Diamond	test	198	0-shot, option rotation	Accuracy
C-EVAL	val	1,346	0-shot Chinese multiple choice	Accuracy
MMLU-Pro	test	12,032	0-shot CoT	Accuracy
TriviaQA	val	17,944	5-shot, short answer	EM (alias match)
PIQA	val	1,838	0-shot MCQ	Accuracy
OCNLI	val	2,950	0-shot Chinese NLI	Accuracy
DROP	val	9,536	3-shot (OpenAI simple-evals)	EM (fuzzy match)
BBH	test	6,511	3-shot CoT (official)	Accuracy
HellaSwag	val	10,042	0-shot MCQ	Accuracy
IFEval	official	541	raw prompt	Prompt-strict (Google checker)

Table S2: Effect of increasing τ_{t2m} on remask volume per strategy.

Strategy	Higher $\tau_{\mathrm{t2m}} \rightarrow$	Sweep range
LowProb	more remasking	$\{0.3, 0.5, 0.7, 0.9\}$
T2T-Remask	less remasking	$\{0.5, 0.7, 0.9\}$
LogitDiff	less remasking	$\{0.05, 0.1, 0.2, 0.3, 0.5\}$

Table S3: Highest accuracy of T2T editing and T2M over ten threshold settings on five full benchmarks ($16{,}384$-token generation budget). The ten settings are five M2T fill thresholds $\{0.5, \ldots, 0.9\}$ with the correction threshold at its default, and five correction thresholds $\{0.5, \ldots, 0.9\}$ (T2T edit threshold or T2M remask threshold) with the fill threshold at 0.7. Entries are accuracy @ NFE/token of the highest-accuracy setting.

Benchmark	Best tested T2T	Best tested T2M	Δ
GSM8K	90.30 @ 0.39	**92.04** @ 0.82	$+1.74$
HumanEval+	81.71 @ 0.19	**84.76** @ 0.37	$+3.05$
MBPP+	70.90 @ 0.23	**71.96** @ 0.50	$+1.06$
C-EVAL	77.86 @ 0.56	**79.12** @ 1.22	$+1.26$
PIQA	83.51 @ 0.71	**84.28** @ 1.93	$+0.77$

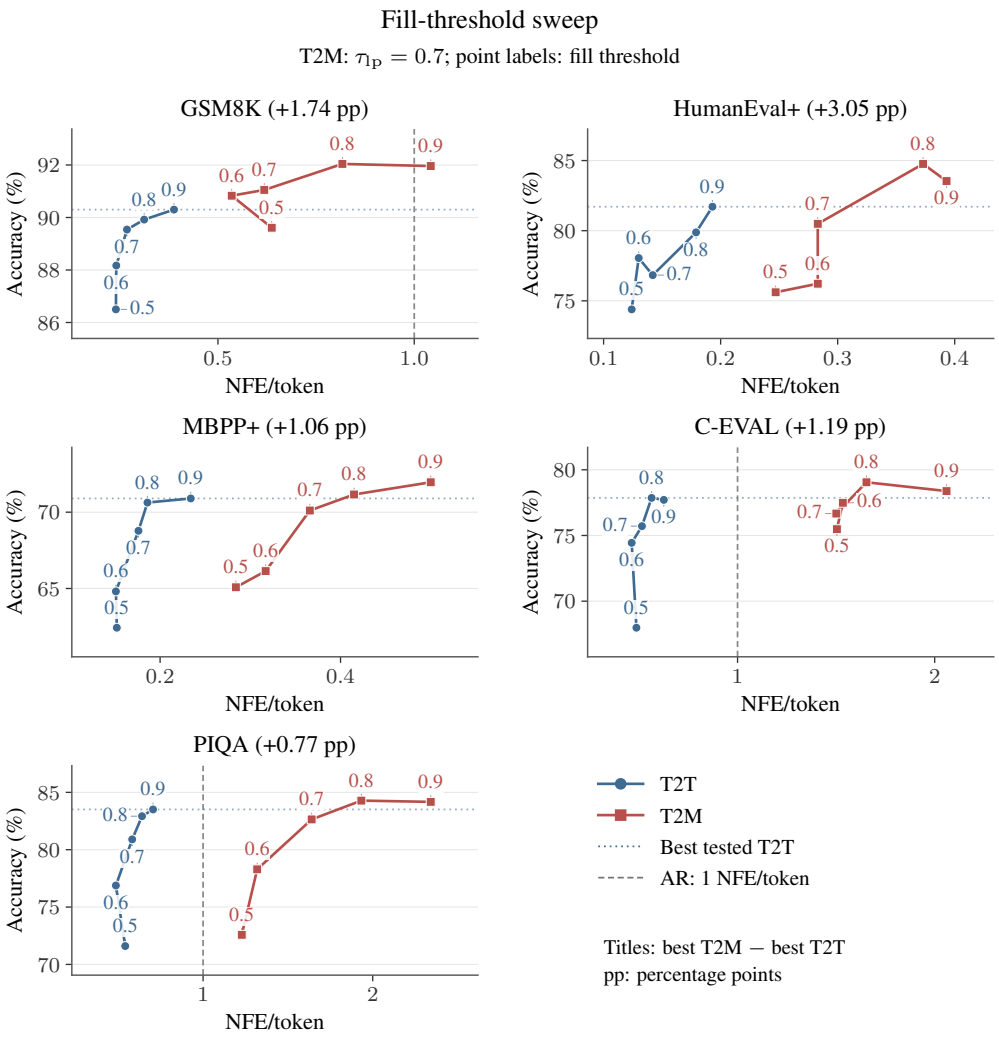


Figure S1: **Fill-threshold sweep on five benchmarks** (16,384-token generation budget). T2M uses τ_{1p}=0.7.

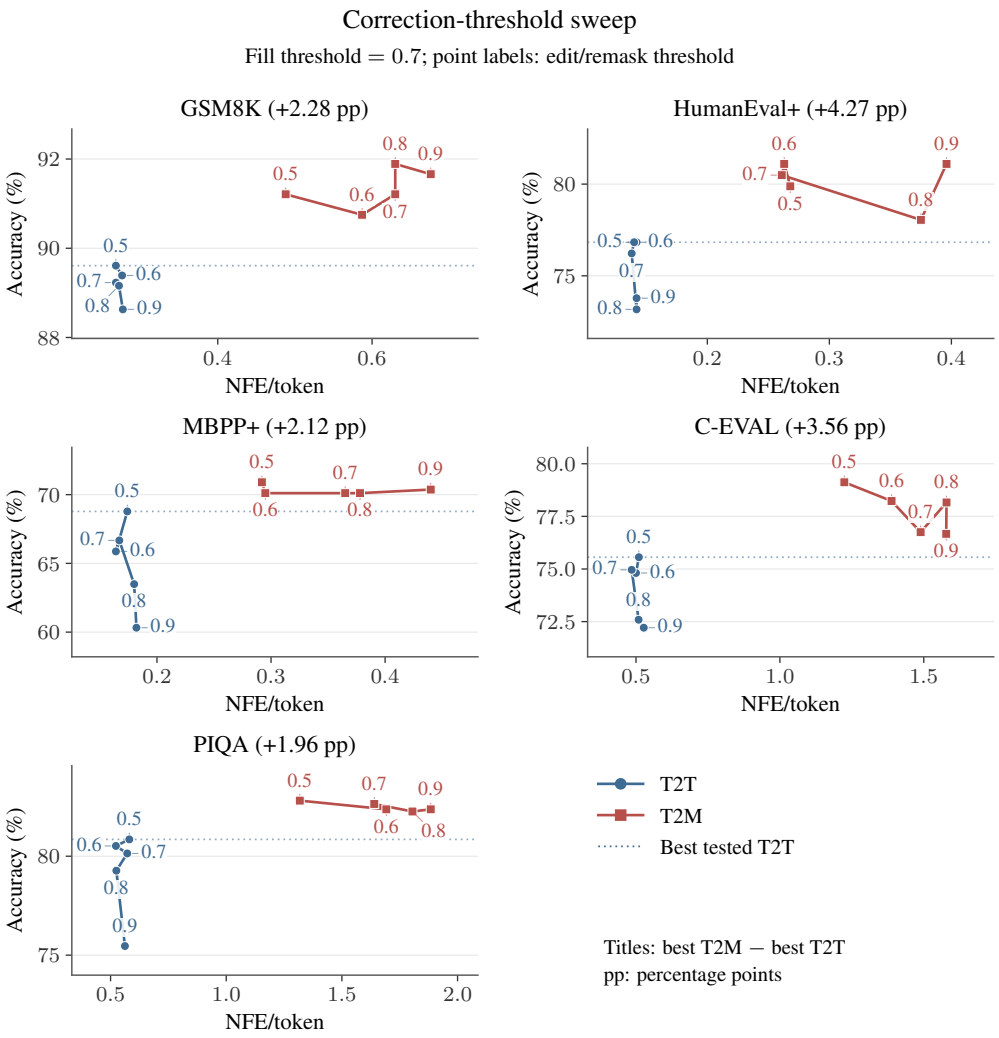


Figure S2: **Correction-threshold sweep on five benchmarks** (16,384-token generation budget). Fill fixed at 0.7.

E DETECTOR TASK-SPECIFICITY

LOGITDIFF and LOWPROB are not interchangeable across task types. Extending LOGITDIFF (τ_{ld}=0.1) beyond GSM8K to six further benchmarks at the same caps as LOWPROB ($C_{\max}$=3, $\rho_{\max}$=0.5) splits by task (Table S4): LOGITDIFF is higher on CMATH and GSM8K, the two tie within 0.02 on GSM-Plus, and LOWPROB is higher on all four non-math tasks. We therefore retain LOWPROB as the task-general default.

Table S4: LOWPROB vs. LOGITDIFF (τ_{ld}=0.1) at matched caps. GSM8K uses the ablation budget; other rows follow Table 1.

Task	LOWPROB	LOGITDIFF	Δ
CMATH	92.35	**93.44**	+1.09
GSM8K	91.13	**91.51**	+0.38
GSM-Plus	**81.79**	81.77	−0.02
GPQA	**45.96**	42.42	−3.54
IFEval	**84.47**	82.99	−1.48
HumanEval+	**80.49**	79.27	−1.22
MBPP+	**70.11**	68.78	−1.33

F TOKEN CHANGES AND ANSWER CORRECTIONS

Most remasks leave the token unchanged: under the τ_{lp}=0.9 configuration on GSM8K, only 1.2% of remasked positions refill with a different token. Under the default configuration, T2M corrects 56 GSM8K answers that the baseline gets wrong and introduces errors on 35 answers that the baseline gets right. HumanEval+ contributes 8 corrections and 2 new errors.

Repeated reopening matters despite this sparsity. Cutting the per-position budget to $C_{\max}$=1 reduces the GSM8K gain to +0.23 at 0.32 NFE/token, suggesting that some positions are refilled before their neighborhoods have stabilized and must be reopened later.

An immediate-refill control separates masking from cross-step deferral. With the same LOWPROB detector, immediate replacement reaches 89.39, immediate masked-context refill reaches 89.99, and deferred refill reaches 91.13. The adjacent differences are not significant (exact McNemar p=.51 and p=.13). Only 8.5% of immediate refills change the original token.

G LATENCY AND THROUGHPUT

Table S5 reports wall-clock latency and throughput under the paired protocols of Table 1 (A100-80GB, batch size 1).

H BASELINE REPRODUCTION QUALITY

All T2T baseline numbers in this paper are our own reproduction with the open-source SGLang dLLM stack under the protocols of Section 4.1. Table S6 compares them against the reported Q Mode values on the sixteen comparable benchmarks; GSM8K is not reported for LLaDA2.1-mini in the technical report. All T2T–T2M comparisons use our reproduction under matched engines, prompts, budgets, and scorers.

Table S5: Per-benchmark latency and throughput (LLaDA2.1-mini, A100-80GB, batch size 1), measured from per-request serving logs.

Benchmark	Latency (s) T2T	Latency (s) T2M	Tokens/s T2T	Tokens/s T2M
GSM8K	0.92	1.99	321	147
OlympiadBench	12.9	27.6	345	159
GSM-Plus	1.16	3.07	289	112
Omni-MATH	13.7	36.3	397	150
CMATH	1.09	2.44	279	121
HumanEval+	0.50	1.09	711	307
MBPP+	0.39	1.04	708	263
GPQA-Diamond	18.0	50.9	256	92
C-EVAL	5.2	11.3	168	72
MMLU-Pro	6.5	18.2	275	105
TriviaQA	0.18	0.23	53	40
PIQA	0.93	2.6	143	52
OCNLI	0.80	2.36	173	58
DROP	0.77	1.7	267	130
BBH	2.4	6.0	246	103
HellaSwag	1.4	4.0	174	63
IFEval	2.9	9.6	161	51

Table S6: Reproduced T2T vs. LLaDA2.1-mini Q Mode in the technical report (Bie et al., 2026). Positive Δ means our reproduction is higher.

Benchmark	Reported Q Mode	Ours (T2T)	Δ
HellaSwag	76.19	79.50	$+3.31$
DROP	82.37	84.14	$+1.77$
BBH	80.58	81.92	$+1.34$
IFEval (prompt-strict)	83.18	83.36	$+0.18$
OCNLI	61.59	60.78	-0.81
TriviaQA	54.24	53.18	-1.06
MMLU-Pro	64.84	63.41	-1.43
CMATH	94.99	92.26	-2.73
C-EVAL	78.59	75.63	-2.96
Omni-MATH	43.60	40.31	-3.29
GSM-Plus (full)	86.55	81.37	-5.18
MBPP+	74.07	68.78	-5.29
PIQA	86.89	80.90	-5.99
HumanEval+	82.93	76.83	-6.10
OlympiadBench	66.67	58.16	-8.51
GPQA-Diamond	53.28	43.94	-9.34