# GUIDING DISCRETE DIFFUSION WITH DISTILLED PROCESS REWARDS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Masked diffusion language models such as LLaDA generate text by iteratively committing tokens from a fully masked canvas. An erroneous early commitment can influence subsequent denoising steps and lead to an incorrect final answer. Process reward models (PRMs) can localize these errors, but running a separate 7B-parameter PRM at inference is expensive. We present a method for distilling process rewards into a frozen diffusion language model. A value head summarizes the quality of a partially masked state and supports post-hoc repair; a step-issue head predicts token-level issue scores and guides commitment order during the original denoising loop. We evaluate a single 256-token canvas to isolate adaptive commitment ordering from fixed block boundaries: lower-risk positions can be committed in parallel, while higher-risk positions wait for additional context. In one reported GSM8K run with LLaDA-8B-Instruct, step-issue guidance changes accuracy from **44.7% to 53.9% (+9.2 pp)** with a 1.1M-parameter MLP. Matched controls in that run do not reproduce this change with random or shuffled issue scores. On MATH-500, the corresponding GSM8K-to-MATH-500 transfer run changes accuracy from **24.0% to 25.8%**. Direct PRM repair reaches a higher point estimate but requires the full PRM at inference and an additional regeneration pass. In a matched 100-example GSM8K timing run, guidance was only 0.4% slower than baseline, versus 23.6% for distilled repair and 35.1% for direct PRM repair.

## 1 INTRODUCTION

Discrete diffusion language models (Nie et al., 2025; Arriola et al., 2025) generate text by iteratively revealing tokens from a masked canvas. Unlike left-to-right decoding, this process exposes a useful control variable: the order in which positions are committed. A token committed before its context has stabilized can influence subsequent denoising decisions, whereas delaying that position gives the model an opportunity to revise its prediction using a more complete canvas. Standard low-confidence remasking uses model confidence to choose this order, but confidence alone need not identify positions that belong to an incorrect reasoning step. Fixed block diffusion further imposes a predefined dependency structure: tokens are parallel within a block but blocks are generated sequentially (Arriola et al., 2025). Those boundaries need not match reasoning dependencies; ideally, a decoder should commit context-independent positions in parallel and defer positions that depend on unresolved intermediate results.

Process reward models (PRMs) (Lightman et al., 2023; Zhang et al., 2025b) provide step-level supervision for mathematical reasoning. Our question is whether this supervision can be transferred into a frozen diffusion language model and used during denoising without running the PRM at inference. The difficulty is that a conventional PRM scores completed reasoning steps, while the diffusion model operates on partially masked canvases. We bridge the two representations during training: complete sampled diffusion states, score the resulting reasoning steps with a PRM, align each step score back to its canvas-token span, and learn lightweight predictors from the corresponding frozen LLaDA hidden states.

Our primary method is a token-level *step-issue head* that predicts positions associated with low-scoring reasoning steps. During the original diffusion loop, we subtract a scaled issue score from each token's commitment priority, delaying suspicious positions until more context is revealed. The



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.

head does not change backbone weights or current-state logits; it changes commitment order, which can change later canvases and predictions. This in-loop intervention requires neither the PRM nor a separate regeneration pass. A state-level *value head* supplies a distilled post-hoc repair baseline for comparing repair quality and inference cost.

> **Question.** Betty and Sam have popsicles in ratio $5{:}6$ and $165$ altogether. How many more does Sam have?
> **Default completion.** "Betty has $75$; Sam has $90$ ... $95 - 75 = 20$."
> **Failure.** The correct final operation is $90 - 75 = 15$; the default completion commits a locally plausible but corrupted subtraction.

This inconsistency motivates guidance; the final output alone does not identify which commitment decisions caused it.

This design differs from denoising-process-reward methods that fine-tune the diffusion backbone with reinforcement learning (Xie et al., 2026). Those methods modify the model's learned distribution; our method freezes the backbone and distills an external PRM into small inference-time heads. The two approaches are complementary: a process-reward-trained model could also serve as the frozen backbone for our guidance policy.

Our contributions are twofold:

1. We introduce PRM-distilled *in-loop token guidance* for masked diffusion decoding: completed-text PRM feedback is aligned to masked-canvas token spans and distilled into a lightweight head over frozen hidden states. The head guides commitment order without changing backbone weights or current-state logits.
2. We evaluate whether learned localization is associated with the observed change beyond an arbitrary commitment-order perturbation. In one reported GSM8K run, guidance changes accuracy from $44.7\%$ to $53.9\%$; constant, hidden-norm, shuffled, and random-score controls do not reproduce that point estimate. We further compare this no-PRM-inference policy with direct PRM repair and distilled repair to expose the quality–cost trade-off. In a matched recorded GSM8K run, guidance was **0.4% slower** than baseline, compared with **23.6%** for distilled repair and **35.1%** for direct PRM repair.

## 2 Related Work

**Masked diffusion language models.** LLaDA (Nie et al., 2025) and MDLM (Sahoo et al., 2024) formulate text generation as iterative unmasking from a fully-masked canvas. Fast-dLLM (Wu et al., 2025) accelerates generation via block-wise approximate KV caching and confidence-aware parallel decoding. These models achieve competitive accuracy but inherit the error-propagation problem of iterative decoding: an erroneous early commitment can affect subsequent predictions at other positions.

**Process reward models.** PRMs (Lightman et al., 2023; Zhang et al., 2025b) score individual reasoning steps for correctness. Qwen2.5-Math-PRM-7B (Qwen Team, 2025) provides step-level feedback for mathematical reasoning. One forward pass can score multiple steps in a completion; repeated evaluations during diffusion still add inference cost.

**Corrective and guidance methods for diffusion LLMs.** RFG (Chen et al., 2025) guides decoding using enhanced and reference model likelihoods. MRO (Wang et al., 2025) optimizes multiple rewards, while reward-guided stitching (Miles et al., 2026) combines PRM-selected steps from diffusion trajectories and uses an autoregressive solver. Corrective diffusion (Zhang et al., 2025a) and self-correcting masked diffusion (Schiff et al., 2026) learn to revise generated errors. BackPlay (Liu et al., 2026) is particularly close: it trains a head with a frozen generator and revisits generated tokens through selective remasking. Our distinction is external PRM supervision projected onto masked canvases and guidance of still-masked positions before commitment.

Denoising process rewards (Xie et al., 2026) estimate contributions of denoising intervals to final outcomes and fine-tune the backbone with RL. Our method retains the frozen backbone and distills

completed-text PRM feedback into a commitment policy. These approaches are complementary: a process-reward-trained model could serve as the backbone for our guidance.

# 3 METHOD

## 3.1 MASKED DIFFUSION DECODING FORMULATION

LLaDA generates text by starting from a fully-masked sequence of length $L$ and iteratively unmasking tokens over $T$ denoising steps. At each step $t$, the model predicts a distribution over tokens for every masked position. A remasking strategy (e.g., low-confidence remasking) selects which positions to commit and which to keep masked. After $T$ steps, all positions are unmasked, yielding the final generation.

Let $\mathbf{x}^{(t)} \in \mathcal{V}^L$ denote the canvas at step $t$, where masked positions hold a special mask token. The model computes hidden states $\mathbf{h}^{(t)} = f_\theta(\mathbf{x}^{(t)})$ and predicts token distributions $p_\theta(v \mid \mathbf{x}^{(t)})$ for each masked position. Under low-confidence remasking, the model commits tokens with the highest predicted probability $x_0^{(p)} = \max_v p_\theta(v \mid \mathbf{x}^{(t)})_p$ at each step, where $p$ indexes positions. This confidence score $x_0^{(p)}$ determines the order in which tokens are revealed.

## 3.2 PRM SCORING AND DISTILLATION DATA

The PRM (Qwen2.5-Math-PRM-7B) scores completed reasoning steps, but diffusion generates partially-masked intermediate states. To bridge this gap, we construct training data by *completing* intermediate states and scoring them with the PRM.

**Trajectory collection.** We run LLaDA on GSM8K training problems with trajectory logging, recording the canvas $\mathbf{x}^{(t)}$ and mask pattern at each denoising step. For each trajectory, we sample states at denoising fractions $f \in \{0.25, 0.5, 0.75\}$.

**Fill and score.** For each sampled state, we complete the masked positions to obtain a full fill. The greedy fill uses the logged argmax token at every masked position; the three stochastic fills independently sample each masked position from the saved normalized top-$k$ distribution ($k = 5$). We decode only the generated suffix (the prompt is retained separately) and segment it at blank-line boundaries, discarding empty or whitespace-only segments. The PRM receives the original question and this generated completion in the Qwen Math-PRM prompt template and returns one score per segment. Scores are already probabilities in $[0, 1]$; we do not threshold or otherwise calibrate them during distillation. A fill is considered malformed if the number of PRM scores differs from the number of nonempty segments; such a fill is rejected rather than assigned an ambiguous label.

To align a score to the masked canvas, we decode every prefix of the exact fill token sequence and find the first token prefix whose decoded character length reaches each segment boundary. This yields half-open token spans $[a_j, b_j)$ in fill-token coordinates. We then add the prompt-token offset $P$ to obtain canvas positions $[P + a_j, P + b_j)$, including positions that were masked in the sampled state. Every canvas token in span $j$ receives the soft issue target

$$y_i = 1 - s_j, \qquad i \in [P + a_j, P + b_j), \tag{1}$$

where $s_j \in [0, 1]$ is the PRM score for segment $j$. Thus, a masked position is labeled using the PRM score of the step containing the token it would receive in the completed fill; it is not scored by the PRM while still masked. Targets for the four fills are averaged at overlapping canvas positions, and positions outside a scored span are excluded from the issue-head loss.

**State-level targets.** In addition to per-token issue targets, we record state-level PRM scores for each sampled state: the greedy-fill PRM score, the mean PRM score across fills, and the minimum PRM score. Only the mean fill-level PRM score supervises the value head (Section 3.3), while the per-token issue targets supervise the step-issue head (Section 3.4). Further corpus construction and span-alignment details appear in Appendix D; the data-creation and head-training steps are summarized in Appendix I.

### 3.3 VALUE HEAD: STATE-LEVEL REPAIR

The value head predicts a scalar quality score for an intermediate diffusion state, enabling post-hoc repair: low predicted PRM quality triggers a search for a suffix boundary, followed by regeneration if the advantage test passes. The corresponding offline-training and inference flow is shown in Figure 1.

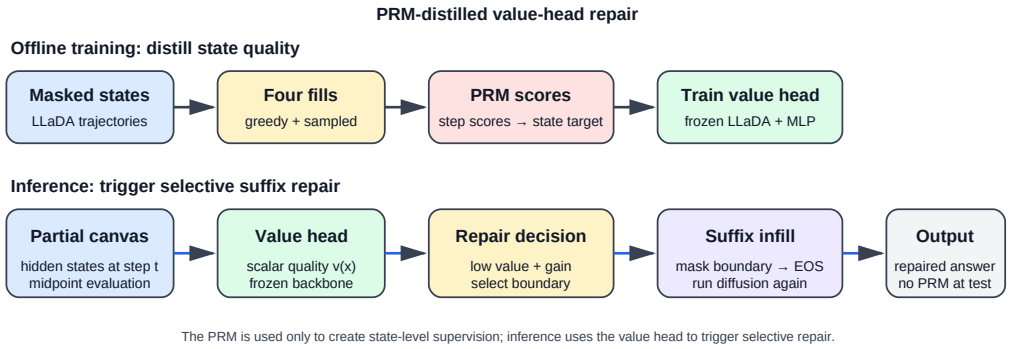


Figure 1: Value-head repair. State-level PRM scores supervise a lightweight value head offline; at inference, low-value intermediate states trigger a suffix-regeneration pass without running the PRM.

**Architecture.** The value head $g_\phi^{\text{val}}$ pools the per-token hidden states into a single state representation and combines it with timestep features $\mathbf{c}^{(t)} = [\text{denoising_fraction}, \text{mask_ratio}]$:

$$v(\mathbf{x}^{(t)}) = g_\phi^{\text{val}}\left(\text{pool}(\mathbf{h}^{(t)}),\ \mathbf{c}^{(t)}\right), \tag{2}$$

where pool is the attention-mask-weighted arithmetic mean over all non-padding positions in the serialized prompt-plus-canvas sequence. Prompt, revealed, and masked-token hidden states therefore all contribute; only padding positions are excluded. If a sequence has no non-padding positions, the pooled vector is defined as zero. The timestep embedding maps $\mathbf{c}^{(t)}$ to a 64-dimensional vector via two linear layers with SiLU activation. The combined representation passes through a 256-unit hidden layer with SiLU activation to produce a scalar prediction. For LLaDA-8B (hidden_size $= 4096$), the head has $\sim$1.1M parameters.

**Training.** The value head is trained with mean-squared error against state-level targets derived from the PRM distillation data (Section 3.2). The training target is the mean PRM score across fills, $\bar{r} = \frac{1}{4}\sum_{j=1}^{4} \text{PRM}(\text{fill}_j)$:

$$\mathcal{L}_{\text{val}} = \left(v(\mathbf{x}^{(t)}) - \bar{r}\right)^2. \tag{3}$$

The LLaDA backbone is frozen; only the head parameters $\phi$ are updated. Training uses AdamW with learning rate $10^{-4}$ for 2 epochs.

**Inference: PRM-distilled repair.** At inference, the value head scores the midpoint masked state. We use $\tau_{\text{val}} = 0.5$ and $\tau_{\text{adv}} = 0$ (the latter is the default in our evaluation wrapper). If the midpoint value is below $\tau_{\text{val}}$, candidates are revealed generated positions whose decoded token contains a digit. Each candidate is temporarily masked and rescored; we select the candidate with the largest $\Delta v_p = v_p - v_{\text{base}}$. If $\Delta v_p > \tau_{\text{adv}}$, we remask from that position through EOS and run one deterministic 64-step, block-size-256 infill pass. The suffix begins at the digit-bearing position with the largest predicted value increase; no additional candidate is sampled. Algorithm 5 gives the complete procedure.

### 3.4 STEP-ISSUE HEAD: TOKEN-LEVEL GUIDANCE

While the value head operates at the state level and triggers post-hoc repair, the step-issue head operates at the token level and guides the diffusion process *as it unfolds*—no regeneration pass is needed.

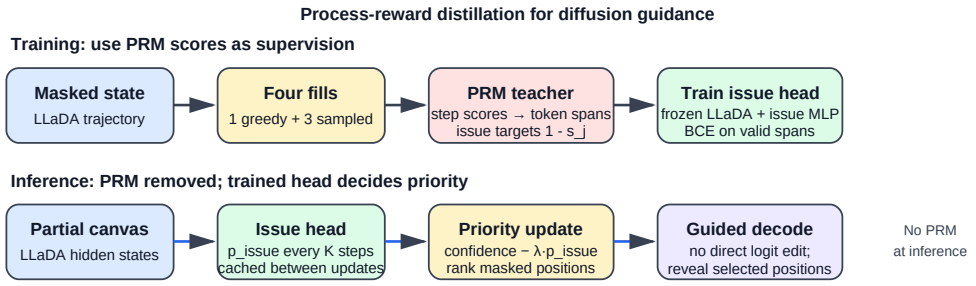


Figure 2: Step-issue guidance. Offline, completed fills of a masked canvas receive PRM step scores that are aligned to canvas-token spans and distilled into the issue head. At inference, the lightweight head runs alongside the frozen backbone; no PRM inference is required. It adjusts commitment priority without directly modifying current-state logits.

**Architecture.** The step-issue head $g_\phi^{\text{issue}}$ predicts a per-token issue score from the frozen hidden state at each position, combined with the same timestep features $\mathbf{c}^{(t)}$:

$$p_i^{\text{issue}} = \sigma\left(g_\phi^{\text{issue}}\left(\mathbf{h}_i^{(t)},\ \mathbf{c}^{(t)}\right)\right), \tag{4}$$

where $\sigma$ is the sigmoid function. Unlike the value head, which pools across positions, the step-issue head operates independently at each token position. The timestep embedding is broadcast across all positions and concatenated with the per-token hidden state before a 256-unit hidden layer with SiLU activation. The architecture is identical in parameter count to the value head ($\sim$1.1M parameters). Additional architecture details for both heads appear in Appendix C.

**Training.** The head is trained with binary cross-entropy against the per-token issue targets from Equation 1:

$$\mathcal{L}_{\text{issue}} = \frac{1}{|\mathcal{M}|} \sum_{i \in \mathcal{M}} \text{BCE}\left(p_i^{\text{issue}}, y_i\right), \tag{5}$$

where $\mathcal{M}$ is the set of tokens with valid PRM-projected targets (i.e., tokens that fall within a scored reasoning step). The LLaDA backbone is frozen; only the head parameters $\phi$ are updated. Training uses AdamW with learning rate $10^{-4}$ for 2 epochs.

**Inference: guidance during denoising.** At inference, the distilled issue head guides the diffusion process by adjusting the commitment priority of suspicious tokens. Recall that under low-confidence remasking, the model commits tokens with the highest predicted probability $x_0^{(p)}$ at each step. We lower the commitment priority of tokens predicted to be problematic:

$$\tilde{x}_0^{(p)} = x_0^{(p)} - \lambda \cdot p_p^{\text{issue}}, \tag{6}$$

where $\lambda$ is the *guidance strength* and $p_p^{\text{issue}}$ is the issue score at position $p$. This lowers the priority of suspicious tokens, delaying their commitment until the surrounding context is more settled. The head does not add a term to the logits computed at a given canvas state; it only changes the priority used to select positions for commitment. Because that selection changes subsequent canvases, later hidden states and logits can differ from the unguided trajectory. The intervention therefore directly changes *when* positions are committed, while leaving the token probabilities at each evaluated state unmodified. A formal distinction between these fixed-state conditionals and the induced trajectory distribution appears in Appendix B.

**Efficiency.** The issue head is evaluated every $K$ denoising steps (we use $K = 16$), using hidden states from the backbone forward pass at that step. The resulting per-token issue scores are cached and reused until the next guidance update. This adds one MLP evaluation at an update step, in contrast to running a 7B PRM forward pass per denoising step.

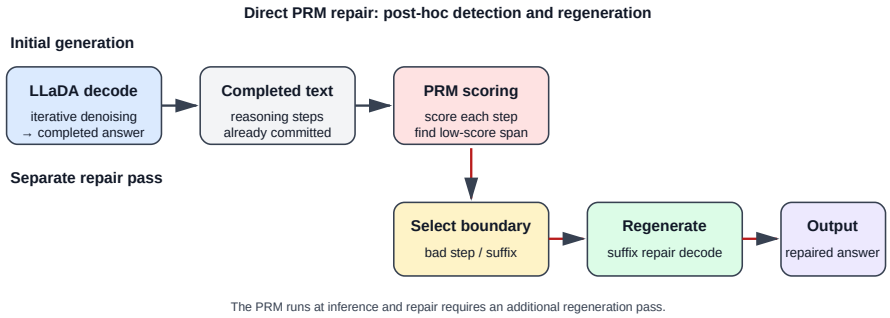


Figure 3: Direct PRM repair baseline. The PRM scores completed text and a separate regeneration pass repairs a selected suffix.

**Entire diffusion process.** The guidance operates within the standard diffusion loop from start to finish—no post-hoc regeneration or separate decoding pass is used. We use one 256-token block to isolate learned commitment ordering from externally imposed left-to-right block boundaries. Low-issue positions may be committed in parallel, while high-issue positions remain masked for additional context. The issue score is a commitment-risk proxy, not an explicit dependency estimate.

# 4 Experiments

## 4.1 Setup

**Model.** We use LLaDA-8B-Instruct (Nie et al., 2025) in bfloat16 with 64 diffusion steps and low-confidence remasking. Our main GSM8K and MATH-500 experiments use a single 256-token block with 256 maximum new tokens, so all response positions share one denoising process. The supplementary ASDiv evaluation uses 128-token blocks, as detailed in Appendix H. The PRM is Qwen2.5-Math-PRM-7B (Qwen Team, 2025), used for training-data generation and the direct PRM-repair baseline at inference.

All experiments were run on a single NVIDIA A6000 GPU with 48 GB of memory.

**Datasets.** We evaluate on GSM8K (Cobbe et al., 2021) (1,319 test examples) via lm-eval-harness with flexible-extract exact-match scoring, and on MATH-500 (Lightman et al., 2023; Hugging Face H4, n.d.) (500 test examples) with math-verify scoring.

**Methods.** We compare:

- **No guidance (baseline)**: Standard LLaDA denoising without any intervention.
- **Step-issue guidance**: LLaDA denoising with the distilled issue head guiding token commitment order (Equation 6).
- **PRM repair**: Post-hoc repair using the full Qwen2.5-Math-PRM-7B. We flag an output when the minimum PRM step score is below $\tau_{\text{PRM}} = 0.5$, choose the first such step, and remask its corresponding suffix through EOS before one deterministic infill pass. The corresponding direct-repair workflow is shown in Figure 3; implementation details appear in Appendix K, including Algorithm 4.
- **PRM-distilled repair**: Post-hoc repair using the distilled value head to decide whether to repair and select a suffix boundary.

**Training data.** We generate one PRM-distillation corpus from logged LLaDA trajectories on GSM8K training problems. For each trajectory we retain three masked states at denoising fractions $f \in \{0.25, 0.5, 0.75\}$, as summarized in Algorithm 1. At each state we generate four fills (one greedy and three samples from the logged top-$k$ distribution) and score each completed fill with Qwen2.5-Math-PRM-7B. The PRM returns a step-level score for each reasoning

step, which we project onto aligned token spans to obtain per-token issue targets $y_i = 1 - \text{PRM}(\text{step containing token } i)$ (Equation 1), averaged across fills. Each row also stores the greedy, mean, minimum, and standard deviation of fill-level PRM scores. The corpus used for the reported heads contains 9,000 state rows (three fractions per source trajectory); the issue head uses these same rows with variable-length span-aligned token labels, not a separate state dataset. Per-token targets supervise the step-issue head, and the mean fill score supervises the value head.

**Head training.** Both heads use frozen LLaDA-8B hidden states (dimension 4096) extracted at the logged denoising steps. Algorithms 2 and 3 summarize training. Each head takes a hidden state concatenated with a timestep embedding $\mathbf{c}^{(t)} = [\text{denoising_fraction}, \text{mask_ratio}]$ (mapped to 64 dimensions via two linear layers with SiLU activation) and passes the combined representation through a 256-unit hidden layer with SiLU activation ($\sim$1.1M parameters per head). The value head applies a attention-mask-weighted mean over all non-padding prompt and canvas positions before the MLP to produce a single scalar; the step-issue head operates independently at each token position to produce a per-token issue score. Training uses AdamW (learning rate $10^{-4}$, batch size 2) for 2 epochs. Greedy and evaluation decoding use temperature 0; stochastic distillation fills sample from the logged $k = 5$ top-token categorical distributions with seed 42 and deterministic per-example derived seeds. The value head is trained with mean-squared error against the mean PRM score across fills (Equation 3); the step-issue head is trained with BCE against per-token issue targets (Equation 5), masking tokens outside scored reasoning-step spans. At inference, the step-issue head uses the fixed operating point $\lambda = 0.75$ and is evaluated every $K = 16$ denoising steps; these values were manually fixed rather than validation-selected. Scores are cached between updates. The value head scores the midpoint state and triggers suffix regeneration when the predicted value falls below $\tau_{\text{val}}$ and a counterfactual advantage threshold is met (Section 3.3).

### 4.2 Main Results

Table 1 compares all methods on GSM8K and MATH-500. These results report one evaluation run per configuration. On GSM8K, step-issue guidance increases flexible accuracy from $44.7\%$ to $53.9\%$ ($+9.2$ pp, using $\lambda = 0.75$ and $K = 16$), operating entirely within the diffusion loop with no additional backbone or PRM forward pass. PRM repair achieves the highest accuracy ($63.5\%$) but requires running the full 7B PRM at inference plus a separate regeneration pass for $52\%$ of examples. PRM-distilled repair ($48.1\%$) improves over the baseline but remains below the reported guidance setting, while requiring a regeneration pass for $28\%$ of examples.

MATH-500 is a cross-dataset transfer evaluation: neither head is trained or tuned on MATH-500; both are trained only from GSM8K trajectories. On MATH-500, the same pattern holds: PRM repair ($29.8\%$) substantially exceeds the baseline ($24.0\%$), while step-issue guidance ($25.8\%$) and PRM-distilled repair ($24.8\%$) stay close to the baseline. The guidance gain is smaller on MATH-500. This may reflect a mismatch between the GSM8K-derived supervision and the MATH-500 distribution, but we do not measure uncertainty or token-level transfer well enough to identify the cause. A separate ASDiv-CoT-LLaMA evaluation is reported in Appendix H; it uses the same frozen heads without in-domain training.

Table 2 reports the control results; definitions appear in Appendix G. Constant subtraction preserves commitment ranking. The control labeled *confidence* in the implementation uses a hidden-state norm proxy, not token probabilities. Shuffling disrupts score-to-position alignment, while random scores perturb priorities. GSM8K accuracy is $44.73\%$ for constant and norm-proxy controls, $44.66\%$ for shuffled scores, and $46.70\%$ for random scores, versus $53.90\%$ for learned guidance. These comparisons support useful learned score assignments, but do not isolate localization from every other score property.

A descriptive guidance-strength and update-interval sweep is provided in Appendix E; the reported operating point ($\lambda = 0.75$, $K = 16$) was manually fixed rather than selected on a validation sweep.

### 4.3 Token-Level Distillation Fidelity

This diagnostic measures how closely the issue head reproduces projected PRM targets within a canvas. We retain only token positions with valid aligned targets and compute Spearman correlation between the issue predictions and teacher targets separately for each canvas. Across 20 evaluated

Table 1: Main results at 64 denoising steps with one 256-token block. Accuracy is flexible-extract exact match on GSM8K and math-verify on MATH-500. "Extra decode" denotes a suffix-regeneration pass; "PRM at inference" denotes use of Qwen2.5-Math-PRM-7B during evaluation.

| Method | Accuracy | PRM at inference | Extra decode | Repair rate |
|---|---|---|---|---|
| *GSM8K* | | | | |
| No guidance (baseline) | 44.73% | × | × | — |
| **Step-issue guidance** | **53.90%** | × | × | — |
| PRM-distilled repair | 48.07% | × | ✓ | 27.9% |
| **PRM repair** | **63.53%** | ✓ | ✓ | 52.3% |
| *MATH-500* | | | | |
| No guidance (baseline) | 24.0% | × | × | — |
| **Step-issue guidance** | **25.8%** | × | × | — |
| PRM-distilled repair | 24.8% | × | ✓ | 56.4% |
| **PRM repair** | **29.8%** | ✓ | ✓ | 67.0% |

Table 2: Commitment-score controls. Accuracy (%) uses the metrics in Table 1.

| Score source | GSM8K | MATH-500 |
|---|---|---|
| Baseline | 44.73 | 24.0 |
| Constant | 44.73 | 24.0 |
| Hidden-norm proxy | 44.73 | 24.0 |
| Shuffled issue scores | 44.66 | 20.8 |
| Random scores | 46.70 | 21.2 |
| Learned issue scores | 53.90 | 25.8 |

canvases, the mean correlation is $0.74$ and the median is $0.79$. Separation from the training data was not verified, so this is a descriptive diagnostic, not evidence of held-out generalization. The targets are projected PRM labels rather than independently annotated token errors; agreement with them does not by itself imply higher final-answer accuracy.

### 4.4 Value-Head Fidelity Across Denoising Stages

This separate diagnostic evaluates the scalar value head against the mean fill-level PRM target, not an average of token-level issue predictions. It comprises 1,350 states, with 150 states at each of nine fractions $f \in \{0.1, 0.2, \ldots, 0.9\}$. This evaluation grid is distinct from the training fractions $\{0.25, 0.5, 0.75\}$ and does not change the training-corpus count. Training–evaluation independence is not established by the available summary, so we do not interpret it as held-out generalization.

At each fixed fraction, Spearman correlation across states ranges from $0.38$ to $0.69$ (Appendix J). These within-fraction results are primary: the pooled Spearman ($0.866$) also reflects systematic score differences across denoising stages. As an auxiliary diagnostic, we use the value prediction to rank correct versus incorrect final answers. The pooled correctness AUC ($0.604$) indicates limited discrimination; teacher fidelity does not establish reliable final-answer verification.

## 5 Analysis

**Why guidance can help.** The issue head lowers commitment priority for positions aligned with teacher-derived issue targets. Delaying them exposes more context before commitment and can change later canvases and backbone predictions; it does not alter logits at the current canvas. The alignment result, matched controls, and cost comparison together support learned localization as a decoding policy, though they do not establish universal accuracy improvement.

**Guidance versus repair.** Guidance changes commitment order within the original diffusion loop, whereas repair regenerates a suffix. PRM repair obtains $63.5\%$ on GSM8K and $29.8\%$ on MATH-500, but regenerates $52\%$ and $67\%$ of examples and runs the 7B PRM. Distilled repair regenerates

Table 3: Matched GSM8K compute-cost comparison (100 completions; 64 steps; 256-token block; seed 0). Counts and timings come from one recorded run on the same device; they are illustrative rather than hardware-normalized throughput measurements.

| Method | Repair rate | Backbone calls | Issue calls | Value calls | Time (s) / overhead |
|---|---|---|---|---|---|
| Baseline | 0% | 6,400 | 0 | 0 | 564.56 / — |
| Step guidance | 0% | 6,400 | 400 | 0 | 566.85 / +0.4% |
| Distilled value repair | 27% | 7,545 | 0 | 611 | 697.73 / +23.6% |
| Direct PRM repair | 43% | 8,580 | 0 | 0 | 762.41 / +35.1% |

28–56% of examples. Table 3 reports the matched 100-example timing comparison; additional cost details and paired generations appear in Appendices F and A.

**Fidelity.** For the value head, fixed-fraction correlations (0.38–0.69) are the primary fidelity measure. The pooled Spearman (0.866) partly reflects denoising fraction, and the correctness AUC (0.604) is weak; teacher alignment does not establish reliable prediction of final correctness.

## 6 Limitations

All accuracy comparisons use one evaluation run. Projected PRM labels are only a proxy for intermediate-state correctness, and the heads are trained on GSM8K then evaluated on MATH-500 without adaptation. Our main experiments use one 256-token block; multi-block decoding may change the baseline and guidance effect. Guidance does not revise already committed tokens, motivating stronger reward models and selective repair.

## 7 Conclusion

We distill process-reward signals into a 1.1M token-level issue head over a frozen diffusion LLM. The head changes commitment order within the original diffusion loop, delaying positions associated with low-scoring reasoning steps without modifying the backbone weights or current-canvas logits. This provides a lightweight alternative to post-hoc repair: it uses neither PRM inference nor a separate regeneration pass.

In the reported run, guidance increases GSM8K accuracy from 44.7% to 53.9% and transfers from GSM8K training to MATH-500 (24.0% to 25.8%). PRM repair is more accurate (63.5% on GSM8K and 29.8% on MATH-500), but requires PRM inference and suffix regeneration. Future work includes multi-block decoding and selective combinations of guidance and repair.

### AI use statement

(This section is **required** and does not count toward the page limit.)

In this work, we used generative AI tools for code generation and experiment scripting. We have not used generative AI tools for writing the core scientific text or analysis, and the rest of the required disclosure tasks are not applicable to this work. Additionally, we used generative AI tools for formatting and LaTeX assistance. We have reviewed all AI-assisted work. We checked AI-generated code for correctness by running all experiments and verifying outputs. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

### Reproducibility statement

The implementation and trained head checkpoints will not be publicly released. To support reproducibility, the appendices provide detailed pseudocode for data creation, training both heads, repair, and guidance, alongside architecture, hyperparameter, and decoding details. The experiment uses LLaDA-8B-Instruct (publicly available) and Qwen2.5-Math-PRM-7B (publicly available). Key hyperparameters: 64 diffusion steps, 256 max new tokens, block size 256, low-confidence

remasking, guidance strength $\lambda = 0.75$, guidance interval $K = 16$. Training uses AdamW with learning rate $10^{-4}$ for 2 epochs on frozen LLaDA hidden states. See Sections 3–4 for full details.

## REFERENCES


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

Tianlang Chen, Minkai Xu, Jure Leskovec, and Stefano Ermon. Rfg: Test-time scaling for diffusion large language model reasoning with reward-free guidance. *arXiv preprint arXiv:2509.25604*, 2025.

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.

Hugging Face H4. MATH-500. `https://huggingface.co/datasets/HuggingFaceH4/MATH-500`, n.d. Dataset.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

Liming Liu, Binxuan Huang, Zixuan Zhang, Xin Liu, Bing Yin, and Tuo Zhao. Backplay: Head-only look-back self-correction for diffusion language models. *arXiv preprint arXiv:2601.06428*, 2026.

Shen-yun Miao, Chao-Chun Liang, and Keh-Yih Su. A diverse corpus for evaluating and developing English math word problem solvers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 975–984. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.acl-main.92. URL `https://aclanthology.org/2020.acl-main.92/`.

Roy Miles, Aysim Toker, Andreea-Maria Oncescu, Songcen Xu, Jiankang Deng, and Ismail Elezi. Test-time scaling with diffusion language models via reward-guided stitching. *arXiv preprint arXiv:2602.22871*, 2026.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. In *Advances in Neural Information Processing Systems*, 2025.

Qwen Team. Qwen2.5-math-prm-7b. `https://huggingface.co/Qwen/Qwen2.5-Math-PRM-7B`, 2025.

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

Yair Schiff, Omer Belhasin, Roy Uziel, Guanghan Wang, Marianne Arriola, Gilad Turok, Ran Zilberstein, Michael Elad, and Volodymyr Kuleshov. Learn from your mistakes: Self-correcting masked diffusion models. *arXiv preprint arXiv:2602.11590*, 2026.

Chenglong Wang, Yang Gan, Hang Zhou, Chi Hu, Yongyu Mu, Kai Song, Murun Yang, Bei Li, Chunliang Zhang, Tongran Liu, Jingbo Zhu, Zhengtao Yu, and Tong Xiao. Mro: Enhancing reasoning in diffusion language models via multi-reward optimization. In *Advances in Neural Information Processing Systems*, 2025.

Chengyue Wu, Hao Zhang, Shuchen Xue, Zhijian Liu, Shizhe Diao, Ligeng Zhu, Ping Luo, Song Han, and Enze Xie. Fast-dllm: Training-free acceleration of diffusion llm by enabling kv cache and parallel decoding. *arXiv preprint arXiv:2505.22618*, 2025.

Table 4: Selected response excerpts at $\lambda = 0.75$, $K = 16$. Reference answers follow the abbreviated questions. No missing intermediate calculations have been reconstructed.

| Question (reference) | Baseline excerpt | Step-guidance excerpt |
|---|---|---|
| Sofie, Anne, and Fawn have 85 books. Sofie has 25 more than Anne; Anne has 12 fewer than Fawn. Fawn's count? (28) | $F + F - 12 + F + 13 = 85$<br>$3F - 23 = 85$<br>...<br>So, Fawn has $\boxed{36}$ books. | $F + F - 12 + F + 13 = 85$<br>$3F + 1 = 85$<br>...<br>So, Fawn has 28 books. |
| Jim has 27 shells, five more than Carlos; Carlos has twice Carrey's count. Equal share among the three? (20) | $27 + 22 + 11 = 63$<br>...<br>$\frac{63}{3} = 21$<br>...<br>Therefore, each person got $\boxed{21}$ seashells. | Total seashells = $27 + 22 + 11 = 60$<br>...<br>Each person gets $\frac{60}{3} = 20$<br>...<br>Therefore, each person got $\boxed{20}$ seashells. |
| Charlie starts with 10 stickers, receives 21 and 23, gives away 9, and uses 28. Remaining? (17) | Charlie has 7 stickers left. | $54 - 9 = 45$<br>...<br>$45 - 28 = 17$<br>...<br>Therefore, Charlie has 17 stickers left. |
| Dylan buys 38 chicken sausages and six more fish sausages than chicken sausages. Total? (82) | Dylan bought 38 + 6 = 44 fish sausages. | Dylan bought 38 chicken sausages + 6 fish sausages = 44 sausages in all. |

Shaoan Xie, Lingjing Kong, Xiangchen Song, Xinshuai Dong, Guangyi Chen, Eric P. Xing, and Kun Zhang. Advancing reasoning in diffusion language models with denoising process rewards. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 42703–42720, 2026.

Shuibai Zhang, Fred Zhangzhi Peng, Yiheng Zhang, Jin Pan, and Grigorios G. Chrysos. Corrective diffusion language models. *arXiv preprint arXiv:2512.15596*, 2025a.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. The lessons of developing process reward models in mathematical reasoning. *arXiv preprint arXiv:2501.07301*, 2025b.

## A Qualitative Examples

Table 4 presents selected paired GSM8K outputs at the main guidance setting: seed 0, 64 steps, block size 256, $\lambda = 0.75$, and $K = 16$. Questions are abbreviated; excerpts preserve generated wording and arithmetic, with mathematics typeset for readability. Ellipses mark omitted output. Three cases show correct guided answers and one shows a shared failure. Selection illustrates behavior, not its frequency or the causal effect of individual commitment decisions.

## B What Step Guidance Does and Does Not Preserve

This appendix distinguishes invariance of predictions at a fixed canvas from possible changes in the distribution of completed sequences.

Let $x$ denote a partially masked canvas and $L_\theta(x)$ the frozen backbone logits, with all other backbone inputs held fixed. The implementation first computes token proposals $z = D(L_\theta(x), u)$, where $D$ is the decoding rule and $u$ is any proposal randomness. Greedy decoding uses a fixed tie-breaking rule and no randomness. It then computes proposal confidence $c_p = P_\theta(z_p \mid x, p)$ at each masked position, where $P_\theta$ is the backbone softmax. Baseline ranks $c_p$, whereas guidance ranks

$c_p - \lambda r_p$, with $r_p$ the current or cached issue score. These ranking scores are not normalized token probabilities.

**Proposition 1 (fixed-canvas prediction invariance).** At the same canvas, with identical backbone inputs, decoding settings, and proposal randomness, baseline and guidance produce the same logits, softmax distributions, and token proposals before selecting positions:

$$L_g(x) = L_0(x) = L_\theta(x), \qquad z_g(x,u) = z_0(x,u) = D(L_\theta(x), u).$$

**Proof.** The issue head does not modify backbone inputs, parameters, or logits. Both decoders therefore evaluate the same function $L_\theta(x)$; applying softmax gives identical distributions. Applying the same proposal rule with the same randomness gives identical $z$. The issue-score subtraction occurs only afterward, when ranking positions for commitment. □

This compares decoder evaluations at an identical input, rather than conditioning on a position being selected. Selection uses proposal confidence, so with stochastic proposals it can bias the distribution of committed tokens. For greedy decoding, the proposal law is a point mass at the argmax, rather than the backbone softmax distribution.

**Proposition 2 (output distributions need not be invariant).** There exist backbone predictions and issue scores for which the policies induce different completed-sequence distributions despite Proposition 1.

**Proof by construction.** Consider a binary vocabulary, two masked positions $a, b$, greedy decoding, and one commitment per step for two steps. Initially, let

$$P_\theta(a = 1 \mid X_0) = 0.9, \qquad P_\theta(b = 1 \mid X_0) = 0.8.$$

Both policies propose $(1, 1)$, and baseline commits $a$ first. Set $\lambda = 0.75$, $r_a = 0.8$, and $r_b = 0.2$. Guided priorities are $0.30$ and $0.65$, so guidance commits $b$ first. At the subsequent canvases, let $P_\theta(b = 1 \mid a = 1) = 0.9$ and $P_\theta(a = 0 \mid b = 1) = 0.9$. Complementary binary probabilities are strictly positive, and all decisions have strict maxima, so tie-breaking is immaterial. Baseline returns $(1, 1)$, while guidance returns $(0, 1)$. Their output laws are distinct point masses. □

The construction treats the backbone as a collection of canvas-dependent predictions; it does not assume they are exact conditionals of one joint distribution. It establishes that preserving logits at a fixed canvas does not guarantee preservation of the output distribution. It neither asserts that every reordering changes the output nor guarantees improved accuracy.

## C Issue Head Architecture

The step-issue head architecture consists of:

- Timestep embedding: Linear$(2 \to 64)$, SiLU, Linear$(64 \to 64)$.
- Issue MLP: Linear(hidden_size $+ 64 \to 256$), SiLU, Linear$(256 \to 1)$.

For LLaDA-8B, hidden_size$= 4096$, giving $\sim$1.1M parameters in the issue head—negligible compared to the 8B backbone. The timestep embedding is broadcast across all token positions and concatenated with the per-token hidden state before the MLP.

The value head uses the same timestep embedding and a separate state-level pooling architecture. It applies an attention-mask-weighted mean over all non-padding prompt and canvas hidden states, concatenates the pooled representation with the 64-dimensional timestep embedding, and applies Linear$(4096 + 64 \to 256)$, SiLU, and Linear$(256 \to 1)$. It therefore also has approximately 1.1M trainable parameters, while the LLaDA backbone remains frozen. Its output is a scalar state-quality estimate used to trigger PRM-distilled suffix repair.

## D PRM Distillation Data Generation

We generate one PRM-distillation corpus from logged GSM8K training trajectories. For each selected masked canvas, we use three sparse denoising fractions, $f \in \{0.25, 0.5, 0.75\}$, and

complete the masked positions to produce four fills: one greedy fill and three stochastic fills. Each completed fill is segmented into blank-line-delimited reasoning steps and scored with Qwen2.5-Math-PRM-7B. The PRM returns one score per reasoning step; the fill score is the mean of its step scores. Across the four fills, we also retain the minimum and standard deviation of these fill-level scores.

The resulting distillation corpus contains 9,000 state rows, corresponding to three fractions per source trajectory, and is used to train the scalar head against the mean fill-level PRM target. The step-issue head is trained from these same state rows; each row contains a variable number of valid token targets after span alignment.

For the step-issue head, we align each PRM step to a half-open token span in the completed fill. If a token belongs to step $j$, its issue target is $y_i = 1 - s_j$, where $s_j$ is the PRM score for that step. Targets are averaged across fills wherever spans overlap, and tokens outside scored spans are excluded from the loss. For reproducibility, each training record retains the per-fill step scores, token spans, and token IDs used for alignment. Thus the value head learns *whether* a canvas is low quality, whereas the step-issue head learns *which token spans* are associated with the low-quality step.

## E Guidance-Strength Sensitivity

We report a descriptive sensitivity analysis over guidance strength and update interval while holding the denoising schedule fixed at 64 steps with a 256-token block. Figure 4 reports the complete two-dimensional sweep. The headline results use the manually fixed operating point $\lambda = 0.75$, $K = 16$, as specified in the main text.

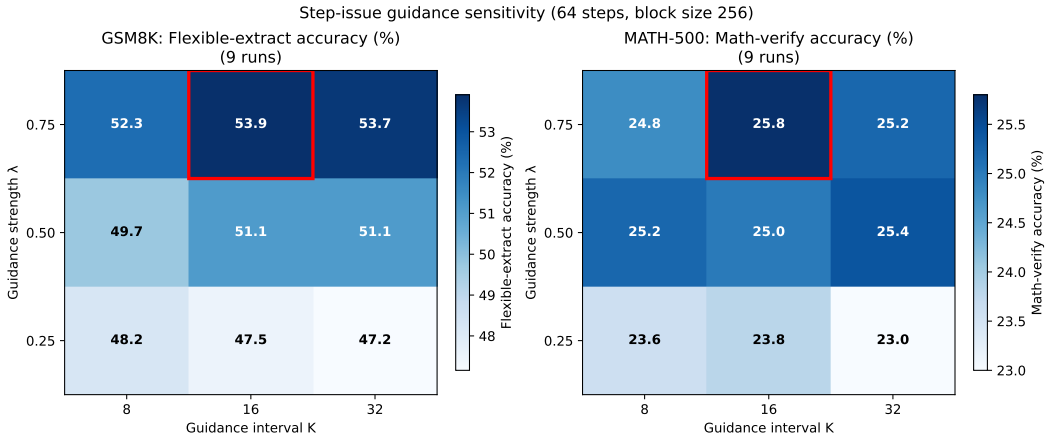


Figure 4: Step-issue guidance sensitivity across guidance strength $\lambda$ and update interval $K$. Each cell reports accuracy for the corresponding full-dataset configuration; the highlighted cell marks the fixed operating point ($\lambda = 0.75$, $K = 16$). The color range is scaled independently for each dataset to make within-dataset differences visible.

## F Compute Cost of Guidance and Repair

We compare the inference cost of guidance and repair in a matched 100-example GSM8K run. All configurations use 64 denoising steps, one 256-token block, and seed 0. The step-guidance policy adds 400 issue-head calls but no extra backbone calls, whereas repair methods launch additional denoising passes for the examples they repair. Direct PRM repair is the most expensive policy because it also requires the external PRM; its wall-clock figure includes that evaluation.

In this recorded run, guidance is only 2.29 seconds (0.4%) slower than baseline, whereas distilled repair and direct PRM repair are 23.6% and 35.1% slower, respectively. These measurements illustrate the quality–cost trade-off but should not be interpreted as a hardware-independent throughput claim.

Table 5: ASDiv-CoT-LLaMA transfer baseline. All methods use LLaDA-8B-Instruct with 64 denoising steps and a 128-token block.

| Method | Accuracy | PRM at inference | Extra decode | Repair rate |
|---|---|---|---|---|
| No guidance (baseline) | 74.19% | × | × | — |
| Step-issue guidance | 75.36% | × | × | — |
| PRM-distilled repair | 74.45% | × | ✓ | 10.3% |
| PRM repair | 77.14% | ✓ | ✓ | 24.2% |

## G COMMITMENT-SCORE CONTROLS

All controls use 64 steps, block size 256, seed 0, $\lambda = 0.75$ and $K = 16$. Scores are cached between updates. The constant control uses $r_p = 0.5$, preserving rankings under fixed-count selection. The control labeled *confidence* computes $u_p = \sigma(\|h_p\|_2)$ and $r_p = (u_p - \min_q u_q)/(\max_q u_q - \min_q u_q + 10^{-6})$ over the full sequence. We call it a hidden-norm proxy; it is not token confidence.

Shuffling independently permutes trained scores across the full prompt-plus-canvas sequence at each update, without separating masked and revealed positions. Only eligible masked positions are committed. Random scores are drawn anew at each update as $r_p = \mathrm{clip}_{[0,1]}(\mu + \max(\sigma_r, 10^{-6})\epsilon_p)$, with independent $\epsilon_p \sim \mathcal{N}(0, 1)$ and trained-score mean $\mu$ and sample standard deviation $\sigma_r$ over the full sequence. Clipping can change the resulting moments. Both use the seeded PyTorch generator. Shuffled and random controls evaluate the issue head as often as learned guidance; constant and norm-proxy controls bypass the head but still request backbone hidden states.

## H ASDIV-COT-LLAMA BASELINE

We additionally evaluate the frozen heads on ASDiv (Miao et al., 2020) using the lm-eval CoT-LLaMA evaluation variant, without any in-domain training or tuning. CoT-LLaMA denotes the evaluation prompt variant, not a separate dataset. Table 5 reports flexible-extract exact match over 2,305 examples at 64 denoising steps and a 128-token block. The small gain from guidance is consistent with transfer, while direct PRM repair yields the largest improvement at the cost of PRM inference and additional decoding. As with the main results, these are single-run point estimates.

# I Data Creation and Head Training Pseudocode

**Algorithm 1: PRM-distillation data creation.** The following pseudocode summarizes the logged-trajectory procedure. A malformed fill is rejected if the number of PRM step scores does not match the number of aligned step spans.

```
INPUT: logged trajectories; fractions F={.25,.50,.75}; fills=4; seed
for each trajectory (question q, states S, prompt length m):
  for f in F:
    s <- S[min(|S|-1, round(f*(|S|-1)))]
    for r=0,...,3:
      z <- greedy_fill(s) if r=0 else sample_logged_topk(s, seed, r)
      text, spans <- segment_and_align(z; blank-line step boundaries)
      scores <- PRM(q, text)                 # one score per step
      reject if len(scores) != len(spans)
      save token ids, spans, and scores for this fill
    save mean/min/std fill score, state metadata, and correctness label
write one state row per (trajectory, fraction)
```

**Algorithm 2: Token-level issue-head training.**

```
INPUT: state rows; frozen LLaDA backbone f; epochs=2; lr=1e-4
initialize token-level issue head g_issue
for each minibatch of state rows:
  H <- f(state_tokens; attention_mask)       # no gradient through f
  c <- [denoising_fraction, mask_ratio]
  y, valid <- average(1-PRM step score) over aligned token spans
  logits <- g_issue(H, c)
  L <- BCE_with_logits(logits, y; weight=valid)
  update only g_issue with AdamW
save g_issue parameters and tokenizer metadata
```

**Algorithm 3: State-level value-head training.**

```
INPUT: state rows; frozen LLaDA backbone f; epochs=2; lr=1e-4
initialize state-level value head g_value
for each minibatch of state rows:
  H <- f(state_tokens; attention_mask)       # no gradient through f
  h_pool <- mask_weighted_mean(H, attention_mask)
  c <- [denoising_fraction, mask_ratio]
  v <- g_value(h_pool, c)
  target <- mean fill-level PRM score
  L <- MSE(v, target)
  update only g_value with AdamW
save g_value parameters and tokenizer metadata
```

# J Value-Head Stage Diagnostic

Table 6: Scalar value-head diagnostic on 1,350 states (150 per fraction). Spearman correlates value predictions with mean fill-level PRM targets at each fixed denoising fraction. AUC measures discrimination of final-answer correctness. Training–evaluation independence is not established.

| Denoising fraction | Spearman vs. mean PRM target | AUC vs. correctness |
|---|---|---|
| 0.1 | 0.383 | 0.636 |
| 0.2 | 0.491 | 0.632 |
| 0.3 | 0.577 | 0.683 |
| 0.4 | 0.599 | 0.669 |
| 0.5 | 0.644 | 0.652 |
| 0.6 | 0.692 | 0.706 |
| 0.7 | 0.686 | 0.725 |
| 0.8 | 0.636 | 0.699 |
| 0.9 | 0.607 | 0.721 |

At a fixed stage, value-head correlations range from $0.383$ to $0.692$. Unlike the token-level diagnostic in Section 4.3, these results measure whether scalar value predictions rank states

consistently with their mean fill-level PRM targets, not whether the issue head localizes problematic spans. The pooled Spearman is $0.866$, but it also reflects between-fraction differences and is therefore secondary to within-fraction fidelity. The pooled correctness AUC of $0.604$ shows that teacher alignment does not make the value head a reliable final-answer verifier.

## K REPAIR PSEUDOCODE

**Algorithm 4: Direct PRM repair.**

```
INPUT: prompt q; completed diffusion output x; PRM; threshold tau=.5
scores <- PRM.step_scores(q, decode(x))
if min(scores) < tau:
  j <- first step with score < tau
  b <- token boundary at the start of step j
  x <- INFILL(mask_suffix(x, boundary=b, through=EOS), steps=64)
return EXTRACT_ANSWER(x)
```

Algorithm 5 gives the complete PRM-distilled repair policy used in our experiments. The value head is evaluated at the midpoint of the recorded denoising trajectory. Candidate boundaries are restricted to revealed generated tokens whose decoded token string contains a digit; this restriction is specific to our arithmetic-reasoning setting.

**Algorithm 5: PRM-distilled suffix repair.**

```
INPUT: prompt q; diffusion model D; value head V;
       value threshold tau_val; advantage threshold tau_adv
OUTPUT: answer a

(x, H) <- DENOISE(q, steps=64, block_size=256, record_trajectory=True)
s <- H[floor(|H| / 2)]
v_base <- V(s, denoising_fraction(s), mask_ratio(s))

if v_base < tau_val:
    C <- {p : p is a revealed generated position in s,
             decoded_token(s[p]) contains a digit}
    for p in C:
        s_p <- s with position p replaced by MASK
        v_p <- V(s_p, denoising_fraction(s), mask_ratio(s_p))
    p_star <- argmax_{p in C} (v_p - v_base)
    if v_p_star - v_base > tau_adv:
        x <- INFILL(x with positions p_star through EOS replaced by MASK,
                    steps=64, block_size=256)

return EXTRACT_ANSWER(x)
```

**Random-repair control.** For every completed response, sample one valid repair boundary uniformly at random, mask the suffix from that boundary through EOS, and apply the same `INFILL` procedure. Random repair therefore differs only in boundary selection; it uses the same 64-step, 256-token suffix-regeneration policy and repairs every example. As a supplementary control, it obtains 48.4% on GSM8K and 26.8% on MATH-500 at 64 steps; the corresponding 32-step results are 45.6% and 17.4%, respectively. These results are not included in the main comparison table because random repair is a diagnostic control rather than a proposed method.

**Algorithm 6: Step-issue guidance.** The pseudocode below follows the guidance rule defined in Section 3.4. The issue head changes commitment priority but does not change the token logits.

```
INPUT: masked canvas x; diffusion model f_theta; issue head g_phi;
       guidance strength lambda; update interval K; denoising steps T
OUTPUT: completed canvas x

for t = 0, ..., T-1:
    logits, hidden <- f_theta(x)

    if t mod K == 0:
        c <- [t / (T-1), mask_ratio(x)]
        p_issue <- sigmoid(g_phi(hidden, c))

```

```
    for each masked position p:
        x0[p] <- argmax_v p_theta(v | x)[p]
        confidence[p] <- max_v p_theta(v | x)[p]
        guided_confidence[p] <- confidence[p] - lambda * p_issue[p]

    reveal positions selected by the standard low-confidence-remasking
    schedule using guided_confidence; assign each revealed position x0[p]

return x
```

We use $T = 64$, $K = 16$, and $\lambda = 0.75$. Between guidance updates, $p_{\text{issue}}$ is held fixed. The token distribution used to produce $x_0$ is unchanged; only the commitment priority is modified.