Pivot-SD: Efficient Self-Distillation for Masked Diffusion Language Models

Anonymous ACL submission

Abstract

Discrete masked diffusion language models (dLMs) offer a promising parallel alternative to sequential autoregressive models for complex reasoning. However, they face a unique credit-assignment challenge: reasoning failures typically stem from a few critical, early denoising commitments that cascade downstream. Existing post-training methods are poorly suited for this dynamic. Supervised fine-tuning (SFT) weights all tokens uniformly, while online reinforcement learning (RL) relies on sparse rewards and discards failed trajectories entirely. To address these limitations, we introduce Pivot-SD, a highly efficient offline self-distillation framework that targets key step-conditioned denoising commitments. Pivot-SD identifies critical reasoning bottlenecks using a normalized information-gain metric. It then reinforces successful pivotal tokens via cross-entropy and suppresses critical errors from failed trajectories via localized unlikelihood training. Using only 200 training questions and four rollouts per question, Pivot-SD improves LLaDA-8B-Instruct over full-sequence SFT and budget-matched diffusion RL baselines across mathematical reasoning and code-generation benchmarks.

1 Introduction

Large language models (LLMs) have achieved remarkable progress on complex mathematical and symbolic reasoning tasks, largely driven by post-training methodologies such as supervised fine-tuning (SFT) and reinforcement learning (RL) (Guo et al., 2025; Ouyang et al., 2022). However, this progress has been fundamentally anchored to autoregressive models (ARMs), where the sequential paradigm causes early mistakes to propagate and collapse the reasoning trace. Discrete masked diffusion language models (dLMs), such as LLaDA (Nie et al., 2026), which employs a forward masking process and reverse masked-token prediction have recently emerged as a compelling parallel, coarse-to-fine alternative to this sequential bottleneck. Yet, while dLMs demonstrate impressive reasoning capabilities, they introduce a fundamentally different credit-assignment problem.

In masked diffusion, reasoning failures are rarely localized to the final generated tokens. Instead, during blockwise denoising, the model makes a small number of high-impact commitments that alter the distribution of the remaining masked block; a single erroneous commitment can collapse the sampling path long before the final answer is produced. Current post-training methodologies fail to capture this step-conditioned dynamic because they treat dLMs too much like their autoregressive counterparts (Zhao et al., 2026; Wang et al., 2026; Rojas et al., 2026; Tang et al., 2026). Standard SFT applies uniform token weighting, failing to distinguish a decisive denoising commitment from a trivial token, while online RL optimizes sparse final-answer rewards without isolating the specific parallel decisions that determined the trajectory's outcome. Ultimately, these conventional approaches severely dilute the learning signal.

Furthermore, existing self-distillation pipelines typically discard failed trajectories entirely, which represents a significant missed opportunity (Zelikman et al., 2022; Singh et al., 2024; Dou et al., 2024). An incorrect trajectory often contains many locally valid reasoning steps, syntactic structures, and algebraic fragments. Broadly penalizing the entire failed sequence destroys these valid reasoning patterns. While unlikelihood training is an established objective to reduce the probability of unlikely generations (Luo et al., 2025), conventional SFT and rejection self-distillation lack a *principled mechanism* to isolate which specific token within a failed reasoning sequence should be treated as the negative candidate.

To address this problem, we propose **Pivot-SD**, an offline self-distillation method that

projects trajectory-level outcomes onto sparse, step-conditioned denoising commitments. For each sampled trajectory, Pivot-SD identifies candidate pivots (t, p, y_p), where the model commits token y_p at position p during denoising step t. Rather than selecting tokens solely by local uncertainty, Pivot-SD scores each candidate by the information gain induced by counterfactually clamping the commitment and measuring the resulting uncertainty reduction over the remaining masked block. The selected pivots then serve as localized training targets: pivots from successful trajectories are reinforced with cross-entropy, while high-impact pivots in failed trajectories are suppressed with targeted unlikelihood. This allows Pivot-SD to exploit both successful and failed samples without applying negative loss to entire incorrect solutions.

Focusing squarely on enhancing the model's fundamental reasoning capabilities, Pivot-SD demonstrates strong data efficiency. Using only 200 training questions and 800 sampled trajectories, Pivot-SD consistently improves over full-sequence SFT baselines and low-budget online RL-style baselines across mathematical reasoning and code-generation benchmarks under our primary 256-token evaluation budget.

In summary, our main contributions are as follows:

- **Step-Conditioned Credit Assignment:** We formulate reasoning supervision for discrete masked diffusion language models around denoising commitments (t, p, y_p), fundamentally shifting the unit of credit assignment from final sequence tokens to structural reasoning bottlenecks.

- **Information-Guided Localized Self-Distillation:** We introduce Pivot-SD, a novel offline objective that selects candidate bottlenecks by measuring counterfactual uncertainty reduction. It extracts dense supervision by reinforcing successful pivots with cross-entropy and suppressing high-impact failed pivots with localized unlikelihood, preserving locally valid sub-steps.

- **Data-Efficient Offline Post-Training:** We empirically demonstrate that Pivot-SD substantially improves structured reasoning using only a minimal cached trajectory pool, consistently outperforming same-data and same-step online RL-style baselines.

2 Related Work

Diffusion language models. Diffusion models have been studied as alternatives to autoregressive language modeling. Early work explored continuous diffusion over word embeddings (Li et al., 2022), and D3PMs extended denoising diffusion to discrete state spaces (Austin et al., 2021a). Later work improved discrete and masked diffusion with score-entropy objectives and masked training recipes (Lou et al., 2024; Sahoo et al., 2024), and LLaDA scaled the paradigm to an 8B instruction-following model trained with forward masking and reverse masked-token prediction (Nie et al., 2026). These models introduce credit-assignment questions because generation unfolds over iterative denoising states rather than a fixed token order (e.g., left-to-right.)

Our work focuses on this denoising-time structure. Instead of treating a sampled response as only a final sequence of tokens, we treat the unmask trajectory itself as a source of supervision. Our central idea is that a token is important not merely because it appears in the final solution, but because it may be committed at a particular denoising step and thereby constrain subsequent masked positions.

Reasoning with masked diffusion language models. Several recent methods adapt these models for reasoning. d1 combines masked SFT with diffu-GRPO, an RL algorithm for masked diffusion (Zhao et al., 2026). DCoLT treats intermediate reverse-diffusion steps as latent thinking actions and optimizes trajectories with outcome-based RL (Huang et al., 2026). d2 proposes policy-gradient techniques that improve reasoning without supervised fine-tuning (Wang et al., 2026), and GDPO revisits likelihood estimation and uses variance-reduced ELBO-based policy optimization (Rojas et al., 2026). These methods rely on online RL. Pivot-SD instead samples trajectories once from a frozen model and converts trajectory outcomes into localized supervision over selected denoising commitments. This gives a stable offline alternative that can use both successful and failed trajectories.

Process supervision and reasoning credit assignment. A common approach to improving mathematical reasoning is to supervise intermediate steps. Lightman et al. (2024) show that step-level feedback can outperform outcome supervision on MATH, and Math-Shepherd automatically constructs process labels and trains process reward

models (Wang et al., 2024). These methods score textual reasoning steps in autoregressive solutions.

Pivot-SD shares the goal of fine-grained credit assignment but changes the unit of supervision. Instead of scoring a sentence or derivation step, it supervises a denoising commitment (t, p, y_p) inside a masked diffusion trajectory. This matters because masked diffusion models commit tokens out of order, and a single commitment can reshape the distribution over many remaining positions.

Token importance and weighted fine-tuning. A related direction reweights tokens during SFT. For diffusion language models, GIFT assigns entropy-based importance weights to tokens (Xu et al., 2026), showing that emphasizing some tokens helps fine-tuning. Pivot-SD differs in what is being weighted: our scores rank tokens by their impact as commitments at specific denoising steps, not by their importance in the final sequence. To separate these effects, our ablations compare pivot-only supervision, selected-step uniform supervision, pivot-weighted supervision, entropy-selected pivots, and random pivots.

Positioning of our work. Pivot-SD sits between standard SFT and online RL. Like SFT, it is stable, offline, and easy to optimize with supervised losses. Like RL, it uses outcome information from successful and failed trajectories. Its key difference is that it maps trajectory outcomes to sparse step-conditioned denoising commitments. This makes it possible to exploit failed trajectories without online policy optimization and to target the decisions that most strongly shape a masked diffusion reasoning trajectory.

3 Methodology

In this section, we present Pivot-SD, an offline self-distillation framework for masked diffusion language models. We first establish the formal mechanics of masked diffusion language models and standard supervised fine-tuning (§3.1). We then introduce our information-gain selection metric and training objectives (§3.2). We detail the complete offline algorithm and contrast its structural properties with standard SFT and online reinforcement learning paradigms (§3.3).

3.1 Preliminaries and Problem Setup

We begin by formalizing the notation for masked diffusion language models and establishing the baseline training objective.

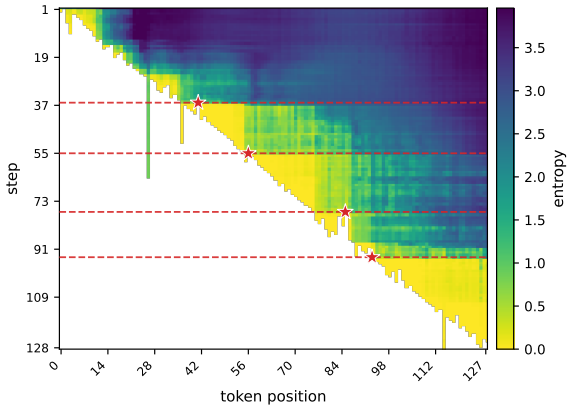

Figure 1: **Predictive Entropy and Pivot Selection.** An entropy heatmap of a masked denoising trajectory for a MATH problem (showing the first 128 tokens). The x-axis represents the token position, and the y-axis denotes the denoising step progressing downward. The color scale indicates the predictive entropy of the active masked positions. Red dashed lines and stars denote the critical denoising steps and the specific pivot tokens selected by our Information Gain metric, respectively.

Setup Let x represent a conditioning prompt, and let $y = (y_1, \ldots, y_L)$ denote a target response sequence of length L. A masked diffusion language model generates y conditioned on x by iteratively denoising an initially fully masked response sequence over T steps. Let M_t represent the complete sequence state at denoising step t (consisting of the unmasked prompt x concatenated with the partially masked response), indexed in the sampler's reverse chronological order. To formalize the active response positions, let $\mathcal{I}_{\text{resp}}$ denote the set of coordinate indices corresponding exclusively to the response tokens. The active response positions that remain masked at step t are defined as

$$\mathcal{B}_t = i \in \mathcal{I}_{\text{resp}} : M_{t,i} = [\text{MASK}]. \qquad (1)$$

At each denoising step, the model parameterizes a distribution over the clean token for each active masked position:

$$P_\theta(x_0^{(i)} \mid M_t), \qquad i \in \mathcal{B}_t, \qquad (2)$$

where $x_0^{(i)}$ is the clean-token random variable at response position i, and its final sampled value in the finalized trajectory is y_i.

Standard SFT Standard masked SFT trains the parameters θ by minimizing the uniform per-

position cross-entropy loss:

$$\mathcal{L}_{\text{SFT}}(\theta) = -\mathbb{E}_{((x,y),t,M_t)}\left[\sum_{i\in\mathcal{B}t} \log P_\theta(y_i \mid M_t)\right]. \tag{3}$$

The baseline loss in Eq. (3) weights every active masked position uniformly. It fails to distinguish between trivial token fills and critical structural commitments that dictate the downstream coherence of the response. Pivot-SD addresses this limitation by restricting supervision to a sparse set of highly informative step-position commitments, where the directional sign of the parameter update is gated by the overall quality of the trajectory outcome.

3.2 The Pivot-SD Framework

In this section, we introduce the core components of Pivot-SD. We outline how we extract localized, step-conditioned units of supervision from raw diffusion trajectories (§3.2.1), isolate high-impact commitments using an information-gain formulation (§3.2.2), and finally the training objective (§3.2.3.)

3.2.1 Trajectory Generation and Candidate Pivots

Sampling a complete response from the frozen base model θ_0 yields a discrete denoising trajectory:

$$\tau = (M_t, \mathcal{B}_t, \mathcal{U}_t)_{t=1}^{T}, \tag{4}$$

where $\mathcal{U}_t \subseteq \mathcal{B}_t$ is the specific subset of positions selected for unmasking and commitment by the sampler at step t. A position p is committed at step t when the sampler selects $p \in \mathcal{U}_t$ and writes the sampled token value y_p into the sequence state. For every committed position $p \in \mathcal{U}_t$, the trajectory generates a candidate pivot c, which we group into a trajectory-wide candidate set:

$$\mathcal{C}(\tau) = (t, p, y_p) : p \in \mathcal{U}_t. \tag{5}$$

The triple in Eq. (5) records what token was committed, where it was committed, and when in the denoising process it was committed. Thus the supervised object is not merely the token y_p, but the event of committing y_p at position p under state M_t. The same token at the same response position can be low-impact late in denoising but high-impact earlier, when many neighboring positions remain masked. This is what separates Pivot-SD from token-level SFT, which supervises only (p, y_p), and from trajectory-level RL, which supervises only the trajectory outcome.

3.2.2 Information-Gain Pivot Selection

Pivot selection ranks candidate commitments by a one-step estimate of how much fixing that commitment reduces uncertainty over the remaining masked block. Intuitively, a useful pivot is not merely a token the model is uncertain about, it is a commitment that changes what the frozen model can infer about the rest of the denoising block. We measure local token uncertainty using the Shannon entropy of the frozen base model's predictions. For any masked state M and index i, the per-position entropy is

$$h_{\theta_0}(i \mid M) = H\left(P_{\theta_0}(x_0^{(i)} \mid M)\right). \tag{6}$$

For a candidate commitment (t, p, y_p), the baseline pre-commitment uncertainty across the remaining uncommitted active block is defined as

$$H_{\text{pre}}(t, p) = \sum_{i\in\mathcal{B}_t\setminus\{p\}} h_{\theta_0}(i \mid M_t). \tag{7}$$

We then evaluate a counterfactual state where position p is explicitly clamped to its sampled token value y_p, yielding a modified context sequence $M_t^{[p\leftarrow y_p]}$. The post-clamp uncertainty over the identical remaining block is calculated as

$$H_{\text{post}}(t, p) = \sum_{i\in\mathcal{B}_t\setminus\{p\}} h_{\theta_0}\left(i \mid M_t^{[p\leftarrow y_p]}\right). \tag{8}$$

By excluding the pivot position p from both terms, this formulation directly isolates how much a localized commitment reduces uncertainty across the rest of the generation block. To prevent early denoising states, which inherently possess larger absolute entropy scales—from mathematically dominating the selection process. We compute a normalized information-gain score:

$$g(t, p) = \frac{H_{\text{pre}}(t, p) - H_{\text{post}}(t, p)}{H_{\text{pre}}(t, p) + \epsilon}, \tag{9}$$

where $\epsilon > 0$ is a small constant ensuring numerical stability. For each trajectory τ, candidates are ranked by $g(t, p)$, and only the top-K scoring pivots are selected for training. The metric $g(t, p)$ is used exclusively as a filtering mechanism and does not scale the loss magnitude during optimization.

Figure 1 visualizes this selection process via the entropy dynamics of a single denoising trajectory. At critical structural bottlenecks, committing a single token drastically reduces the uncertainty of the

remaining sequence. Our Information Gain metric effectively captures these moments of sharp entropy collapse (marked by red dashed lines and stars). This confirms that Pivot-SD isolates the decisive commitments that dictate the logical outcome, rather than uniformly weighting the entire sequence.

3.2.3 Training objective

Each selected pivot is mapped to a trajectory outcome score $s(\tau) \in \{+1, -1\}$, which evaluates the correctness of the final response sequence y generated by trajectory τ:

$$s(\tau) = \begin{cases} +1, & \text{if } y \text{ is correct,} \\ -1, & \text{otherwise.} \end{cases} \tag{10}$$

Every chosen pivot yields a training tuple $z = (M_t, y, p, y_p, s(\tau))$. For a selected pivot token y_p, we define the positive cross-entropy loss as

$$\ell_p^+(\theta; M_t, y_p) = -\log P_\theta(y_p \mid M_t), \tag{11}$$

and the token-level unlikelihood penalty as

$$\ell_p^-(\theta; M_t, y_p) = -\log\left(1 - P_\theta(y_p \mid M_t)\right). \tag{12}$$

During training, the predicted probability $P_\theta(y_p \mid M_t)$ in Eq. (12) is clipped away from 1 to maintain numerical stability. Failed trajectories often contain structurally sound intermediate sequences, such as valid syntactic phrasing or correct localized algebraic steps. To avoid penalizing these helpful components, Pivot-SD applies negative supervision exclusively to the highest-impact pivots extracted from failed generations. The complete outcome-conditioned pivot loss for a single tuple z is formulated as

$$\begin{aligned} L_{\text{pivot}}(z) = \mathbf{1}[r = +1]\, \ell_p^+(\theta; M_t, y_p) \\ + \lambda_{\text{neg}}\, \mathbf{1}[r = -1]\, \ell_p^-(\theta; M_t, y_p), \end{aligned} \tag{13}$$

where λ_{neg} is a hyperparameter scaling the unlikelihood penalty. Let $\mathcal{D}_{\text{pivot}}$ be the offline dataset compiled by executing this selection process across all training prompts $\mathcal{X}$ and generated trajectories. The final fine-tuning objective minimizes the expected pivot loss over this curated dataset:

$$\min_\theta \frac{1}{|\mathcal{D}_{\text{pivot}}|} \sum_{z \in \mathcal{D}_{\text{pivot}}} L_{\text{pivot}}(z). \tag{14}$$

We initialize θ from the base model θ_0 and optimize Eq. (14) using standard gradient-based optimization.

Algorithm 1: Pivot-SD: Offline Outcome-Conditioned Pivot Self-Distillation

[1] Training prompts $\mathcal{X}$, frozen base model θ_0, trajectories per prompt G, pivots per trajectory K Fine-tuned model θ Initialize pivot dataset $\mathcal{D}_{\text{pivot}} \leftarrow \emptyset$ **for** *each prompt* $x \in \mathcal{X}$ **do**

$g = 1, \ldots, G$ Sample a denoising trajectory $\tau = \{(M_t, \mathcal{B}_t, \mathcal{U}_t)\}_{t=1}^{T}$ and response y from θ_0 Assign trajectory outcome $s(\tau)$ using Eq. (10) Construct candidate pivots $\mathcal{C}(\tau) = \{(t, p, y_p) : p \in \mathcal{U}_t\}$ **for** *each candidate* $(t, p, y_p) \in \mathcal{C}(\tau)$ **do**

Compute information-gain score $g(t, p)$ using Eq. (9) Select the top-K candidates by $g(t, p)$ **for** *each selected pivot* (t, p, y_p) **do**

Add $(M_t, y, p, y_p, s(\tau))$ to $\mathcal{D}_{\text{pivot}}$ Fine-tune θ on $\mathcal{D}_{\text{pivot}}$ using Eq. (13) **return** θ

3.3 Algorithmic Implementation

We synthesize the full training pipeline in Algorithm 1, detailing the offline pipeline from data collection to fine-tuning.

3.4 Relation to SFT and Online RL

Pivot-SD occupies a distinct methodological space between standard SFT and online RL. Standard masked SFT applies uniform token-level supervision, ignoring the variance in causal impact across different denoising positions. Rejection-based self-distillation improves on this by filtering for successful trajectories, but completely discards the valuable structural failures present in unsuccessful generations. Conversely, online policy-gradient RL methods utilize trajectory-level rewards to optimize execution paths, but they lack the localized precision required to determine which specific spatial or temporal token commitment triggered the final outcome. In contrast, Pivot-SD extracts discrete credit assignments from global outcomes without requiring online exploration. Successful trajectories provide target contexts for positive cross-entropy updates, while failed trajectories provide focused unlikelihood constraints at localized points of failure. Because all trajectories are sampled statically from the initial frozen base model θ_0, the compiled dataset $\mathcal{D}_{\text{pivot}}$ remains invariant, allowing for reproducible offline hyperparameter tuning and optimization sweeps without the instabilities typical of online policy-gradient methods.

4 Experiments

We evaluate whether outcome-conditioned pivot self-distillation can improve masked diffusion lan-

Table 1: **Performance comparison of LLaDA-8b-instruct on math and coding tasks.** We report the accuracy evaluated at generation lengths of 256 and 512. The baselines include both SFT and RL methods. All non-base methods are trained under the same controlled setup whenever applicable.

Method	MATH		GSM8K		HumanEval+		MBPP+	
	256	512	256	512	256	512	256	512
Base	31.4	38.8	75.13	83.40	32.32	**42.07**	43.12	44.44
SFT-GT	35.6	38.6	75.82	83.17	32.93	36.59	**46.56**	42.86
SFT-SD	31.4	38.6	74.91	81.96	35.37	40.85	45.77	44.97
diffu-GRPO (budget-matched)	34.6	38.2	75.74	82.11	31.10	**42.07**	43.65	45.24
diffu-GRPO	35.2	40.6	77.63	83.62	34.15	40.85	42.59	43.65
wd1++ (budget-matched)	33.8	39.2	77.10	82.71	32.93	40.24	43.12	43.92
wd1++	35.4	39.4	76.65	82.71	34.15	38.41	44.97	44.44
Pivot-SD (ours)	**38.6**	**42.6**	**79.53**	**84.38**	**40.85**	41.46	46.03	**48.15**

Table 2: **Ablation study of pivot selection and credit assignment.** "IG state" denotes denoising states selected by our information-gain criterion. "Phase-matched" denotes random pivot states sampled from a matched denoising phase. "All" applies the loss to all supervised masked tokens in the selected state, while "Pivot" applies the loss only to the selected pivot token. CE denotes cross-entropy and UL denotes unlikelihood.

Method	State	Target	Pos.	Neg.	MATH	GSM8K	HumanEval+	MBPP+
Positive All-token	IG	All	CE	-	31.6	74.37	36.59	39.68
Positive Pivot	IG	Pivot	CE	-	32.2	72.48	35.37	43.65
Pos+Neg All-token	IG	All	CE	State-UL	34.6	74.53	37.8	**47.09**
Random-Step Pivot	Random	Pivot	CE	Pivot-UL	26.6	76.88	35.37	46.03
Random-Token Pivot	IG	Random	CE	Pivot-UL	33.2	75.74	37.8	45.77
Pivot-SD: (ours)	IG	Pivot	CE	Pivot-UL	**38.6**	**79.53**	**40.85**	46.03

guage models under a small post-training budget. Our main experiments use LLaDA-8B-Instruct as the backbone.

Training data. For each math reasoning dataset, we sample 200 training questions from the corresponding training split: GSM8K for GSM8K evaluation (Cobbe et al., 2021) and MATH for MATH500 (Hendrycks et al., 2021) evaluation. For code tasks, we sample 200 training questions from AceCode (Yang et al., 2024) and evaluate out-of-domain on HumanEval+ and MBPP+ (Chen et al., 2021; Liu et al., 2023; Austin et al., 2021b). For each training question, we generate four denoising trajectories from the frozen base model, yielding only 800 sampled trajectories per dataset. Each trajectory is labeled as successful or failed using the task verifier: exact-answer matching for math and unit-test execution for code.

Baselines. We compare against offline SFT baselines and online RL-style baselines. SFT-GT trains on ground-truth solutions with standard random masking. SFT-SD uses the same generated rollout pool as Pivot-SD, but trains on successful generated solutions with ordinary masked SFT. For RL-style baselines, we include diffu-GRPO (Zhao et al., 2026) and wd1++ (Tang et al., 2026). Their budget-matched variants use the same 200-question, four-rollout, 1,000-step budget as Pivot-SD, while the larger variants use 2,000 questions and 5,000 optimizer steps.

Evaluation. All rollout-based methods collect training trajectories with a maximum generation length of 256 tokens. SFT-GT does not use model rollouts and instead trains on budget-feasible ground-truth solutions with the same response budget. We evaluate all trained models on MATH500, GSM8K, HumanEval+, and MBPP+. Unless otherwise stated, all methods use the same decoding configuration and answer-extraction or unit-test evaluation pipeline. The 256-token setting is our primary controlled setting, while 512-token evaluation is used to measure length transfer. Detailed experimental configurations and hyperparameters are provided in Appendix B.

4.1 Main Results

Table 1 reports the main comparison. We use the 256-token response budget as the primary controlled setting, since it matches the budget used to collect pivot supervision. Under this small-budget

Table 3: **Compute accounting.** Wall-clock hours measured on a 2-GPU setup. Unlike RL baselines that interleave costly online rollouts, Pivot-SD cleanly decouples the pipeline into offline generation and lightweight offline training. RL baseline times are averaged across evaluated tasks.

Method	Steps	Gen. (h)	Train (h)	Total (h)
diffu-GRPO (matched)	1000	0	5.4	5.4
diffu-GRPO	5000	0	22.9	22.9
wd1++ (matched)	1000	0	3.6	3.6
wd1++	5000	0	17.3	17.3
Pivot-SD (ours)	1000	2.5	0.3	**2.8**

Table 4: **Dream generalization results.** We keep the same baseline categories as in the LLaDA experiments and report the same evaluation metrics.

Method	M500	GSM	HE	MBPP
Base (Dream-v0-Instruct-7B)	37.97	79.55	54.27	60.58
SFT-GT	35.40	61.49	54.27	59.26
SFT-SD	40.80	70.74	48.78	61.38
Pivot-SD (ours)	**42.20**	**81.88**	**58.54**	**61.64**

setting, Pivot-SD achieves the strongest average performance across the four benchmarks. It is best on MATH, GSM8K, and HumanEval+, and remains competitive with the strongest SFT baseline on MBPP+.

The key comparison is with methods trained under the same budget. SFT-SD uses the same self-generated rollout pool as Pivot-SD but applies ordinary masked SFT to successful generated solutions. Budget-matched Diffu-GRPO and wd1++ use the same 200 training questions and 1,000 optimizer steps. Pivot-SD outperforms these budget-matched baselines overall, indicating that localizing credit and blame to high-information pivots is a more efficient use of the small rollout pool.

The 512-token columns evaluate length transfer. Pivot-SD continues to perform strongly on math reasoning and obtains the best overall average, but the code results are mixed, with HumanEval+ slightly favoring other 512-token baselines. We therefore use the 512-token results as supporting length-transfer evidence, while our main claim is based on the matched 256-token setting.

4.2 What Makes Pivot-SD Work?

Ablation setup. Table 2 isolates two design choices: the denoising state used for training and the token locations that receive credit (via cross-entropy, CE) or blame (via unlikelihood, UL). **Positive All-token** applies CE to all supervised tokens in IG-selected states from successful trajectories. **Positive Pivot** applies CE only to the IG pivot from successful trajectories. **Pos+Neg All-token** applies full-state CE/UL on IG-selected states. **Random-Step Pivot** applies the same pivot-local CE/UL objective to a token decoded at a random denoising step. **Random-Token Pivot** keeps the IG-selected state but replaces the IG pivot with a randomly selected decoded token. **Pivot-SD (ours)** applies pivot-local CE/UL to the selected high-information pivot.

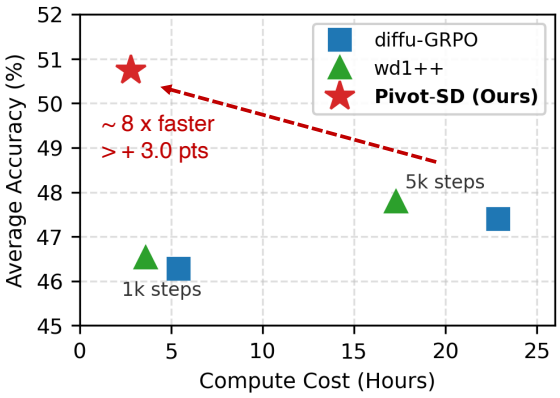

Figure 2: **Accuracy vs. Compute Cost.** Average reasoning accuracy versus total 2-GPU wall-clock hours. Pivot-SD achieves a > 3.0 absolute point improvement while requiring only a fraction of the compute time compared to fully converged RL baselines.

Table 5: **Average Out-of-Distribution (OOD) performance.** Each column reports the average accuracy on target domains unseen during the corresponding training phase. The "Overall" column denotes the micro-average across all evaluated OOD source-target pairs.

Method	OOD Average by Training Domain			Overall
	MATH	GSM8K	AceCode	
Base	50.19	35.61	53.27	45.49
SFT-GT	**53.57**	39.46	51.80	47.83
SFT-SD	51.13	35.44	51.21	45.27
diffu-GRPO (matched)	50.89	36.45	**55.49**	46.62
diffu-GRPO	53.02	37.99	54.66	47.79
wd1++ (matched)	50.71	38.06	53.76	46.73
wd1++	52.55	37.60	53.81	47.25
Pivot-SD (ours)	52.87	**39.82**	55.42	**48.61**

Pivot-local credit assignment matters. Table 2 ablates where the supervision is applied. Positive All-token and Pos+Neg All-token spread the loss over all supervised masked tokens in the selected denoising state. In contrast, Pivot-SD applies credit and blame only to the selected pivot token. The gains over all-token variants show that trajectory

feedback should not be distributed uniformly over the whole state; concentrating the update on high-information pivot decisions is more effective.

Information-gain pivot selection matters. The random pivot controls test whether any decoded token can serve as a useful local update. Random-Step Pivot chooses pivots from random denoising steps, while Random-Token Pivot keeps the IG-selected state but replaces the selected pivot with a random decoded token. Pivot-SD outperforms these controls on the overall trend, indicating that both the denoising state and the selected pivot token matter. The MBPP+ exception, where the all-token variant achieves the highest score, suggests that code completion may occasionally benefit from a broader full-output stability anchor.

4.3 Compute Efficiency

Offline pivot distillation avoids online rollout overhead. Table 3 compares the wall-clock training costs measured on a 2-GPU setup. RL-style baselines inherently interleave generation and optimization, making their online training loop substantially expensive (up to 22.9 hours for diffu-GRPO at 5,000 steps). In contrast, Pivot-SD cleanly decouples the pipeline into a small offline rollout collection stage (2.5 hours) followed by an extremely lightweight offline fine-tuning phase (0.3 hours). This brings the total time to 2.8 hours, making the method substantially cheaper than online RL-style training under the same hardware setup.

Superior compute-performance trade-off. As illustrated in Figure 2, this structural efficiency does not come at the cost of performance. Pivot-SD significantly shifts the Pareto frontier of accuracy versus compute cost. It improves the average score while requiring roughly $8\times$ less wall-clock time than the larger diffu-GRPO run. This shows that a small offline pool of 800 verifier-labeled trajectories can provide a favorable compute-performance trade-off compared with repeated online rollout collection.

4.4 Backbone Generalization

Pivot-SD transfers to another diffusion backbone. Table 4 evaluates the same training recipe on the Dream-v0-Instruct-7B base model (Ye et al., 2025). Pivot-SD improves over the Dream base model and the SFT baselines on all four benchmarks. This suggests that the benefit of pivot-local credit assignment is not specific to LLaDA, but transfers to another masked diffusion language model.

4.5 OOD Transfer

Table 5 reports cross-domain OOD averages grouped by the training domain. For MATH-trained models, we average over GSM8K, HumanEval+, and MBPP+; for GSM8K-trained models, we average over MATH, HumanEval+, and MBPP+; and for AceCode-trained models, we average over the non-code targets MATH and GSM8K. Thus, the AceCode column measures transfer from code training to math reasoning, while same-family code transfer is reported in the main results. The Overall score is the micro-average over all listed OOD source-target pairs.

Using the same small 200-question, four-rollout budget, Pivot-SD achieves the best overall OOD average. Although it is not the top method for every individual source-target group, it improves over the base model, SFT-SD, and SFT-GT on the aggregate OOD score. This suggests that pivot-local credit assignment learns transferable denoising preferences rather than merely fitting the source training questions. The full source-target breakdown is provided in Appendix C.

5 Conclusion

We introduced Pivot-SD, a data-efficient offline self-distillation framework for masked diffusion LMs that resolves the credit-assignment bottleneck via step-conditioned denoising commitments. By localizing structural bottlenecks through counterfactual information-gain, Pivot-SD applies targeted unlikelihood to erroneous pivots, converting failed trajectories into precise training signals while preserving valid reasoning sub-steps. Empirical results demonstrate that Pivot-SD outperforms sample-intensive online RL and uniform SFT baselines using only 200 training prompts, establishing a robust, compute-efficient paradigm for structured reasoning post-training.

6 Limitations

While Pivot-SD demonstrates exceptional data efficiency, future work will address three scalability paths. First, replacing the static pivot budget ($K = 10$) with an adaptive mechanism based on step-wise entropy collapse will better accommodate varying task complexities. Second, extending our step-conditioned credit assignment to long-

form generation via block-wise information-gain metrics remains a key avenue for scaling to long-context reasoning.

References

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

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

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

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

Zi-Yi Dou, Cheng-Fu Yang, Xueqing Wu, Kai-Wei Chang, and Nanyun Peng. 2024. Re-ReST: Reflection-reinforced self-training for language agents. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 15394–15411, Miami, Florida, USA. Association for Computational Linguistics.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the MATH dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Zemin Huang, Zhiyang Chen, Zijun Wang, Tiancheng Li, and Guo-Jun Qi. 2026. Reinforcing the diffusion chain of lateral thought with diffusion language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori Hashimoto. 2022. Diffusion-LM improves controllable text generation. In *Advances in Neural Information Processing Systems*.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. 2023. Is your code generated by chatGPT really correct? rigorous evaluation of large language models for code generation. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Aaron Lou, Chenlin Meng, and Stefano Ermon. 2024. Discrete diffusion language modeling by estimating the ratios of the data distribution.

Wen Luo, Feifan Song, Wei Li, Guangyue Peng, Shaohang Wei, and Houfeng Wang. 2025. Odysseus navigates the sirens' song: Dynamic focus decoding for factual and diverse open-ended text generation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 27200–27218, Vienna, Austria. Association for Computational Linguistics.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, JUN ZHOU, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. 2026. Large language diffusion models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Gray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*.

Kevin Rojas, Jiahe Lin, Kashif Rasul, Anderson Schneider, Yuriy Nevmyvaka, Molei Tao, and Wei Deng. 2026. Improving reasoning for diffusion language models via group diffusion policy optimization. In *The Fourteenth International Conference on Learning Representations*.

Subham Sekhar Sahoo, Marianne Arriola, Aaron Gokaslan, Edgar Mariano Marroquin, Alexander M Rush, Yair Schiff, Justin T Chiu, and Volodymyr Kuleshov. 2024. Simple and effective masked diffusion language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Avi Singh, John D Co-Reyes, Rishabh Agarwal, Ankesh Anand, Piyush Patil, Xavier Garcia, Peter J Liu, James Harrison, Jaehoon Lee, Kelvin Xu, Aaron T Parisi, Abhishek Kumar, Alexander A Alemi, Alex

Rizkowsky, Azade Nova, Ben Adlam, Bernd Bohnet, Gamaleldin Fathy Elsayed, Hanie Sedghi, and 21 others. 2024. Beyond human data: Scaling self-training for problem-solving with language models. *Transactions on Machine Learning Research*. Expert Certification.

Xiaohang Tang, Rares Dolga, Sangwoong Yoon, and Ilija Bogunovic. 2026. wd1: Weighted policy optimization for reasoning in diffusion language models. In *The Fourteenth International Conference on Learning Representations*.

Guanghan Wang, Gilad Turok, Yair Schiff, Marianne Arriola, and Volodymyr Kuleshov. 2026. d2: Improved techniques for training reasoning diffusion language models. *Preprint*, arXiv:2509.21474.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2024. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9426–9439, Bangkok, Thailand. Association for Computational Linguistics.

Guowei Xu, Wenxin Xu, Jiawang Zhao, and Kaisheng Ma. 2026. Gift: Guided importance-aware fine-tuning for diffusion language models. *Preprint*, arXiv:2509.20863.

Chengran Yang, Hong Jin Kang, Jieke Shi, and David Lo. 2024. Acecode: A reinforcement learning framework for aligning code efficiency and correctness in code language models. *Preprint*, arXiv:2412.17264.

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

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. 2022. STar: Bootstrapping reasoning with reasoning. In *Advances in Neural Information Processing Systems*.

Siyan Zhao, Devaansh Gupta, Qinqing Zheng, and Aditya Grover. 2026. d1: Scaling reasoning in diffusion large language models via reinforcement learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

A Trajectory-Level Entropy Analysis

In this section, we provide a more detailed visualization of how Pivot-SD identifies structural bottlenecks during the masked denoising process. As illustrated in Figure 3, the commitment of a high-impact pivot (t, p, y_p)—such as a core mathematical operator—drastically reduces the uncertainty of the remaining masked positions compared to low-impact token assignments.

B Experimental Details and Configurations

In this section, we provide detailed configurations for our training and evaluation pipelines to ensure reproducibility.

Hardware Infrastructure. All training, trajectory generation, and evaluation pipelines across all methods were executed on a hardware setup consisting of two NVIDIA L40 GPUs.

Generation and Rollout Limits. For all training methodologies—including our offline trajectory sampling for Pivot-SD, the success-only sampling for SFT-SD, and the online rollouts for the RL baselines (diffu-GRPO and wd1++)—we strictly enforce a maximum generation length of **256 tokens**. This ensures a compute-matched and controlled environment during the training phase.

Optimization and Hyperparameters. All offline methods (SFT-GT, SFT-SD, and Pivot-SD) are trained for 1,000 optimizer steps with a batch size of 8. The baseline RL methods are trained under both a budget-matched setting (1,000 steps) and an extended setting (5,000 steps) to observe convergence scaling. For Pivot-SD, we empirically adjust the negative unlikelihood weight λ_{neg} according to the task distribution. We set $\lambda_{\text{neg}} = 1$ for MATH, and $\lambda_{\text{neg}} = 5$ for GSM8K to account for an approximate 5:1 ratio of positive to negative trajectories. For coding tasks, we significantly lower the penalty to $\lambda_{\text{neg}} = 0.02$ to mitigate the impact of inherently noisy and unstructured failed code states.

Evaluation Protocol and Decoding Strategy. During the evaluation phase on MATH500, GSM8K, HumanEval+, and MBPP+, we evaluate the models under two different maximum generation length constraints: **256 tokens** and **512 tokens**. The 256-token evaluation tests the models' performance under the exact same constraints seen during training, while the 512-token evaluation assesses length generalization and the models' ability to sustain reasoning over longer contexts without suffering from degradation. We evaluate and report the performance at intermediate checkpoints of 500 and 1,000 optimization steps. To guarantee fully deterministic and reproducible evaluation results, all evaluations using the LLaDA-8B-Instruct backbone employ top-1 confidence decoding with a temperature of 0.

C Out-of-Distribution Generalization

To understand the robustness of our learned representations, we evaluate the Out-of-Distribution (OOD) generalization of models trained on a single domain and tested on unseen domains. The results are summarized in Table 6.

As shown in the table, Pivot-SD maintains highly competitive performance across unseen domains. By avoiding the destruction of locally valid representations during negative updates, our surgical credit assignment prevents catastrophic forgetting of domain-agnostic syntax and structural knowledge.

D Extended Ablation Study Settings

In Table 6 and our main experiments, we evaluate several loss variants to dissect the contribution of our pivot selection strategy. The nomenclature of our ablation variants (e.g., '[positive_loss]_[negative_loss]') is defined as follows:

- **Pivot-SD (`pivot_pivot`):** Applies positive Cross-Entropy (CE) to high-IG pivots in successful trajectories, and localized Unlikelihood (UL) to high-IG pivots in failed trajectories.
- **`uniform_uniform`:** Applies CE uniformly across all active masked tokens for successful trajectories, and UL uniformly for failed trajectories. This represents a full-sequence update lacking spatial localization.
- **`pivot` (positive-only):** Applies CE to high-IG pivots in successful trajectories but entirely drops the negative UL updates for failed trajectories.
- **`block_random_pivot_pivot`:** Retains the exact denoising step selected by Information Gain but randomly selects an active masked

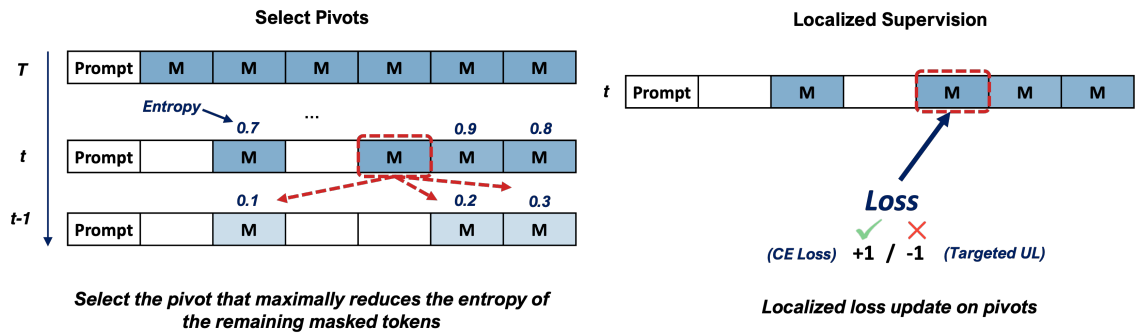

Figure 3: **Information Gain and Entropy Reduction.** Visualization of information gain and entropy reduction across masked positions after a high-impact pivot commitment. The red dashed box highlights the selected pivot commitment, which maximally reduces the predictive entropy over the remaining masked block.

Table 6: Out-of-Distribution (OOD) generalization results. Models are trained on one domain and evaluated on others. "budget-matched" denotes models trained with the 1,000-step budget configuration matching Pivot-SD.

Method	MATH Trained				GSM8K Trained				AceCode Trained		
	GSM8K	HumanEval+	MBPP+	Avg.	MATH	HumanEval+	MBPP+	Avg.	MATH	GSM8K	Avg.
SFT-GT	76.35	37.80	46.56	53.57	35.40	37.20	45.77	39.46	30.20	73.39	51.80
SFT-SD	75.74	32.93	44.71	51.13	30.60	30.49	45.24	35.44	29.40	73.01	51.21
diffu-GRPO (budget-matched)	75.21	33.54	43.92	50.89	33.80	31.10	44.44	36.45	34.40	76.57	55.49
diffu-GRPO	76.35	37.20	45.50	53.02	35.80	32.93	45.24	37.99	33.80	75.51	54.66
wd1++ (budget-matched)	76.42	32.32	43.39	50.71	35.00	36.59	42.59	38.06	32.60	74.91	53.76
wd1++	78.09	35.37	44.18	52.55	35.60	33.54	43.65	37.60	32.40	75.21	53.81
Pivot-SD (pivot_pivot)	78.09	**36.59**	43.92	**52.87**	**36.60**	**38.41**	44.44	**39.82**	**34.80**	**76.04**	**55.42**
Ablation Variants (Budget-matched)											
pivot	73.46	33.54	41.27	49.42	31.40	36.59	41.01	36.33	30.80	73.92	52.36
uniform	75.36	34.76	42.86	50.99	29.60	36.59	43.92	36.70	32.60	73.77	53.19
uniform_uniform	75.44	38.41	43.39	52.41	33.00	34.76	44.44	37.40	33.60	75.44	54.52
block_random_pivot_pivot	77.10	36.59	46.03	53.24	31.80	37.80	45.77	38.46	34.60	74.07	54.34
random_step_pivot_pivot	75.06	32.93	37.57	48.52	33.40	29.27	41.27	34.65	35.00	76.42	55.71

position as the pivot, rather than the position with highest impact.

- **`random_step_pivot_pivot`:** Randomly samples a denoising step, breaking the temporal localization provided by Information Gain, while keeping the max-entropy position within that step.

E Addressing Reward Sparsity in Low-Data RL: The wd1++ Bootstrap Analysis

A critical challenge when applying online reinforcement learning (RL) to discrete diffusion models in low-data regimes is reward sparsity. During our experiments with the `wd1++` baseline on the 200-prompt AceCode subset ($s200$), we observed a pathological training collapse.

Because AceCode is a highly challenging domain, the base model frequently failed to pass any unit tests across all generated samples within a batch. Consequently, the group reward was entirely zero, rendering the advantage uniformly zero. Under the `wd1++` dual-objective formulation (which combines positive weighted likelihood and negative weighted unlikelihood), a zero-advantage batch causes the softmax weights to become uniform, perfectly canceling out the positive and negative gradients. As a result, the model experiences a complete gradient collapse, evidenced by identical checkpoint hashes and completely stagnant evaluation metrics between step 500 and 1000 (see "w/o bootstrap" in Table 7).

To ensure a rigorous and fair baseline comparison—and to verify that our proposed Pivot-SD's sample efficiency is not merely an artifact of a broken RL baseline—we implemented a **Bootstrap Reward** shaping strategy for the $s200$ `wd1++` runs. Specifically, if the unit test pass rate was zero, we injected a small dense structural reward (up to $+0.05$) if the generated code met basic syntactic requirements (e.g., proper `def` initialization, valid `return` statements, and general syntax compliance).

Variant	Math		Code (256)		Code (512)	
	MATH	GSM	MBPP+	HE+	MBPP+	HE+
w/o bootstrap						
Step 500	32.60	74.91	44.18	32.93	45.50	40.24
Step 1000	-	-	44.18	32.93	-	-
w/ bootstrap						
Step 500	33.20	74.53	44.71	31.10	46.56	39.63
Step 1000	31.20	74.45	**47.09**	**33.54**	**46.03**	**40.85**

Table 7: Effect of Bootstrap Reward on `wd1++` trained on 200 AceCode prompts. GSM and HE+ denote GSM8K and HumanEval+, respectively. Without bootstrap, the model completely stagnates between step 500 and 1000 due to zero-advantage gradient cancellation. With the bootstrap reward, training resumes, yielding a $+2.91$ improvement on MBPP+ at step 1000. However, this heavily shaped baseline still falls short of Pivot-SD.

As shown in Table 7, injecting the bootstrap reward successfully revived the learning process. The model broke out of the flatline, achieving notable performance gains by step 1000, most prominently a $+2.91$ point improvement on MBPP+ (256 length) and consistent gains in the 512-length evaluations.

We confirmed that online RL baselines like `wd1++` are inherently fragile in low-data, sparse-reward environments and require aggressive reward shaping to function. Remarkably, even when we explicitly rescue the baseline with structural dense rewards, its peak performance (47.09 on MBPP+, 33.54 on HumanEval+) is still comprehensively outperformed by our proposed Pivot-SD without requiring any such reward engineering. This further validates that the high-impact pivot selection in Pivot-SD provides a fundamentally richer and more robust learning signal than trajectory-level RL.

F Artifact Licenses and Intended Use

In our experiments, we utilize publicly available mathematical and code generation benchmarks, including GSM8K (MIT License), MATH (MIT License), HumanEval (MIT License), and MBPP (CC-BY-4.0). We also utilize the pre-trained masked diffusion language model, LLaDA, which is distributed under the Llama 3 Community License.

Our use of these artifacts strictly aligns with their intended use cases as standard structural reasoning and evaluation benchmarks for language modeling research. These datasets do not contain personally identifying information or offensive content. Furthermore, the trajectories and evaluation scripts generated during the development of Pivot-SD will be open-sourced under the MIT License to facilitate reproducibility and future research in masked diffusion language models.

G Potential Risks

While Pivot-SD significantly improves the compute efficiency and reasoning capabilities of masked diffusion language models, it carries certain potential risks that merit consideration:

- **Bias and Hallucination Amplification:** As an offline self-distillation framework, Pivot-SD relies on trajectories generated by the frozen base model. Although our method filters for successful outcomes, it may still inherit and potentially amplify subtle biases or hallucinations deeply ingrained in the base model's pre-trained representations.

- **Dependence on Automated Verifiers:** Our trajectory labeling currently relies on exact-match checking for mathematics and unit-test execution for code. Applying this framework to open-ended generation tasks without clear objective verifiers may lead to reward hacking or optimization toward superficial proxies of quality.

H Broader Impacts

The development of Pivot-SD offers practical benefits for the computational efficiency and accessibility of reasoning research in language models. Standard post-training methods, particularly online reinforcement learning (RL), are highly compute-intensive and require extensive hardware resources. By demonstrating that step-conditioned pivot supervision can extract dense learning signals from a minimal data budget (e.g., 200 prompts), Pivot-SD significantly reduces the computational overhead required to improve reasoning capabilities. In our empirical setting, this yielded an approximate 8x reduction in compute time compared to conventional RL baselines. This efficiency supports the principles of Green AI and lowers the barrier to entry, enabling researchers in resource-constrained academic institutions to conduct experiments on diffusion language model alignment and reasoning.

Additionally, the localized credit assignment mechanism presents useful properties for model alignment and safety. Standard rejection fine-tuning or full-sequence unlikelihood training risks

over-correcting the model by suppressing locally valid syntactic fragments alongside undesirable behavior. Pivot-SD's localized unlikelihood objective suggests a path for isolating and suppressing only the specific denoising commitments that lead to logical failures or policy violations, preserving the model's general capabilities. While enhancing reasoning and code synthesis naturally carries dual-use risks—such as the potential generation of malicious code or sophisticated misinformation—the offline, verifier-dependent nature of Pivot-SD is compatible with existing safety guardrails. Its sample efficiency also facilitates the rapid, low-cost deployment of defensive alignment updates to mitigate such downstream risks.