# MATCH THE CANVAS, NOT JUST THE ANSWER: PREDICTABILITY-AWARE SFT FOR DIFFUSION LANGUAGE MODELS

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

## ABSTRACT

Diffusion language models (DLMs) enable flexible parallel generation by resolving tokens according to the model's evolving predictability rather than a fixed left-to-right order. This flexibility, however, introduces training choices absent from standard autoregressive teacher forcing: which reference tokens are visible when a target is supervised and how much generation space is allocated. Standard supervised fine-tuning (SFT) typically makes these choices through random masking, independently of inference. Our analysis shows how this training–inference mismatch can misdirect supervision. Random masking can conceal prediction errors or penalize uncertainty that decoding would otherwise resolve, while the attention-visible input length can change which response formulations the model favors, even when the underlying solution is unchanged. Motivated by these findings, we propose *Predictability-Aware Canvas Selection and Supervision (PACS)*, an SFT method. PACS uses the model's own predictability to select each example's attention-visible input length and supervises reference tokens immediately before their confidence-guided revelation. Across reasoning and code-generation benchmarks, PACS consistently achieves the highest mean accuracy among the compared SFT methods, with gains of up to 12.84 percentage points over vanilla SFT on individual benchmarks. (Code is available at `https://anonymous.4open.science/r/pacs-FB4E/README.md`.)

## 1 INTRODUCTION

Diffusion language models (DLMs) offer a promising route to parallel text generation (Lou et al., 2024; Sahoo et al., 2024; Shi et al., 2024). Their flexible generation order allows tokens to be generated in parallel according to the model's evolving predictability rather than a fixed left-to-right schedule.

At inference time, a widely used way to realize this flexibility is *confidence-guided decoding*, which uses token-level confidence to decide which predictions to commit (Wu et al., 2026a). Standard autoregressive (AR) decoding extends a contiguous prefix one token at a time. Masked DLMs such as LLaDA (Nie et al., 2025) and Dream (Ye et al., 2025), by contrast, begin with an allocated response region filled with `[MASK]` tokens. We call the full attention-visible input—the prompt, visible response tokens, and remaining masked positions—the *canvas*, illustrated in Fig. 1. As confident predictions replace selected masks, the canvas evolves and conditions subsequent predictions.

At training time, this flexibility makes DLM SFT inherently more difficult to specify. In AR teacher forcing, once a reference answer is given, the context for predicting each target is fixed by the preceding reference tokens. For a masked DLM, however, the reference fixes the target tokens but does not itself specify the canvas from which they should be predicted. The same reference can therefore be paired with different canvas lengths, visible reference tokens, and mask patterns, each defining a different prediction problem.

Full-attention masked DLMs such as LLaDA and Dream accommodate variable-length responses within a preallocated generation canvas, with `EOS` marking the response boundary. During SFT, their training recipes pad shorter examples with the same `EOS` token for batched computation. These positions remain attention-visible and participate in masking and reconstruction, teaching the model



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

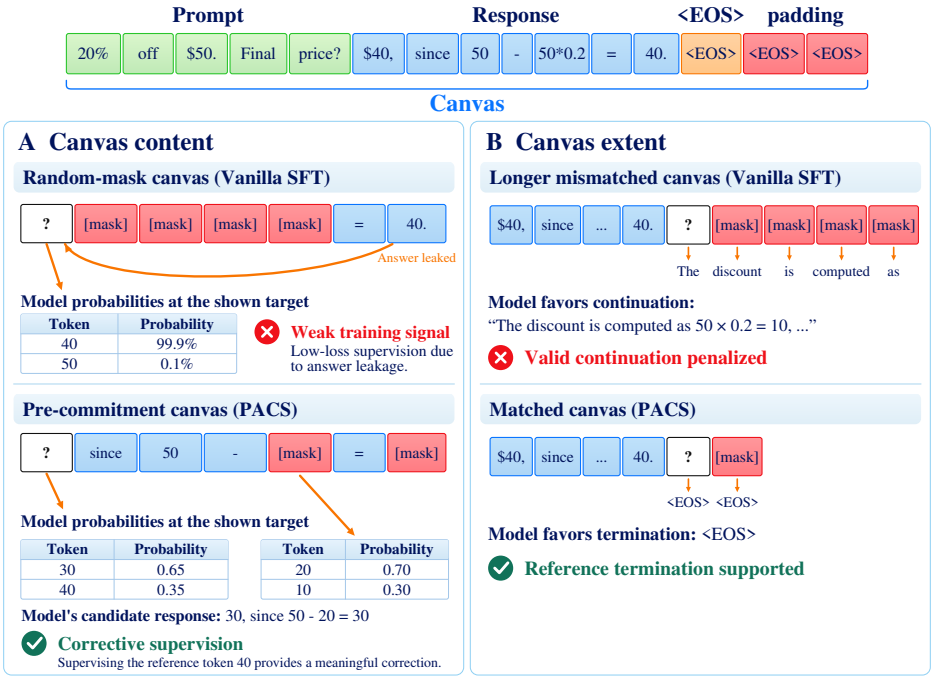



Figure 1: **The training canvas changes reference supervision. Top:** A canvas includes prompt tokens (green), visible answer tokens (blue), and masked positions (red). **A:** Visible reference content can conceal or expose a commitment-time disagreement. **B:** Canvas extent can alter the prediction problem, inducing an extra stopping signal on a mismatched canvas but not on the matched canvas. to predict `EOS` beyond the response. The allocated extent therefore contributes to the context in which the reference is predicted. As we show, this choice can introduce training signals that adjust how a correct solution is expressed, rather than correct its reasoning.

Vanilla masked-diffusion SFT typically constructs the visible context by randomly masking the reference response, independently of the model's confidence-guided decoding decisions. During inference, by contrast, those decisions determine which predictions are committed and hence what context becomes available next. We therefore study how the model's own decoding behavior can guide training-canvas construction:

*A reference answer tells us which tokens to teach—but how can the model's own predictions help us choose the training canvas?*

We investigate how the canvas's revealed content and length affect supervision for the same reference. These choices determine which reference tokens are visible when a target is supervised and how many input positions the model can attend to.

First, random masking can omit reference content that the model's reveal process would establish before selecting a target, or expose later content prematurely. These cases can respectively penalize missing-context uncertainty or conceal errors present at token selection (Figure 1A).

Second, canvas extent can change reference predictability even when the solution is fixed. In our setting, longer canvases produce longer responses (Figure 2), and changing extent shifts reference loss, particularly in later response blocks (Figure 4). Supervision may therefore compensate for differences in response formulation rather than reasoning errors; Figure 1B illustrates this at termination.

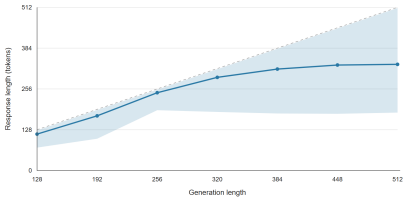



Figure 2: **Longer canvases induce longer responses.** Realized response length increases with allocated generation length.

These observations motivate supervision at token selection, rather than uniformly stronger or weaker supervision. We therefore develop PACS, which selects a canvas using commitment-level reference predictability along a confidence-guided, reference-conditioned reveal process and fine-tunes on the corresponding pre-commitment states, without changing the reference.

Our contributions are as follows:

- We identify two sources of how training-canvas choices can misdirect supervision in full-attention DLM: the visible reference content revealed when a target is supervised, and the attention-visible extent of the training canvas. Both shift the prediction problem induced by the same fixed reference.
- We introduce *PACS*, a predictability-aware SFT procedure that addresses these two mismatches by selecting the initial canvas through commitment-level reference predictability and supervising the corresponding pre-commitment states along the resulting reference-conditioned trajectory.
- Across four reasoning and code-generation benchmarks, PACS achieves the highest average accuracy among the compared SFT methods on both LLaDA-8B-Instruct and Dream-7B-Instruct at every evaluated generation budget, with individual-benchmark gains of up to 12.84 percentage points over vanilla SFT.

## 2 PRELIMINARIES

### 2.1 MASKED DIFFUSION LANGUAGE MODELS AND CANVASES

We consider full-attention masked DLMs that operate on a prompt followed by a preallocated region of token positions. This region can extend beyond the eventual response and include attention-visible padding, such as positions added for batching. At the start of decoding, all non-prompt positions contain [mask] tokens; generation proceeds by iteratively revealing selected predictions.

We call the complete attention-visible input the *canvas*. For a tokenized prompt $P$, we write the canvas as $C = (C_1, \ldots, C_L)$, where $L$ counts all attention-visible positions. The non-prompt canvas positions are $\mathcal{I}_L = \{|P| + 1, \ldots, L\}$ and contain visible tokens or [mask]. We write $p_{\theta,i}(\cdot \mid C)$ for the token distribution at position $i$ produced by the model with parameters $\theta$.

For a prompt–response pair $(P, A)$, where $A$ includes its terminating EOS token, the clean canvas has token values

$$(y_1, \ldots, y_L) = [P, A, \texttt{EOS}^{\,L-|P|-|A|}], \qquad L \geq |P| + |A|. \tag{1}$$

The EOS ending $A$ marks the reference boundary; subsequent EOS copies fill the remaining attention-visible positions. Termination and padding are thus different roles of the same token value. Here $y_i$ specifies the clean target at each non-prompt position.

### 2.2 CONFIDENCE-GUIDED DECODING

For a fixed canvas length $L$, let $C^{(s)}$ denote the canvas before decoding step $s$, initialized as $C^{(0)} = [P, \texttt{[mask]}^{\,L-|P|}]$. Full-canvas decoding makes all masked positions in $\mathcal{I}_L$ eligible for revelation. Semi-AR decoding instead processes contiguous response blocks from left to right, restricting eligibility to the first unfinished block; future blocks remain attention-visible as masks.

At each eligible position $i$, confidence is defined as the probability of the model's most likely token:

$$c_i(C) = \max_{v \in \mathcal{V}} p_{\theta,i}(v \mid C), \tag{2}$$

where $\mathcal{V}$ is the non-mask output vocabulary. For a threshold $\tau \in (0, 1)$, the commitment set $S^{(s)}$ contains eligible positions with $c_i(C^{(s)}) \geq \tau$. Their predicted tokens are revealed together to form $C^{(s+1)}$, after which confidence is recomputed.

### 2.3 TRAINING CANVASES IN SUPERVISED FINE-TUNING

During supervised fine-tuning (SFT), padding accommodates variable-length prompt–response pairs in batched training. Some recipes set $L$ to the longest prompt–response sequence in each minibatch (Hu et al., 2026; Li et al., 2025), while others use a dataset-level or otherwise globally fixed training extent (Zhao et al., 2025; Wang et al., 2026a; Parashar et al., 2026; Chen et al., 2026b).

Vanilla SFT samples a masking probability $t \sim \mathrm{Unif}(0, 1)$ and constructs a training canvas $C$ from the clean tokens in Equation 1. Each $y_i$ at a non-prompt position $i \in \mathcal{I}_L$ is independently replaced

by [mask] with probability $t$, including the EOS-padding tokens; the prompt is unchanged. The objective is

$$\mathcal{L}_{\mathrm{SFT}}(\theta) = -\mathbb{E}_{(P,A),\,t,\,C}\left[\frac{1}{t(L-|P|)}\sum_{i\in\mathcal{I}_L}\mathbf{1}[C_i=\texttt{[mask]}]\log p_{\theta,i}(y_i \mid C)\right]. \tag{3}$$

The expectation is over training pairs, masking probabilities, and sampled training canvases. Here, $\mathbf{1}[\cdot]$ denotes an indicator. Under this objective, every masked padding position is supervised to reconstruct EOS and contributes to the loss. Both masked and unmasked padding positions remain attention-visible.Thus, padding determines the attention-visible extent, while random masking determines which token values are visible and which positions receive supervision.

## 3 Related Work

**Supervised fine-tuning for diffusion language models.** Vanilla masked-diffusion SFT reconstructs randomly masked reference tokens (Sahoo et al., 2024; Shi et al., 2024; Nie et al., 2025). Because masks are sampled independently of model predictions, training contexts need not reflect confidence-guided decoding. Recent methods incorporate decoding structure or token predictability. Block-based approaches condition prediction on preceding clean blocks (Arriola et al., 2025; Chen et al., 2026b; Jain, 2026). Chen et al. (2026a) guide masking with teacher-derived difficulty rankings, while Parashar et al. (2026) adapt target selection to model-estimated difficulty and visible context. PACS shares this motivation of using model predictability to shape supervision, but follows confidence-guided decoding more directly: after each reference revelation, confidence is recomputed to determine which positions are supervised next.

**Trajectory-aware training and reference supervision.** Recent work brings training closer to confidence-guided inference by learning from states visited during the model's own decoding process (Wang et al., 2026b; Shao et al., 2026; Ren et al., 2026). For on-policy reinforcement learning and distillation, these trajectories can be followed directly because model-generated tokens remain part of subsequent states. Fixed-reference SFT cannot follow such trajectories exactly: teacher forcing replaces model predictions with reference tokens, changing the states from which later decisions are made. PACS therefore takes a middle ground, retaining reference supervision while letting model predictability guide the evolving reveal process.

**From response compatibility to canvas compatibility.** Recent SFT methods adapt supervision to the learner's response distribution through likelihood-based response selection (Zhang et al., 2026), reference rewriting (Yang et al., 2024), or selective use of correct model-generated answers (Gupta et al., 2025). These approaches seek to teach the task while limiting unnecessary changes to the model's existing response patterns. PACS extends this principle to a degree of freedom absent from standard AR teacher forcing: *the attention-visible extent of the training canvas*. In full-attention masked DLMs, changing this extent can alter reference predictability even when the solution itself is unchanged. Compatibility therefore depends not only on the reference response, but also on the canvas on which it is taught. PACS preserves the reference and selects its canvas using commitment-level reference predictability, adapting the prediction conditions rather than the answer.

## 4 How Canvas Shapes Reference Supervision

In this section, we examine the canvas mismatches that motivate PACS. We show how changing the reference content available at supervision or the canvas length changes predictions for the same reference response. These observations motivate when PACS supervises each target and how it selects the training canvas. All diagnostics use a frozen model; experimental details appear in Appendix C.

### 4.1 The Canvas Can Create or Conceal Commitment Disagreement

We construct reference-conditioned trajectories by selecting masked positions according to model confidence and revealing their reference tokens before the next selection. A target's prediction immediately before revelation is its *commitment prediction*. We call it *correct* (reference-agreeing)

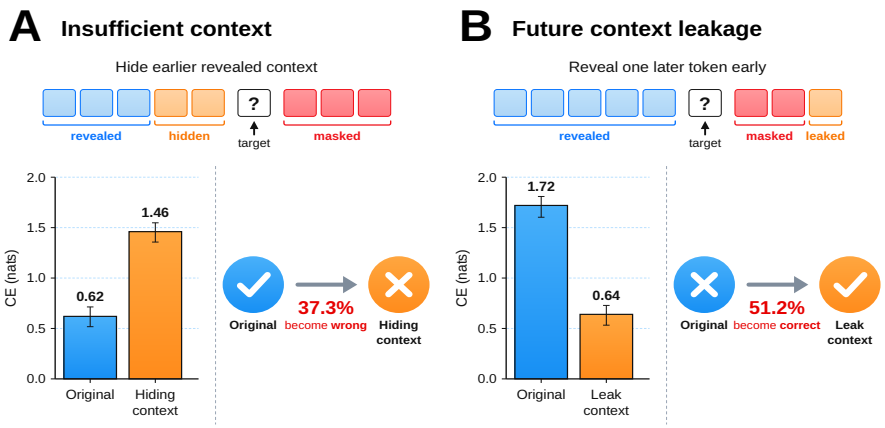



Figure 3: **Changing context around commitment can create or conceal reference disagreement.** LLaDA-8B-Instruct on GSM8K. **A: Insufficient context.** Hiding reference tokens revealed in the preceding step makes 37.3% of initially correct predictions wrong. **B: Future context leakage.** Revealing one reference token scheduled for the following reveal step early makes 51.2% of initially wrong predictions correct.

when the most likely token matches the reference, and *wrong* otherwise. Starting from this original canvas, we change the surrounding content while keeping the target masked and the canvas length fixed. Figure 3 compares mean reference CE before and after each change and reports the corresponding correct-to-wrong or wrong-to-correct transitions.

**Insufficient context.** We consider targets predicted correctly at commitment that were already eligible in the immediately preceding reveal step. For each target, we hide the reference tokens revealed in that step by replacing them with [mask], leaving the target masked and all other positions unchanged. After hiding this context, 37.3% of these initially correct predictions become wrong, and mean reference CE increases by 0.84 nats (Figure 3A). The additional supervision thus penalizes uncertainty from missing context, rather than an error at the target's original selection step.

**Future context leakage.** We instead consider targets predicted incorrectly at commitment. Starting from each target's original canvas, we reveal one reference token selected in the following reveal step of the original trajectory, leaving the target masked and all other positions unchanged. Revealing this token early makes 51.2% of these initially wrong predictions correct and reduces mean reference CE by 1.08 nats (Figure 3B). Conversely, future context leakage can weaken corrective supervision at positions the model still predicts incorrectly at commitment. By making these targets appear easier, the leaked information suppresses the reference-loss signal for errors that remain on the original canvas.

Together, these results show that canvas content can misdirect corrective supervision relative to the model's errors at commitment. These findings motivate supervising each target on the canvas present when the confidence-guided rule selects it for revelation.

### 4.2 Canvas Extent Changes Reference Predictability

Even when the reference response is unchanged, varying the canvas length can change the probabilities assigned to its tokens. Section 4.1 motivates commitment-level scoring as a way to assess how well a canvas supports the reference when its tokens are selected for revelation. We therefore using mean CE at commitment to examine how this predictability changes with canvas extent. Let $C^{(s)}$ be the canvas before reveal step $s$, $S_L^{(s)}$ the positions selected by the confidence-guided rule, and $T_L$ the total number of reveal steps:

$$\mathrm{CE}_{\mathrm{tr}}(P, A; L) = -\frac{1}{|\mathcal{R}|} \sum_{s=0}^{T_L-1} \sum_{i \in S_L^{(s)} \cap \mathcal{R}} \log p_{\theta,i}\left(y_i \mid C^{(s)}\right). \tag{4}$$

Here, $\mathcal{R}$ contains the reference positions through the first response-termination token, inclusive, and $y_i$ is the reference token at position $i$. Each target is scored once, immediately before its revelation.

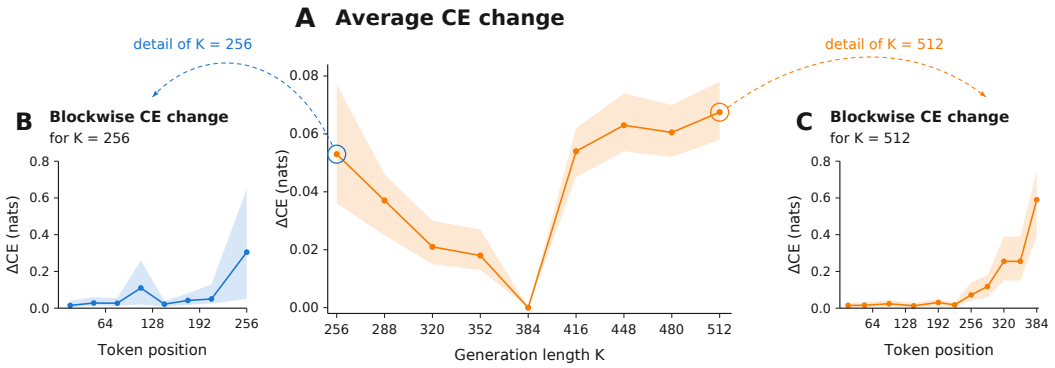



Figure 4: **Canvas-extent mismatch increases reference cross-entropy.** Target responses are generated at source generation length $K_0 = 384$, and $\Delta$CE is the change of CE compared to that evaluated at $K_0$. **A (center):** Mean $\Delta$CE across generation lengths, evaluated along each candidate's own reference-conditioned trajectory. **B (left) and C (right):** Detailed blockwise $\Delta$CE across response positions for $K = 256$ (blue) and $K = 512$ (orange).

**Algorithm 1** PACS: canvas selection and commitment-level SFT

**Require:** Training pairs $\mathcal{D}$, frozen model $p_{\theta_0}$, candidate lengths $\Lambda(P, A)$

1: **for all** $(P, A) \in \mathcal{D}$ **do**
2: Construct a trajectory $\mathcal{T}_L(P, A)$ for each $L \in \Lambda(P, A)$ using confidence-guided decoding
3: Score each candidate with $\mathrm{CE}_{\mathrm{tr}}(P, A; L)$
4: Select $L^*(P, A)$ by equation 6 and retain $\mathcal{T}_{L^*}(P, A)$
5: **end for**
6: Initialize $\theta \leftarrow \theta_0$
7: Fine-tune $p_\theta$ on the retained pre-commitment canvases, supervising only selected reference positions using equation 7
8: **return** $p_\theta$

To study this effect, we generate references at source generation length $K_0 = 384$ and reevaluate them at other lengths, where $K = L - |P|$. Each candidate uses its own reference-conditioned trajectory, and CE changes are paired against the same reference evaluated at $K_0$.

Figure 4A shows positive mean paired $\Delta$CE at both shorter and longer lengths. Panels B and C show that the excess CE is particularly pronounced in later response blocks for $K = 256$ and $K = 512$. This additional loss need not reflect a reasoning error. It may also reflect differences between the model's response behavior under a given canvas and the formulation of the fixed reference. The termination example in Figure 1B illustrates one such discrepancy. For these source-generated references, the observed sensitivity to extent motivates using $\mathrm{CE}_{\mathrm{tr}}$ as an empirical measure of compatibility between a fixed reference and a candidate canvas.

## 5 Predictability-Aware Canvas Selection and Supervision

Section 4 shows that both canvas content and canvas extent shape reference supervision. Accordingly, PACS has two components: *commitment-based canvas selection*, which chooses a canvas under which the reference is well supported along the model's reveal trajectory, and *commitment-level SFT*, which teaches the reference at the corresponding pre-commitment states.

### 5.1 Commitment-Based Canvas Selection

Recall that the clean canvas extends the reference with attention-visible `EOS` padding. When replaced by `[mask]` and included in supervision, these positions are trained to reconstruct `EOS` and contribute to the loss. PACS selects this attention-visible canvas extent for each fixed reference using commitment-level predictability, rather than inheriting it from a padding convention.

Let $\mathcal{D}$ contain the training pairs $(P, A)$. For each pair, let $\mathcal{R}$ be the scored reference positions defined in Section 4.2, and let $\Lambda(P, A)$ be a finite, nonempty set of candidate total canvas lengths satisfying

$L \geq |P| + |A|$. Each candidate retains the same prompt and reference but allocates $L - |P| - |A|$ attention-visible positions beyond the reference.

For each $L \in \Lambda(P, A)$, the frozen model $p_\theta$ constructs the reference-conditioned trajectory from Section 4.1 (details in Appendix C). The padding positions begin as [mask] and take their clean EOS values when revealed. We record the pre-commitment canvases and commitment sets as

$$\mathcal{T}_L(P, A) = \left(C^{(s)}, S_L^{(s)}\right)_{s=0}^{T_L - 1}, \tag{5}$$

with every position in $\mathcal{R}$ committed once along the trajectory.

We compare candidates using $\mathrm{CE}_{\mathrm{tr}}(P, A; L)$ from Equation 4, scoring only the same reference span $\mathcal{R}$ across all lengths. Thus, the additional padding affects the conditioning canvas but does not add targets to the selection score. We select

$$L^*(P, A) \in \operatorname*{arg\,min}_{L \in \Lambda(P,A)} \mathrm{CE}_{\mathrm{tr}}(P, A; L), \tag{6}$$

and retain the selected canvas length and trajectory $\mathcal{T}_{L^*}(P, A)$ for SFT in Section 5.2.

### 5.2 Commitment-Level Supervised Fine-Tuning

For each training pair, let $L^* = L^*(P, A)$. The selected trajectory $\mathcal{T}_{L^*}(P, A)$ supplies the pre-commitment canvases for fine-tuning. Rather than applying loss to all unresolved positions on each canvas, we supervise only the positions selected for commitment:

$$\mathcal{L}_{\mathrm{commit}}(\theta) = -\mathbb{E}_{(P,A)\sim\mathcal{D}} \left[ \frac{1}{|\mathcal{R}|} \sum_{s=0}^{T_{L^*}-1} \sum_{i \in S_{L^*}^{(s)} \cap \mathcal{R}} \log p_{\theta,i}\left(y_i \mid C^{(s)}\right) \right]. \tag{7}$$

Each reference position in $\mathcal{R}$ is therefore supervised once along the selected trajectory.

This implements *confidence-guided teacher forcing*: the frozen model determines which positions are selected at each step, while the reference supplies the values revealed before the next prediction. The resulting canvases remain reference-consistent, but their reveal order follows the model's own confidence-guided decoding behavior. Supervising the pre-commitment prediction focuses the loss on the prediction problem encountered when the decoder elects to commit. The method therefore changes the context in which each reference token is taught, rather than removing difficult targets.

Canvas selection and commitment-level SFT operate on the same conditional prediction tasks: candidate canvases are assessed through their pre-commitment reference predictions, and the selected trajectory supplies those same conditioning states for fine-tuning. Integrating these two components, Algorithm 2 summarizes the complete decoding-aware SFT procedure.

## 6 Experiments

We evaluate downstream generation, isolate the contribution of each PACS component to examine how canvas selection changes reference supervision.

### 6.1 Experimental Setup

We evaluate LLaDA-8B-Instruct and Dream-7B-Instruct on GSM8K, MATH500, MBPP, and HumanEval. Each checkpoint is evaluated at $K_{\mathrm{eval}} \in \{128, 256, 512\}$. Tables 1 and 2 show the $K_{\mathrm{eval}} = 256$ results; bold marks the highest score in each benchmark column. Complete three-length scores and the relative percentage gains $\Delta$ of PACS over the corresponding initial checkpoint appear in Tables A.1 and A.2. Additional training details are provided in Section A.1.

**Data and comparisons.** For a fair comparison, all SFT methods use the same training data, prepared from the training splits of GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021) and from OpenCoder (Huang et al., 2025). We follow the reference-guided response-construction procedure of ReGFT (Wu et al., 2026b), using confidence-guided decoding (Wu et al., 2026a) to generate model-written reference responses. We compare vanilla random-mask SFT(Sahoo et al., 2024; Shi et al., 2024; Nie et al., 2025), semi-AR block SFT (Jain, 2026; Chen et al., 2026b), LIFT(Parashar et al., 2026), TRIMS(Chen et al., 2026a), and PACS.

**Canvas search.** For each pair $(P, A)$, let $L_{\text{ref}} = |P| + |A|$ be the minimum feasible total canvas length and $L_{\min} = B\lceil L_{\text{ref}}/B \rceil$ its rounded value, with $B = 32$. We consider the nested families

$$\Lambda_K(P, A) = \begin{cases} \{L_{\text{ref}}\}, & K = 1, \\ \{L_{\text{ref}}\} \cup \{L_{\min} + jB : j = 0, \ldots, K-2\}, & K = 2, \ldots, 5. \end{cases}$$

Duplicate lengths are evaluated once. Our main family is $\Lambda_{\text{main}} = \Lambda_4$. PACS selects each example's canvas using commitment-level reference CE (Section 5.1); the search extent is chosen on held-out validation (Section 6.4).

**Main results.** PACS outperforms vanilla SFT in 22 of 24 model–benchmark–length settings and ties in the remaining two, achieving the highest accuracy among all compared methods in 19 settings (Tables A.1 and A.2). At $K_{\text{eval}} = 256$, it improves MATH500 by 2.00 and 5.00 percentage points and HumanEval by 9.15 and 6.71 points on LLaDA and Dream, respectively (Tables 1 and 2). The full results show that these gains extend across both backbones and all three generation budgets.

Table 1: LLaDA-8B-Instruct results (%) at $K_{\text{eval}} = 256$.

| Method | GSM8K | MATH500 | MBPP | HumanEval |
|---|---|---|---|---|
| Initial | 75.44 | 36.40 | 40.86 | 37.80 |
| Vanilla SFT | 76.27 | 37.00 | 49.03 | 38.41 |
| Semi-AR block | **78.20** | 38.40 | 48.60 | 43.90 |
| LIFT | 77.63 | 37.00 | 46.69 | 39.63 |
| TRIMS | 77.63 | 38.00 | **52.92** | 40.24 |
| PACS | 77.94 | **39.00** | 49.42 | **47.56** |

Table 2: Dream-7B-Instruct results (%) at $K_{\text{eval}} = 256$.

| Method | GSM8K | MATH500 | MBPP | HumanEval |
|---|---|---|---|---|
| Initial | **80.52** | 43.80 | 45.53 | 51.83 |
| Vanilla SFT | 80.06 | 44.20 | 60.31 | 54.27 |
| Semi-AR block | 80.21 | 43.60 | 61.09 | 58.53 |
| LIFT | 80.36 | 44.80 | 50.58 | 52.44 |
| TRIMS | 80.44 | 45.60 | 52.14 | 53.66 |
| PACS | **80.52** | **49.20** | **61.48** | **60.98** |

### 6.2 Component Analysis

We cross two canvas rules with random or confidence-guided ordering. Within each canvas rule, random ordering matches the corresponding confidence trajectory's per-step reveal counts. Appendix C.3 details the schedule-matched control.

Table 3 shows that canvas selection improves both reveal-order variants, while confidence-guided ordering outperforms schedule-matched random ordering under either canvas rule. Combining both components achieves the highest accuracy on all four benchmarks, supporting complementary benefits from selecting the canvas extent and the pre-commitment states used for supervision.

Table 3: Contributions of canvas selection and reveal ordering. LLaDA at $K_{\text{eval}} = 256$.

| Canvas rule | Reveal order | GSM8K | MATH500 | MBPP | HumanEval |
|---|---|---|---|---|---|
| Batch-max | Random | 76.27 | 37.00 | 45.14 | 39.63 |
| Batch-max | Confidence-guided (ours) | 77.48 | 38.00 | 47.08 | 40.90 |
| Trajectory-selected (ours) | Random | 77.03 | 37.40 | 47.86 | 44.51 |
| Trajectory-selected (ours) | Confidence-guided (ours) | 77.94 | 39.00 | 49.42 | 47.56 |

### 6.3 Canvas Choice and Reference Predictability

We compare predictability-based canvas selection with length rules motivated by existing SFT recipes. Batch-max uses the longest example in each batch, representing batch-dependent padding. Dataset-max uses a shared maximum length across training examples, providing a control for globally specified training extents. We additionally include Random visible length, which samples feasible canvas lengths without predictability-based selection (Ye et al., 2025). Figure 5 compares target CE and downstream test performance after SFT on MATH500 and HumanEval at $K_{\text{eval}} = 256$.

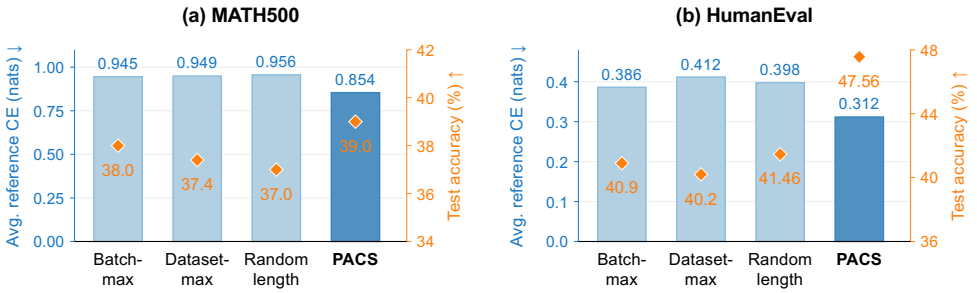



Figure 5: **Canvas choice and reference predictability.** Target CE (blue bars, left) and post-SFT test accuracy (orange diamonds, right) for four canvas-length rules on MATH500 and HumanEval with LLaDA-8B-Instruct at $K_{\text{eval}} = 256$.

With the reference responses unchanged, PACS yields lower reference CE and higher test accuracy than batch-max, dataset-max, and random visible lengths (Figure 5).

### 6.4 Ablation on Canvas Search Extent

We vary the nested candidate family $\Lambda_K$ defined in Section 6.1, holding the commitment-level reference CE selection criterion fixed.

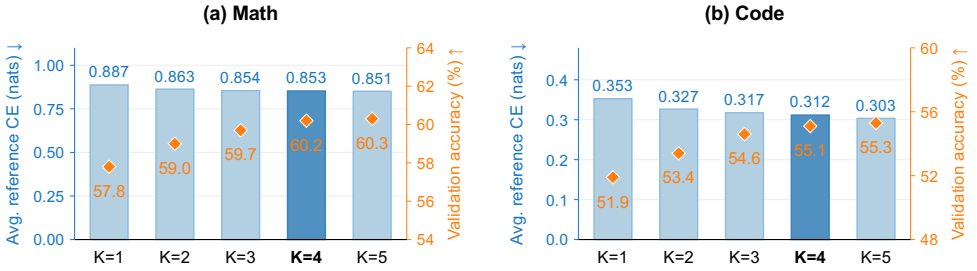



Figure 6: Canvas search extent at $K_{\text{eval}} = 256$. Bars show average reference CE (left axis); the line shows held-out validation accuracy (right axis).

Most validation improvement is obtained by our main family $\Lambda_4$; expanding to $\Lambda_5$ adds only 0.1 percentage points on Math and 0.2 points on Code (Figure 6). A compact candidate family therefore captures most of the observed benefit while limiting per-example search.

## 7 Conclusion

This work highlights training-canvas construction as an integral part of masked-DLM fine-tuning: revealed reference content and attention-visible extent shape the learning problem even when the answer is fixed. PACS couples canvas selection and reference-consistent pre-commitment supervision through commitment-level reference predictability, bringing the model's decoding behavior into the construction of its training tasks. Experiments across mathematical reasoning, code generation, and multiple generation lengths support this decoding-aware perspective on supervision. More broadly, this work opens promising directions for future research on predictability-aware canvas construction across diverse DLM architectures, including block diffusion and hybrid diffusion–AR models.

## AI USE STATEMENT

AI assistance with manuscript writing was limited to language polishing, including improvements to grammar, clarity, and readability. The authors reviewed all AI-assisted revisions and take full responsibility for the final manuscript.

## REFERENCES

Marianne Arriola, Aaron Gokaslan, Justin Chiu, Zhihan Yang, Zhixuan Qi, Jiaqi Han, Subham Sahoo, and Volodymyr Kuleshov. Block diffusion: Interpolating between autoregressive and diffusion language models. In *International Conference on Learning Representations*, volume 2025, pp. 50726–50753, 2025.

Lingjie Chen, Ruizhong Qiu, Yuyu Fan, Yanjun Zhao, and Hanghang Tong. TRIMS: Trajectory-ranked instruction masked supervision for diffusion language models. *arXiv preprint arXiv:2604.00666*, 2026a. URL https://arxiv.org/abs/2604.00666.

Zigeng Chen, Gongfan Fang, Xinyin Ma, Ruonan Yu, and Xinchao Wang. dParallel: Learnable parallel decoding for dLLMs. In *The Fourteenth International Conference on Learning Representations*, 2026b. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/57250222014c35949476f3f272c322d2-Abstract-Conference.html.

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.

Sonam Gupta, Yatin Nandwani, Asaf Yehudai, Dinesh Khandelwal, Dinesh Raghu, and Sachindra Joshi. Selective self-to-supervised fine-tuning for generalization in large language models. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 6255–6264, 2025.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021.

Yuezhou Hu, Harman Singh, Monishwaran Maheswaran, Haocheng Xi, Coleman Hooper, Jintao Zhang, Aditya Tomar, Michael W. Mahoney, Sewon Min, Mehrdad Farajtabar, Kurt Keutzer, Amir Gholami, and Chenfeng Xu. Residual Context Diffusion Language Models. *arXiv preprint arXiv:2601.22954*, 2026. URL https://arxiv.org/abs/2601.22954v2.

Siming Huang, Tianhao Cheng, Jason Klein Liu, Weidi Xu, Jiaran Hao, Liuyihan Song, Yang Xu, Jian Yang, Jiaheng Liu, Chenchen Zhang, Linzheng Chai, Ruifeng Yuan, Xianzhen Luo, Qiufeng Wang, YuanTao Fan, Qingfu Zhu, Zhaoxiang Zhang, Yang Gao, Jie Fu, Qian Liu, Houyi Li, Ge Zhang, Yuan Qi, Xu Yinghui, Wei Chu, and Zili Wang. OpenCoder: The open cookbook for top-tier code large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 33167–33193, 2025. doi: 10.18653/v1/2025.acl-long.1591. URL https://aclanthology.org/2025.acl-long.1591/.

Gagan Jain. Adaptive block diffusion: Resolving training-inference mismatch in diffusion language models, 2026.

Shufan Li, Konstantinos Kallidromitis, Hritik Bansal, Akash Gokul, Yusuke Kato, Kazuki Kozuka, Jason Kuen, Zhe Lin, Kai-Wei Chang, and Aditya Grover. LaViDa: A Large Diffusion Language Model for Multimodal Understanding. *arXiv preprint arXiv:2505.16839*, 2025. URL https://arxiv.org/abs/2505.16839v4.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32819–32848. PMLR, 21–27 Jul 2024.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-1689. Oral presentation.

Shubham Parashar, Atharv Chagi, Jacob Helwig, Lakshmi Jotsna Madhavarapu, Sushil Vemuri, James Caverlee, Dileep Kalathil, and Shuiwang Ji. Learnability-informed fine-tuning of diffusion language models. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=IgzMfT6eUx.

Haolin Ren, Ziyang Huang, Chenhao Yuan, Jun Zhao, and Kang Liu. Trace-based on-policy distillation for masked diffusion language models. *arXiv preprint arXiv:2607.16872*, 2026. URL https://arxiv.org/abs/2607.16872.

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. In *Advances in Neural Information Processing Systems*, volume 37, pp. 130136–130184, 2024. doi: 10.52202/079017-4135. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/eb0b13cc515724ab8015bc978fdde0ad-Abstract-Conference.html.

Yawen Shao, Jie Xiao, Kai Zhu, Yu Liu, Hongchen Luo, Xueyang Fu, Yang Cao, Wei Zhai, and Zheng-Jun Zha. Back on track: Aligning rewards and states for reasoning in diffusion large language models. *arXiv preprint arXiv:2606.08501*, 2026.

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

Shuaidi Wang, Zhan Zhuang, Ruping Huang, and Yu Zhang. NaRA: Noise-Aware LoRA for Parameter-Efficient Fine-Tuning of Diffusion LLMs. *arXiv preprint arXiv:2605.29716*, 2026a. URL https://arxiv.org/abs/2605.29716v1.

Yinjie Wang, Ling Yang, Bowen Li, Ye Tian, Ke Shen, and Mengdi Wang. Revolutionizing reinforcement learning framework for diffusion large language models. In *International Conference on Learning Representations*, volume 2026, pp. 101476–101501, 2026b.

Chengyue Wu, Hao Zhang, Shuchen Xue, Zhijian Liu, Shizhe Diao, Ligeng Zhu, Ping Luo, Song Han, and Enze Xie. Fast-dllm: Training-free acceleration of diffusion llm by enabling kv cache and parallel decoding. In *International Conference on Learning Representations*, volume 2026, pp. 57027–57051, 2026a.

Yangzhen Wu, Shanda Li, Zixin Wen, Xin Zhou, Ameet Talwalkar, Yiming Yang, Wenhao Huang, and Tianle Cai. Learn hard problems during rl with reference guided fine-tuning. *arXiv preprint arXiv:2603.01223*, 2026b.

Zhangchen Xu, Yang Liu, Yueqin Yin, Mingyuan Zhou, and Radha Poovendran. Kodcode: A diverse, challenging, and verifiable synthetic dataset for coding. *arXiv preprint arXiv:2503.02951*, 2025.

Zhaorui Yang, Tianyu Pang, Haozhe Feng, Han Wang, Wei Chen, Minfeng Zhu, and Qian Liu. Self-distillation bridges distribution gap in language model fine-tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1028–1043, 2024.

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

Dylan Zhang, Qirun Dai, and Hao Peng. The best instruction-tuning data are those that fit. *Advances in Neural Information Processing Systems*, 38:141172–141208, 2026.

Siyan Zhao, Devaansh Gupta, Qinqing Zheng, and Aditya Grover. d1: Scaling Reasoning in Diffusion Large Language Models via Reinforcement Learning. *arXiv preprint arXiv:2504.12216*, 2025. URL https://arxiv.org/abs/2504.12216v2.

# A Additional Experiment Details

## A.1 Training configurations

Table 4 reports the SFT hyperparameters. The shared GSM8K, MATH, and OpenCoder training corpus is described in Section 6.1; PACS's four-candidate canvas family with $B = 32$ is defined in Section 6.1. Decoding settings and prompt formats are given in Sections A.2 and A.3.

Table 4: Hyperparameters used for PACS.

| **Hyperparameter** | **Value** |
|---|---|
| Learning rate | $5 \times 10^{-6}$ |
| Learning rate scheduler type | Cosine |
| Adam $\beta$ parameters | $\beta_1 = 0.9,\ \beta_2 = 0.999$ |
| Gradient accumulation steps | 4 |
| Per-device train batch size | 2 |
| Epochs | 1 |
| Precision | bf16 |
| LoRA $r$ | 32 |
| LoRA $\alpha$ | 64 |
| Weight decay | 0.0 |
| Maximum gradient norm | 1.0 |
| Warmup ratio | 5% |
| Random seed | 42 |

## A.2 Decode Configuration

The following settings are shared by all four benchmarks.

| **Parameter** | **Value** |
|---|---|
| `gen_length` | $\{128, 256, 512\}$ |
| `block_length` | 32 |
| `temperature` | 0.0 (greedy) |
| `cfg_scale` | 0.0 |
| `confidence_threshold` | 0.9 |
| `seed` | 42 |

At temperature zero, `add_gumbel_noise` is a no-op. Decoding uses blockwise confidence-threshold parallel unmasking: at each forward pass, every masked position in the active block whose top-1 softmax probability is at least 0.9 is committed. If no position meets the threshold, the single highest-confidence masked position is committed Wu et al. (2026a).

The same confidence-guided selection rule is used to construct PACS's reference-conditioned training trajectories: the frozen model determines which positions are revealed at each step, while the reference supplies their token values.

### A.3 GENERATION PROMPTS

All four benchmarks use a single user turn, with no added task instruction or few-shot exemplars.

```
msgs = [{"role": "user", "content": benchmark_item}]
s = tokenizer.apply_chat_template(msgs, add_generation_prompt=True, tokenize=
↪ False)
```

#### A.3.1 HUMANEVAL

**LLaDA-8B-Instruct** Source:

```
<|startoftext|><|start_header_id|>user<|end_header_id|>

from typing import List


def has_close_elements(numbers: List[float], threshold: float) -> bool:
    """ Check if in given list of numbers, are any two numbers closer to each
↪ other than
    given threshold.
    >>> has_close_elements([1.0, 2.0, 3.0], 0.5)
    False
    >>> has_close_elements([1.0, 2.8, 3.0, 4.0, 5.0, 2.0], 0.3)
    True
    """<|eot_id|><|start_header_id|>assistant<|end_header_id|>
```

#### A.3.2 MBPP

**LLaDA-8B-Instruct**

```
<|startoftext|><|start_header_id|>user<|end_header_id|>

def remove_Occ(s,ch):
    """Write a python function to remove first and last occurrence of a given
↪ character from the string.

    >>> remove_Occ('hello', 'l')
    'heo'
    >>> remove_Occ('abcda', 'a')
    'bcd'
    >>> remove_Occ('PHP', 'P')
    'H'
    """<|eot_id|><|start_header_id|>assistant<|end_header_id|>
```

### A.4 SELECTED-BIN DISTRIBUTIONS

Figure A.1 shows which canvas bins are selected within each cumulative candidate family for Math and OpenCoder. Let $T = |P| + |A|$ and $L = L_{\min}(P, A)$. The families are

$$\Lambda_1 = \{T\},$$
$$\Lambda_K = \{T\} \cup \{L + 32j \mid j = 0, \ldots, K-2\}, \qquad K \in \{2, 3, 4, 5\}.$$

Each family retains all earlier candidates; duplicate lengths are considered once. Here $K$ indexes the nominal family, not its cardinality after deduplication, and $K = 4$ is the main configuration.

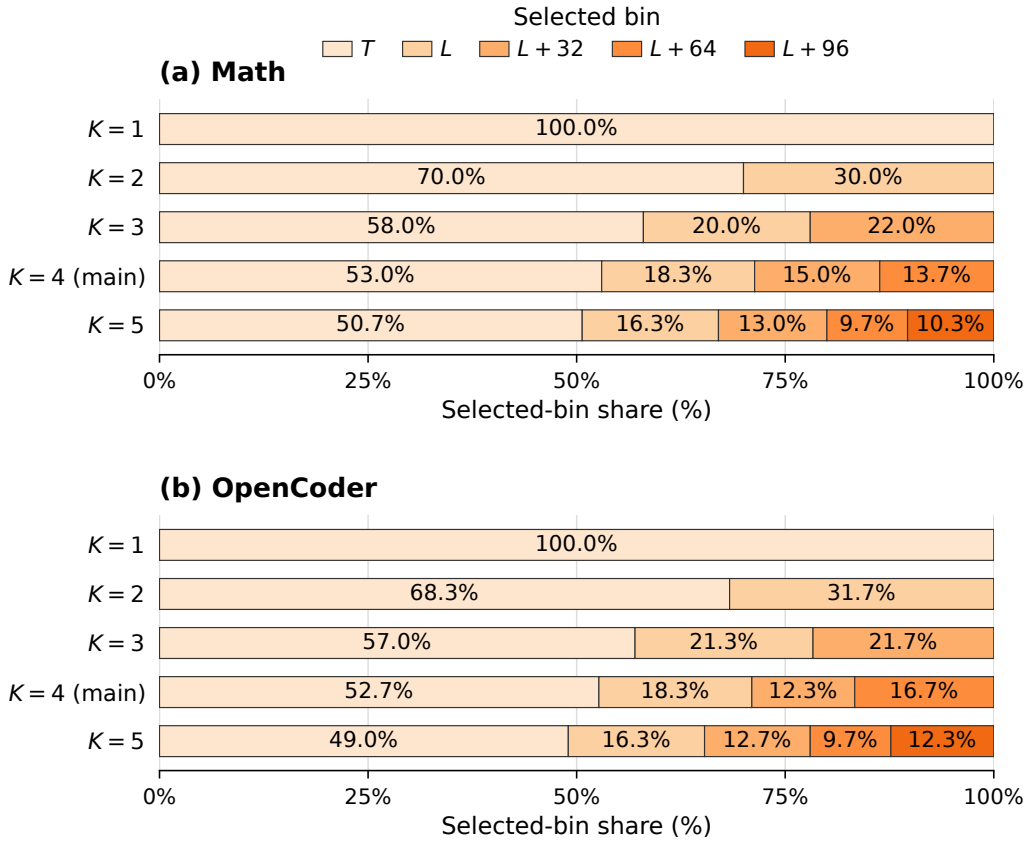



Figure A.1: **Selected-bin proportions across candidate families.** Each horizontal bar is normalized independently to $100\%$ and shows the fraction of examples for which PACS selects each candidate in the nested family $\Lambda_K(P, A)$. We use $L_{\text{ref}} = |P| + |A|$, $L_{\min} = B\lceil L_{\text{ref}}/B \rceil$, and $B = 32$, so the bins correspond to $L_{\text{ref}}$, $L_{\min}$, $L_{\min} + B$, $L_{\min} + 2B$, and $L_{\min} + 3B$. Segment labels give percentages rounded to one decimal place. Bins not contained in a given $\Lambda_K$ have no segment, and colors identify the same candidate lengths across datasets and families. $\Lambda_4$ is the main configuration.

The selected bins show that predictability-based canvas selection does not collapse to the shortest feasible canvas or to a single fixed offset. Under our main family $\Lambda_4$, PACS selects $L_{\text{ref}}$ for only $53.0\%$ of Math and $52.7\%$ of OpenCoder examples; the remaining $47.0\%$ and $47.3\%$, respectively, are distributed across $L_{\min}$, $L_{\min} + B$, and $L_{\min} + 2B$. Thus, for nearly half of the training examples, the reference is better supported on a canvas extending beyond its exact length, and the preferred extent varies across examples. This heterogeneity is precisely why a single global canvas rule is insufficient for our setting: exploiting the candidate family requires selecting, for each reference, the canvas on which its commitment-level predictions are best supported. PACS performs this selection using reference predictability rather than imposing the same extent on every example. The similar distributions across Math and OpenCoder suggest that this need for example-specific canvas selection is not confined to one domain. Finally, expanding to $\Lambda_5$ assigns only $10.3\%$ of

Math and 12.3% of OpenCoder examples to the newly added $L_{\min} + 3B$ candidate, consistent with diminishing returns from searching substantially beyond the main family.

### A.5 Complete Results Across Generation Lengths

Tables A.1 and A.2 report the complete $K_{\text{eval}} \in \{128, 256, 512\}$ comparisons. $\Delta$ is PACS's relative percentage gain over the corresponding initial checkpoint.

Table A.1: Complete LLaDA-8B-Instruct results (%) across $K_{\text{eval}} \in \{128, 256, 512\}$.

| | GSM8K | | | MATH500 | | |
|---|---|---|---|---|---|---|
| **Model / Seq Len** | **128** | **256** | **512** | **128** | **256** | **512** |
| **LLaDA-Instruct** | 76.12 | 75.44 | 79.08 | 35.60 | 36.40 | 38.80 |
| + Vanilla | 77.03 | 76.27 | 78.77 | 35.60 | 37.00 | 39.40 |
| + Semi-AR Block | 76.12 | **78.20** | 79.23 | 34.00 | 38.40 | 39.20 |
| + LIFT | 75.89 | 77.63 | 78.85 | 35.40 | 37.00 | 38.00 |
| + TRIMS | 76.88 | 77.63 | 79.53 | 34.80 | 38.00 | 39.60 |
| + PACS (Ours) | **77.10** | 77.94 | **80.97** | **35.80** | **39.00** | **40.00** |
| $\Delta(\%)$ | +1.29% | +3.31% | +2.39% | +0.56% | +7.14% | +3.09% |

| | MBPP | | | HumanEval | | |
|---|---|---|---|---|---|---|
| **Model / Seq Len** | **128** | **256** | **512** | **128** | **256** | **512** |
| **LLaDA-Instruct** | 24.12 | 40.86 | 49.81 | 28.05 | 37.80 | 43.90 |
| + Vanilla | 24.90 | 49.03 | 50.19 | 26.83 | 38.41 | 46.95 |
| + Semi-AR Block | 34.24 | 48.60 | 50.20 | 27.40 | 43.90 | **48.20** |
| + LIFT | 27.63 | 46.69 | 48.25 | 25.00 | 39.63 | 46.30 |
| + TRIMS | 34.24 | **52.92** | 50.58 | 23.78 | 40.24 | 45.73 |
| + PACS (Ours) | **37.74** | 49.42 | **52.14** | **30.49** | **47.56** | 46.95 |
| $\Delta(\%)$ | +56.47% | +20.95% | +4.68% | +8.70% | +25.82% | +6.95% |

Table A.2: Complete Dream-7B-Instruct results (%) across $K_{\text{eval}} \in \{128, 256, 512\}$.

| | GSM8K | | | MATH500 | | |
|---|---|---|---|---|---|---|
| **Model / Seq Len** | **128** | **256** | **512** | **128** | **256** | **512** |
| **Dream-Instruct-7B** | 70.66 | **80.52** | 80.74 | 37.20 | 43.80 | 44.20 |
| + Vanilla | 70.35 | 80.06 | 80.36 | **37.20** | 44.20 | 44.80 |
| + Semi-AR Block | 70.43 | 80.21 | 80.06 | 37.00 | 43.60 | 43.40 |
| + LIFT | 71.41 | 80.36 | 80.81 | 36.00 | 44.80 | 45.20 |
| + TRIMS | 71.26 | 80.44 | 80.59 | 37.40 | 45.60 | 45.40 |
| + PACS (Ours) | **72.33** | **80.52** | **80.89** | **37.20** | **49.20** | **46.20** |
| $\Delta$ (%) | +2.36% | 0.00% | +0.19% | 0.00% | +12.33% | +4.52% |

| | MBPP | | | HumanEval | | |
|---|---|---|---|---|---|---|
| **Model / Seq Len** | **128** | **256** | **512** | **128** | **256** | **512** |
| **Dream-Instruct-7B** | 31.52 | 45.53 | 46.69 | 45.73 | 51.83 | 48.17 |
| + Vanilla | 48.25 | 60.31 | 59.92 | 42.07 | 54.27 | 50.61 |
| + Semi-AR Block | **49.81** | 61.09 | 58.75 | 45.73 | 58.53 | 56.71 |
| + LIFT | 33.07 | 50.58 | 51.75 | 46.34 | 52.44 | 54.26 |
| + TRIMS | 32.30 | 52.14 | 51.36 | 41.46 | 53.66 | 56.10 |
| + PACS (Ours) | 48.64 | **61.48** | **61.48** | **47.56** | **60.98** | **57.93** |
| $\Delta$ (%) | +54.31% | +35.03% | +31.68% | +4.00% | +17.65% | +20.26% |

# B SUPPLEMENTARY EXPERIMENTS

## B.1 ROBUSTNESS TO TRAINING DATASETS

To test whether the gains of PACS depend on the training-data source, we repeat the comparison using an alternative mathematics–code corpus. For mathematical reasoning, we use Math14k, following the training setup of NaRA (Wang et al., 2026a); Math14k combines GSM8K and AQuA examples with model-generated rationales. For code generation, we use KodCode, a large-scale synthetic coding dataset with verified solutions and unit tests (Xu et al., 2025).

All methods are trained on the same retained examples and data splits under this alternative corpus, and are evaluated with the same decoding configuration as in the main experiments. This comparison tests whether the relative benefit of PACS persists when the supervision data are changed, rather than being specific to the training sources used in the main results.

Table B.1: Performance on the alternative training corpus. All methods use Math14k and KodCode and are evaluated using LLaDA-8B-Instruct at $K_{\text{eval}} = 256$. Bold marks the highest score in each benchmark column.

| Method | GSM8K | MATH500 | MBPP | HumanEval |
|---|---|---|---|---|
| Vanilla SFT | 75.82 | 36.20 | 42.41 | 36.59 |
| Semi-AR block | **77.56** | 37.60 | 41.63 | 37.80 |
| LIFT | 77.10 | 36.40 | 40.08 | 38.41 |
| TRIMS | 77.03 | 37.20 | **43.58** | 35.98 |
| PACS | 77.41 | **38.20** | 42.80 | **40.24** |

PACS improves over vanilla SFT on all four benchmarks and achieves the highest accuracy on MATH500 and HumanEval under the alternative training corpus, supporting the persistence of its benefits across training-data sources.

## B.2 SANITY REPRODUCTION OF PRIOR BASELINES

To validate our implementations of the strongest SFT baselines, we reproduce representative results from LIFT and TRIMS using LLaDA-8B-Instruct. For each method, we follow the corresponding paper's training data, number of fine-tuning epochs, and evaluation configuration rather than our PACS training recipe. For LIFT, we reproduce the reported $\text{LIFT}_3$ results on GSM8K and MATH500 using the 20-epoch training setup and $K_{\text{eval}} = 512$ (Parashar et al., 2026). For TRIMS, we reproduce its coding results on MBPP and HumanEval using the reported 30-epoch training recipe and the same decoding configuration as the original evaluation (Chen et al., 2026a).

Table B.2: Sanity reproduction of prior SFT baselines on LLaDA-8B-Instruct. Reported values are taken from the corresponding papers; each reproduction follows the training and evaluation configuration of the original result.

| **Method** | **Benchmark** | **Reported** | **Reproduction** |
|---|---|---|---|
| $\text{LIFT}_3$ | GSM8K | 79.40 | 79.30 |
| $\text{LIFT}_3$ | MATH500 | 38.40 | 38.20 |
| TRIMS | MBPP | 41.80 | 41.63 |
| TRIMS | HumanEval | 37.20 | 36.59 |

The reproduced scores are within 0.61 percentage points of the corresponding reported results across all four comparisons (Table B.2). This close agreement provides a sanity check for our implementations before the controlled comparisons in the main experiments.

For the controlled comparisons in the main experiments, we retain each baseline's paper-selected training configuration rather than retuning it under the PACS setup. For LIFT, we use the $\text{LIFT}_3$ variant reported as the strongest configuration in the original study; TRIMS likewise uses the configuration selected in its paper. This keeps the baseline comparisons tied to the methods' reported training choices while changing only the shared data and evaluation protocol used in our study.

### B.3 Computational Cost

All methods use a common prepared corpus, whose construction cost is excluded from the runtime comparison. Table B.3 summarizes runtime on the OpenCoder workload. Vanilla and semi-autoregressive SFT use 20-epoch schedules. For TRIMS and LIFT, we adopt the 30-epoch and 20-epoch main training schedules reported in their respective papers (Chen et al., 2026a; Parashar et al., 2026). Vanilla SFT samples the masking probability $\alpha \sim \mathrm{Unif}(0.1, 1)$.

Table B.3: Runtime on the OpenCoder workload.

| **Method** | **Runtime** |
|---|---|
| Vanilla SFT | ≈1 h 13 min |
| Semi-AR block | ≈1 h 16 min |
| LIFT | ≈8 h 24 min |
| TRIMS | ≈1 h 21 min |
| PACS | ≈14 h 55 min |

Timings use one H200. The timed LIFT variant is $\mathrm{LIFT}_3$. The timed PACS run uses four candidate lengths.

PACS allocates additional computation to constructing and learning from successive reference-conditioned prediction states. Confidence-guided position selection is recomputed as reference tokens are revealed, and canvas selection compares this process across candidate lengths. Its higher computational cost therefore reflects multiple prediction states per reference, rather than simply more epochs of independently sampled masking configurations.

**Matched post-training budgets.** To distinguish the effect of the supervision procedure from that of additional computation, the matched-budget comparison includes both method-specific preprocessing and optimization, while excluding shared corpus construction. Comparisons use common hardware and validation-selected checkpoints, evaluated at $K_{\mathrm{eval}} = 256$. Table B.4 reports the results.

Table B.4: Compute-matched comparison on LLaDA-8B-Instruct at $K_{\mathrm{eval}} = 256$. Budgets include method-specific preprocessing and optimization, excluding shared corpus construction. Accuracy is reported on MATH500 ($n = 500$) and HumanEval ($n = 164$). Bold marks the highest score for each benchmark and budget.

| | 4 GPU-hours | | 8 GPU-hours | | 12 GPU-hours | |
|---|---|---|---|---|---|---|
| Method | MATH500 | HumanEval | MATH500 | HumanEval | MATH500 | HumanEval |
| Vanilla SFT | 37.20 | 39.02 | 37.40 | 39.63 | 37.20 | 39.02 |
| Semi-AR block | **38.40** | **43.90** | 38.60 | **44.51** | 38.60 | 43.90 |
| LIFT | 36.40 | 38.41 | 37.00 | 39.63 | 37.20 | 40.85 |
| TRIMS | 37.60 | 40.24 | 38.20 | 40.85 | 38.20 | 39.63 |
| PACS | 37.40 | 40.24 | **38.80** | 43.90 | **39.00** | **46.34** |

Table B.4 shows that PACS makes effective use of additional post-training computation. It achieves the highest MATH500 accuracy at 8 GPU-hours and leads both benchmarks at 12 GPU-hours, surpassing the strongest baseline by 0.40 percentage points on MATH500 and 2.44 points on HumanEval. These gains under matched total budgets support the effectiveness of PACS beyond simply allocating more computation. Its growing advantage at larger budgets highlights the value of directing post-training computation toward predictability-aware canvas selection and supervision.

### B.4 Canvas Selection under Other Supervision Rules

Within each supervision pair, we hold the references, loss support, normalization, and optimization fixed while changing only the canvas rule. Table B.5 compares batch-max and PACS-selected lengths under random-mask SFT, semi-AR block SFT, and confidence-guided reference revelation. Figure B.1 further compares four canvas-length rules under semi-AR block SFT.

Table B.5: Canvas selection under different supervision rules. LLaDA-8B-Instruct at $K_{\text{eval}} = 256$. For trajectory-selected rows, canvas lengths are chosen from the same four-candidate family used in the main experiments.

| Supervision | Length rule | GSM8K | MATH500 | MBPP | HumanEval |
|---|---|---|---|---|---|
| Random-mask SFT | Batch-max | 76.27 | 37.00 | 49.03 | 38.41 |
| Random-mask SFT | Trajectory-selected | **76.50** | **37.20** | **49.42** | **39.02** |
| Semi-AR block | Batch-max | 78.20 | 38.40 | 48.60 | 43.90 |
| Semi-AR block | Trajectory-selected | **78.39** | **38.80** | **49.03** | **45.73** |
| Confidence trajectory | Batch-max | 77.48 | 38.00 | 47.08 | 40.90 |
| Confidence trajectory | Trajectory-selected | **77.94** | **39.00** | **49.42** | **47.56** |

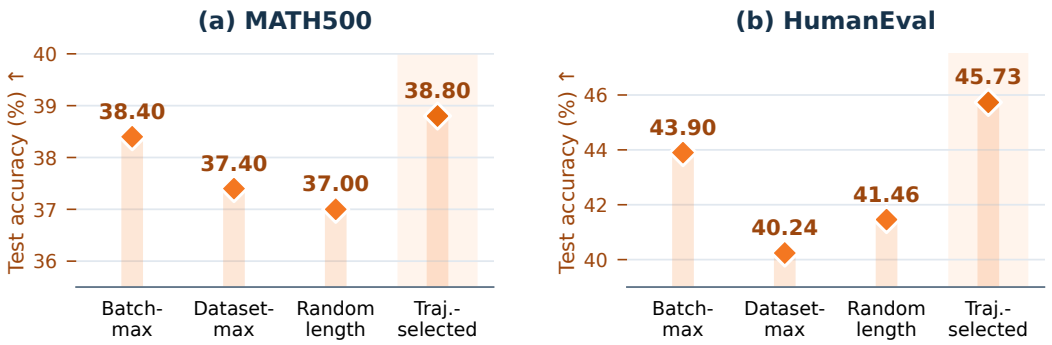



Figure B.1: **Semi-AR block canvas-choice test accuracy.** Post-SFT test accuracy (orange diamonds) on MATH500 and HumanEval at $K_{\text{eval}} = 256$ for four canvas-length rules.

PACS-selected canvases improve or match batch-max accuracy on all four benchmarks under each supervision rule. On HumanEval, the gains increase from 0.61 percentage points under random masking to 1.83 under semi-AR block SFT and 6.66 under confidence-guided reference revelation. The four-rule semi-AR comparison further shows that selected canvases outperform batch-max, dataset-max, and random lengths on both MATH500 and HumanEval. These results support the applicability of predictability-based canvas selection across supervision rules, with the largest observed gains when paired with confidence-guided reference revelation.

# C Details of the Canvas Diagnostics

**Experimental settings.** We use LLaDA-8B-Instruct on 300 prompts from GSM8K. Reference-conditioned trajectories follow the confidence-guided decoding rule in Section 2.3, with confidence threshold $\tau = 0.9$ and block size $B = 32$. Source responses are generated at temperature 0.

## C.1 Interventions on Revealed Content

Both interventions start from a target's original pre-commitment canvas. We leave the target masked and modify only the specified surrounding reference content. The target is then reevaluated on the modified canvas; its identity and reference value remain unchanged. For each candidate generation length $K$, we retain only examples whose scored reference length satisfies $m < K$. Both the candidate and source-length evaluations use this same subset and score the same complete reference spans. The retained subset may therefore vary across candidate lengths.

**Withholding established content.** We consider targets whose original commitment predictions agree with the reference. For a target selected at iteration $s$, we replace the reference tokens revealed at iteration $s - 1$ with [MASK], leaving all other positions unchanged. We restrict the comparison to targets that were already eligible at iteration $s - 1$. This restriction excludes targets that become eligible only after the preceding step, for example when decoding advances to a new block.

**Revealing subsequent content.** We consider targets whose original commitment predictions disagree with the reference. Starting from the original pre-commitment canvas, we reveal exactly one reference token randomly selected at iteration $s + 1$ of the original trajectory, keeping the current target masked. All other positions remain unchanged. The next step is defined by the original reveal schedule, not by sequence position.

**Reported quantities.** For each intervention, we report the fraction of targets whose reference-agreement status changes and the mean paired CE difference. The difference is computed as the intervention CE minus the original commitment CE. The two interventions use different initial cohorts: reference-agreeing targets for withholding and reference-disagreeing targets for additional revelation. Confidence intervals are obtained by prompt-level bootstrap resampling.

### C.2 Interventions on Canvas Extent

The frozen model first generates each reference response at source generation length $K_0 = 384$. We then keep the reference and model parameters fixed while varying the evaluation extent $K$, corresponding to total canvas length $L = |P| + K$.

**Fixed-trajectory control.** We retain the source reveal trajectory and reevaluate its pre-commitment predictions after changing the attention-visible canvas extent. The reveal schedule is not recomputed. This control examines the effect of extent without allowing confidence-based selection to produce a different trajectory.

**Candidate-trajectory comparison.** For each candidate length, we construct a new reference-conditioned trajectory. Both the predictions and the subsequent selections can therefore change with the canvas extent. We score each reference target on its pre-commitment canvas and aggregate using Equation 4.

For each construction, Figure 4A reports the change in mean CE relative to the source-length evaluation. Figure 4B reports the corresponding position-wise changes for candidate trajectories at $K = 256$ and $K = 512$. The increase toward later response positions is consistent with the continuation and termination effects illustrated in Figure 1B.

### C.3 Schedule-Matched Random-Order Control

We construct a random-order trajectory that matches the reveal schedule of the corresponding frozen PACS trajectory while removing confidence-based position selection.

For block $b$ and step $s$, let

$$q_{b,s} = \left|S_{b,s}^{\mathrm{PACS}}\right|$$

denote the number of positions selected by PACS at that step. Let $\mathcal{E}_{b,s}^{\mathrm{rand}}$ be the currently eligible masked positions in the active block of the random trajectory. We sample

$$S_{b,s}^{\mathrm{rand}} \sim \mathrm{Unif}\left\{S \subseteq \mathcal{E}_{b,s}^{\mathrm{rand}} : |S| = q_{b,s}\right\}. \tag{8}$$

Thus, PACS and the random-order control reveal the same number of positions at every step, but differ in which eligible positions are selected.

For each $i \in S_{b,s}^{\mathrm{rand}}$, the reference-token loss is computed on the pre-revelation canvas. The selected masks are then replaced by their corresponding reference values before the next step:

$$C_i^{(s+1)} = \begin{cases} y_i, & i \in S_{b,s}^{\mathrm{rand}}, \\ C_i^{(s)}, & \text{otherwise.} \end{cases} \tag{9}$$

Consequently, both variants use reference-consistent intermediate canvases and identical per-step reveal counts; the only difference is whether the revealed positions are chosen by model confidence or uniformly at random.

Equivalently, the random trajectory can be constructed by sampling one uniform random permutation of the eligible positions within each block and partitioning that permutation according to the PACS reveal counts $(q_{b,1}, q_{b,2}, \ldots)$. This implements sampling without replacement while exactly preserving the number of reveals at each step.

This control is intentionally schedule matched rather than an independent random decoding rule: its purpose is to isolate the contribution of confidence-based position ordering from differences in reveal rate, trajectory length, or amount of visible reference context.

# D ALGORITHM

**Algorithm 2** Canvas selection and commitment-level SFT

**Require:** Training pairs $\mathcal{D}$, frozen model $p_{\theta_0}$, candidate sets $\Lambda(P, A)$

Initialize $\mathcal{D}_{\mathrm{PACS}} \leftarrow \varnothing$
**for all** $(P, A) \in \mathcal{D}$ **do**
    Determine the scored reference span $\mathcal{R}$ and $m = |\mathcal{R}|$
    **for all** $L \in \Lambda(P, A)$ **do**
        Construct $\mathcal{T}_L(P, A)$ using confidence-guided reference revelation
        Compute $\mathrm{CE}_{\mathrm{tr}}(P, A; L)$
    **end for**
    Select $L^*(P, A)$ using equation 6
    Add $(P, A, \mathcal{R}, m, L^*, \mathcal{T}_{L^*}(P, A))$ to $\mathcal{D}_{\mathrm{PACS}}$
**end for**
Initialize $p_\theta$ from $p_{\theta_0}$
Fine-tune $p_\theta$ on $\mathcal{D}_{\mathrm{PACS}}$ using equation 7
**return** $p_\theta$