# Self-Distilled Trajectory-Aware Boltzmann Modeling: Bridging the Training-Inference Discrepancy in Diffusion Language Models

**Kecheng Chen**[1,★], **Ziru Liu**[2,★,♠], **Xijia Tao**[3], **Hui Liu**[1], **Yibing Liu**[1], **Xinyu Fu**[2], **Shi Wu**[2], **Suiyun Zhang**[2], **Dandan Tu**[2,♣], **Lingpeng Kong**[3], **Rui Liu**[2,♣], **Haoliang Li**[1,♣]

[1]City University of Hong Kong, [2]Huawei Research, [3]The University of Hong Kong
Email: tudandan@huawei.com; liu.rui2@huawei.com; haoliang.li@cityu.edu.hk

## Abstract

Diffusion Language Models (DLMs) have recently emerged as a promising alternative to autoregressive language models, offering stronger global awareness and highly parallel generation. However, post-training DLMs with standard Negative Evidence Lower Bound (NELBO)-based supervised fine-tuning remains inefficient: training reconstructs randomly masked tokens in a single step, whereas inference follows a confidence-guided, multi-step easy-to-hard denoising trajectory. Recent trajectory-based self-distillation methods exploit such inference trajectories mainly for sampling-step compression and acceleration, often improving decoding efficiency without substantially enhancing the model's underlying capability, and may even degrade performance under full diffusion decoding. In this work, we ask whether self-distilled trajectories can be used not merely for faster inference, but for genuine knowledge acquisition. Although these trajectories lie on the pretrained DLM's own distributional manifold and thus offer a potentially lower optimization barrier, we find that naively fine-tuning on them with standard NELBO objectives yields only marginal gains. To address this limitation, we propose **T**rajectory-**A**ligned optimization via **Bo**ltzmann **M**odeling (**TABOM**), a self-distilled trajectory-based post-training framework that aligns training with the easy-to-hard structure of inference. TABOM models the inference unmasking preference as a Boltzmann distribution over predictive entropies and derives a tractable pairwise ranking objective to align the model's certainty ordering with the observed decoding trajectory. Empirically, TABOM achieves substantial gains in new domains, expands the effective knowledge boundary of DLMs, and significantly mitigates catastrophic forgetting compared with standard SFT.

## 1 Introduction

Modern Large Language Models (LLMs) have demonstrated exceptional capabilities across diverse tasks, including mathematical reasoning, coding, and multi-turn dialogue [1–3]. While the prevailing paradigm relies on autoregressive transformer architectures [4], Diffusion Language Models (DLMs) have emerged as a compelling alternative [5, 6]. By leveraging bidirectional contexts and incorporating information from all positions simultaneously, DLMs offer superior global awareness and the potential for highly parallel token generation.

To further enhance pretrained DLMs, supervised fine-tuning (SFT) [5] is commonly applied by training the model to predict randomly masked tokens in a single forward pass. However, this objective is

[1]★ Contribute equally to this work. ♣ Corresponding authors. ♠ Project Lead.

Preprint.

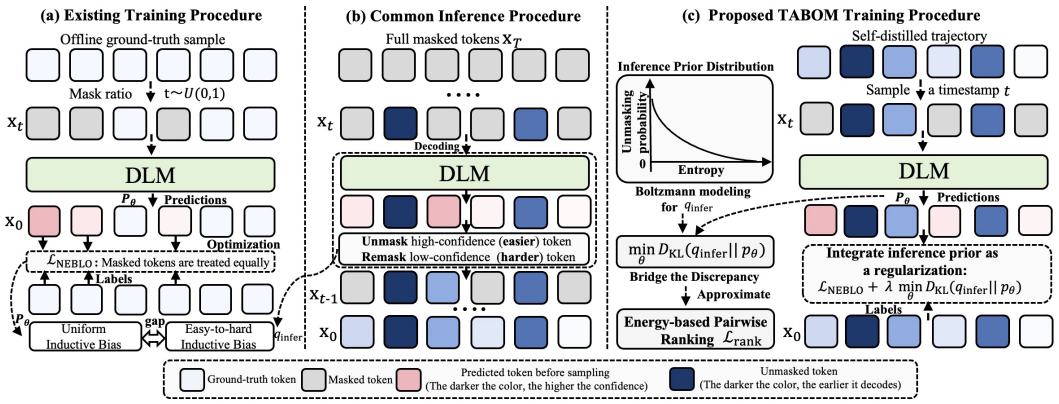



Figure 1: Overview of the proposed TABOM framework. Standard SFT trains DLMs with uniformly random masking, which induces a mismatch with the easy-to-hard unmasking trajectory used during inference. TABOM instead learns from self-distilled trajectories by combining trajectory-aware reconstruction with an energy-based pairwise ranking loss, aligning the model's entropy landscape with the inference-time Boltzmann unmasked distribution.

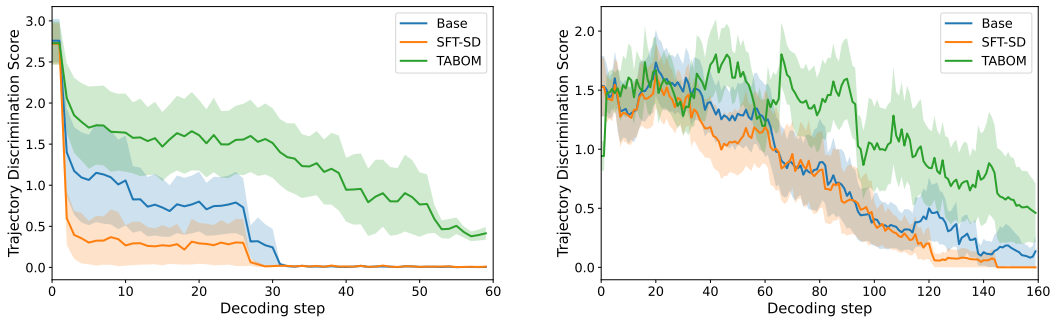
substantially less sample-efficient than autoregressive training and introduces a pronounced training-inference mismatch, since inference requires iterative denoising along a decoding trajectory. A natural remedy is therefore to directly learn high-quality decoding trajectories, which requires both a scalable trajectory generation pipeline and an efficient trajectory-level learning objective.

Although reinforcement learning (RL) may seem relevant to this goal [7–10], it is not a direct solution. Outcome-reward RL provides only coarse sequence-level supervision, making credit assignment over intermediate denoising decisions difficult. It also requires repeated rollouts and reward evaluation, which is especially costly for DLMs since each rollout already involves multi-step denoising. Recent work has explored trajectory-based self-distillation for DLM post-training [11–13], where the model learns from decoding traces generated by itself. However, existing methods are primarily designed for sampling-step compression. By learning shortcut transitions from later diffusion states to earlier ones, they mainly target *inference efficiency*, allowing DLMs to decode with fewer steps while maintaining acceptable performance trade-offs, as demonstrated by Seed Diffusion [12] and dInfer [11]. Despite these efficiency gains, recent evidence suggests (see Table 2 in Zhang et al. [13] paper) that DLMs fine-tuned on these trajectories may suffer from degraded performance under full diffusion decoding, i.e., decoding one token per step [13].

This observation motivates a more fundamental question: *Beyond improving inference efficiency, can self-distilled trajectories enable genuine knowledge acquisition and performance gains?* Intuitively, because self-distilled trajectories are generated from the pretrained DLM's own distributional manifold, they may present a lower optimization barrier than externally constructed targets, thereby facilitating smoother absorption of new knowledge during fine-tuning. However, our preliminary investigations show that naively fine-tuning DLMs on self-distilled trajectories with the standard Negative Evidence Lower Bound (NELBO) objective [8] yields only marginal improvements.

To fully exploit self-distilled trajectories for performance enhancement, we propose **Trajectory-Aligned Optimization via Boltzmann Modeling** (TABOM), a novel post-training framework for DLMs. TABOM leverages the structured decoding patterns embedded in self-distilled trajectories to align the model's predictive distribution with its actual inference-time behavior. We theoretically formulate this target behavior as a Boltzmann distribution over the ideal predictive entropy of each token, which serves as a surrogate for the easy-to-hard inductive bias and effectively mitigates the training-inference discrepancy. The main contributions of this work are summarized as follows:

- (Section 3.1) Theoretically, we demonstrate that the inference unmasked distribution under easy-to-hard decoding schedules can be explicitly modeled as a Boltzmann distribution over the ideal predictive entropy of each token. This allows us to formulate the training-inference alignment as a direct Kullback-Leibler (KL) divergence minimization problem.
- (Section 3.2) Methodologically, we introduce a tractable surrogate objective based on Pairwise Ranking to optimize the intractable global KL divergence. This mechanism enforces local entropy




(a) MBPP (b) GSM8K

Figure 2: Trajectory Discrimination Score during decoding on Dream. We compute the variance of predictive entropy over decoding-time masked tokens, average it over 64 sampled trajectories, and exclude steps after the first EOS token in each trajectory. Higher curves indicate stronger token-level uncertainty discrimination along the trajectory. “Base" denotes the original model without SFT.

gradients that strictly adhere to the easy-to-hard decoding schedule, smoothly transferring new knowledge into the model without disrupting its inherent predictive manifold.

- (Section 4.1) Experimentally, we show that TABOM resolves the central SFT dilemma identified in this paper: it turns self-distilled trajectories into substantial performance gains, while preserving the pretrained model's out-of-distribution capabilities and avoiding the catastrophic forgetting often induced by standard supervised fine-tuning.
- (Section 4.2) We introduce Trajectory Discrimination Score (TDS) to quantify whether a model preserves token-level uncertainty differences along the decoding trajectory. Qualitative (see Figure 2) and quantitative (see Table 6) TDS results reveal that TABOM reshapes the entropy landscape toward the easy-to-hard inference bias, rather than merely reusing self-distilled samples.

## 2 Empirical Observations on Self-Distilled Trajectories

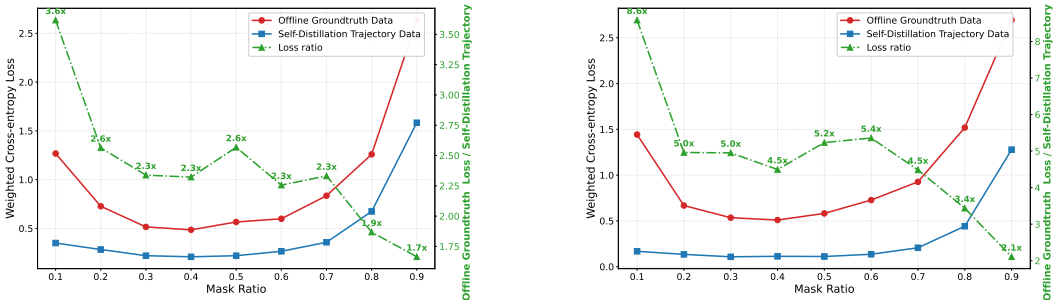
To understand the impact of self-distillation on model optimization and its limitations under standard SFT paradigms, we conduct a comparative analysis between self-distilled (SD) data and offline ground-truth (GT) data. Specifically, we conduct experiments on the Dream model [5] across two domains: code generation and mathematical reasoning. For code generation, we randomly sample 17K queries from Ling-Coder-SFT [14]. For mathematical reasoning, we utilize queries from MixChain-Z-PRM12K [15]. For each query, the offline ground-truth (GT) data consists of the standard problem-answer pairs originally provided in the datasets. In contrast, the self-distilled (SD) data is generated by having the base model decode the answers using entropy-based decoding, yielding ∼3.8K and ∼5.1K valid SD trajectories for code generation and mathematical reasoning, respectively.

| Method | HumanEval | MBPP | GSM8K | MATH500 |
|---|---|---|---|---|
| *Code SFT (Ling-Coder-SFT)* | | | | |
| No-SFT | 52.66 | 58.00 | 81.41 | 39.80 |
| SFT-GT | 61.55$_{+8.89}$ | 58.00$_{+0.00}$ | 52.33$_{-29.08}$ | 32.40$_{-7.40}$ |
| SFT-SD | 53.66$_{+1.00}$ | 59.20$_{+1.20}$ | 81.81$_{+0.40}$ | 41.60$_{+1.80}$ |
| TABOM (Ours) | 60.36$_{+7.70}$ | 60.60$_{+2.60}$ | 81.73$_{+0.32}$ | 42.40$_{+2.60}$ |
| *Math SFT (MixChain-Z-PRM12K)* | | | | |
| No-SFT | 52.66 | 58.00 | 81.41 | 39.80 |
| SFT-GT | 46.34$_{-6.32}$ | 58.00$_{+0.00}$ | 80.12$_{-1.29}$ | 37.40$_{-2.40}$ |
| SFT-SD | 57.92$_{+5.26}$ | 58.60$_{+0.60}$ | 81.95$_{+0.54}$ | 39.80$_{+0.00}$ |
| TABOM (Ours) | 58.54$_{+5.88}$ | 59.20$_{+1.20}$ | 84.31$_{+2.90}$ | 41.10$_{+1.30}$ |

Table 1: Performance comparison on Dream. SFT-SD avoids catastrophic forgetting but yields marginal in-domain gains compared to SFT-GT. TABOM achieves the best of both worlds.

**Lower Optimization Barrier.** First, we evaluate the model on the same queries using either GT data or SD data across various mask ratios. As illustrated in Figure 3, the cross-entropy (CE) loss for SD data is consistently lower than that of GT data. This suggests that self-distillation aligns the target distribution with the model's intrinsic predictive manifold, providing a smoother optimization landscape and lowering the optimization barrier.

**The Dilemma: Forgetting vs. Marginal Gains.** To investigate how this lower barrier translates to downstream performance, we evaluate the models fine-tuned on these datasets. As shown in Table




(a) Code Generation (b) Math Reasoning

Figure 3: Comparison of Cross-Entropy loss between GT and SD data across different mask ratios.

1, directly fine-tuning on GT data (SFT-GT) improves in-domain performance (e.g., HumanEval increases to 61.55) but suffers from severe catastrophic forgetting out-of-domain (e.g., GSM8K drops to 52.33). Conversely, fine-tuning on self-distilled trajectories (SFT-SD) effectively prevents catastrophic forgetting (GSM8K remains at 81.81). This is reasonable as the self-distilled trajectories are generated by the model itself, thereby preserving its inherent predictive manifold. However, the in-domain performance improvement is marginal (HumanEval with only 53.66). This empirical dilemma implies the following observation:

*Despite offering a lower optimization barrier, naively fine-tuning on self-distilled trajectories is insufficient to unlock substantial performance gains.*

## 3 Self-Distilled Trajectory-Aware Boltzmann Modeling

**Motivation.** In this paper, we argue that the fundamental training-inference discrepancy inherent in the current NELBO-based SFT paradigm leads to marginal performance gains, despite utilizing self-distilled trajectories with a lower optimization barrier. We provide specific analysis as follows.

Let $\overline{\mathbb{X}} = \mathbb{X} \cup \{\mathrm{M}\}$ be the extended vocabulary, where $\mathrm{M}$ denotes an absorbing mask token. Let $\mathbf{s} \in \mathbb{X}^L$ be a prompt sequence, and let $\mathbf{x}_t = \{x_t^1, \ldots, x_t^N\} \in \overline{\mathbb{X}}^N$ denote the response sequence at time step $t \in \{0, \ldots, T\}$, where $x_t^r$ is the token at position $r$. The process evolves from a fully masked state $\mathbf{x}_T = \{\mathrm{M}\}^N$ to a fully unmasked state $\mathbf{x}_0 \in \mathbb{X}^N$ drawn from a training dataset $\mathcal{D}$. We use $\mathcal{I} = \{1, \ldots, N\}$ for the token index set, reserve $t, t'$ for decoding timesteps, and reserve $r, s, k$ for token indices. Let $U_t \subseteq \mathcal{I}$ and $M_t = \mathcal{I} \setminus U_t$ denote the unmasked and masked index sets at timestep $t$, respectively. For convenience, we use $\mathbf{x}_0^{U_t}$ to represent the visible context formed by the unmasked tokens in $U_t$. Standard training of DLMs typically uses the NELBO objective with uniform masking, where the unmasked set $U$ is sampled uniformly to serve as the visible context, and the masked tokens in $M = \mathcal{I} \setminus U$ are reconstructed in a single step:

$$\mathcal{L}_{\mathrm{NELBO}} = \mathbb{E}_{\mathbf{x}_0 \sim \mathcal{D}} \, \mathbb{E}_{U \sim q_{\mathrm{unif}}(U)} \left[ \frac{1}{|M|} \sum_{r \in M} - \log p_\theta(x_0^r \mid \mathbf{x}_0^U, \mathbf{s}) \right] \tag{1}$$

**Proposition 1** (Uniform Inductive Bias of NELBO)**.** *Under the NELBO objective, the model is optimized to reconstruct all masked tokens simultaneously given a uniformly sampled context. This optimization inherently instills a uniform inductive bias that treats all tokens equally regardless of their inherent prediction difficulty, thereby failing to capture the sequential dependencies and certainty gradients required for coherent generation.*

Proposition 1 mainly implies the uniform inductive bias of the existing NELBO objective.

In contrast, actual inference decoding adopted by existing DLM models [16, 5, 17] is a confidence- or entropy-guided multi-step process. Specifically, starting from an empty unmasked set $U_T = \emptyset$, each reverse decoding step $t$ selects a subset of tokens $\mathcal{J}_t$ of size $b_t$ from the masked set $M_t$ to

unmask by minimizing the predictive entropy:

$$\mathcal{J}_t = \underset{\mathcal{S}\subset M_t, |\mathcal{S}|=b_t}{\arg\min} \sum_{r\in\mathcal{S}} \mathcal{H}_\theta(x_0^r \mid \mathbf{x}_0^{U_t}, \mathbf{s}), \quad U_{t-1} = U_t \cup \mathcal{J}_t. \tag{2}$$

where $\mathcal{H}_\theta$ denotes the predictive entropy, which can also be served as a smooth and robust approximation to confidence-based metrics (e.g., margin confidence), particularly given that LLM distributions tend to be highly overconfident [18]. By iteratively selecting tokens based on this certainty metric, the decoding mechanism fundamentally shapes the generation trajectory, which can be formally summarized as follows.

**Proposition 2** (Easy-to-Hard Inductive Bias of Inference)**.** *The entropy-guided decoding schedule inherently instills a strong easy-to-hard inductive bias, forming a stark contrast to the uniform inductive bias of the NELBO objective.*

*Proof.* Let the prediction difficulty of a token $r$ be quantified by its predictive entropy $\mathcal{H}_\theta(x_0^r \mid \mathbf{x}_0^{U_t}, \mathbf{s})$. A lower entropy indicates higher predictive certainty, corresponding to an "easier" token. At any step $t$, the decoding objective (Eq. 2) explicitly minimizes this entropy, thereby selecting the easiest remaining tokens in the masked set $M_t$ to transition into the next unmasked set $U_{t-1}$. Consequently, the unmasking sequence monotonically progresses from the most certain (easiest) tokens to the least certain (hardest) tokens, establishing an implicit easy-to-hard inductive bias. □

To formalize this discrepancy between the uniform training bias and the easy-to-hard inference bias, we explicitly define the target distribution generated during inference.

**Definition 1** (Inference Unmasked Distribution)**.** *For a base DLM $\theta_{\text{base}}$, let a self-distilled trajectory be $\tau = (U_T, U_{T-1}, \ldots, U_0)$, where $U_T = \emptyset$, $U_0 = \mathcal{I}$, and $U_{t-1} \supseteq U_t$. We denote the collection of such trajectories by $\mathcal{T}_{gold}$. The distribution $q_{\text{infer}}(U \mid \mathbf{x}_0, \theta_{\text{base}})$ denotes the marginal distribution over unmasked states $U$ visited by these entropy-guided trajectories. For a token $r$, $\text{step}_\tau(r)$ denotes the reverse timestep immediately before $r$ is decoded; if several tokens are decoded in the same model step, ties are resolved by their entropy ranks within that step. Thus, $q_{\text{infer}}$ concentrates probability mass on states reachable through high-certainty, easy-to-hard unmasking decisions.*

Consequently, a severe distribution discrepancy exists between training and inference: $D_{\text{KL}}(q_{\text{infer}}(U \mid \mathbf{x}_0, \theta_{\text{base}}) \,\|\, q_{\text{unif}}(U)) \gg 0$. This large divergence arises because the training distribution $q_{\text{unif}}$ assigns equal probability to all possible unmasking combinations, whereas the inference distribution $q_{\text{infer}}$ is highly peaked, concentrating its mass on a narrow set of easy-to-hard trajectories. This theoretical gap derives the following observation:

*Although directly utilizing inference trajectories for SFT provides the model with exposure to intermediate reasoning states, this fundamental distribution discrepancy prevents the model from fully exploiting the structured information embedded within the trajectories, leading to sub-optimal performance gains.*

### 3.1 Trajectory-Aligned Optimization via Boltzmann Modeling

To resolve this distribution discrepancy and fully unlock the potential of self-distilled trajectories, our core idea is to explicitly model the inference-time decoding behavior $q_{\text{infer}}(U \mid \mathbf{x}_0, \theta_{\text{base}})$ and integrate it into the SFT process. By doing so, we can reduce the discrepancy between training and inference, aligning the model's optimization landscape with its actual generation dynamics.

**Proposition 3** (Boltzmann Unmasked Distribution)**.** *Under the easy-to-hard decoding assumption, the idealized target inference distribution $q^\star_{\text{infer}}$ can be modeled as a Boltzmann distribution over the ideal predictive entropy $\mathcal{H}_{\text{ideal}}(x_0^r)$ of each token with an inverse temperature parameter $\beta$:*

$$q^\star_{\text{infer}}(U \mid \mathbf{x}_0) = \frac{1}{Z} \exp\left(-\beta \sum_{r \in U} \mathcal{H}_{\text{ideal}}(x_0^r)\right) \tag{3}$$

*where $Z = \sum_{U' \subseteq \mathcal{I}} \exp\left(-\beta \sum_{r \in U'} \mathcal{H}_{\text{ideal}}(x_0^r)\right)$ is the partition function.*

*Proof.* Intuitively, we can view a token's predictive entropy as its "cost". The easy-to-hard decoding process naturally gravitates towards paths with the lowest total cost. Therefore, an unmasked set $U$ consisting of easy tokens (low total entropy) is highly likely to be formed, whereas a set containing hard tokens is exponentially less likely. This exponential preference for low-energy states is exactly described by the Boltzmann distribution, where the probability of a state is proportional to $\exp(-\beta \cdot \text{cost})$. Substituting the total entropy $\sum_{r \in U} \mathcal{H}_{\text{ideal}}(x_0^r)$ as the cost yields Eq. 3. □

Here, $\mathcal{H}_{\text{ideal}}(x_0^r)$ denotes the ideal predictive entropy under the true generative manifold (i.e., the inherent structural uncertainty of predicting $x_0^r$ given its optimally decoded context). In practice, we approximate $\mathcal{H}_{\text{ideal}}(x_0^r)$ using the base model's predicted entropy $\mathcal{H}_{\theta_{\text{base}}}(x_0^r \mid \mathbf{x}_0^{U_t}, \mathbf{s})$ on its own self-distilled trajectories. This derived formulation assigns high probability mass to unmasked sets dominated by low-entropy tokens, and near-zero probability mass to unmasked sets consisting entirely of high-entropy tokens, effectively capturing the easy-to-hard inductive bias.

**The Ideal Objective.** Therefore, to close the gap between the uniform training distribution and the easy-to-hard inference distribution, we propose to minimize the Kullback-Leibler (KL) divergence between the model's predicted unmasked distribution $p_\theta$ and the target inference Boltzmann distribution $q^\star_{\text{infer}}$, and integrate this alignment term into the standard NELBO-based SFT objective:

$$\mathcal{L}_{\text{TD}}^{\text{ideal}} = \mathcal{L}_{\text{NELBO}} + \lambda \min_\theta D_{\text{KL}}(q^\star_{\text{infer}} \,\|\, p_\theta) \tag{4}$$

By optimizing this objective, we explicitly force the model's predictive certainty to match the actual easy-to-hard decoding trajectory. To derive a tractable form for this KL divergence, we first parameterize the model's predicted probability of an unmasked set $U$ using the energy score induced by its predictive entropies:

$$p_\theta(U \mid \mathbf{x}_0, \mathbf{s}) = \frac{1}{Z_\theta} \exp\left(-\beta \sum_{r \in U} h_\theta(r; \tau)\right), \quad h_\theta(r; \tau) \triangleq \mathcal{H}_\theta(x_0^r \mid \mathbf{x}_0^{U_{\text{step}_\tau(r)}}, \mathbf{s}), \tag{5}$$

where $\text{step}_\tau(r)$ denotes the timestep immediately before token $r$ is decoded along trajectory $\tau$. Based on this parameterization, the KL divergence term expands as follows,

$$\begin{aligned}
\min_\theta D_{\text{KL}}(q^\star_{\text{infer}} \,\|\, p_\theta) &= \min_\theta \mathbb{E}_{U \sim q^\star_{\text{infer}}} \left[\log q^\star_{\text{infer}}(U \mid \mathbf{x}_0) - \log p_\theta(U \mid \mathbf{x}_0, \mathbf{s})\right] \\
&= \max_\theta \mathbb{E}_{U \sim q^\star_{\text{infer}}} [\log p_\theta(U \mid \mathbf{x}_0, \mathbf{s})] \\
&= \max_\theta \mathbb{E}_{U \sim q^\star_{\text{infer}}} \left[-\beta \sum_{r \in U} h_\theta(r; \tau) - \log Z_\theta\right]
\end{aligned} \tag{6}$$

Directly optimizing this expectation is intractable because we cannot easily sample from the global dynamic distribution $q^\star_{\text{infer}}$, and computing the partition function $Z_\theta$ requires a combinatorial sum over all possible unmasked states.

### 3.2 The Tractable Surrogate: Pairwise Ranking

To bypass the intractable partition function and the need for global sampling, we shift our perspective from explicit likelihood maximization to implicit distribution matching via ranking.

**Theorem 1** (Energy-Based Ranking [19])**.** *Maximum likelihood estimation, which is equivalent to minimizing the KL divergence, can be effectively transformed into an energy-based ranking problem. By enforcing correct relative rankings between states, one can shape the energy landscape and implicitly match the target distribution without the intractable partition function.*

Based on Theorem 1, we derive the following lemma specifically for our objective of aligning unmasked distributions.

**Lemma 1** (Ranking Unmasked States)**.** *Let $S_\theta(U;\tau) = -\beta \sum_{r\in U} h_\theta(r;\tau)$ denote the unnormalized energy score of an unmasked set $U$ along trajectory $\tau$. To align the model distribution $p_\theta$ with the target inference distribution $q^\star_{\text{infer}}$, for any pair of unmasked sets where $q^\star_{\text{infer}}(U_b) > q^\star_{\text{infer}}(U_a)$, it is sufficient to enforce the corresponding model ranking $S_\theta(U_b;\tau) > S_\theta(U_a;\tau)$.*

Lemma 1 implies that optimizing our objective relies on identifying pairs of unmasked states with a known relative preference under the target inference distribution. Therefore, **self-distilled trajectories** provide the exact structured data required to construct these pairs and learn this ranking objective. Because these trajectories record the actual step-by-step unmasking process of the base model, they naturally reflect the relative predictive certainty between different tokens.

Specifically, consider a self-distilled trajectory $\tau \sim \mathcal{T}_{\text{gold}}$ and two token indices $r$ and $s$ decoded at different positions along this trajectory. If $r$ is decoded before $s$ (`i.e.`, $\text{step}_\tau(r) > \text{step}_\tau(s)$ under reverse-time decoding), then $r$ is treated as an easier token with lower ideal predictive entropy. Let $U_{\text{base}}$ be a shared subset of unmasked tokens, and define two unmasked sets differing by only one token: $U_r = U_{\text{base}} \cup \{r\}$ and $U_s = U_{\text{base}} \cup \{s\}$. Under the target Boltzmann distribution (Eq. 3), the unmasked set containing the easier token has a higher probability, so $q^\star_{\text{infer}}(U_r) > q^\star_{\text{infer}}(U_s)$. Enforcing corresponding model ranking $S_\theta(U_r;\tau) > S_\theta(U_s;\tau)$ allows us to bypass the intractable $Z_\theta$. Since $U_r$ and $U_s$ share the common subset $U_{\text{base}}$, their scores simplify:

$$S_\theta(U_r;\tau) > S_\theta(U_s;\tau) \implies h_\theta(r;\tau) < h_\theta(s;\tau). \tag{7}$$

To enforce this inequality, we adopt a pairwise hinge loss that penalizes the model whenever the predicted entropy of an easier token $r$ is not sufficiently lower than that of a harder token $s$. For a local trajectory segment $(U_t, U_{t'})$ with $t' < t$, define the newly unmasked token set as $\Delta_{t\to t'} = U_{t'} \setminus U_t$ and the local ordered pair set as

$$\mathcal{P}_{t,t'} = \{(r,s) : r,s \in \Delta_{t\to t'},\ \text{step}_\tau(r) > \text{step}_\tau(s)\}.$$

When $|\Delta_{t\to t'}| = W$, we have $|\mathcal{P}_{t,t'}| = W(W-1)/2$. The ranking loss over this segment is

$$\mathcal{L}_{\text{rank}}(t,t') = \frac{1}{|\mathcal{P}_{t,t'}|} \sum_{(r,s)\in\mathcal{P}_{t,t'}} \max\Big(0,\ h_\theta(r;\tau) - h_\theta(s;\tau) + \gamma\Big), \tag{8}$$

where $\gamma > 0$ is the ranking margin. Since the surrogate only depends on relative ordering, the inverse-temperature scale $\beta$ is absorbed into the margin $\gamma$ and the ranking weight $\lambda$ in practice. As guaranteed by Lemma 1, this ranking mechanism serves as an effective surrogate for the intractable KL divergence (Eq. 6). It avoids global sampling and partition function computation by implicitly matching the target Boltzmann distribution through entropy landscape shaping. Integrating this ranking regularization with the reconstruction loss, our final objective is

$$\begin{aligned}\mathcal{L}_{\text{TABOM}} = \mathbb{E}_{\tau\sim\mathcal{T}_{\text{gold}}}\,\mathbb{E}_{(t,t')\sim\tau}\Bigg[&\frac{1}{W}\sum_{k\in\Delta_{t\to t'}} -\log p_\theta(x_0^k \mid \mathbf{x}_0^{U_t}, \mathbf{s}) \\ &+ \frac{2\lambda}{W(W-1)}\sum_{(r,s)\in\mathcal{P}_{t,t'}} \max\Big(0,\ h_\theta(r;\tau) - h_\theta(s;\tau) + \gamma\Big)\Bigg].\end{aligned} \tag{9}$$

where $(t,t') \sim \tau$ means sampling a valid local segment from trajectory $\tau$ such that $t' < t$ and $|\Delta_{t\to t'}| = W$. The first term denotes the trajectory-aware reconstruction loss, which replaces the uniform masking ($U \sim q_{\text{unif}}$) with a trajectory-aware masking strategy as widely adopted by previous works [11] for fully leveraging the self-distilled trajectories. Specifically, the unmasked tokens at an earlier state $U_t$ are used as context to predict the newly unmasked tokens $\Delta_{t\to t'} = U_{t'} \setminus U_t$ in the subsequent state $U_{t'}$ ($t' < t$). Note that we do not construct the pair set globally across all unmasked tokens. Comparing tokens decoded at vastly different stages (e.g., the first vs. the last step) is less informative, as their prediction difficulties are inherently disparate (see analysis in Appendix C). Instead, we apply the ranking loss strictly over a local decoding window containing $W = |\Delta_{t\to t'}|$ newly unmasked tokens. To avoid the instability of the ranking loss that occurs when the target state is either too far from $U_t$ or too close to $U_t$, we set a fixed window size $W$ (e.g., $W = 32$). Consequently, during training, we only need to randomly sample the initial timestep $t$ and its context $U_t$; the target state $U_{t'}$ is then determined by $W$ newly unmasked tokens.

# 4 Experiments

**Models.** We employ two state-of-the-art diffusion language models as our primary experimental testbeds: Dream-7B-Instruct [5] and LLaDA-8B-Instruct [6]. These models serve as the baselines upon which we apply our post-training recipes.

**Self-distilled and Offline Ground-truth Dataset.** We focus on two core domains: mathematical reasoning and code generation. For mathematical reasoning, we utilize 12K queries from MixChain-Z-PRM12K [15]. For code generation, we sample 18K queries from Ling-Coder-SFT [14]. For each query, the offline ground-truth data consists of the standard problem-answer pairs originally provided in the datasets. To construct the self-distilled data, we generate trajectories using the base models' default decoding strategies. Specifically, we employ entropy-based decoding for Dream and confidence-based decoding for LLaDA, limiting the generation sequence length to 256 tokens.

**Training.** By following previous post-training-related counterparts [20], we train a separate model for each domain dataset. We employ LoRA [21] for parameter-efficient fine-tuning, with rank $r = 16$, LoRA scaling $\alpha = 16$, and target modules `q_proj` and `v_proj`. The training is conducted across 8 GPUs with a per-device mini-batch size of 4. We optimize the models with AdamW using a peak learning rate of 2e-5, together with a cosine learning rate decay schedule and a warm-up period of 50 steps. All models are trained for 5 epochs and we report the best results for all baselines. For TABOM, the window size $W$ is fixed to 32 across all tasks. The margin $\gamma$ and the ranking loss weight $\lambda$ in Eq. 9 are tuned via a grid search over $\{0.1, 0.2, 0.3\}$ and $\{1, 2\}$, respectively, for each task. Specific hyperparameters, training details, and evaluation setups are detailed in Appendix.

**Evaluation Tasks.** We conduct experiments on five main tasks grouped into three categories: (1) *Mathematical reasoning*: GSM8K [22] and MATH500 [23]; (2) *Code generation*: HumanEval [24] and MBPP [25]; (3) *Instruction following*: IFEval [26], which evaluates the model's ability to follow verifiable instructions. Note that for each trained model, we evaluate both its in-domain and out-of-distribution (OOD) performance. For instance, for a model trained on the code generation dataset, HumanEval and MBPP serve as in-domain evaluations, while the remaining three tasks are considered OOD.

**Baselines.** We evaluate original DLMs under its official default inference setting (**No-SFT**), and further compare our approach with four post-training baselines. The LoRA configuration and the training data for all post-training baselines are the same as our method (i.e., using the self-distilled data), except for SFT-GT which utilizes the offline ground-truth data: (1) **SFT-GT**: Standard supervised fine-tuning using offline ground-truth data, where the DLM is trained to reconstruct randomly masked tokens. (2) **SFT-SD**: Identical to SFT-GT, but trained on the self-distilled data. (3) **dInfer** [11]: Instead of learning the standard single-step transition, dInfer conducts compressed transition learning by randomly sampling two timesteps from self-distilled trajectories. (4) **T3D** [13]: Training on the self-distilled data with a randomly sampling timestep, which applies direct discriminative optimization and path-consistency weighting to the objective.

## 4.1 Main Results

Table 2 and Table 3 summarize the performance of our proposed TABOM framework compared to various baselines on the code generation and mathematical reasoning datasets, respectively. We observe several key findings: **(1) SFT-GT suffers from severe catastrophic forgetting.** While it improves in-domain performance (e.g., HumanEval on Dream code increases by 8.89%), it drastically degrades OOD capabilities (e.g., GSM8K drops by 29.08%). This highlights the optimization barrier of directly injecting offline ground-truth data into the diffusion model's manifold. **(2) SFT-SD prevents forgetting but yields marginal gains.** By utilizing self-distilled trajectories, SFT-SD effectively preserves the model's inherent predictive manifold, maintaining OOD performance. However, due to the training-inference misalignment, its in-domain improvements are minimal. **(3) Trajectory-based baselines (dInfer, T3D) offer limited improvements.** While they attempt to leverage decoding trajectories, they still struggle to fully unlock the potential of the self-distilled data, resulting in sub-optimal average gains. Finally, **TABOM achieves the best of both worlds.** By explicitly modeling the inference-time decoding behavior and aligning the predictive certainty via pairwise ranking, TABOM consistently outperforms all baselines across both base models and domains. It not only achieves the highest in-domain performance (e.g., +5.15% average gain on

Table 2: Performance comparison of models fine-tuned on the **Code Generation** dataset. We report the absolute scores and their relative gains/losses compared to the No-SFT baseline. The highest and second-highest values in each column are highlighted in **bold** and underlined, respectively.

| Method | In-Domain | | | Out-Of-Distribution (OOD) | | | |
|---|---|---|---|---|---|---|---|
| | HumanEval | MBPP | Avg. | GSM8K | MATH500 | IFEval | Avg. |
| *Base Model: Dream-7B-Instruct* | | | | | | | |
| No-SFT | 52.66 | 58.00 | 55.33 | 81.41 | 39.80 | 56.56 | 59.26 |
| SFT-GT | **61.55**$_{+8.89}$ | 58.00$_{+0.00}$ | 59.78$_{+4.45}$ | 52.33$_{-29.08}$ | 32.40$_{-7.40}$ | 46.21$_{-10.35}$ | 43.65$_{-15.61}$ |
| SFT-SD | 53.66$_{+1.00}$ | 59.20$_{+1.20}$ | 56.43$_{+1.10}$ | 81.81$_{+0.40}$ | 41.60$_{+1.80}$ | 57.10$_{+0.54}$ | **60.17**$_{+0.91}$ |
| dInfer | 57.31$_{+4.65}$ | 58.20$_{+0.20}$ | 57.76$_{+2.43}$ | **81.88**$_{+0.47}$ | 39.80$_{+0.00}$ | **57.30**$_{+0.74}$ | 59.66$_{+0.40}$ |
| T3D | 55.48$_{+2.82}$ | 58.70$_{+0.70}$ | 57.09$_{+1.76}$ | 81.84$_{+0.43}$ | 40.70$_{+0.90}$ | 57.20$_{+0.64}$ | 59.91$_{+0.65}$ |
| TABOM (Ours) | 60.36$_{+7.70}$ | **60.60**$_{+2.60}$ | **60.48**$_{+5.15}$ | 81.73$_{+0.32}$ | **42.40**$_{+2.60}$ | 55.45$_{-1.11}$ | 59.86$_{+0.60}$ |
| *Base Model: LLaDA-8B-Instruct* | | | | | | | |
| No-SFT | 36.01 | 39.20 | 37.61 | 76.12 | 36.20 | 33.08 | 48.47 |
| SFT-GT | 42.01$_{+6.00}$ | 32.80$_{-6.40}$ | 37.41$_{-0.20}$ | 70.73$_{-5.39}$ | 35.40$_{-0.80}$ | **34.93**$_{+1.85}$ | 47.02$_{-1.45}$ |
| SFT-SD | 39.63$_{+3.62}$ | 38.80$_{-0.40}$ | 39.22$_{+1.61}$ | 76.95$_{+0.83}$ | 35.80$_{-0.40}$ | 32.90$_{-0.18}$ | 48.55$_{+0.08}$ |
| dInfer | 41.46$_{+5.45}$ | 38.60$_{-0.60}$ | 40.03$_{+2.42}$ | **77.33**$_{+1.21}$ | 36.60$_{+0.40}$ | 33.82$_{+0.74}$ | 49.25$_{+0.78}$ |
| T3D | 40.54$_{+4.53}$ | 38.70$_{-0.50}$ | 39.62$_{+2.01}$ | 77.14$_{+1.02}$ | 36.20$_{+0.00}$ | 33.36$_{+0.28}$ | 48.90$_{+0.43}$ |
| TABOM (Ours) | **42.68**$_{+6.67}$ | **40.00**$_{+0.80}$ | **41.34**$_{+3.73}$ | **77.33**$_{+1.21}$ | **38.20**$_{+2.00}$ | 34.19$_{+1.11}$ | **49.91**$_{+1.44}$ |

Table 3: Performance comparison of models fine-tuned on the **Mathematical Reasoning** dataset. We report the absolute scores and their relative gains/losses compared to the No-SFT baseline. The highest and second-highest values in each column are highlighted in **bold** and underlined, respectively.

| Method | In-Domain | | | Out-Of-Distribution (OOD) | | | |
|---|---|---|---|---|---|---|---|
| | GSM8K | MATH500 | Avg. | HumanEval | MBPP | IFEval | Avg. |
| *Base Model: Dream-7B-Instruct* | | | | | | | |
| No-SFT | 81.41 | 39.80 | 60.61 | 52.66 | 58.00 | 56.56 | 55.74 |
| SFT-GT | 80.12$_{-1.29}$ | 37.40$_{-2.40}$ | 58.76$_{-1.85}$ | 46.34$_{-6.32}$ | 58.00$_{+0.00}$ | 53.23$_{-3.33}$ | 52.52$_{-3.22}$ |
| SFT-SD | 81.95$_{+0.54}$ | 39.80$_{+0.00}$ | 60.88$_{+0.27}$ | 57.92$_{+5.26}$ | 58.60$_{+0.60}$ | 56.01$_{-0.55}$ | 57.51$_{+1.77}$ |
| dInfer | 82.33$_{+0.92}$ | **41.60**$_{+1.80}$ | 61.97$_{+1.36}$ | 56.11$_{+3.45}$ | 58.80$_{+0.80}$ | 55.82$_{-0.74}$ | 56.91$_{+1.17}$ |
| T3D | 82.14$_{+0.73}$ | 40.70$_{+0.90}$ | 61.42$_{+0.81}$ | 57.01$_{+4.35}$ | 58.70$_{+0.70}$ | 55.91$_{-0.65}$ | 57.21$_{+1.47}$ |
| TABOM (Ours) | **84.31**$_{+2.90}$ | 41.10$_{+1.30}$ | **62.71**$_{+2.10}$ | **58.54**$_{+5.88}$ | **59.20**$_{+1.20}$ | **56.19**$_{-0.37}$ | **57.98**$_{+2.24}$ |
| *Base Model: LLaDA-8B-Instruct* | | | | | | | |
| No-SFT | 76.12 | 36.20 | 56.16 | 36.01 | 39.20 | 33.08 | 36.10 |
| SFT-GT | 74.29$_{-1.83}$ | 35.50$_{-0.70}$ | 54.90$_{-1.26}$ | 31.09$_{-4.92}$ | **40.60**$_{+1.40}$ | 26.98$_{-6.10}$ | 32.89$_{-3.21}$ |
| SFT-SD | 75.96$_{-0.16}$ | 35.70$_{-0.50}$ | 55.83$_{-0.33}$ | 36.58$_{+0.57}$ | 39.80$_{+0.60}$ | **34.38**$_{+1.30}$ | 36.92$_{+0.82}$ |
| dInfer | 76.72$_{+0.60}$ | 36.50$_{+0.30}$ | 56.61$_{+0.45}$ | 38.90$_{+2.89}$ | 39.40$_{+0.20}$ | **34.38**$_{+1.30}$ | 37.56$_{+1.46}$ |
| T3D | 76.34$_{+0.22}$ | 36.10$_{-0.10}$ | 56.22$_{+0.06}$ | 37.74$_{+1.73}$ | 39.60$_{+0.40}$ | **34.38**$_{+1.30}$ | 37.24$_{+1.14}$ |
| TABOM (Ours) | **78.62**$_{+2.50}$ | **36.80**$_{+0.60}$ | **57.71**$_{+1.55}$ | **40.30**$_{+4.29}$ | 40.10$_{+0.90}$ | 32.98$_{-0.10}$ | **37.79**$_{+1.69}$ |

Dream code) but also expands the knowledge boundaries, yielding positive OOD gains without catastrophic forgetting.

**Robustness in Parallel Decoding.** Although our primary focus is not on accelerating inference, we are also interested in the performance of these models under highly parallel generation settings. To assess this, we evaluate the models under a fixed 2-token parallel decoding setting. Table 4 presents the results on the Dream model fine-tuned on the mathematical reasoning dataset. Traditional methods like SFT-GT suffer from a severe performance drop when forced to decode multiple tokens simultaneously, highlighting their fragility. In contrast, our TABOM and other trajectory-based baselines (dInfer, T3D) exhibit similar robustness and maintain high performance even under parallel decoding. This is reasonable, as learning from self-distilled trajectories inherently aligns the model closer to its actual inference distribution, thereby expanding the subset of high-certainty tokens at each step and enabling multi-token decoding.

Table 4: Performance under 2-token parallel decoding.

| Method | GSM8K | MATH500 | HumanEval | MBPP |
|---|---|---|---|---|
| No-SFT | 74.37 | 31.60 | 43.29 | 43.60 |
| SFT-GT | 72.33$_{-2.04}$ | 25.40$_{-6.20}$ | 40.85$_{-2.44}$ | 40.12$_{-3.48}$ |
| SFT-SD | 74.91$_{+0.54}$ | **34.80**$_{+3.20}$ | **47.56**$_{+4.27}$ | 46.40$_{+2.80}$ |
| dInfer | 77.63$_{+3.26}$ | 33.20$_{+1.60}$ | 40.24$_{-3.05}$ | 49.20$_{+5.20}$ |
| T3D | 76.72$_{+2.35}$ | 32.60$_{+1.00}$ | 41.46$_{-1.83}$ | **49.20**$_{+5.60}$ |
| TABOM | **77.79**$_{+3.42}$ | 34.40$_{+2.80}$ | 45.73$_{+2.44}$ | 47.60$_{+4.00}$ |

**Component Analysis.** To validate the necessity of each component in TABOM, we conduct an ablation study on the Dream model fine-tuned on the mathematical reasoning dataset., as shown in Table 5. Starting from the SFT-SD baseline, we evaluate two settings: a global timestep window and a local timestep window ($W = 32$). In both settings, simply applying trajectory-aware masking (which is similar to the reconstruction objective in dInfer [11]) yields limited improvements and even slight degradation on OOD tasks like HumanEval. However, adding the pairwise ranking loss consistently and significantly boosts performance in both settings, demonstrating its crucial role in aligning the model with the easy-to-hard inference distribution. Furthermore, comparing the two settings reveals that applying the ranking loss globally introduces noise due to comparing tokens with vastly disparate prediction difficulties (e.g., MATH500 score drops to 40.20). In contrast, our full TABOM, which applies the ranking loss strictly within a local timestep window, achieves the best balance and optimal performance across various tasks.

Table 5: Component analysis of TABOM on Dream.

| Method | In-Domain | | OOD | |
|---|---|---|---|---|
| | GSM8K | MATH500 | HumanEval | MBPP |
| Base (SFT-SD) | 81.95 | 39.80 | 57.92 | 58.60 |
| *Global Timestep Window* | | | | |
| Traj. Masking (only $\mathcal{L}_{\text{NELBO}}$) | 82.18 | **41.20** | 56.45 | 58.70 |
| + Pairwise Ranking | 83.10 | 40.20 | 57.50 | 58.20 |
| *Local Timestep Window ($W = 32$)* | | | | |
| Traj. Masking (only $\mathcal{L}_{\text{NELBO}}$) | 82.50 | 40.40 | 56.80 | 58.80 |
| + Pairwise Ranking (TABOM) | **84.31** | 41.10 | **58.54** | **59.20** |

## 4.2 Trajectory Discrimination Score

To directly examine whether a trained model preserves the easy-to-hard structure of the decoding trajectory, we introduce a diagnostic metric named *Trajectory Discrimination Score* (TDS). For a decoding trajectory $\tau$, let $M_t^\tau$ denote the set of still-masked response positions at timestep $t$, excluding positions after the first generated EOS token. For each masked token $i \in M_t^\tau$, we compute its predictive entropy

$$H_\theta(i, t; \tau) = -\sum_{v \in \mathcal{V}} p_\theta(v \mid \mathbf{x}_t^\tau)_i \log p_\theta(v \mid \mathbf{x}_t^\tau)_i. \tag{10}$$

We define the per-step TDS as the trajectory-averaged variance of these entropies:

$$\mathrm{TDS}_\theta(t) = \frac{1}{|\mathcal{T}|} \sum_{\tau \in \mathcal{T}} \mathrm{Var}_{i \in M_t^\tau} \left[ H_\theta(i, t; \tau) \right]. \tag{11}$$

A larger TDS indicates that the model assigns more differentiated uncertainty to masked tokens at the same decoding step, which is consistent with an easy-to-hard decoding preference; a near-zero TDS suggests that the model treats masked positions with nearly uniform confidence, revealing a residual uniform reconstruction bias inherited from the NELBO objective.

Table 6 quantitatively confirms that TABOM produces the most discriminative entropy landscape. Across both LLaDA and Dream, TABOM achieves the highest TDS on every task. The gap is especially pronounced on Dream code tasks: compared with SFT-SD, TABOM increases TDS from 0.035 to 0.711 on HumanEval and from 0.138 to 0.929 on MBPP. This indicates that standard NELBO-style learning, even when using self-distilled trajectories, still tends to assign nearly uniform uncertainty to masked tokens, whereas TABOM separates easy and hard tokens more clearly at decoding time.

Table 6: Quantitative Trajectory Discrimination Score. Higher values indicate stronger token-level uncertainty discrimination along the decoding trajectory. The largest value in each column within each base-model block is highlighted in **bold**.

| Base Model | Method | GSM8K | HumanEval | MBPP | MATH500 |
|---|---|---|---|---|---|
| LLaDA | Base | 0.891 | 1.533 | 1.207 | 0.943 |
| | SFT-SD | 0.927 | 1.623 | 1.223 | 0.924 |
| | TABOM | **0.963** | **1.900** | **1.336** | **1.076** |
| Dream | Base | 1.167 | 0.036 | 0.294 | 1.413 |
| | SFT-SD | 1.082 | 0.035 | 0.138 | 1.428 |
| | TABOM | **1.469** | **0.711** | **0.929** | **1.471** |

Figure 2 further shows the temporal pattern behind this aggregate score. On MBPP, the TDS of NELBO-based baselines quickly collapses to a small value, suggesting that the model behaves as if masked tokens should be reconstructed in a largely uniform manner. In contrast, TABOM maintains a substantially larger TDS throughout decoding, meaning that tokens decoded at the same step receive more diverse confidence levels. This directly supports our motivation: self-distilled trajectories alone do not remove the uniform inductive bias of random-mask reconstruction, while the pairwise ranking term reshapes the entropy landscape toward the easy-to-hard bias captured by the inference trajectory. Therefore, this evidence demonstrates that TABOM's gains come from better trajectory alignment rather than merely from reusing self-generated samples.

# 5 Related Work

## 5.1 Diffusion Language Models

The application of diffusion processes within discrete domains, specifically for text generation, traces its origins to seminal studies by [27] and [28]. A comprehensive probabilistic schema, known as D3PM [25], extended this paradigm by employing a discrete state Markov chain which systematically injects noise into the original input sequence over sequential time steps during the forward phase. Following this initial discrete formulation, the methodology was also investigated within a continuous-time setting [29]. Concurrently, an alternative strategy, SEDD [30], took a different approach by directly calculating likelihood ratios and incorporating a novel denoising score entropy objective for model optimization. Recent theoretical and empirical investigations, exemplified by works such as MDLM [31–33] and RADD [34], have demonstrated a crucial equivalence: various distinct parameterizations of discrete diffusion models are, in fact, mathematically identical. This finding is central to simplifying the understanding of these architectures. Inspired by these advancements in algorithms and theory, the scale of diffusion model application has significantly expanded, reaching the 7-8 billion parameter level. For instance, LLaDA [6] was trained from scratch with weighted cross-entropy loss, while Dream [5] involved adaptation from the established Qwen2.5 base model with a substantially smaller data requirement. Prominent commercial implementations in this space include Mercury [17], Gemini Diffusion [35], and Seed Diffusion [12]. A notable characteristic of these proprietary models is that their external benchmark performance is comparable to that of larger autoregressive (AR) language models. Furthermore, they exhibit superior decoding efficiency, achieving fast sampling speeds. Collectively, these results strongly affirm the substantial promise of diffusion language models as a compelling and viable alternative within the landscape of generative AI.

## 5.2 Trajectory Distillation for DLMs

Despite the theoretical advantages of Diffusion Language Models (DLMs) in parallel decoding, their practical inference efficiency is often bottlenecked by the requirement of numerous iterative denoising steps. To mitigate this latency, recent research has increasingly focused on trajectory distillation and inference acceleration techniques.

A primary line of work aims to compress the generation trajectory to enable few-step or even one-step decoding. For instance, Seed Diffusion [12] proposes a two-stage curriculum that incorporates an edit-based forward process and constrained-order trajectory distillation, significantly boosting inference speed to over 2,000 tokens per second on code generation tasks. Similarly, T3D [13] introduces a trajectory self-distillation framework equipped with Direct Discriminative Optimization (DDO) to alleviate the mean-field approximation error under tight step budgets, effectively pushing the limits of few-step decoding. At the system and framework level, dInfer [11] provides a modularized inference engine that integrates algorithmic innovations—such as hierarchical decoding, credit decoding, and iteration smoothing—with system-level optimizations like vicinity KV-cache refresh and loop unrolling. This holistic approach achieves substantial speedups over existing frameworks like Fast-dLLM [36] without compromising output quality.

**Differences from our work:** While the aforementioned methods predominantly focus on compressing the trajectory to accelerate inference speed (i.e., efficiency enhancement), our work takes a fundamentally different perspective. Rather than viewing self-distilled trajectories as decoding shortcuts, we treat them as high-quality demonstrations of the model's own correct generation process. The objective is to efficiently absorb the distributional structure of trajectories that the model can already complete successfully, including their decoding order and uncertainty patterns. This allows the model to better align its predictive landscape with its native inference dynamics, improving generation quality without reducing the problem to sampling-speed optimization.

# 6 Conclusion

In this paper, we introduced Trajectory-Aligned Boltzmann Modeling (TABOM), a novel post-training framework that addresses the severe training-inference misalignment in Diffusion Language Models (DLMs). By explicitly modeling the easy-to-hard inductive bias of inference as a Boltzmann

distribution, TABOM leverages self-distilled trajectories to construct a tractable pairwise ranking objective. This approach effectively shapes the local entropy landscape without requiring intractable partition function computations. Extensive experiments across mathematical reasoning and code generation domains demonstrate that TABOM not only achieves state-of-the-art in-domain performance but also successfully expands the model's knowledge boundaries to out-of-distribution tasks, completely mitigating the catastrophic forgetting observed in standard supervised fine-tuning. Furthermore, TABOM exhibits strong robustness under highly parallel decoding settings. We believe our insights into the inference unmasked distribution and trajectory-aware optimization will inspire future advancements in the efficient and effective training of DLMs.

## References


[1] Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, Zhuofu Chen, Jialei Cui, Hao Ding, Mengnan Dong, Angang Du, Chenzhuang Du, Dikang Du, Yulun Du, Yu Fan, Yichen Feng, Kelin Fu, Bofei Gao, Hongcheng Gao, Peizhong Gao, Tong Gao, Xinran Gu, Longyu Guan, Haiqing Guo, Jianhang Guo, Hao Hu, Xiaoru Hao, Tianhong He, Weiran He, Wenyang He, Chao Hong, Yangyang Hu, Zhenxing Hu, Weixiao Huang, Zhiqi Huang, Zihao Huang, Tao Jiang, Zhejun Jiang, Xinyi Jin, Yongsheng Kang, Guokun Lai, Cheng Li, Fang Li, Haoyang Li, Ming Li, Wentao Li, Yanhao Li, Yiwei Li, Zhaowei Li, Zheming Li, Hongzhan Lin, Xiaohan Lin, Zongyu Lin, Chengyin Liu, Chenyu Liu, Hongzhang Liu, Jingyuan Liu, Junqi Liu, Liang Liu, Shaowei Liu, T. Y. Liu, Tianwei Liu, Weizhou Liu, Yangyang Liu, Yibo Liu, Yiping Liu, Yue Liu, Zhengying Liu, Enzhe Lu, Lijun Lu, Shengling Ma, Xinyu Ma, Yingwei Ma, Shaoguang Mao, Jie Mei, Xin Men, Yibo Miao, Siyuan Pan, Yebo Peng, Ruoyu Qin, Bowen Qu, Zeyu Shang, Lidong Shi, Shengyuan Shi, Feifan Song, Jianlin Su, Zhengyuan Su, Xinjie Sun, Flood Sung, Heyi Tang, Jiawen Tao, Qifeng Teng, Chensi Wang, Dinglu Wang, Feng Wang, Haiming Wang, Jianzhou Wang, Jiaxing Wang, Jinhong Wang, Shengjie Wang, Shuyi Wang, Yao Wang, Yejie Wang, Yiqin Wang, Yuxin Wang, Yuzhi Wang, Zhaoji Wang, Zhengtao Wang, Zhexu Wang, Chu Wei, Qianqian Wei, Wenhao Wu, Xingzhe Wu, Yuxin Wu, Chenjun Xiao, Xiaotong Xie, Weimin Xiong, Boyu Xu, Jing Xu, Jinjing Xu, L. H. Xu, Lin Xu, Suting Xu, Weixin Xu, Xinran Xu, Yangchuan Xu, Ziyao Xu, Junjie Yan, Yuzi Yan, Xiaofei Yang, Ying Yang, Zhen Yang, Zhilin Yang, Zonghan Yang, Haotian Yao, Xingcheng Yao, Wenjie Ye, Zhuorui Ye, Bohong Yin, Longhui Yu, Enming Yuan, Hongbang Yuan, Mengjie Yuan, Haobing Zhan, Dehao Zhang, Hao Zhang, Wanlu Zhang, Xiaobin Zhang, Yangkun Zhang, Yizhi Zhang, Yongting Zhang, Yu Zhang, Yutao Zhang, Yutong Zhang, Zheng Zhang, Haotian Zhao, Yikai Zhao, Huabin Zheng, Shaojie Zheng, Jianren Zhou, Xinyu Zhou, Zaida Zhou, Zhen Zhu, Weiyu Zhuang, and Xinxing Zu. Kimi k2: Open agentic intelligence, 2025. URL `https://arxiv.org/abs/2507.20534`.

[2] OpenAI. OpenAI o3 and o4-mini system card. *https://openai.com/index/o3-o4-mini-system-card/*, 2025.

[3] DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runxin Xu, Ruoyu Zhang, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Shuting Pan, T. Wang, Tao Yun, Tian Pei, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wanjia Zhao, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaokang Zhang, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xingkai Yu, Xinnan Song,

Xinxia Shan, Xinyi Zhou, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. X. Zhu, Yang Zhang, Yanhong Xu, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Li, Yaohui Wang, Yi Yu, Yi Zheng, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Ying Tang, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yu Wu, Yuan Ou, Yuchen Zhu, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yukun Zha, Yunfan Xiong, Yunxian Ma, Yuting Yan, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Z. F. Wu, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao, Zhipeng Xu, Zhiyu Wu, Zhongyu Zhang, Zhuoshu Li, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Ziyi Gao, and Zizheng Pan. Deepseek-v3 technical report, 2025. URL https://arxiv.org/abs/2412.19437.

[4] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023. URL https://arxiv.org/abs/1706.03762.

[5] 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.

[6] 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. *arXiv preprint arXiv:2502.09992*, 2025.

[7] Jingyang Ou, Jiaqi Han, Minkai Xu, Shaoxuan Xu, Jianwen Xie, Stefano Ermon, Yi Wu, and Chongxuan Li. Principled rl for diffusion llms emerges from a sequence-level perspective, 2025. URL https://arxiv.org/abs/2512.03759.

[8] Jingyi Yang, Yuxian Jiang, Xuhao Hu, Shuang Cheng, Biqing Qi, and Jing Shao. Dare: Diffusion large language models alignment and reinforcement executor, 2026. URL https://arxiv.org/abs/2604.04215.

[9] Siyan Zhao, Devaansh Gupta, Qinqing Zheng, and Aditya Grover. d1: Scaling reasoning in diffusion large language models via reinforcement learning, 2025. URL https://arxiv.org/abs/2504.12216.

[10] Jingyi Yang, Guanxu Chen, Xuhao Hu, and Jing Shao. Taming masked diffusion language models via consistency trajectory reinforcement learning with fewer decoding step, 2025. URL https://arxiv.org/abs/2509.23924.

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

[12] Yuxuan Song, Zheng Zhang, Cheng Luo, Pengyang Gao, Fan Xia, Hao Luo, Zheng Li, Yuehang Yang, Hongli Yu, Xingwei Qu, Yuwei Fu, Jing Su, Ge Zhang, Wenhao Huang, Mingxuan Wang, Lin Yan, Xiaoying Jia, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Yonghui Wu, and Hao Zhou. Seed diffusion: A large-scale diffusion language model with high-speed inference, 2025. URL https://arxiv.org/abs/2508.02193.

[13] Tunyu Zhang, Xinxi Zhang, Ligong Han, Haizhou Shi, Xiaoxiao He, Zhuowei Li, Hao Wang, Kai Xu, Akash Srivastava, Vladimir Pavlovic, et al. T3d: Few-step diffusion language models via trajectory self-distillation with direct discriminative optimization. *arXiv preprint arXiv:2602.12262*, 2026.

[14] inclusionAI. Ling-coder-sft. https://huggingface.co/datasets/inclusionAI/Ling-Coder-SFT, 2024.

[15] horseee. Mixchain-z-prm12k. https://huggingface.co/datasets/horseee/MixChain-Z-PRM12K, 2024.

[16] Gen Li and Changxiao Cai. A convergence theory for diffusion language models: An information-theoretic perspective, 2025. URL https://arxiv.org/abs/2505.21400.

[17] Inception Labs, Samar Khanna, Siddhant Kharbanda, Shufan Li, Harshit Varma, Eric Wang, Sawyer Birnbaum, Ziyang Luo, Yanis Miraoui, Akash Palrecha, Stefano Ermon, Aditya Grover, and Volodymyr Kuleshov. Mercury: Ultra-fast language models based on diffusion, 2025. URL https://arxiv.org/abs/2506.17298.

[18] Fengfei Sun, Ningke Li, Kailong Wang, and Lorenz Goette. Large language models are overconfident and amplify human bias, 2025. URL https://arxiv.org/abs/2505.02151.

[19] Yann Lecun, Sumit Chopra, and Raia Hadsell. *A tutorial on energy-based learning*. 01 2006.

[20] 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.

[21] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021. URL https://arxiv.org/abs/2106.09685.

[22] 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.

[23] 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. *NeurIPS*, 2021.

[24] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

[25] Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 17981–17993, 2021.

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

[27] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015.

[28] Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. *Advances in Neural Information Processing Systems*, 34:12454–12465, 2021.

[29] Andrew Campbell, Joe Benton, Valentin De Bortoli, Thomas Rainforth, George Deligiannidis, and Arnaud Doucet. A continuous time framework for discrete denoising models. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

[30] Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion language modeling by estimating the ratios of the data distribution. *arXiv preprint arXiv:2310.16834*, 2023.

[31] Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis K Titsias. Simplified and generalized masked diffusion for discrete data. *arXiv preprint arXiv:2406.04329*, 2024.

[32] Subham Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. *Advances in Neural Information Processing Systems*, 37:130136–130184, 2024.

[33] Kaiwen Zheng, Yongxin Chen, Hanzi Mao, Ming-Yu Liu, Jun Zhu, and Qinsheng Zhang. Masked diffusion models are secretly time-agnostic masked models and exploit inaccurate categorical sampling. *arXiv preprint arXiv:2409.02908*, 2024.

[34] Jingyang Ou, Shen Nie, Kaiwen Xue, Fengqi Zhu, Jiacheng Sun, Zhenguo Li, and Chongxuan Li. Your absorbing discrete diffusion secretly models the conditional distributions of clean data. *arXiv preprint arXiv:2406.03736*, 2024.

[35] Google DeepMind. Gemini-diffusion, 2025. URL `https://blog.google/technology/google-deepmind/gemini-diffusion/`.

[36] 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. *CoRR*, abs/2505.22618, 2025.

## A Sensitivity to $\lambda$ and $\gamma$

We perform a $2\times3$ sensitivity analysis over $\lambda \in \{1, 2\}$ and $\gamma \in \{0.1, 0.2, 0.3\}$, where $\gamma$ denotes the margin. For each $(\lambda, \gamma)$ pair, we report the best GSM8K score across available checkpoints (epochs). The No-SFT LLaDA baseline is 76.12.

Table 7: LLaDA GSM8K sensitivity to $\lambda$ and $\gamma$ (best checkpoint per combination).

| $\lambda\backslash\gamma$ | 0.1 | 0.2 | 0.3 |
|---|---|---|---|
| 1 | 77.94 | **78.62** | 78.09 |
| 2 | 77.18 | **78.62** | 77.48 |

All six settings outperform No-SFT (76.12), with gains from +1.06 to +2.50 points. The most robust choice is $\gamma = 0.2$, which reaches the top score (78.62) under both $\lambda = 1$ and $\lambda = 2$. Compared with $\lambda = 2$, $\lambda = 1$ is more stable across margins (77.94/78.62/78.09 vs. 77.18/78.62/77.48), suggesting that moderate ranking strength with moderate margin provides the best accuracy-stability trade-off in this grid.

## B Hyperparameter for inference

As shown in Table 8, we provide the full list of used hyperparameters for inference.

Table 8: Hyperparameter settings across different base models and benchmarks. "–" indicates the hyperparameter is not applicable or not used.

| Base Model | Hyperparameter | GSM8K | MATH500 | MBPP | HumanEval |
|---|---|---|---|---|---|
| Dream-7B-Instruct | Max New Tokens | 256 | 512 | 1024 | 768 |
| | Diffusion Steps | 256 | 512 | 1024 | 768 |
| | Temperature | 0.1 | 0.1 | 0.1 | 0.1 |
| | Top-$p$ | 0.9 | 0.9 | 0.9 | 0.9 |
| | Top-$k$ | – | – | – | – |
| LLaDA-8B-Instruct | Max New Tokens | 512 | 512 | 256 | 512 |
| | Diffusion Steps | 512 | 512 | 256 | 512 |
| | Block Length | 8 | 64 | 256 | 512 |
| | Temperature | 0.0 | 0.0 | 0.0 | 0.0 |
| | Top-$p$ | 0.9 | 0.9 | 0.9 | 0.9 |
| | Top-$k$ | – | – | – | – |

## C Sensitivity to window size $W$.

We further summarize the effect of timestep window size and use $W \in \{16, 32, 48, 64, 256\}$ as a compact stress-test axis. By conducting experiments on Dream model fine-tuned on MixChain-Z-PRM12K dataset, we observe a clear sweet spot at $W = 32$: both GSM8K and HumanEval peak at $W = 32$, then consistently decrease as $W$ increases to 48/64, and degrade most severely at $W = 256$.

This trend supports using a moderate local window in practice: too small a window under-utilizes cross-step supervision, while too large a window introduces noisy pairwise relations and hurts both in-domain reasoning and code generalization.

Table 9: Window-size sensitivity ($W$). We report the best checkpoint score for each $W$. The $W = 64$ and $W = 256$ rows are long-window stress-test references; $W = 256$ is set as a conservative worst-case anchor.

| W | GSM8K (flexible-extract, %) | HumanEval (pass@1, %) |
|---|---|---|
| 16 | 83.78 | 59.15 |
| 32 | **84.08** | **60.37** |
| 48 | 83.95 | 59.76 |
| 64 | 82.79 | 59.15 |
| 256 | 75.36 | 58.00 |