DIFFCOT: Diffusion-styled Chain-of-Thought Reasoning in LLMs

Shidong Cao[1], **Hongzhan Lin**[1], **Yuxuan Gu**[2], **Ziyang Luo**[1], **Jing Ma**[1*]
[1]Hong Kong Baptist University
[2]Harbin Institute of Technology
{cssdcao,cshzlin,majing}@comp.hkbu.edu.hk

Abstract

Chain-of-Thought (CoT) reasoning improves multi-step mathematical problem solving in large language models but remains vulnerable to exposure bias and error accumulation, as early mistakes propagate irreversibly through autoregressive decoding. In this work, we propose DIFFCOT, a diffusion-styled CoT framework that reformulates CoT reasoning as an iterative denoising process. DIFFCOT integrates diffusion principles at the reasoning-step level via a sliding-window mechanism, enabling unified generation and retrospective correction of intermediate steps while preserving token-level autoregression. To maintain causal consistency, we further introduce a causal diffusion noise schedule that respects the temporal structure of reasoning chains. Extensive experiments on three multi-step CoT reasoning benchmarks across diverse model backbones demonstrate that DIFFCOT consistently outperforms existing CoT preference optimization methods, yielding improved robustness and error-correction capability in CoT reasoning.

1 Introduction

Large Language Models (LLMs) (Brown et al., 2020; Achiam et al., 2023; Guo et al., 2025) have marked a major breakthrough in natural language processing, achieving competitive performance across a wide range of mathematical reasoning tasks. A widely adopted technique in LLMs, Chain-of-Thought (CoT) reasoning (Wei et al., 2022; Kojima et al., 2022), enhances mathematical problem-solving by decomposing tasks into step-by-step intermediate reasoning. However, CoT reasoning is highly sensitive to potential errors, where mistakes introduced in the early stages can propagate through later steps, often leading to incorrect final answers (Wang et al., 2023; Lyu et al., 2023). This highlights a central research challenge: reducing errors by guiding LLMs to identify and follow reliable reasoning paths in long thought chains.

Recent work has sought to optimize the exploration of reasoning branches in LLMs to reduce errors in CoT, by coupling generation with external critics or search procedures, such as Monte Carlo Tree Search (MCTS), to prune faulty trajectories (Li et al., 2024; Qin et al., 2024; Xi et al., 2024). In contrast, Preference Optimization (PO) methods (Lai et al., 2024; Zhang et al., 2024; Xu et al., 2025) aligned the policy with implicit preferences distilled from reasoning chains, so that decoding itself favors correct trajectories rather than relying on post-hoc filtering. However, beyond the substantial inference-time overhead of MCTS, these methods share a fundamental limitation that motivates rethinking of the CoT paradigm. Specifically, they all execute CoT reasoning in a strictly step-by-step manner, with each step conditioned on previous ones. Under static teacher-forcing supervision, the model is exposed only to correct prefixes during training, whereas at inference time the prefixes may contain errors that accumulate across steps, resulting in exposure bias that commonly plagues autoregressive generation (Bengio et al., 2015) and imitation learning (Ross et al., 2011).

To mitigate the limitations above, we formulate CoT reasoning as a globally revisable trajectory, in which intermediate steps are not fixed once generated but can be adaptively corrected in light of later context. In contrast to autoregressive likelihood maximization, which optimizes conditional factors independently, score-based modeling optimizes a global objective over the full trajectory distribution, implicitly coupling all positions through the gradient field of the log-density. Rather than defining reasoning as a strictly forward, prefix-conditioned process driven by human priors (see Figure 1), our method jointly models all reasoning steps within a diffusion paradigm (Dhariwal and Nichol, 2021), allowing robust reasoning un-

*Corresponding author.

Findings of the Association for Computational Linguistics: ACL 2026, pages 38942–38957

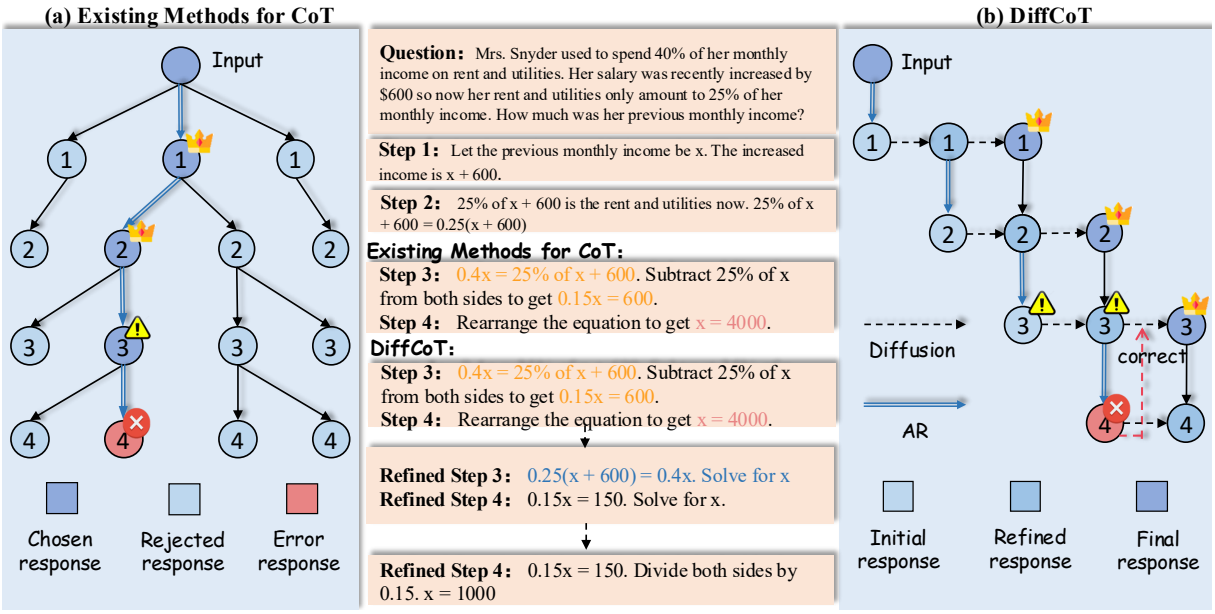

Figure 1: Comparison of our proposed DIFFCOT with existing CoT reasoning approaches: **(a)** Existing step-by-step CoT Reasoning methods adopt teacher-forcing training, where each step depends on the ground-truth output of the previous one. At inference time, this assumption breaks, causing exposure bias and leading to error accumulation. **(b)** DIFFCOT performs CoT reasoning along both the noise (diffusion) and temporal (autoregressive) dimensions, enabling iterative correction of prior mistakes and effectively mitigating exposure bias.

der realistic noise from a global perspective with long-term revision guided by future signals. Starting from a corrupted or noisy reasoning chain, the entire reasoning trajectory can be iteratively updated, where intermediate revisions and forward progression are seamlessly integrated into a unified refinement process. By unifying iterative revision and forward continuation within the same diffusion-based refinement process, our design philosophy aims to reduce the discrepancy between training-time supervision and inference-time reasoning dynamics, leading to more stable reasoning behavior during inference.

To this end, we propose Diffusion-Styled Chain of Thought, **DIFFCOT**, which integrates diffusion principles into the CoT framework to mitigate the exposure bias of static teacher-forcing and address the pointwise and local-step supervision issues of prior preference optimization solutions. Specifically, 1) DIFFCOT modifies only the step-level generation strategy while preserving token-level autoregression, making it straightforward to fine-tune from existing autoregressive models. 2) By incorporating sliding windows and causal noise masks, DIFFCOT unifies generation and revision within a versatile framework, to balance autoregressive and diffusion-styled reasoning. 3) Moreover, DIFFCOT employs sentence-level noise injection to convert the reasoning paradigm into iterative denoising, progressively refining corrupted trajectories into coherent chains of thought, which captures the distributional evolution of reasoning errors to enable recovery from diverse and compounding mistakes. Altogether, DIFFCOT could establish a unified and adaptive paradigm to CoT reasoning in LLMs.

Our contributions are summarized as follows:

- We propose DIFFCOT, a novel diffusion-styled CoT framework that reformulates multi-step reasoning as an iterative denoising process, effectively mitigating exposure bias and error accumulation in autoregressive CoT reasoning.[1]
- We introduce a step-level diffusion-based learning strategy that unifies generation and revision through a sliding-window mechanism, enabling retrospective correction of earlier reasoning steps while preserving token-level autoregression.
- We design a causal noise schedule that explicitly encodes the temporal dependency of reasoning steps, balancing global error correction with the causal structure required for coherent reasoning.
- Extensive experiments on three public mathematical reasoning benchmarks demonstrate that DIFFCOT outperforms State-of-The-Art (SoTA) PO methods, yielding robust gains and substantially improved error-correction capability in CoT.

[1]The source code is released via https://github.com/caoshidong66/DiffCoT.

2 Preliminaries

Chain-of-Thought Reasoning Given a question prompt p, the CoT paradigm (Wei et al., 2022) explicitly unfolds the reasoning process into a sequence of reasoning steps $s_{1:K}$ whose final element s_K corresponds to the answer a. The conditional distribution of a complete reasoning trace can be expressed as:

$$p_\theta(s_{1:K} \mid p) = \prod_{t=1}^{K} \pi_\theta(s_k \mid p, s_{<k}), \quad (1)$$

where s_k denotes the k-th reasoning step, $k \leq K$ is the step budget, and π_θ is the Auto-Regressive (AR) policy, i.e., the conditional distribution over the next step given the prompt p and the previously generated steps.

On annotated CoT data $(p, s_{1:K})$, the model is trained by maximizing the conditional likelihood:

$$\mathcal{L}_{\text{CoT}} = -\sum_{k=1}^{K} \log \pi_\theta(s_k \mid p, s_{<k}). \quad (2)$$

Diffusion Models Diffusion models (Dhariwal and Nichol, 2021) define a generative framework where data samples are gradually corrupted into a tractable noise distribution through a forward process, and a model is trained to approximate the corresponding reverse process. Let $x_0 \sim \varphi(x_0)$ denote a data sample from real data distribution φ. The forward process produces a sequence $\{x_t\}_{t=1}^{T}$ by applying a noise operator $\mathcal{O}$ at each step:

$$x_t = \mathcal{D}(x_{t-1}, \eta_t), \quad t = 1, \ldots, T, \quad (3)$$

where t denotes the diffusion step index, η_t is a noise variable. $\mathcal{O}$ could be a deterministic or structured corruption operator.

The generative process is defined by learning a parameterized reverse transition $p_\theta(x_{t-1} \mid x_t)$ that reconstructs the original sample through step-wise denoising. Training minimizes the discrepancy between the predicted $\hat{x}_{t-1} = f_\theta(x_t, t)$ and the ground-truth corrupted data x_{t-1} obtained from the forward process:

$$\mathcal{L}_{\text{Diffusion}} = \mathbf{E}_{x_0 \sim \varphi(x_0),\, t \sim \{1,\ldots,T\}} \Big[\ell\big(f_\theta(x_t, t),\, x_{t-1}\big) \Big], \quad (4)$$

where $f_\theta(x_t, t)$ predicts the reconstruction of x_{t-1} from x_t, $\ell(\cdot, \cdot)$ is a reconstruction loss. In inference, sampling begins from x_T and iteratively applies the learned reverse transitions until x_0 is obtained.

3 Methodology

During CoT reasoning, LLMs are typically trained only on correct trajectories (Wei et al., 2022), while at inference time they may condition on erroneous intermediate steps (Lyu et al., 2023). Moreover, step-wise optimization focuses on local alignment and ignores future signals within the reasoning trajectory, limiting global consistency (Zhang et al., 2024). Together, these issues lead to error accumulation (Yoon et al., 2025; Chen et al., 2024a), motivating a rethinking of CoT reasoning under preference optimization. To address these issues, we argue that effective CoT reasoning requires trajectory-level revision to enforce global consistency and recover from corrupted intermediate steps. As error accumulation leads to noisy reasoning chains at inference time, a mechanism for global recovery becomes essential (Lyu et al., 2023; Wang et al., 2023). Unlike autoregressive decoding that commits to each step irrevocably, diffusion operates through iterative denoising over the entire sequence, enabling global revision and correction of intermediate errors. This property naturally aligns with the need for trajectory-level recovery in CoT reasoning. Motivated by the robustness of diffusion models in reconstructing structured data from noise (Ho et al., 2020; Li et al., 2022), we aim to develop a diffusion-styled preference optimization paradigm to reformulate CoT reasoning.

To this end, we propose *Diffusion-styled Chain of Thought* (DiffCoT), which models CoT reasoning in a diffusion-styled framework. We first present a step-level forward noising process for CoT using reward-ranked candidates (§3.1), and then apply a diffusion sliding window to iteratively denoise past steps while generating new ones (§3.2). We further introduce causal diffusion noise to strengthen causal consistency across reasoning steps (§3.3). An overview is shown in Figure 2.

3.1 Diffusion-Styled Noising for CoT

Previous preference optimization methods overlooked the distribution between high-reward and low-reward reasoning steps and relied on isolated step-wise adjustments that fail to mitigate the exposure bias inherent in teacher forcing. In this section, we design a diffusion-styled forward noising process for CoT reasoning at the reasoning-step level, where step-level noise is induced by ranking candidate responses for the same step according to their reward scores. In our principle, higher-reward can-

didates are treated as lower-noise reasoning states, while lower-reward candidates correspond to progressively higher-noise states in the forward process. This ranking forms a progression from clean to corrupted reasoning states under diffusion-styled noise, enabling distribution-aware modeling of reasoning steps.

Specifically, we implement forward noising by first collecting CoT reasoning trajectories for each question to construct the training set $\mathcal{D}$. As shown in Figure 2(a), given a question prompt p, we follow the standard CoT data generation process by employing MCTS (Browne et al., 2012), to gradually build a search tree for step-by-step solution exploration. The root corresponds to p, and each child node represents an intermediate step s. A complete path from the root to a terminal node s_K yields a trajectory $\mathbf{tra} = p \oplus s_1 \oplus s_2 \oplus \cdots \oplus s_K$, where each step s_k is annotated with its reward $r(s_k)$.

In contrast to conventional CoT trajectory construction (Zhang et al., 2024), at each step k we collect several candidate responses. These responses are scored using either an external reward model or rollout-based success rates. In our forward noising view, the candidate with the highest reward, denoted as $s_k^{\sigma^0=w}$, is regarded as the lowest-noise state for step k, and the remaining candidates are ordered by their rewards to form $\{s_k^{\sigma^{1:T}}\}$, which we interpret as states with gradually increasing noise. Step-level noise is thus measured by the deviation of lower-reward responses from this best trajectory, indicating the corruption level at the step granularity. This design ensures that the final generation remains coherent while providing a set of forward diffusion states for subsequent denoising and preference optimization.

Under the step-level forward noising, DIFFCOT defines a diffusion-styled distribution that spans low-noise to high-noise reasoning states (Chen et al., 2025). Note that our data construction does not distinguish between "correct" and "incorrect" labels at the collection stage as in Lai et al. (2025); instead, it uniformly gathers diverse responses for each step throughout the entire trajectory. The global view could provide distribution-aware data to support both generation and refinement. In this way, error patterns are implicitly encoded in trajectory-level modeling, allowing robust reasoning without additional manual annotations of corrective steps.

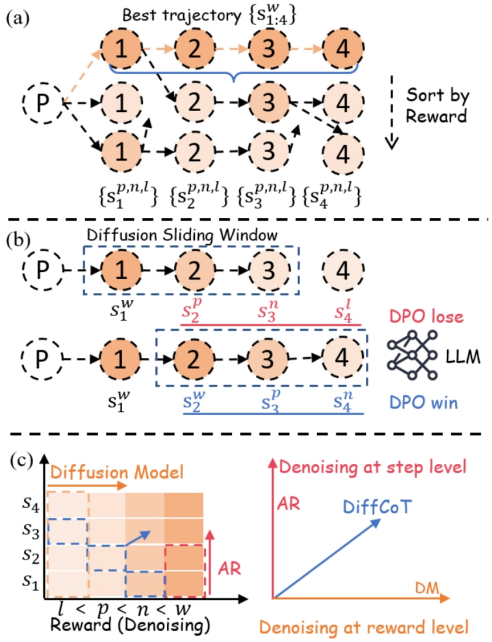

Figure 2: DIFFCOT Framework and Training Data Construction: **(a)** Step-level forward noising: MCTS-based data generation defines step-level noise by reward-ranking multiple candidates, yielding states ranging from clean to corrupted. **(b)** Sliding-window denoising: a diffusion sliding window refines previously generated CoT steps while producing the next step in an autoregressive manner. **(c)** Causal diffusion noise: a step-dependent schedule assigns stronger noise to later steps to encode the causal order of the reasoning chain.

3.2 Reformulating CoT Reasoning as Denoising

In this section, our DIFFCOT framework reformulates CoT reasoning beyond the teacher-forcing paradigm as a diffusion-styled denoising process to mitigate exposure bias at inference time. To couple this with the autoregressive nature of CoT, we introduce a diffusion sliding-window mechanism that operates directly on the CoT reasoning process. Within the devised sliding window, previously generated CoT steps are progressively denoised from high-noise toward low-noise reasoning states, thereby facilitating self-correction, while advancing the window naturally aligns with the autoregressive generation of subsequent CoT steps. Thus, instead of costly training a separate diffusion language model from scratch, we leverage diffusion-styled data to efficiently fine-tune pre-trained LLM π for CoT reasoning.

Specifically, for the iterative denoising process, the model π takes the question prompt p together with the preceding reasoning steps $\mathbf{s}_{1:k}$ as input. To enable variable-length generation when applying

diffusion to thought chains while integrating generation with self-correction, the model maintains a diffusion sliding window of size m and stride n. At denoising iteration t, the window containing previously generated CoT steps $\{s^{\sigma}_{k-m}, \ldots, s^{\sigma}_{k}\}$ is updated to a lower-noise version $\{s^{\sigma'}_{k-m}, \ldots, s^{\sigma'}_{k}\}$, where $\sigma = \sigma^{(t)}_{k}$ is defined as a function of the step index k and the denoising iteration t, while σ and σ' denote the noise levels before and after refinement, respectively. Simultaneously, as the window shifts forward by one step, the model predicts the next step s^{σ}_{k+1}, initialized at a high-noise state. Iterating this denoising process eventually yields a clean trajectory $s^{\sigma^0}_{1:K}$:

$$\pi_\theta\big(p,\, \mathbf{s}^{\sigma}_{1:k}\big) \;\mapsto\; \underbrace{\mathbf{s}^{\sigma'}_{k-m:k}}_{\text{refined past}} \;\oplus\; \underbrace{\mathbf{s}^{\sigma}_{k+1}}_{\text{predicted future}} \quad . \tag{5}$$

An illustration of the denoising process is shown in Figure 2(b). In sequence modeling, teacher-forcing next-token prediction can be viewed as masking along the reasoning-step axis s, whereas diffusion corresponds to masking along the noise axis σ, where σ^T approaches pure white noise after T iterations. To unify both views, we denote s^{σ}_{k} as the k-th CoT step under noise level σ, where σ is instantiated as σ^{t}_{k}, i.e., the diffusion noise strength assigned to the k-th step at denoising iteration t.

Building on the above reasoning process, we now introduce the training objective in DiffCoT. Our goal is to optimize a model π_θ that takes the question prompt p together with the preceding reasoning steps $\mathbf{s}_{1:k}$ as input.

We construct the *win* sequence $s^{w}_{k-m:k+1}$ by combining the denoised steps $\{s^{\sigma'}_{k-m}, \ldots, s^{\sigma'}_{k}\}$ with a lower-noise variant of $s^{\sigma'_{k+1}}_{k+1}$. In practice, following DiffPO (Chen et al., 2025), we use the best sample s^{σ^0} as the winning target and consistently align all intermediate generations to this target. Similarly, the *lose* sequence $s^{l}_{k-m:k+1}$ is formed by combining the unrefined steps $\{s^{\sigma}_{k-m}, \ldots, s^{\sigma}_{k}\}$ with the highest-noise variant of $s^{\sigma_{k+1}}_{k+1}$. The prefix condition is the past text $s_{1:k-1}$. To optimize the LLM on this preference pair, we adopt the Direct Preference Optimization (DPO) loss (Rafailov et al., 2023):

$$\mathcal{L}_i(\pi_\theta; \pi_{\text{ref}}) = -\log \phi\Big(\beta \log \frac{\pi_\theta(s^{w}_{:k+1} \mid p, s_{1:k})}{\pi_{\text{ref}}(s^{w}_{:k+1} \mid p, s_{1:k})} - \beta \log \frac{\pi_\theta(s^{l}_{:k+1} \mid p, s_{1:k})}{\pi_{\text{ref}}(s^{l}_{:k+1} \mid p, s_{1:k})}\Big), \tag{6}$$

where $s_{:k+1}$ denotes the subsequence $s_{k-m:k+1}$, $\phi(\cdot)$ denotes the sigmoid function, and β is a factor controlling the strength of the preference signal.

Note that our conditional prefix $s^{\sigma}_{1:k-1}$ differs from that used in standard DPO: instead of being composed exclusively of preferred (i.e., clean) steps, it combines clean steps from 1 to $k-m$ with noisy steps within the sliding window. In this manner, the hybrid construction could alleviate the exposure bias by training the model to make preference-consistent updates even when conditioned on partially corrupted reasoning prefixes.

3.3 Causal Diffusion Noise

Modeling CoT reasoning with diffusion poses a significant challenge to causality. Conventional full-sequence diffusion models are inherently non-causal (Ho et al., 2020; Dhariwal and Nichol, 2021), which contrasts sharply with the causal nature of CoT reasoning. While our backbone π retains an AR token-level generation process, prior studies on diffusion models indicate that relying solely on the model's ability to capture causality is inadequate, especially when reasoning must be performed over noisy or perturbed data (Li et al., 2022).

Inspired by Diffusion Forcing (Chen et al., 2024a), as shown in Figure 2(c), we leverage noise as a mechanism to inject causal priors into diffusion sliding window.In conventional full-sequence diffusion, the noise strength $\sigma^{(t)}$ depends solely on the denoising iteration t and is shared across all tokens. Such uniform noise injection is ill-suited for CoT reasoning, as it limits the model's ability to capture step-wise causal dependencies. To overcome this, we redefine the noise schedule as σ^{t}_{k}, a joint function of reasoning step k and iteration t (see §3.2). Within the diffusion sliding window, σ^{t}_{k} follows a progressive schedule, where earlier steps are perturbed with weaker noise, while later steps are perturbed with stronger noise. The noise schedule is formally defined as follows:

$$F(s_{j-m+1}, \ldots, s_j; j) = \big(s^{\sigma^0}_{j-m+1},\; s^{\sigma^1}_{j-m+2},\; \ldots,\; s^{\sigma^T}_{j}\big), \tag{7}$$

where the diffusion sliding window has size m and stride 1, and at the j-th denoising iteration the window advances to generate s_j. In this manner, our framework could better stabilize the causal chain and enhance self-correction in subsequent reasoning steps even if the current step is erroneous.

Model	Llama3-8B							Qwen3-4B						
	CoT	TSFT	CPO	ToT	Step	FStep	DiffCoT	CoT	TSFT	CPO	ToT	Step	FStep	DiffCoT
GSM8K	62.5	62.7	61.6	61.7	63.0	63.2	**64.4**	84.7	84.5	85.7	85.4	87.5	87.2	**88.5**
SVAMP	72.2	73.8	73.3	73.0	75.7	76.2	**76.9**	86.8	86.8	87.2	87.9	88.3	89.7	**90.2**
M-L1	43.3	39.1	39.0	39.5	**43.5**	39.6	39.3	59.3	58.4	60.1	60.9	61.9	64.5	**75.0**
M-L2	29.0	28.4	28.7	29.3	28.9	**29.7**	26.6	35.0	35.5	35.8	36.0	34.1	39.9	**48.7**
M-L3	13.0	13.6	14.0	13.5	15.7	14.8	**17.2**	21.6	20.7	21.5	21.8	24.5	27.6	**33.1**
M-L4	7.4	7.8	7.0	7.4	8.0	7.8	**9.5**	10.5	10.4	11.3	9.5	12.3	18.4	**24.4**
M-L5	2.1	1.7	1.8	1.4	1.9	2.2	**3.8**	3.9	4.4	4.3	3.2	5.0	5.1	**13.2**
Model	**Qwen3-8B**							**Ministral3-8B**						
	CoT	TSFT	CPO	ToT	Step	FStep	DiffCoT	CoT	TSFT	CPO	ToT	Step	FStep	DiffCoT
GSM8K	87.3	87.5	88.0	86.3	86.4	88.7	**91.5**	61.5	61.8	63.8	61.5	66.3	65.9	**67.4**
SVAMP	86.4	87.0	86.4	86.7	87.7	87.9	**89.3**	82.6	81.6	82.5	82.7	79.1	79.9	**83.9**
M-L1	69.7	69.9	70.4	70.0	71.2	74.3	**77.0**	50.8	50.6	50.3	50.2	57.2	57.5	**63.0**
M-L2	51.1	51.7	52.0	51.5	51.9	53.2	**58.1**	30.2	30.1	29.8	30.0	37.0	37.2	**39.6**
M-L3	31.3	30.3	31.6	30.5	32.3	35.4	**42.8**	22.9	22.5	22.5	22.8	24.7	28.1	**31.9**
M-L4	14.0	14.3	13.5	14.7	14.8	18.8	**26.6**	6.1	6.8	7.1	7.3	**14.5**	13.7	11.5
M-L5	4.9	4.3	5.0	4.6	5.5	7.1	**14.9**	2.4	2.9	2.7	2.6	5.3	5.4	**7.0**

Table 1: Test reasoning accuracy (%) on GSM8K, SVAMP, and MATH. M-L1 to M-L5 denote different difficulty levels in the MATH dataset, where L1 is the easiest and L5 is the most challenging, while Step denotes Step-DPO and FStep denotes Full-Step-DPO. The best results are in **bold**.

4 Experiment

4.1 Settings

Models and Datasets We conduct experiments on four representative backbone models: Llama3-8B (Touvron et al., 2023), Qwen3-8B (Yang et al., 2025), Ministral3-8B (Liu et al., 2026) and Qwen3-4B (Yang et al., 2025). We primarily evaluate our DiffCoT on three public mathematical reasoning benchmarks, specifically GSM8K (Cobbe et al., 2021), SVAMP (Patel et al., 2021), and MATH (Hendrycks et al., 2021). The MATH is divided into five difficulty levels, as defined in the original dataset. More data statistics and preparation are provided in Appendix §A.

Baselines We compare DiffCoT with the following SoTA baselines: 1) **CoT** (Wei et al., 2022): generates step-by-step reasoning before the final answer, evaluated with greedy decoding. 2) **ToT** (Yao et al., 2023): explores multiple reasoning paths via tree search. 3) **TS-SFT** (Feng et al., 2023): applies supervised fine-tuning on reasoning paths obtained from ToT. 4) **CPO** (Zhang et al., 2024): performs preference optimization at the step level, directly aligning intermediate reasoning steps via contrastive pairs. 5) **Step-DPO** (Lai et al., 2025): step-level preference optimization by explicitly collecting erroneous steps and applying preference optimization to correct them. **6) Full-Step-DPO** (Xu et al., 2025): further generalizes step-wise optimization to the entire reasoning trajectory. More implementation details regarding model training and data construction are provided in Appendix §B.

4.2 Main Results

As shown in Table 1, DiffCoT improves reasoning performance on a range of modern instruction-tuned language models, such as Qwen3, Ministral3 and Llama3, under standard fine-tuning. This design enables seamless integration into existing training pipelines and allows DiffCoT to deliver consistent performance improvements across models of varying sizes, demonstrating strong scalability across different base models.

From the result, we can observe that: 1) Baselines guided by LLM self-verification signals, such as CPO and ToT, exhibit notable instability across models and datasets, likely because the quality of self-verification is inherently model and task dependent. When the verification signal is poorly calibrated, it may favor plausible but incorrect reasoning paths, leading to inconsistent gains. For example, on Llama3-8B, CPO degrades performance on MATH-L1 but remains competitive on SVAMP. 2) Step-level preference learning baselines, including Step-DPO and Full-Step-DPO, generally achieve great improvements but still suffer from occasional performance drops under certain settings. This suggests that relying solely on local step-wise op-

timization may not be sufficient to ensure stable reasoning behavior across diverse evaluation scenarios. 3) Overall, DIFFCOT outperforms existing PO approaches on most benchmarks and evaluation settings, although it does not achieve the single best result in every case. For example, on MATH-L1 with the Llama3-8B backbone, Step-DPO attains slightly higher accuracy. Beyond this overall trend, we observe that the gains of DIFFCOT are more pronounced on stronger backbones, such as Qwen3-8B and Qwen3-4B, suggesting that its more complex reasoning trajectories are better utilized by models with stronger instruction following and reasoning capabilities. We also find that DIFFCOT brings larger improvements on harder problems, such as MATH-L4 and MATH-L5, where correct reasoning paths are sparser. This highlights that DIFFCOT enables the model to leverage information across different solution paths, making broader exploration more effective, whereas for easier problems, standard CoT-style depth search is often sufficient.

4.3 Ablation Study

To validate the effectiveness of the proposed DIFFCOT method, we perform an ablation study on its key components. We select two representative base language models from different model families with varied sizes, Llama3-8B and Qwen3-4B, and evaluate them on two general mathematical reasoning benchmarks, GSM8K and SVAMP.

To investigate the impact of incorporating diffusion into CoT reasoning, we evaluate various window sizes to explore how this factor influences the model's performance. Specifically, when the diffusion window size and stride are set to 1, our approach essentially degenerates into an AR method, albeit with differently constructed prefixes. On the other hand, when the sliding diffusion window and stride are set to the number of steps K, the method reverts to a purely diffusion-based approach.

As shown in Table 2, it can be observed that performance degrades when the window size and stride is set to 1 or when it becomes too large. We attribute this phenomenon to a trade-off between causal connectivity and error-correction capability. Strengthening the ability to revise earlier steps often comes at the expense of weakening top-down causal reasoning. When the model operates entirely in AR mode, it exhibits the strongest causal reasoning ability but suffers from exposure bias during testing. In contrast, when the model fully adopts the diffusion mode, an excessively long window introduces noise: it disrupts the structural coherence of reasoning and prolonged denoising fluctuations undermine the causal nature of inference.

Model	GSM8K (%)	SVAMP (%)
Llama3-8B	**64.4**	**76.9**
- window size, stride=1	62.9 -1.5	75.8 -1.1
- window size, stride=K	55.4 -9.0	68.5 -8.4
- causal noise	62.6 -1.8	75.5 -1.4
Qwen3-4B	**88.5**	**90.2**
- window size, stride=1	87.3 -1.2	88.2 -2.0
- window size, stride=K	80.0 -8.5	82.5 -7.7
- causal noise	86.7 -1.8	87.7 -2.5

Table 2: Ablative results on the general datasets GSM8K and SVAMP, where K is the number of reasoning steps. Numbers shown in the upper-right corner of each cell indicate the relative change in accuracy rate compared to the full DIFFCOT model.

We further conduct an ablation study to verify the effectiveness of the causal diffusion noise. Specifically, we disrupt our proposed noise scheduling by randomly shuffling the data order used for noising, instead of following the accuracy-based progression. This modification effectively breaks the causal structure of the noise. Results in Table 2 show that this ablation leads to substantial performance degradation across different models and datasets, demonstrating that our causal noise scheduling is a critical component of DIFFCOT.

4.4 Analysis

Case Study To further illustrate the model's robustness to accumulated imperfections in intermediate reasoning, we present qualitative case studies that contrast effective and suboptimal reasoning trajectories. In particular, we focus on challenging examples in which the model introduces semantically irrelevant or weakly informative steps at an early stage of reasoning. Although such steps are not necessarily incorrect in isolation, they tend to accumulate and hinder progress toward the correct solution under standard step-by-step reasoning.

As shown in Figure 3, DIFFCOT is able to progressively refine earlier reasoning steps through the diffusion process. Instead of rigidly committing to the initial reasoning prefix, the model gradually improves previously generated steps while producing subsequent ones, effectively revising semantically unhelpful content introduced earlier in the trajectory. This allows the model to move away from locally coherent but globally suboptimal reasoning paths and ultimately reach the correct solution. It highlights a key distinction between DIFFCOT and conventional autoregressive reasoning. While

Question: Carla is downloading a 200 GB file. Normally she can download 2 GB/minute, but 40\% of the way through the download, Windows forces a restart to install updates, which takes 20 minutes. Then Carla has to restart the download from the beginning. How load does it take to download the file?

Past Reasoning Steps:
Step 1: 40\% of the way through the download is 0.4 * 200 GB = 80 GB.
Step 2: 80 GB / 2 GB per minute = 40 minutes to download the first 80 GB.
Step 3: 20 minutes for the restart.
Step 4: 200 GB - 80 GB = 120 GB remaining after the restart.

Next Generation:
Step 3: 20 minutes for the restart, so total time so far is 40 + 20 = 60 minutes.
Step 4: Then she downloads the full 200 GB from scratch, taking 100 minutes after the restart starts.
Step 5: The total time can be calculated by adding the time before and after the interruption.

Figure 3: Example illustrating how DIFFCOT modifies early-stage reasoning shift steps. The steps highlighted in blue represent the diffusion sliding window.

autoregressive models tend to propagate early semantic noise forward without revision, DIFFCOT leverages its denoising dynamics to refine the entire reasoning trajectory, including previously generated steps, resulting in more robust and globally consistent reasoning behavior.

Error Accumulation Analysis We further analyze the model's ability to recover from accumulated imperfections in intermediate reasoning steps. We consider a correction-oriented setting in which the model is deliberately conditioned on prefixes that contain semantically suboptimal or noisy reasoning steps, and is then required to continue the reasoning process to reach a correct final answer.

To this end, we introduce controlled perturbations at an intermediate stage of the reasoning process. Specifically, when the reasoning reaches approximately half of the trajectory, each preceding step is independently perturbed with probability ω. A perturbation replaces the original step with a low-reward alternative sampled from model-generated trajectories that are semantically plausible but less aligned with the optimal reasoning path. The noise strength ω therefore governs the degree of accumulated semantic drift in the prefix, enabling a systematic evaluation of robustness under varying levels of intermediate reasoning noise.

We compare different training strategies under this protocol using the correction success rate, defined as the proportion of cases in which the model produces a correct final answer despite being conditioned on a perturbed prefix. Experiments are conducted on 300 GSM8K problems with multiple backbone models, comparing our approach against Full-Step-DPO. As shown in Figure 4, our method consistently achieves substantially higher correction success rates across all settings, demonstrating a stronger ability to compensate for accumulated intermediate noise and to maintain globally coherent reasoning, rather than brittle continuation of imperfect early steps. In other words, our model can mitigate the impact of early-stage reasoning drift, rather than rigidly propagating locally coherent but globally suboptimal trajectories in CoT reasoning.

5 Related Work

Preference Learning Preference learning has recently emerged as an effective paradigm for aligning LLMs with human preferences by contrasting desirable and undesirable responses (Ouyang et al., 2022; Liu et al., 2025), with methods such as DPO showing strong performance on general language tasks (Rafailov et al., 2023). However, extending these gains to mathematical reasoning remains challenging, largely due to the coarse granularity of solution-level supervision, which fails to localize and correct intermediate reasoning errors (Chen et al., 2024b). To address this limitation, recent work incorporates step-level preference signals to align intermediate reasoning processes (Lai et al., 2024; Lu et al., 2024), while Full-Step-DPO further optimizes entire reasoning trajectories using global objectives (Xu et al., 2025). Despite these advances, most existing step-wise methods adopt a teacher-forcing paradigm and reason exclusively over clean prefixes during training, leading to error accumulation and degraded robustness during inference. Accordingly, our work reconceptualizes CoT reasoning as a globally revisable trajectory, in which previously generated steps remain malleable to correction in light of future context.

Mathematical Reasoning Mathematical reasoning is widely regarded as a challenging capability for large language models, and prior work has explored several directions to improve it. One line of research strengthens base models through continual pre-training on large-scale mathematical corpora or supervised fine-tuning on synthetic datasets distilled from stronger models (Azerbayev et al., 2023; Zhihong et al., 2024; Xu et al., 2024; Mitra et al., 2024; Tian et al., 2025a). Another line improves test-time performance by increasing computational

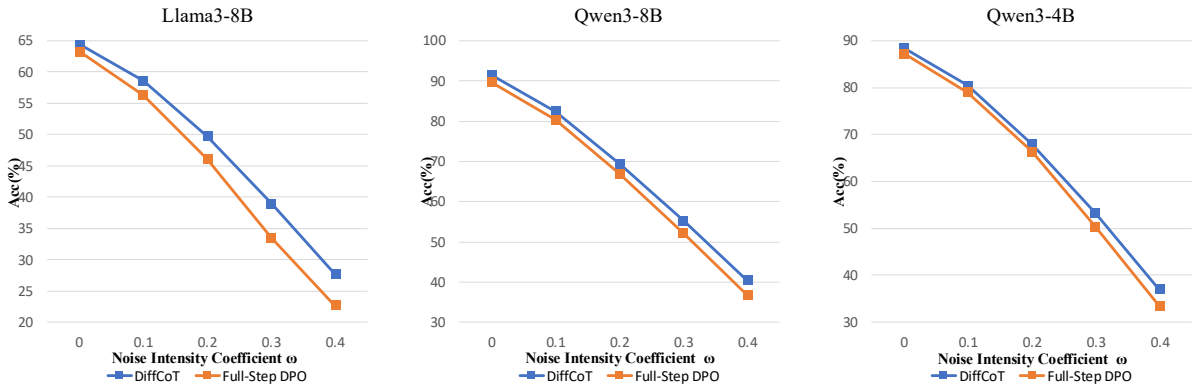

Figure 4: Correction success rate under stochastic prefix corruption, where noise is injected at the midpoint of the reasoning trajectory with probability ω.

budgets, such as generating and reranking multiple solutions using outcome- or process-level rewards, or adopting reward-guided decoding strategies (Guan et al., 2025; Wu et al., 2024; Wang et al., 2024). Recently, reinforcement learning and preference-based optimization have been explored to directly align reasoning behaviors via trajectory- or step-level supervision, aiming to improve robustness beyond supervised objectives alone (Pal et al., 2024; Wang et al., 2024). But they largely operate in a forward-only manner and lack mechanisms for revising corrupted intermediate reasoning, motivating our diffusion-styled reformulation of CoT reasoning to enable effective error correction.

6 Conclusion and Future Work

In this work, we proposed a novel DiffCoT framework to enhance mathematical reasoning and alleviate error accumulation by integrating diffusion steps into autoregressive generation. Our method utilizes a sliding diffusion window with causal noise to refine intermediate steps while maintaining consistent reasoning chains. Extensive experiments confirm that our proposed DiffCoT outperforms existing CoT methods, offering a robust solution for multi-step CoT reasoning. Future work will explore the scalability of our paradigm across more backbone models and broader reasoning domains.

Limitations

Although we have conducted extensive experiments on DiffCoT and observed strong empirical performance, several limitations remain. First, the data construction and training of DiffCoT follow an off-policy paradigm, where preference data are collected using a policy that differs from the one being optimized. Such a mismatch between the behavior policy and the training policy may introduce distribution shift, biased value estimation, and training instability, especially when scaling to more difficult datasets or longer reasoning chains. Second, similar to Diffusion Forcing, DiffCoT breaks the local Markov property of prefix conditioned generation by revisiting and modifying historical reasoning steps. While this violation enables stronger capabilities, it also increases the uncertainty and controllability challenges during generation, typically requiring more training iterations and larger amounts of data to achieve stable convergence.

Acknowledgments

This work is partially supported by National Natural Science Foundation of China Young Scientists Fund (No. 62206233) and RMGS (2025 First Processing Cycle).

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Zhangir Azerbayev, Hailey Schoelkopf, Keiran Paster, Marco Dos Santos, Stephen McAleer, Albert Q. Jiang, Jia Deng, Stella Biderman, and Sean Welleck. 2023. Llemma: An open language model for mathematics. *arXiv preprint arXiv:2310.06786*.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in neural information processing systems*, 28.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind

Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Cameron B Browne, Edward Powley, Daniel Whitehouse, Simon M Lucas, Peter I Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. 2012. A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in games*, 4(1):1–43.

Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. 2024a. Diffusion forcing: Next-token prediction meets full-sequence diffusion. *Advances in Neural Information Processing Systems*, 37:24081–24125.

Guoxin Chen, Minpeng Liao, Chengxi Li, and Kai Fan. 2024b. Step-level value preference optimization for mathematical reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 7889–7903, Miami, Florida, USA. Association for Computational Linguistics.

Ruizhe Chen, Wenhao Chai, Zhifei Yang, Xiaotian Zhang, Ziyang Wang, Tony Quek, Joey Tianyi Zhou, Soujanya Poria, and Zuozhu Liu. 2025. DiffPO: Diffusion-styled preference optimization for inference time alignment of large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 18910–18925, Vienna, Austria. Association for Computational Linguistics.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Prafulla Dhariwal and Alexander Quinn Nichol. 2021. Diffusion models beat gans on image synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Xidong Feng, Ziyu Wan, Muning Wen, Stephen Marcus McAleer, Ying Wen, Weinan Zhang, and Jun Wang. 2023. Alphazero-like tree-search can guide large language model decoding and training. *arXiv preprint arXiv:2309.17179*.

Zeyu Gan, Yun Liao, and Yong Liu. 2025. Rethinking external slow-thinking: From snowball errors to probability of correct reasoning. In *International Conference on Machine Learning*, pages 18170–18188.

Xinyu Guan, Li Lyna Zhang, Yifei Liu, Ning Shang, Youran Sun, Yi Zhu, Fan Yang, and Mao Yang. 2025. rStar-math: Small LLMs can master math reasoning with self-evolved deep thinking. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 20640–20661. PMLR.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, and 1 others. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

Takeshi Kojima, Shixiang Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Preprint*, arXiv:2205.11916.

Xin Lai, Zhuotao Tian, Yukang Chen, Senqiao Yang, Xiangru Peng, and Jiaya Jia. 2024. Step-dpo: Step-wise preference optimization for long-chain reasoning of llms. *arXiv preprint arXiv:2406.18629*.

Xin Lai, Zhuotao Tian, Yukang Chen, Senqiao Yang, Xiangru Peng, and Jiaya Jia. 2025. Step-DPO: Step-wise preference optimization for long-chain reasoning of LLMs.

Xiang Lisa Li, Tianyu Gong, Tian He, Dan Jurafsky, Percy Liang, and Steven CH Zhao. 2022. Diffusion-lm improves controllable text generation. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Yanhong Li, Chenghao Yang, and Allyson Ettinger. 2024. When hindsight is not 20/20: Testing limits on reflective thinking in large language models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3741–3753.

Alexander H Liu, Kartik Khandelwal, Sandeep Subramanian, Victor Jouault, Abhinav Rastogi, Adrien Sadé, Alan Jeffares, Albert Jiang, Alexandre Cahill, Alexandre Gavaudan, and 1 others. 2026. Ministral 3. *arXiv preprint arXiv:2601.08584*.

Tianqi Liu, Zhen Qin, Junru Wu, Jiaming Shen, Misha Khalman, Rishabh Joshi, Yao Zhao, Mohammad Saleh, Simon Baumgartner, Jialu Liu, and 1 others. 2025. Lipo: Listwise preference optimization through learning-to-rank. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 2404–2420.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Zimu Lu, Aojun Zhou, Ke Wang, Houxing Ren, Weikang Shi, Junting Pan, Mingjie Zhan, and Hongsheng Li. 2024. Step-controlled dpo: Leveraging stepwise error for enhanced mathematical reasoning. *CoRR*.

Qing Lyu, Tongshuang Wu, Fei Sha, and Graham Neubig. 2023. Faithful chain-of-thought reasoning. *Preprint*, arXiv:2301.13379.

Mathematical Association of America. 2025. American invitational mathematics examination. https://maa.org/.

Arindam Mitra, Hamed Khanpour, Corby Rosset, and Ahmed Awadallah. 2024. Orca-math: Unlocking the potential of slms in grade school math. *Preprint*, arXiv:2402.14830.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Arka Pal, Deep Karkhanis, Samuel Dooley, Manley Roberts, Siddartha Naidu, and Colin White. 2024. Smaug: Fixing failure modes of preference optimisation with dpo-positive. *arXiv preprint arXiv:2402.13228*.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are nlp models really able to solve simple math word problems? *arXiv preprint arXiv:2103.07191*.

Yiwei Qin, Xuefeng Li, Haoyang Zou, Yixiu Liu, Shijie Xia, Zhen Huang, Yixin Ye, Weizhe Yuan, Hector Liu, Yuanzhi Li, and 1 others. 2024. O1 replication journey: A strategic progress report–part 1. *arXiv preprint arXiv:2410.18982*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. 2011. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings.

Yuxi Sun, Aoqi Zuo, Haotian Xie, Wei Gao, Mingming Gong, and Jing Ma. 2026. Fact-e: Causality-inspired evaluation for trustworthy chain-of-thought reasoning. *Preprint*, arXiv:2604.10693.

Yuchen Tian, Ruiyuan Huang, Xuanwu Wang, Jing Ma, Zengfeng Huang, Ziyang Luo, Hongzhan Lin, Da Zheng, and Lun Du. 2025a. Evolprover: Advancing automated theorem proving by evolving formalized problems via symmetry and difficulty. *arXiv preprint arXiv:2510.00732*.

Yuchen Tian, Weixiang Yan, Qian Yang, Xuandong Zhao, Qian Chen, Wen Wang, Ziyang Luo, Lei Ma, and Dawn Song. 2025b. Codehalu: Investigating code hallucinations in llms via execution-based verification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 25300–25308.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

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

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations (ICLR)*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Zhenyu Wu, Qingkai Zeng, Zhihan Zhang, Zhaoxuan Tan, Chao Shen, and Meng Jiang. 2024. Enhancing mathematical reasoning in llms by stepwise correction. *Preprint*, arXiv:2410.12934.

Zhiheng Xi, Dingwen Yang, Jixuan Huang, Jiafu Tang, Guanyu Li, Yiwen Ding, Wei He, Boyang Hong, Shihan Do, Wenyu Zhan, and 1 others. 2024. Enhancing llm reasoning via critique models with test-time and training-time supervision. *arXiv preprint arXiv:2411.16579*.

Huimin Xu, Xin Mao, Feng-Lin Li, Xiaobao Wu, Wang Chen, Wei Zhang, and Anh Tuan Luu. 2025. Full-step-DPO: Self-supervised preference optimization with step-wise rewards for mathematical reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 24343–24356, Vienna, Austria. Association for Computational Linguistics.

Yifan Xu, Xiao Liu, Xinghan Liu, Zhenyu Hou, Yueyan Li, Xiaohan Zhang, Zihan Wang, Aohan Zeng, Zhengxiao Du, Zhao Wenyi, Jie Tang, and Yuxiao Dong. 2024. ChatGLM-math: Improving math problem-solving in large language models with a self-critique pipeline. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 9733–9760, Miami, Florida, USA. Association for Computational Linguistics.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822.

Jaesik Yoon, Hyeonseo Cho, Doojin Baek, Yoshua Bengio, and Sungjin Ahn. 2025. Monte carlo tree diffusion for system 2 planning. In *Forty-second International Conference on Machine Learning*.

Xuan Zhang, Chao Dut, Tianyu Pang, Qian Liu, Wei Gao, and Min Lin. 2024. Chain of preference optimization: improving chain-of-thought reasoning in llms. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, pages 333–356.

Shao Zhihong, Wang Peiyi, Zhu Qihao, Xu Runxin, Song Junxiao, Zhang Mingchuan, Y. K. Li, Y. Wu, and Guo Daya. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models.

A Data Statistics

We primarily evaluate our DIFFCOT on mathematical reasoning benchmarks, specifically SVAMP (Patel et al., 2021), MATH (Hendrycks et al., 2021) and GSM8K (Cobbe et al., 2021). The MATH is divided into five difficulty levels, as defined in the original dataset. To reduce randomness and better showcase the experimental results, we sample test instances separately from each of five levels. To maintain a reasonable computational budget, particularly given the high cost of Tree-of-Thought (Yao et al., 2023) search, we restrict each dataset to at most 300 test samples through random sampling. For training, we similarly sample 300 instances from each dataset to construct the training set. For a fair comparison, all baseline methods use the same dataset configuration. We provide the detailed data statistics in Table 3.

Dataset	#Train	#Test	#Step K	#window
GSM8K	7473	1319	5	2
SVAMP	794	206	3	2
MATH-L1	564	437	3	2
MATH-L2	1348	894	4	2
MATH-L3	1592	1131	5	2
MATH-L4	1690	1214	6	3
MATH-L5	2304	1324	6	3

Table 3: Dataset-specific configurations for the reasoning experiments. $\#\text{Train}$ and $\#\text{Test}$ denote the numbers of training and test samples, respectively. $\#\text{Step}\ K$ denotes the maximum number of reasoning steps, while $\#\text{window}$ denotes the number of reasoning windows.

B Implementation Details

Data Generation via Rollout-Based Thought Search We generate candidate contexts by conducting an iterative thought search starting from each original problem. Specifically, at reasoning step t, we maintain a prefix that contains all previously selected thoughts.

Conditioned on the current prefix, we sample five candidate thoughts from the model. Figure 5 provides a representative example illustrating the constructed candidate thoughts and their corresponding step-wise evaluations under this procedure.

To evaluate each candidate thought, we estimate its utility via Monte-Carlo rollouts. Concretely, we perform $R = 8$ independent rollouts conditioned on the concatenation of the prefix and the candidate thought. Each rollout is decoded until termination

to obtain a final answer. We then compute the empirical success rate as the fraction of rollouts that yield the correct answer.

We continue the search by greedily selecting the candidate with the highest success rate, appending it to the prefix, and repeating the above procedure until the reasoning process terminates. We provide a detailed example with step-wise reasoning annotations in Figure 5.

Model Training For efficient fine-tuning, we employ Low-Rank Adaptation (LoRA) (Hu et al., 2022) with a rank of 8 and $\alpha = 16$, where LoRA adapters are inserted into the `q_proj` and `v_proj` linear projections of every self-attention layer. Model training is conducted using the DPO (Rafailov et al., 2023) loss with a regularization coefficient of $\beta = 0.1$, optimized by AdamW (Loshchilov and Hutter, 2017) with a cosine learning rate schedule.We train for only one epoch. For the MATH dataset, we use a learning rate of 1×10^{-5} for Qwen3-8B, 2×10^{-5} for Qwen3-4B, 1.5×10^{-5} for Llama3-8B, and 2.5×10^{-5} for Ministral. For SVAMP and GSM8K, we use a learning rate of 2×10^{-5}. We set the warm-up ratio to 0.05 and the global batch size to 8. During decoding, the temperature is fixed at 0. Compared results ($p < 0.05$ under t-test) are averaged over three random runs. All experiments are conducted on NVIDIA A100 GPUs 80GB. Training and inference on 300 GSM8K samples require approximately 11 GPU-hours in total.

Baseline Implementation The instruction-tuned backbone models are implemented by adopting the 'Meta-Llama-3-8B-Instruct[2]', 'Qwen3-8B[3]', 'Ministral3-8B-Instruct-2512[4]' and 'Qwen3-4B-Instruct-2507[5]' versions.

For Full-Step-DPO, we do not retrain the Process Reward Model (PRM) introduced in Full-Step-DPO. In our implementation, we retain the proposed global step-wise loss formulation, but replace the PRM-based step rewards with empirical success rates estimated from Monte-Carlo rollouts. Specifically, we collect extra complete reasoning trajectories from rollouts and use their final outcomes to compute success-based step scores, which serve as substitutes for PRM outputs in the global loss.

Moreover, for CPO and ToT baselines, we follow the implementation described in the CPO paper, where preference pairs are constructed via LLM self-evaluation of intermediate reasoning steps. This design choice highlights the fundamental distinction between CPO-style self-judgment-based supervision and Step-DPO-style supervision derived from execution or outcome feedback.

[2] https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct

[3] https://huggingface.co/Qwen/Qwen3-8B

[4] https://huggingface.co/mistralai/Ministral-3-8B-Instruct-2512-BF16

[5] https://huggingface.co/Qwen/Qwen3-4B-Instruct-2507

C Discussion with Reinforcement Learning

PO methods generally follow a paradigm: trajectory generation followed by trajectory-level reweighting. The model first expands a full reasoning path via longitudinal generation. Alignment is then applied horizontally across completed trajectories using either reinforcement learning or preference optimization, discouraging low-reward reasoning paths and favoring high-reward ones.

While this paradigm of vertical generation followed by horizontal RL-based refinement has achieved broad empirical success, it inevitably suffers from error accumulation (Gan et al., 2025; Tian et al., 2025b; Sun et al., 2026), as early mistakes in the generated trajectory can propagate and constrain subsequent optimization.

Chen et al. (2025) theoretically showed that the denoising process of diffusion models, which gradually transforms low-quality samples into high-quality ones, is equivalent to preference optimization. Building on this insight, DIFFCOT introduces a diffusion sliding window, tightly coupling longitudinal reasoning generation with horizontal preference optimization within a unified framework.

From this perspective, we believe that future extensions of our method can naturally integrate reinforcement learning, leveraging RL to directly optimize the denoising dynamics of the diffusion sliding window.

D Additional Experiments

In this section, we present additional experiments to further evaluate our method. Specifically, we analyze the sensitivity of parameter β in §D.1, report results on the full test sets in §D.2, examine performance under different evaluation protocols in §D.3, and provide additional results on AIME in §D.4.

D.1 Sensitivity Analysis of Parameter β

We analyze the sensitivity of the β parameter in the DPO loss, which controls the difference between the new and old policies. Specifically, we conduct this experiment with Qwen3-4B and Llama3-8B on the SVAMP and MATH-L4 benchmarks.

As shown in Table 4, the results show that our method remains stable within a reasonable variation range, without suffering a sharp performance drop, which demonstrates its robustness.

Model	SVAMP (%)	MATH-L4 (%)
Llama3-8B		
- $\beta = 0.05$	75.0 -1.9	8.3 -1.2
- $\beta = 0.1$	**76.9**	**9.5**
- $\beta = 0.15$	76.5 -0.4	9.0 -0.5
- $\beta = 0.2$	74.7 -1.8	8.7 -0.8
Qwen3-4B		
- $\beta = 0.05$	87.2 -3.0	22.1 -2.3
- $\beta = 0.1$	**90.2**	**24.4**
- $\beta = 0.15$	89.9 -0.3	24.2 -0.2
- $\beta = 0.2$	88.8 -1.4	22.9 -1.5

Table 4: Sensitivity Analysis of Parameter β. The best results are highlighted in **bold**. Numbers shown in the upper-right corner of each cell indicate the relative change in accuracy rate compared to the default $\beta = 0.1$ setting.

D.2 Results on the Full Test Set

To mitigate the potential impact of dataset splitting, we additionally evaluate our method on the full test sets. Specifically, we use Llama3-8B, Qwen3-4B, and Qwen3-8B for evaluation on the full SVAMP test set and the full MATH-L4 test set.

Method	SVAMP (%)	MATH-L4 (%)
Llama3-8B		
CoT	74.7	6.0
Step-DPO	75.8	**8.6**
Full-Step-DPO	75.9	7.4
DiffCoT	**77.1**	7.8
Qwen3-4B		
CoT	80.5	8.2
Step-DPO	88.5	9.4
Full-Step-DPO	87.3	15.6
DiffCoT	**90.2**	**22.6**
Qwen3-8B		
CoT	85.7	12.6
Step-DPO	85.9	12.8
Full-Step-DPO	87.1	17.0
DiffCoT	**89.3**	**23.2**

Table 5: Results on the full SVAMP and MATH-L4 test sets. To mitigate the potential impact of dataset splitting, we additionally evaluate our method using Llama3-8B, Qwen3-4B, and Qwen3-8B on the full test sets. The best results are highlighted in **bold**.

As shown in Table 5, our model achieves the best performance on most model–dataset combinations. In particular, on the Qwen models with MATH-L4, it substantially outperforms previous methods.

D.3 Results under Different Evaluation Protocols

To further assess the generation quality of DiffCoT, we evaluate it under multiple evaluation protocols. In the main paper, we adopt a strict criterion, where a prediction is counted as correct only if the model outputs the correct numerical answer in the required format. To better reflect the actual reasoning ability of DiffCoT, we further consider two additional protocols: *format_agnostic*, which ignores the formatting requirement and extracts the last occurring number in the output as the predicted answer, and *contains_gt*, which checks whether the generated text contains the ground-truth answer anywhere in the response.

Method	Strict (%)	Agnostic(%)	Contain (%)
GSM8K			
CoT	84.7	86.1	88.7
Step-DPO	87.5	88.9	90.7
FStep	87.2	88.4	89.1
DiffCoT	**88.5**	**90.8**	**91.6**
SVAMP			
CoT	86.8	87.8	89.2
Step-DPO	88.3	89.3	**94.7**
FStep	89.7	90.2	92.0
DiffCoT	**90.2**	**90.8**	92.2
MATH-L1			
CoT	59.3	62.7	88.3
Step-DPO	61.9	66.3	89.0
FStep	64.5	67.1	88.7
DiffCoT	**75.0**	**88.0**	**91.7**
MATH-L2			
CoT	35.0	38.9	75.6
Step-DPO	34.1	38.8	75.4
FStep	39.9	41.7	76.0
DiffCoT	**48.7**	**77.0**	**87.7**
MATH-L3			
CoT	21.6	26.2	71.7
Step-DPO	24.5	25.6	75.4
FStep	27.6	29.1	77.0
DiffCoT	**33.1**	**73.8**	**86.3**
MATH-L4			
CoT	10.5	11.6	59.6
Step-DPO	12.3	13.0	59.4
FStep	18.4	17.6	62.1
DiffCoT	**24.4**	**62.9**	**78.3**
MATH-L5			
CoT	3.9	6.8	48.0
Step-DPO	5.0	7.3	51.7
FStep	5.1	8.4	58.9
DiffCoT	**13.2**	**37.3**	**61.9**

Table 6: Qwen3-4B results under different evaluation protocols. The best results are highlighted in **bold**. FStep denotes Full-Step-DPO. Agnostic denotes *format_agnostic*, and Contain denotes *contains_gt*.

The results of Qwen3-4B and Qwen3-8B on different datasets under these evaluation protocols are shown in Table 6 and Table 7. As can be seen, DIFFCOT consistently achieves clear improvements under all three evaluation protocols. This indicates that the gains of our method do not merely come from better instruction following or answer formatting, but reflect a real enhancement in the model's underlying reasoning capability.

Method	Strict (%)	Agnostic(%)	Contain (%)
GSM8K			
CoT	87.3	88.1	90.8
Step-DPO	86.4	87.6	**93.4**
FStep	88.7	89.0	93.3
DIFFCOT	**91.3**	**91.4**	92.9
SVAMP			
CoT	86.4	88.3	91.7
Step-DPO	87.7	88.9	92.1
FStep	87.9	89.4	92.3
DIFFCOT	**89.3**	**89.8**	**93.6**
MATH-L1			
CoT	69.7	75.6	87.0
Step-DPO	71.2	76.7	88.0
FStep	74.3	78.5	**88.3**
DIFFCOT	**77.0**	**79.9**	85.1
MATH-L2			
CoT	51.1	58.0	81.3
Step-DPO	51.9	59.4	**82.2**
FStep	53.2	61.7	81.9
DIFFCOT	**58.1**	**74.3**	78.4
MATH-L3			
CoT	31.3	37.0	77.8
Step-DPO	32.3	37.2	78.1
FStep	35.4	38.0	78.3
DIFFCOT	**42.8**	**71.5**	**79.0**
MATH-L4			
CoT	14.0	16.4	62.4
Step-DPO	14.8	16.5	62.7
FStep	18.8	19.4	63.9
DIFFCOT	**26.6**	**43.9**	**70.1**
MATH-L5			
CoT	4.9	6.9	51.2
Step-DPO	5.5	7.4	52.8
FStep	7.1	8.6	59.2
DIFFCOT	**14.9**	**49.0**	**65.7**

Table 7: Qwen3-8B results under different evaluation protocols. The best results are highlighted in **bold**. FStep denotes Full-Step-DPO. Agnostic denotes *format_agnostic*, and Contain denotes *contains_gt*.

D.4 Results on AIME

To further evaluate the effectiveness of our method on more challenging mathematical reasoning tasks, we additionally conduct experiments on the AIME benchmark (Mathematical Association of America, 2025). Specifically, we train our method on AIME problems from 1983 to 2024 and evaluate it on AIME 2025. We use the experimental setting of step = 6 and window = 3, while keeping all other hyperparameters the same as in the other experiments.

However, since the base reasoning abilities of Llama3-8B and Ministral3-8B are too weak, both models achieve zero correct answers on AIME 1983–2025. As a result, even with multiple rollouts, they produce only a very limited number of correct response samples, which makes preference optimization training infeasible. Therefore, we only report the results of Qwen3-4B and Qwen3-8B.

As shown in Table 8, our method remains competitive on this benchmark and achieves strong performance across different backbones. These results further demonstrate that our method generalizes well to harder mathematical reasoning benchmarks beyond SVAMP and MATH.

Method	Strict (%)	Agnostic(%)	Contain (%)
Qwen3-4B			
CoT	5.5	6.6	23.3
Step-DPO	10.0	10.0	28.8
FStep	12.2	13.3	30.0
DIFFCOT	**30.0**	**32.2**	**35.5**
Qwen3-8B			
CoT	8.8	10.0	17.7
Step-DPO	14.4	14.4	24.4
FStep	12.2	13.3	20.0
DIFFCOT	**20.0**	**20.0**	**26.6**

Table 8: Result on AIME Benchmark. The best results are highlighted in **bold**, FStep denotes Full-Step-DPO. Agnostic denotes *format_agnostic*, and Contain denotes *contains_gt*.

E Generative AI Usage

AI assistants were used in a limited and supportive manner during the preparation of this manuscript, primarily for language polishing, formatting suggestions, and improving clarity of presentation.

Question. In a truck, there are 26 pink hard hats, 15 green hard hats, and 24 yellow hard hats. If Carl takes away 4 pink hard hats, and John takes away 6 pink hard hats and twice as many green hard hats as the number of pink hard hats that he removed, calculate the total number of hard hats that remained in the truck.

Prompt. You are an expert math problem solver. You must reason step by step and avoid logical or arithmetic mistakes. Solve the following math problem. You MUST use exactly 5 reasoning steps. After the reasoning, output the final answer in the last line using the format: `Answer: <final_answer>`.

Step 1.

[0] 26 pink hard hats $-$ 4 pink hard hats $=$ 22 pink hard hats `success_rate=0.0`

[1] $26 - 4 =$? (number of pink hats after Carl's removal) `success_rate=0.125`

[2] Initial total number of hard hats: 26 + 15 + 24 = 65. `success_rate=0.5`

[3] Initial number of pink hard hats is 26. `success_rate=0.5`

[4] We have 26 pink hard hats initially. Carl takes away 4 pink hard hats, leaving us with $26 - 4 = 22$ pink hard hats. `success_rate=0.75`

Step 2.

[0] John removed 6 pink hats and 6 green hats, leaving $15 - 6 = 9$ green hats. `success_rate=0.0`

[1] John removes 6 pink hats, so he removes 6 green hats as well. Thus $15 - 6 = 9$ green remain. `success_rate=0.125`

[2] Carl removes 4 pink hard hats: $26 - 4 = 22$. `success_rate=0.5`

[3] 6 pink hats are removed by John, leaving $22 - 6 = 16$ pink hats. `success_rate=0.625`

[4] John removes 6 pink hats: $22 - 6 = 16$ pink left. He removes twice as many green hats: $2 \times 6 = 12$,so $15 - 12 = 3$ green left. `success_rate=1.0`

Step 3.

[0] 24 yellow hard hats remain in the truck `success_rate=0.75`

[1] We still have 24 yellow hard hats. `success_rate=0.75`

[2] Yellow hard hats remain unchanged, so we still have 24 yellow hard hats. `success_rate=0.875`

[3] 24 yellow hard hats remain in the truck and are not affected by the removals. `success_rate=0.875`

[4] 24 yellow hard hats remain unchanged. `success_rate=1.0`

Step 4.

[1] 19 pink, 3 green, and 24 yellow hard hats remain in the truck. `success_rate=0.0`

[1] 16 pink, 3 green, and 24 yellow hard hats remain in the truck. `success_rate=0.75`

[2] 16 pink and 3 green hard hats remain; $16 + 3 = 19$. `success_rate=0.875`

[3] Total remaining hats: $16 + 3 + 24 = 43$. `success_rate=0.875`

[4] 16 pink and 3 green hard hats remain, so $16 + 3 = 19$ hats across these two colors. `success_rate=1.0`

Step 5.

[0] Total: $3 + 19 + 24 =$? `success_rate=0.0`

[1] Total remaining hats: $3 + 19 + 24 =$? `success_rate=0.0`

[2] Total: $19 + 24 = 43$. `success_rate=0.875`

[3] Combining 19 (pink+green) with 24 yellow gives $19 + 24 = 43$. `success_rate=1.0`

[4] Total remaining hats: $19 + 24 = 43$. `success_rate=1.0`

Figure 5: Representative dataset example with step-wise reasoning annotations.