ENHANCING REASONING FOR DIFFUSION LLMS VIA DISTRIBUTION MATCHING POLICY OPTIMIZATION

Yuchen Zhu*, **Wei Guo***, **Jaemoo Choi, Petr Molodyk, Bo Yuan, Molei Tao**†, **Yongxin Chen**†
Georgia Institute of Technology
{yzhu738, wei.guo, mtao, yongchen}@gatech.edu

ABSTRACT

Diffusion large language models (dLLMs) are promising alternatives to autoregressive large language models (AR-LLMs), as they potentially allow higher inference throughput. Reinforcement learning (RL) is a crucial component for dLLMs to achieve comparable performance with AR-LLMs on important tasks, such as reasoning. However, RL algorithms that are well-suited for dLLMs' unique characteristics have yet to be developed. This paper proposes **Distribution Matching Policy Optimization (DMPO)**, a principled and theoretically grounded RL fine-tuning method specifically designed to enhance the reasoning capabilities of dLLMs by matching the dLLM policy distribution to the optimal, reward-tilted one through cross-entropy optimization. We identify a key challenge in the implementation with a small training batch size and propose several effective solutions through a novel weight baseline subtraction technique. DMPO exhibits superior performance on multiple reasoning benchmarks without supervised fine-tuning, with an accuracy improvement of up to 54.3% over previously SOTA baselines and 66.41% over the base model, underscoring the effectiveness of the distribution matching framework. Our code is available at https://github.com/yuchen-zhu-zyc/DMPO.

1 INTRODUCTION

Autoregressive large language models (AR-LLMs) have demonstrated remarkable capabilities in addressing sophisticated reasoning tasks, such as solving challenging math questions and completing coding tasks (Jaech et al., 2024; Anthropic, 2025; Guo et al., 2025a; Novikov et al., 2025; Kimi Team et al., 2025). While these models form their amazing capabilities from pretraining on massive text corpora, the main powerhouse behind the success is scaling the post-training phase with reinforcement learning (RL) techniques, such as Proximal Policy Optimization (PPO, Schulman et al. (2017)) and Group Relative Policy Optimization (GRPO, Shao et al. (2024)), which enhance model abilities through exploration of reward functions and go beyond static datasets. While possessing extraordinary competence, AR-LLMs are known to be expensive for inference due to their sequential, fixed left-to-right generation order, which currently prohibits large-scale deployment.

With the aim of addressing such issues, diffusion large language models (dLLMs) have been investigated as an alternative to the AR models. Unlike their counterparts, dLLMs iteratively refine a sequence from a masked state, allowing for any-order generation, and have shown promising performance in text generation tasks. dLLMs, such as LLaDA (Nie et al., 2025b) and Dream (Ye et al., 2025), have demonstrated competitive performances on many tasks compared to similar-size AR baselines. Recently, commercial models such as Mercury (Inception Labs et al., 2025) and Gemini Diffusion (DeepMind) have demonstrated the capability to achieve a magnitude higher inference throughput without sacrificing generation quality, suggesting dLLM as a promising future direction for language modeling. However, one question that remains largely unanswered is how to transfer the success of RL on LLM to dLLM, thereby further scaling up the model's skills.

Designing RL algorithms for dLLMs faces two major challenges. Due to the bidirectional nature of dLLMs, estimating the log probability of the generated sequences is more expensive than for AR models, making it less favorable to naively adapt LLM post-training algorithms like GRPO to dLLMs,

*Core contributors, equal contribution. †Equal advising.

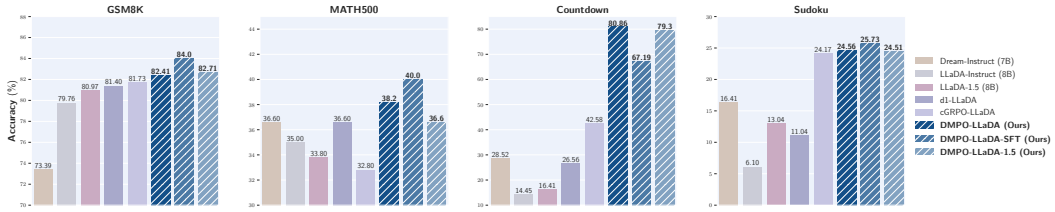


Figure 1: Performance on reasoning benchmarks evaluated with generation length 256. DMPO consistently achieves the best performance across dLLMs, outperforming d1 and cGRPO.

as they heavily rely on such estimation. The GRPO-style algorithms also do not leverage dLLM's unique characteristic of having a *forward* noising process, as they are backward-only algorithms when using generated rollouts. Moreover, existing RL frameworks for enhancing LLM reasoning capabilities overly focus on reward maximization (Guo et al., 2025a; Liu et al., 2025c; Zheng et al., 2025a). By targeting only the reward mode, these approaches do not properly utilize dLLM's potential in generating more diverse responses than LLMs due to the random-order nature (Gong et al., 2025).

To jointly address these challenges, we propose **Distribution Matching Policy Optimization (DMPO)**, a principled and efficient RL fine-tuning method specifically designed for dLLMs. DMPO is designed based on a novel framework theoretically grounded in stochastic optimal control (SOC), which shifts away from the conventional reward maximization paradigm and targets a new goal of matching the entire reward-tilted policy distribution. This enables the model to explore diverse, high-quality reasoning paths and responses during training, addressing concerns about over-focusing on absolute reward values and modes. Moreover, DMPO training leverages importance sampling and a novel weighted denoising cross-entropy (WDCE) loss, which enjoys the key advantage of operating in an *off-policy* manner, allowing the use of replay buffers for improved sample efficiency. More importantly, WDCE is a *forward-only* objective that relies solely on the obtained clean samples and the inexpensive, forward-noising process unique to diffusion LLMs. DMPO largely discards the dependence on rollout trajectories, enabling it to potentially enjoy more speed-up than other dLLM RL algorithms when employed with fast inference techniques.

Contributions. The core contributions of this paper are summarized as follows: **(I)** We propose a novel RL learning framework for dLLMs that targets distribution matching rather than reward maximization (Sec. 3.1). **(II)** We propose Distribution Matching Policy Optimization (DMPO), a principled, theoretically-grounded fine-tuning strategy for enhancing dLLM's reasoning capabilities, supported by importance sampling and weighted denoising cross-entropy (Sec. 3.2). **(III)** We identify a special challenge that occurred for WDCE due to the use of a limited training batch size, and propose two novel techniques to address it: weight baseline subtraction (Sec. 3.3) and weighted direct discriminative optimization (Sec. 3.4). **(IV)** DMPO exhibits superior performances on multiple reasoning benchmarks without supervised fine-tuning (SFT), with an accuracy improvement up to 54.3% over previously SOTA baselines and 66.41% over the base model, being top-performing across bi-directional dLLMs (Sec. 4).

2 Preliminaries

2.1 Masked Diffusion Models for Language Modeling

The **masked (discrete) diffusion models (MDM)** (Lou et al., 2024; Ou et al., 2025; Sahoo et al., 2024; Shi et al., 2024; Zheng et al., 2025f) is a novel method for learning high-dimensional categorical distributions with application to text (Nie et al., 2025b), images (Chang et al., 2022; Bai et al., 2025), DNAs (Hayes et al., 2025), etc. Essentially, it learns the one-dimensional conditional distributions of the data given any subset of observed dimensions. Suppose the data are finite-length sequences with vocabulary $\mathcal{V} = \{1, 2, ..., V\}$. Include the mask token M into the $\mathcal{V}$ and let $\overline{\mathcal{V}} = \{1, 2, ..., V, \mathsf{M}\}$. The MDM takes a partially masked sequence $\boldsymbol{x} = (x_1, ..., x_D) \in \overline{\mathcal{V}}^D$ as an input, and outputs $\boldsymbol{\pi}_\theta(\boldsymbol{x}) \in \mathbb{R}^{D\times V}$, whose (d, u)-th entry $\boldsymbol{\pi}_\theta(\boldsymbol{x})_{d,u}$ is set to $1_{x_d=u}$ if $x_d \neq \mathsf{M}$, and if $x_d = \mathsf{M}$, is trained to approximate the conditional probability

$$\Pr_{\boldsymbol{X}\sim p_{\text{data}}}(X_d = u | \boldsymbol{X}_{\text{UM}} = \boldsymbol{x}_{\text{UM}}), \quad \text{where } \boldsymbol{x}_{\text{UM}} = (x_d : x_d \neq \mathsf{M}).$$

By definition, we assume each row of $\boldsymbol{\pi}_\theta(\boldsymbol{x})$ is a valid probability vector. The probability of a unmasked sequence $\boldsymbol{x} \in \mathcal{V}^D$ under the MDM $\boldsymbol{\pi}_\theta$ is defined through **random-order autoregressive**

(AR) generation: choosing a uniformly random order of the D positions, and autoregressively sampling each position conditional on the previously sampled ones. Formally,

$$p_\theta(\boldsymbol{x}) = \mathbb{E}_{\boldsymbol{\sigma}}\, p_\theta(\boldsymbol{x};\boldsymbol{\sigma}), \quad \text{where } \boldsymbol{\sigma} \sim \mathrm{Unif}(S_D) \text{ and } p_\theta(\boldsymbol{x};\boldsymbol{\sigma}) = \prod_{d=1}^{D} \boldsymbol{\pi}_\theta(x_{\sigma_d}|\boldsymbol{x}_{\boldsymbol{\sigma}_{<d}}). \tag{1}$$

Here, S_D is the set of all permutations of $\{1, ..., D\}$; $\boldsymbol{\pi}_\theta(x_{\sigma_d}|\boldsymbol{x}_{\boldsymbol{\sigma}_{<d}})$ means input $\boldsymbol{x}$ with all positions except $\boldsymbol{\sigma}_{<d} = \{\sigma_1, ..., \sigma_{d-1}\}$ masked into the MDM and take the output at position (σ_d, x_{σ_d}).

The standard way to train a masked discrete diffusion model given i.i.d. samples from p_{data} is to minimize the **denoising cross-entropy (DCE)** loss $\mathbb{E}_{p_{\mathrm{data}}(\boldsymbol{x})}\, \mathcal{L}_\theta(\boldsymbol{x})$, which involves the following definition of the (negative) **evidence lower bound (ELBO)** $\mathcal{L}_\theta$:

$$\begin{aligned} -\log p_\theta(\boldsymbol{x}) &= -\log \mathbb{E}_{\boldsymbol{\sigma}}\, p_\theta(\boldsymbol{x};\boldsymbol{\sigma}) \le -\,\mathbb{E}_{\boldsymbol{\sigma}} \log p_\theta(\boldsymbol{x};\boldsymbol{\sigma}) \qquad (\textit{Jensen's inequality}) \\ &= \mathbb{E}_{m\sim\mathrm{Unif}\{1,\ldots,|\boldsymbol{x}|\}} \left[\frac{|\boldsymbol{x}|}{m}\, \mathbb{E}_{\mu_m(\widetilde{\boldsymbol{x}}|\boldsymbol{x})} \sum_{d:\widetilde{x}_d=\mathsf{M}} -\log \boldsymbol{\pi}_\theta(\widetilde{\boldsymbol{x}})_{d,x_d} \right] =: \mathcal{L}_\theta(\boldsymbol{x}), \end{aligned} \tag{2}$$

where the transition distribution $\mu_m(\cdot|\boldsymbol{x})$ means to sample a uniformly random subset of $\{1, ..., |\boldsymbol{x}|\}$ of size m and mask the corresponding entries in $\boldsymbol{x}$, and $|\boldsymbol{x}|$ is the length of $\boldsymbol{x}$. The proof of the last equation can be found in Uria et al. (2016); Ou et al. (2025).

When applying to text data, the MDM is also referred to as the **diffusion large language model (dLLM)** (Nie et al., 2025b; Ye et al., 2025; Inception Labs et al., 2025; Song et al., 2025). For the purpose of reasoning, we typically write $\boldsymbol{x} = (\boldsymbol{q}, \boldsymbol{o})$, where $\boldsymbol{q}$ is the **prompt** (or query, which is always assumed to contain no mask state) and $\boldsymbol{o}$ is the **response** (or output). We use $\pi_\theta(\boldsymbol{o}|\boldsymbol{q}) \in \mathbb{R}^{|\boldsymbol{o}|\times V}$ to denote the policy model output of the dLLM given a prompt $\boldsymbol{q}$ and a partially masked response $\boldsymbol{o}$. The conditional sequence probability of a clean model $\boldsymbol{o}$ given a prompt $\boldsymbol{q}$, denoted as $p_\theta(\boldsymbol{o}|\boldsymbol{q})$, is similarly defined through (1), where we now use notations $p_\theta(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})$ and $\boldsymbol{\pi}_\theta(o_{\sigma_d}|\boldsymbol{q}, \boldsymbol{o}_{\boldsymbol{\sigma}_{<d}})$ to emphasize the dependence on the prompt $\boldsymbol{q}$. The negative ELBO will be written as $\mathcal{L}_\theta(\boldsymbol{o}|\boldsymbol{q})$.

2.2 Reinforcement Learning for Enhancing Reasoning

We first present the **Group Relative Policy Optimization** (**GRPO**, Shao et al. (2024)) method for LLMs, which is the basis of most of the existing RL methods for dLLMs. Given a pretrained LLM with policy π_{ref} that samples from the distribution $p_{\mathrm{ref}}(\boldsymbol{o}|\boldsymbol{q}) = \prod_{d=1}^{|\boldsymbol{o}|} \pi_{\mathrm{ref}}(o_d|\boldsymbol{q}, \boldsymbol{o}_{<d})$, a reward function $r : (\boldsymbol{q}, \boldsymbol{o}) \mapsto \mathbb{R}$, a set of prompts $\mathcal{D}$, and a regularization parameter $\alpha \ge 0$, each step of the GRPO aims to solve the following problem: sample $\boldsymbol{q} \sim \mathcal{D}$, $\boldsymbol{o}^{(1:G)} \overset{\text{i.i.d.}}{\sim} p_{\theta_{\mathrm{old}}}(\boldsymbol{o}|\boldsymbol{q})$, and maximize

$$\mathbb{E}\left\{ \frac{1}{G}\sum_{i=1}^{G} \frac{1}{|\boldsymbol{o}^{(i)}|} \sum_{d=1}^{|\boldsymbol{o}^{(i)}|} \left[\min\left(\rho_d^{(i)} A_i, \mathrm{clip}(\rho_d^{(i)})_{1\pm\epsilon} A_i \right) - \alpha\, \mathrm{KL}(p_\theta(\boldsymbol{o}^{(i)}|\boldsymbol{q}) \| p_{\mathrm{ref}}(\boldsymbol{o}^{(i)}|\boldsymbol{q})) \right] \right\}, \tag{3}$$

where the advantages[1] are $A_i = r(\boldsymbol{q}, \boldsymbol{o}^{(i)}) - \mathrm{mean}(r(\boldsymbol{q}, \boldsymbol{o}^{(1:G)}))$, the per-token probability ratios are $\rho_d^{(i)} = \frac{\pi_\theta(o_d^{(i)}|\boldsymbol{q}, \boldsymbol{o}_{<d}^{(i)})}{\pi_{\theta_{\mathrm{old}}}(o_d^{(i)}|\boldsymbol{q}, \boldsymbol{o}_{<d}^{(i)})}$, and the KL regularization term is estimated similarly by the per-token probability ratios between π_θ and π_{ref}. The clipping threshold ϵ prevents overly large policy updates.

While (3) works well for LLMs, it is not directly applicable to dLLMs due to mismatch between the *dLLM policy (model output)* $\boldsymbol{\pi}_\theta(\boldsymbol{o}|\boldsymbol{q})$ and the *sequence likelihood* $p_\theta(\boldsymbol{o}|\boldsymbol{q})$: unlike in LLMs where these two quantities are easily connected through the chain rule, it is generally non-trivial to compute the per-token probability given the dLLM model output, and only ELBO (2) is available as a surrogate. To tackle this issue, diffu-GRPO (Zhao et al., 2025a) proposed to fully mask all response positions and partially masks the prompt $\boldsymbol{q}$, and feed this sequence into the model to obtain the approximate probability $p_\theta(o_d|\boldsymbol{q})$. Next, the sequence probability $p_\theta(\boldsymbol{o}|\boldsymbol{q})$ is approximated by mean-field decomposition: $p_\theta(\boldsymbol{o}|\boldsymbol{q}) \approx \prod_{d=1}^{|\boldsymbol{o}|} p_\theta(o_d|\boldsymbol{q})$. Such approximations do not capture the correlation between different positions in the response, which produces imprecision. A similar technique is employed in coupled-GRPO (cGRPO) for code generation tasks in Gong et al. (2025).

[1] As suggested by Liu et al. (2025c), we list here the version without normalization by standard deviation.

3 Distribution Matching Policy Optimization

3.1 From Reward Maximization to Distribution Matching

To incentivize the reasoning capabilities of large language models, reward-maximizing reinforcement learning finetuning algorithms, such as TRPO (Schulman et al., 2015), PPO (Schulman et al., 2017), and GRPO (Shao et al., 2024), are often employed, with an additional entropy regularization term that penalizes the deviation of the model from the pretrained one. This process amounts to solving the following optimization problem,

$$\max_{\theta} \mathbb{E}_{\boldsymbol{q}\sim\mathcal{D}}\left[\mathbb{E}_{p_\theta(\boldsymbol{o}|\boldsymbol{q})}[r(\boldsymbol{q},\boldsymbol{o})] - \alpha\,\mathrm{KL}(p_\theta(\cdot|\boldsymbol{q})\|p_{\mathrm{ref}}(\cdot|\boldsymbol{q}))\right]. \quad (4)$$

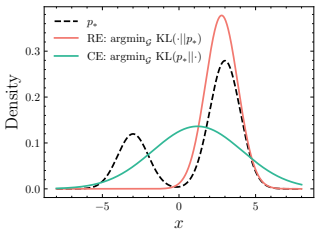


Figure 2: Illustration of relative entropy (mode-seeking) and cross-entropy (mass-covering) for fitting a target p_* ($\mathcal{G}$ is the set of Gaussian distributions).

However, existing techniques over-focus on finding and optimizing the **reward mode** and adopt many heuristic techniques to accelerate the mode searching process, neglecting the exploration of the entire distribution landscape, and often result in model mode collapse or reward hacking, causing the model to produce undesirable responses (Weng, 2024). A simple fix to this issue and to encourage diverse model responses is to enforce the optimality of the target policy distribution during the training. It can be shown that the optimal sequence distribution that solves the problem (4) is the following **reward-tilted distribution**:

$$p_*(\boldsymbol{o}|\boldsymbol{q}) = \frac{1}{Z(\boldsymbol{q})} p_{\mathrm{ref}}(\boldsymbol{o}|\boldsymbol{q}) \mathrm{e}^{r(\boldsymbol{q},\boldsymbol{o})/\alpha}, \quad \text{where } Z(\boldsymbol{q}) = \sum_{\boldsymbol{o}} p_{\mathrm{ref}}(\boldsymbol{o}|\boldsymbol{q}) \mathrm{e}^{r(\boldsymbol{q},\boldsymbol{o})/\alpha}. \quad (5)$$

That is to say, we want to use the optimal sequential distribution $p_*(\boldsymbol{o}|\boldsymbol{q})$ as the **supervision signal** throughout the learning process, so that we can learn a dLLM policy $\boldsymbol{\pi}_\theta$ which produces a sequence distribution p_θ matching p_*. We can thus obtain a policy that not only explores the dominant reward mode, but is guaranteed to sample other high-reward trajectories with a likelihood proportional to the reward value. This motivates us to consider the following task of **policy distribution matching**,

Policy Distribution Matching Learning: Given a pretrained dLLM policy $\boldsymbol{\pi}_{\mathrm{ref}}(\boldsymbol{o}|\boldsymbol{q})$ that samples from $p_{\mathrm{ref}}(\boldsymbol{o}|\boldsymbol{q})$, a reward function $r : (\boldsymbol{q},\boldsymbol{o}) \mapsto \mathbb{R}$, a set of prompts $\mathcal{D}$, and temperature $\alpha > 0$, learn a dLLM policy $\boldsymbol{\pi}_\theta(\boldsymbol{o}|\boldsymbol{q})$ to produce the desired optimal distribution $p_*(\boldsymbol{o}|\boldsymbol{q})$ in (5) by

$$\min_{\boldsymbol{\pi}_\theta} \mathbb{E}_{\boldsymbol{q}\sim\mathcal{D}}\, \mathcal{F}(p_\theta(\cdot|\boldsymbol{q}), p_*(\cdot|\boldsymbol{q})). \quad (6)$$

Here, $\mathcal{F}$ is a class of functionals such that $\mathrm{argmin}_p \mathcal{F}(p, p_*) = p_*$. Note that the original entropy-regularized entropy optimization problem is equivalent to choosing $\mathcal{F}$ to be the reverse KL between p and p_*, i.e., $\mathcal{F}(p_\theta, p_*) = \mathrm{KL}(p_\theta\|p_*) = \mathbb{E}_{p_\theta}[\log \frac{p_\theta}{p_*}]$. While this objective in theory can also lead to the same optimal distribution with the desired property, it is widely known that reverse KL is *mode-seeking*, i.e., it tends to match the most dominant mode in p_* while potentially neglecting other modes, which may lead to reward hacking.

To address this issue, we consider a series of new objectives $\mathcal{F}$ with more desirable convergence guarantees that steadily lead to optimization towards the desired sequence distribution, and propose **Distribution Matching Policy Optimization (DMPO)** (Alg. 1), which targets matching the entire reward-tilted policy distribution. In Sec. 3.2, we introduce **weighted denoising cross-entropy (WDCE)**, a **scalable** implementation of the forward KL using importance sampling. In Secs. 3.3 and 3.4, we discuss an important failure case of forward KL with **small training batch size**, and propose a series of novel techniques such as **weight baseline subtraction** (Sec. 3.3) and **weighted direct discriminative optimization** (Sec. 3.4) to address it.

3.2 Weighted Denoising Cross-entropy

Unlike the reverse KL objective considered by many existing works, which are known to be prone to mode seeking and collapse, one alternative choice is to use the forward KL divergence (or **cross-entropy, CE**) for the functional, i.e., $\mathcal{F}(p_\theta, p_*) = \mathrm{KL}(p_*\|p_\theta)$, which tends to cover all the modes

of the optimal distribution and can retain the response diversity. The CE loss is also widely used in another domain known as stochastic optimal control (SOC) (Domingo-Enrich et al., 2024; 2025), which is also closely connected with our work. This amounts to solving the following task,

$$\min_{\theta} \mathbb{E}_{\boldsymbol{q}\sim\mathcal{D}} \mathbb{E}_{p_*(\boldsymbol{o}|\boldsymbol{q})} \left[\log \frac{p_*(\boldsymbol{o}|\boldsymbol{q})}{p_\theta(\boldsymbol{o}|\boldsymbol{q})}\right]. \tag{7}$$

However, objective (7) is not directly amenable to practical implementation, as we do not have access to real samples from the p_*, nor can we exactly compute $\log p_*$ due to the presence of the unknown partition function $Z(\boldsymbol{q})$. To bypass this issue, we draw inspiration from the recent work masked diffusion neural sampler (MDNS, Zhu et al. (2025g)), which proposes a training framework for learning a masked diffusion neural sampler with stochastic optimal control and cross-entropy minimization. While targeting a different task, the core of MDNS resides in solving the same distribution matching problem with cross-entropy loss, and it proposes a practically implementable and scalable variant of (7), named **weighted denoising cross-entropy (WDCE)** loss. The central idea is to introduce a reference policy and leverage *importance sampling*, so that we can treat i.i.d. samples as importance-weighted samples from p_*. Taking advantage of this approach, we now derive WDCE for the purpose of dLLM policy learning.

First, given the relationship between the policy output and sequence distribution of the masked dLLM (1), it is clear that we can match the correct target sequence distribution $p_*(\boldsymbol{o}|\boldsymbol{q})$ as long as we train $p_\theta(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})$ to match the *order-specific* ones, i.e., $p_*(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})$, given by

$$p_*(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma}) = \frac{1}{Z(\boldsymbol{q})} p_{\mathrm{ref}}(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma}) \mathrm{e}^{r(\boldsymbol{q},\boldsymbol{o})/\alpha}. \tag{8}$$

Leveraging this fact, given any prompt $\boldsymbol{q}$, we can express the cross-entropy loss as follows:

$$\begin{aligned} \mathrm{KL}(p_*(\cdot|\boldsymbol{q})\|p_\theta(\cdot|\boldsymbol{q})) &= \mathbb{E}_{p_*(\boldsymbol{o}|\boldsymbol{q})}[-\log p_\theta(\boldsymbol{o}|\boldsymbol{q})] + \mathrm{const} \\ &= \mathbb{E}_{\boldsymbol{\sigma}} \mathbb{E}_{p_v(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})} \frac{p_*(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})}{p_v(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})}[-\log p_\theta(\boldsymbol{o}|\boldsymbol{q})] + \mathrm{const}, \end{aligned} \tag{9}$$

where p_v is the sequence probability under a reference policy model v that does not involve gradient computation, and in practice, one often chooses $v \leftarrow \bar{\theta} := \mathrm{stopgrad}(\theta)$ to be a copy of the policy model detached from the computation graph, and periodically synchronizes with the current model policy p_θ, which is also commonly referred to as $p_{\theta_{\mathrm{old}}}$ in the literature. The importance weight $w(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma}) := \frac{p_*(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})}{p_v(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})}$ captures the mismatch between p_v and p_* and ensures the mathematical correctness of the objective, and $\log p_\theta(\boldsymbol{o}|\boldsymbol{q})$ is an intractable sequence log probability under the current dLLM policy. We discuss the computation of these two components in parallel below.

Importance weight $w(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})$. We simplify it with the pretrained model and the reward:

$$w(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma}) = \frac{1}{Z(\boldsymbol{q})} \frac{p_{\mathrm{ref}}(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})}{p_v(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})} \mathrm{e}^{\frac{r(\boldsymbol{q},\boldsymbol{o})}{\alpha}} \propto \exp\left(\frac{r(\boldsymbol{q},\boldsymbol{o})}{\alpha} + \log \frac{p_{\mathrm{ref}}(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})}{p_v(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})}\right) =: \mathrm{e}^{\ell(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})}. \tag{10}$$

Recall that the order-specific probability of a sequence is computed via (1). To ensure that the sample distribution after importance sampling is valid and normalized, we keep track of the **log weights** $\ell(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})$, and taking softmax among those corresponding to the same prompt $\boldsymbol{q}$ to compute the real weight $w(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})$. This is equivalent to estimating the unknown partition function $Z(\boldsymbol{q})$ using an empirical estimator of the following expectation:

$$Z(\boldsymbol{q}) = \mathbb{E}_{\boldsymbol{\sigma}} \mathbb{E}_{p_v(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})} \left[\frac{p_{\mathrm{ref}}(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})}{p_v(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})} \mathrm{e}^{r(\boldsymbol{q},\boldsymbol{o})/\alpha}\right],$$

The need to estimate partition functions is common in RL algorithms for LLM, such as in GflowNet (Bengio et al., 2021; Kimi Team et al., 2025). In contrast to these approaches that learn such functions independently, our estimation approach is training-free and more efficient.

Sequence log probability $\log p_v(\boldsymbol{o}|\boldsymbol{q})$. Unlike the case of LLM, the exact sequence log probability is intractable due to the presence of expectation over the random order $\boldsymbol{\sigma}$. However, similar to the training of dLLM, we can leverage the negative ELBO (2) as a surrogate. Combined with the importance weight $w(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})$, we introduce the **weighted denoising cross-entropy (WDCE)** loss for dLLM policy distribution matching:

$$\min_{\theta} \mathop{\mathbb{E}}_{\boldsymbol{q}\sim\mathcal{D}} \mathop{\mathbb{E}}_{\boldsymbol{\sigma}} \mathop{\mathbb{E}}_{p_v(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})} \left\{ w(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma}) \mathop{\mathbb{E}}_{m\sim\mathrm{Unif}\{1,\ldots,|\boldsymbol{o}|\}} \left[\frac{|\boldsymbol{o}|}{m} \mathop{\mathbb{E}}_{\mu_m(\widetilde{\boldsymbol{o}}|\boldsymbol{o})} \sum_{d:\widetilde{o}_d=\mathsf{M}} -\log \boldsymbol{\pi}_\theta(\widetilde{\boldsymbol{o}}|\boldsymbol{q})_{d,o_d}\right]\right\}. \tag{11}$$

Notably, this loss highly resembles the DCE loss used in pre-training and the supervised fine-tuning (SFT) phase of dLLM. One major difference is that instead of using i.i.d. samples from p_*, we use importance sampling to weight samples from p_v and obtain a valid training objective with theoretical guarantees. WDCE differs significantly from other popular RL training techniques such as PPO/GRPO in two key aspects.

WDCE is an *off-policy* loss. The WDCE loss remains valid as the model parameter θ gets updated, since both the sampling policy p_v and the important sampling target policy p_* are independent of the current model policy p_θ. This allows us to save generated rollouts in a replay buffer and reuse them for multiple training updates, without worrying excessively about numerical instability, leading to improved sample efficiency. On the other hand, for on-policy methods, to use a replay buffer, one would need to estimate importance weights with respect to the current model policy $p_\theta(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})$, i.e., $\frac{p_\theta(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})}{p_v(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})}$. Different from the case of LLM, where such estimation can be done in one model forward pass, an accurate estimation in dLLM **per training update** is expensive, rendering the on-policy method less efficient. Moreover, for large models, where rollout generation and sequence likelihood estimation are typically handled by different implementations (such as vLLM and FSDP), this could lead to more nuanced, hard-to-detect biases that secretly undermine the algorithm's performance (Yao et al., 2025). With WDCE, we are largely free of such concerns.

WDCE is a *forward* loss. Unlike the GRPO-style of algorithms that typically require keeping track of the entire rollout trajectories, WDCE leverages the forward noising process in training, which is a characteristic unique to **diffusion** LLMs. Once we obtain the final samples and their associated weights, we can discard the trajectories and perform training using the cheap forward process by randomly masking the data. This implies that the training speed when using WDCE largely depends on the model inference speed. With the advances of dLLM efficient inference techniques such as fast decoding algorithms and KV-cache techniques (Ma et al., 2025; Hu et al., 2025; Wu et al., 2025; Liu et al., 2025b), WDCE could also enjoy a great boost in efficiency. This method also effectively utilizes dLLM's potential in surpassing LLMs in inference throughput, distinguishing it from other RL baselines that merely adapt LLM algorithms to dLLM. We defer a more detailed discussion of such properties to App. B.2.

Finally, we remark that while we developed the WDCE loss through the lens of policy distribution matching learning (6), it can also be derived through the perspective of **stochastic optimal control (SOC)** as in Wang et al. (2025a); Zhu et al. (2025g), which we detail in App. B.1. In particular, we emphasize that the ELBO approximation of *sequence-level* log probabilities in matching KL divergence in (9) is equivalent to precisely matching *path-level* probabilities in the SOC framework, justifying the validity of such a heuristic approximation.

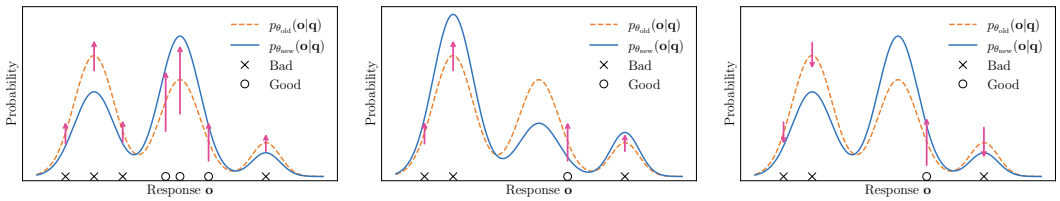
3.3 Effective Training with Negative Gradient Insertion

While theoretically, minimizing the WDCE loss (11) probably leads to convergence of the model sequence distribution to $p_*(\boldsymbol{o}|\boldsymbol{q})$, this could face practicality issues in real implementation due to the often limited number of rollouts generated per prompt. Ideally, we would want to promote the likelihood of "good" responses while decreasing those of "bad" responses. However, with WDCE, any response $\boldsymbol{o}$ will be associated with a positive weight $w(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})$ due to the softmax operation, which may lead to ineffective learning in the low-batch-size scenario.

We note that this issue does not arise when the batch size is sufficiently large for the following reason. When having a large batch of diverse responses that make up a good coverage of the sample space, despite having all positive weights, since the model cannot increase likelihood on all responses (as it is a probability distribution that sums up to 1), the "bad" responses will be automatically and implicitly penalized due to not having larger weights than the "good" responses.

When the batch size is small, the scenario is different as is illustrated in Fig. 3. In such a case, the model will tend to promote **both "good" and "bad" responses** due to the positive weights, and potentially penalize the likelihood of other unseen responses to maintain a valid probability distribution. This could be detrimental to achieving distribution matching, as these unseen responses may have high reward values and correspond to an undiscovered distribution mode.

To address this issue, we inject negative gradient (Ren & Sutherland, 2025; Deng et al., 2025) by designing a **weight baseline** and subtract it from the obtained weights to facilitate an effective



(a) Large batch size. (b) Small batch size. (c) Small batch size with baseline.

Figure 3: Demonstration of the effect of weight baseline. The orange and blue curves represent the probability $p_\theta(\boldsymbol{o}|\boldsymbol{q})$ before and after update, and the magenta arrows represent the weights. (a) When batch size is large, distribution mode coverage is good. Though bad responses have positive weights, the correct ones will have larger weights to force the distribution updates towards the right direction. (b) When batch size is small, some modes (e.g., the good one in the middle) may not be sampled. Without **weight baseline subtraction**, the dominant positive weights of the bad responses lead to wrong update directions. (c) With **weight baseline subtraction**, the bad responses will appropriately be penalized, leading to the desired update direction.

reinforcement on the good samples, i.e.,

$$w_{\text{real}}(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma}) = w(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma}) - w_{\text{base}}(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma}). \tag{12}$$

This approach resembles that adopted by PPO/GRPO. However, distinct from these methods, we rate responses based on the log weights $\ell(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})$, whose larger values indicate a better alignment with the target optimal distribution. As a consequence, we promote responses that are more likely to be sampled from p_* and penalize those that are less likely. Based on this perspective, we consider the following three methods for choosing $w_{\text{base}}(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})$.

Group weight baseline. When the dLLM policy is close to optimal, the original log weight $\ell(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})$ should behave approximately like constants for a group of different responses $\{\boldsymbol{o}^{(n)}\}_{1\le n\le N}$, leading to nearly uniform weight value for $\{w(\boldsymbol{o}^{(n)}|\boldsymbol{q};\boldsymbol{\sigma}^{(n)})\}_{1\le n\le N}$ after group softmax. We can thus choose the baseline as 1 to encourage convergence to this optimal situation:

$$w_{\text{base}}(\boldsymbol{o}^{(n)}|\boldsymbol{q};\boldsymbol{\sigma}^{(n)}) = 1,\ \forall n. \tag{13}$$

Individual weight baseline. We can also consider the individual weight value of each response. For samples with smaller log weights, a stronger penalization is more desirable. A natural, adaptive way of designing penalization strength is to use softmax over the log weights with *negative* reward: let $\ell_-(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma}) := -\frac{r(\boldsymbol{q},\boldsymbol{o})}{\alpha} + \log\frac{p_{\text{ref}}(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})}{p_v(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})}$, and define

$$w_{\text{base}}(\boldsymbol{o}^{(n)}|\boldsymbol{q};\boldsymbol{\sigma}^{(n)}) = \frac{N\exp(\ell_-(\boldsymbol{o}^{(n)}|\boldsymbol{q};\boldsymbol{\sigma}^{(n)}))}{\sum_k \exp(\ell_-(\boldsymbol{o}^{(k)}|\boldsymbol{q};\boldsymbol{\sigma}^{(k)}))},\ \forall n. \tag{14}$$

Note that this $w_{\text{base}}(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})$ now corresponds to a bad target distribution given by $p_{*-}(\boldsymbol{o}|\boldsymbol{q}) \propto_{\boldsymbol{o}} p_{\text{ref}}(\boldsymbol{o}|\boldsymbol{q})\mathrm{e}^{-r(\boldsymbol{q},\boldsymbol{o})/\alpha}$ which is tilted by the negative reward. The minus sign in the loss before $w_{\text{base}}(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})$ means we want to drive the dLLM policy away from this bad distribution.

Model weight baseline. Finally, we can determine whether to promote or penalize specific responses by comparing $w(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})$ with the importance weight under the current model policy $p_\theta(\boldsymbol{o}|\boldsymbol{q})$. This pushes the model further towards the optimal one $p_*(\boldsymbol{o}|\boldsymbol{q})$. Note that this does not incur additional computation overhead as we can estimate $\log p_{\bar\theta}(\boldsymbol{o}|\boldsymbol{q})$ using negative ELBO (2), which is already computed in the WDCE loss. Let $\ell_\theta(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma}) := \log\frac{p_{\bar\theta}(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})}{p_v(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})}$, and define

$$w_{\text{base}}(\boldsymbol{o}^{(n)}|\boldsymbol{q};\boldsymbol{\sigma}^{(n)}) = \frac{N\exp(\ell_\theta(\boldsymbol{o}^{(n)}|\boldsymbol{q};\boldsymbol{\sigma}^{(n)}))}{\sum_k \exp(\ell_\theta(\boldsymbol{o}^{(k)}|\boldsymbol{q};\boldsymbol{\sigma}^{(k)}))},\ \forall n. \tag{15}$$

We remark that the group weight and model weight baselines (13) and (15) can also be interpreted as an *approximate variance reduction*. See App. B.3 for discussion.

3.4 Weighted Direct Discriminative Optimization

To explore the full potential of the distribution matching framework in (6), we also investigate other choices for the potential $\mathcal{F}$ other than the cross-entropy. One particularly interesting objective is the

Table 1: Model performances on reasoning benchmarks. **best** and second best results are highlighted. DMPO consistently outperforms other baselines across different generation length.

Task	GSM8K			MATH500			Countdown			Sudoku		
Sequence Length	128	256	512	128	256	512	128	256	512	128	256	512
Dream-Instruct (7B)	56.63	73.39	76.65	31.00	36.60	36.40	22.66	28.52	27.34	14.45	16.41	11.77
LLaDA-Instruct (8B)	71.87	79.76	83.62	28.20	35.00	38.80	23.44	14.45	14.84	12.94	6.10	7.37
LLaDA-1.5 (8B)	73.09	80.97	84.38	26.80	33.80	40.00	26.17	16.41	23.83	15.19	13.04	8.98
d1-LLaDA	75.28	81.40	84.38	30.00	36.60	40.80	34.38	26.56	30.47	21.97	11.04	8.69
cGRPO-LLaDA	67.40	81.73	84.23	21.40	32.80	38.40	30.08	42.58	37.11	24.17	24.17	21.97
DMPO-LLaDA (Ours)	74.83	82.41	**85.22**	30.00	38.20	**42.80**	**67.19**	**80.86**	82.81	**32.76**	24.56	19.97
DMPO-LLaDA-SFT (Ours)	**80.06**	**84.00**	84.09	**31.80**	**40.00**	41.20	54.69	67.19	77.34	25.20	**25.73**	**23.78**
DMPO-LLaDA-1.5 (Ours)	77.56	82.71	84.61	30.20	36.60	41.00	59.77	79.30	**83.20**	25.34	24.51	23.34

following **direct discriminative optimization (DDO)** loss,

$$\mathcal{F}(p_\theta(\cdot|\boldsymbol{q}), p_*(\cdot|\boldsymbol{q})) = -\mathbb{E}_{p_*(\boldsymbol{o}|\boldsymbol{q})} \log \sigma \left(\log \frac{p_\theta(\boldsymbol{o}|\boldsymbol{q})}{p_v(\boldsymbol{o}|\boldsymbol{q})} \right) - \mathbb{E}_{p_v(\boldsymbol{o}|\boldsymbol{q})} \log \sigma \left(-\log \frac{p_\theta(\boldsymbol{o}|\boldsymbol{q})}{p_v(\boldsymbol{o}|\boldsymbol{q})} \right), \quad (16)$$

where $\sigma(t) = 1/(1+\mathrm{e}^{-t})$. The global optimum of (16) is also $p_*(\cdot|\boldsymbol{q})$, thus being a valid functional for distribution matching learning. For a more detailed justification, see App. B.4.

This is inspired by Zheng et al. (2025e), which proposes a GAN-like (Goodfellow et al., 2014) training loss for the SFT of vision models. One interesting trait of this objective is its natural incorporation of negative gradients for bad samples due to the GAN nature, as is shown in the analysis therein:

$$\nabla_\theta \mathcal{F}(p_\theta(\cdot|\boldsymbol{q}), p_*(\cdot|\boldsymbol{q})) = \sum_{\boldsymbol{o}} \sigma \left(-\log \frac{p_\theta(\boldsymbol{o}|\boldsymbol{q})}{p_v(\boldsymbol{o}|\boldsymbol{q})} \right) (p_\theta(\boldsymbol{o}|\boldsymbol{q}) - p_*(\boldsymbol{o}|\boldsymbol{q})) \nabla_\theta \log p_\theta(\boldsymbol{o}|\boldsymbol{q}).$$

From the expression, as the first term is always non-negative, and the middle term $p_\theta(\boldsymbol{o}|\boldsymbol{q}) - p_*(\boldsymbol{o}|\boldsymbol{q})$ applies a penalty for bad response $\boldsymbol{o}$, thus providing a gradient direction for increasing $p_\theta(\boldsymbol{o}|\boldsymbol{q})$. Leveraging this property, we adapt it for RL finetuning of dLLM and introduce the **weighted direct discriminative optimization (WDDO)** loss, again through importance sampling to represent $p_*(\boldsymbol{o}|\boldsymbol{q})$,

$$\mathcal{F}(p_\theta(\cdot|\boldsymbol{q}), p_*(\cdot|\boldsymbol{q})) = -\mathbb{E}_{\boldsymbol{\sigma}} \mathbb{E}_{p_v(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})} \left[w(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma}) \log \sigma \left(\log \frac{p_\theta(\boldsymbol{o}|\boldsymbol{q})}{p_v(\boldsymbol{o}|\boldsymbol{q})} \right) + \log \sigma \left(-\log \frac{p_\theta(\boldsymbol{o}|\boldsymbol{q})}{p_v(\boldsymbol{o}|\boldsymbol{q})} \right) \right],$$

where $w(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})$ is the importance weight defined in (10).

4 Experiments

Model and baselines. We apply DMPO to LLaDA-8B-Instruct (Nie et al., 2025b), a state-of-the-art open-sourced, native masked dLLM that has not been post-trained with RL techniques. To clearly demonstrate the potential of DMPO, we follow an R1-Zero-like training recipe (Guo et al., 2025a; Liu et al., 2025c) and directly apply DMPO to the LLaDA model without first performing SFT on curated datasets. We refer to the model obtained via this pipeline as **DMPO-LLaDA**. We benchmark our method against a series of top-performing dLLM base models of comparable model size, such as Dream-Instruct (7B, Ye et al. (2025)), LLaDA-Instruct (8B), and LLaDA-1.5 (8B, Zhu et al. (2025a)). Our main RL baseline is d1 (Zhao et al., 2025a), a state-of-the-art RL finetuning approach developed for dLLMs that combines both SFT and diffu-GRPO (an adapted version of GRPO). We also compare with cGRPO, which was used to fine-tune a Dream-based coding dLLM in Gong et al. (2025).

DMPO incentivizes superior reasoning capabilities. We report in Tab. 1 the performance of DMPO together with that of the base model LLaDA-Instruct (8B), LLaDA-1.5 (8B), the models obtained by d1 and cGRPO post-training strategies, and other pretrained dLLM models. DMPO consistently outperforms both the LLaDA-Instruct baseline and the d1/cGRPO models, achieving the best performance among the listed state-of-the-art dLLMs. Notably, DMPO achieves excellent gains over the LLaDA-Instruct baseline, with an accuracy improvement of an average of $+\mathbf{2.40}\%$ on GSM8K, $+\mathbf{3.00}\%$ on MATH500, $+\mathbf{59.38}\%$ on Countdown, $+\mathbf{16.96}\%$ on Sudoku. DMPO also demonstrates superior performance over d1, the current SOTA RL baseline for dLLM, especially on planning tasks, with an increase of $+\mathbf{46.48}\%$ on Countdown and $+\mathbf{11.86}\%$ on Sudoku. This underscores the overall effectiveness of DMPO for enhancing model reasoning capabilities.

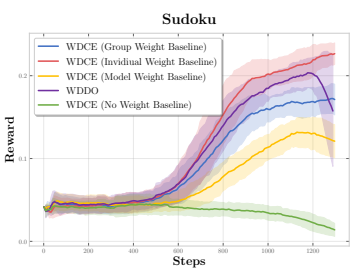


Figure 4: Effects of negative gradient insertion on Sudoku.

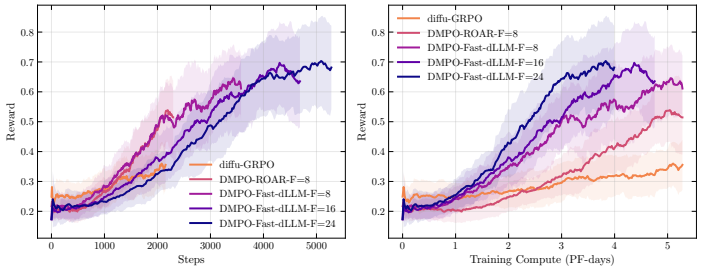


Figure 5: Comparison of training dynamics on Countdown. F is the frequency of sampling the buffer.

Weight baseline subtraction is crucial for small batch size training. We test the different choices presented for negative gradient insertion in Secs. 3.3 and 3.4 when training on the Sudoku dataset with a small batch size, and the result is visualized in Fig. 4. As shown by the curves, without weight baseline subtraction, the model does not improve as training progresses. All the proposed weight baselines in (13), (14), and (15) effectively increase the reward value during training. Weighted DDO achieves the fastest reward increase during the initial 1k steps but suffers from instability afterwards.

DMPO enables efficient and fast training. Due to its *off-policy* and *forward* nature, DMPO achieves considerable training acceleration compared with GRPO-type methods. In Fig. 5, we compare head-to-head the training dynamics of diffu-GRPO, DMPO with random-order autoregressive (ROAR) sampler, and DMPO with Fast-dLLM (an approximate KV-cache mechanism enabled, confidence-based heuristic sampler for dLLMs, from Wu et al. (2025)) on Countdown under the same amount of training compute (measured in PF-days, where 1 PF-day $= 8.64 \times 10^{19}$ floating point operations). Due to its off-policy nature, DMPO enables heavy reuse of each sampled buffer of rollouts and achieves a sample efficiency $\mathbf{2 \sim 3}\times$ that of diffu-GRPO. Regarding training-compute efficiency, as a forward-loss-based algorithm, DMPO enjoys a flexible choice of rollout sampler. With fast-dLLM, DMPO gains an acceleration of up to $\mathbf{8}\times$ per rollout sampling, and achieves the same level of reward as d1 with only $\mathbf{31}\%$ of the training budget (1.8 PF-days v.s. 5.78 PF-days). This empirical evidence emphasizes that DMPO is not only an effective algorithm but also highly sample- and compute-efficient.

DMPO exhibits stable training despite highly stale data. As is evident from Figs. 5 and 6, DMPO enjoys a largely stable dynamics despite using up to $24\times$ stale data (which means 24 parameter updates on the same batch of rollouts), without suffering from high variance of importance sampling. While this seems to contradict the general belief that on-policy learning beats off-policy learning for LLM RL, we argue that this is not the case because the off-policy in DMPO is inherently different from that used in diffu-GRPO or GRPO. Note that the latter considers the importance weight of the form $\frac{\pi_\theta}{\pi_{\text{old}}}$, which **inevitably diverges** as the number of parameter updates on θ increases. However, DMPO uses importance weight of the form $\frac{p_*}{p_{\theta^{\text{old}}}}$, which is independent of the current policy model π_θ and remains stable over a long horizon of training, enabling the use of a low buffer sampling frequency and highly stale rollouts without sacrificing performances. Moreover, DMPO adopts *sequence-level* importance sampling, in contrast to the *token-level* importance sampling used in diffu-GRPO or cGRPO, thereby providing an additional layer of stability. This advantage is also discussed in depth in Group Sequence Policy Optimization (GSPO, Zheng et al. (2025a)), which similarly considers *sequence-level* importance sampling.

Additional experimental results and discussion can be found in App. C.3.

5 Conclusion

This paper proposed Distribution Matching Policy Optimization (DMPO), a novel RL fine-tuning framework for dLLMs. DMPO leverages the unique characteristics of dLLMs via importance sampling and a WDCE loss, enabling off-policy training and forward-only computation that naturally exploit dLLM inference capabilities. The main limitation of this work is that we focus on a single pretrained dLLM and four elementary reasoning benchmarks, and DMPO's performance on other pretrained dLLMs and tasks in different domains remains unknown. Our work opens several promising directions for future research, such as investigating the distribution matching framework for other sequence models and studying the design of more effective weight baseline techniques.

REFERENCES

Anthropic. Introducing claude 4, May 2025. URL https://www.anthropic.com/news/claude-4. Accessed: 2025-09-01.

Arel. Arel's sudoku generator. https://www.ocf.berkeley.edu/~arel/sudoku/main.html, 2025. Accessed: 2025-07-01.

Marianne Arriola, Subham Sekhar Sahoo, Aaron Gokaslan, Zhihan Yang, Zhixuan Qi, Jiaqi Han, Justin T Chiu, and Volodymyr Kuleshov. Block diffusion: Interpolating between autoregressive and diffusion language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=tyEyYT267x.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=h7-XixPCAL.

Jinbin Bai, Tian Ye, Wei Chow, Enxin Song, Qing-Guo Chen, Xiangtai Li, Zhen Dong, Lei Zhu, and Shuicheng YAN. Meissonic: Revitalizing masked generative transformers for efficient high-resolution text-to-image synthesis. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=GJsuYHhAga.

Heli Ben-Hamu, Itai Gat, Daniel Severo, Niklas Nolte, and Brian Karrer. Accelerated sampling from masked diffusion models via entropy bounded unmasking. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=WBcBhT1NKO.

Emmanuel Bengio, Moksh Jain, Maksym Korablyov, Doina Precup, and Yoshua Bengio. Flow network based generative models for non-iterative diverse candidate generation. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 27381–27394. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/e614f646836aaed9f89ce58e837e2310-Paper.pdf.

Victor Besnier, Mickael Chen, David Hurych, Eduardo Valle, and Matthieu Cord. Halton scheduler for masked generative image transformer. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=RDVrlWAb7K.

Andrew Campbell, Joe Benton, Valentin De Bortoli, Thomas Rainforth, George Deligiannidis, and Arnaud Doucet. A continuous time framework for discrete denoising models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 28266–28279. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/b5b528767aa35f5b1a60fe0aaeca0563-Paper-Conference.pdf.

Andrew Campbell, Jason Yim, Regina Barzilay, Tom Rainforth, and Tommi Jaakkola. Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=kQwSbv0BR4.

Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. MaskGIT: Masked generative image transformer. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11315–11325, 2022. doi: 10.1109/CVPR52688.2022.01103.

Chen-Hao Chao, Wei-Fang Sun, Hanwen Liang, Chun-Yi Lee, and Rahul Krishnan. Beyond masked and unmasked: Discrete diffusion models via partial masking. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=nqpbbvEZwF.

Haoxuan Chen, Yinuo Ren, Martin Renqiang Min, Lexing Ying, and Zachary Izzo. Solving inverse problems via diffusion-based priors: An approximation-free ensemble sampling approach. *arXiv preprint arXiv:2506.03979*, 2025a.

Tong Chen, Yinuo Zhang, Sophia Tang, and Pranam Chatterjee. Multi-objective-guided discrete flow matching for controllable biological sequence design. *arXiv preprint arXiv:2505.07086*, 2025b.

Xiangxiang Chu, Hailang Huang, Xiao Zhang, Fei Wei, and Yong Wang. GPG: A simple and strong reinforcement learning baseline for model reasoning. *arXiv preprint arXiv:2504.02546*, 2025.

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

Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=mZn2Xyh9Ec.

DeepMind. Gemini diffusion. https://deepmind.google/models/gemini-diffusion/. Accessed: 2025-09-24.

Wenlong Deng, Yi Ren, Muchen Li, Danica J. Sutherland, Xiaoxiao Li, and Christos Thrampoulidis. On the effect of negative gradient in group relative deep reinforcement optimization. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=2K9QsDaqkM.

Justin Deschenaux and Caglar Gulcehre. Beyond autoregression: Fast LLMs via self-distillation through time. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=uZ5K4HeNwd.

Carles Domingo-Enrich, Jiequn Han, Brandon Amos, Joan Bruna, and Ricky T. Q. Chen. Stochastic optimal control matching. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 112459–112504. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/cc32ec39a5073f61d38c338d963df30d-Paper-Conference.pdf.

Carles Domingo-Enrich, Michal Drozdzal, Brian Karrer, and Ricky T. Q. Chen. Adjoint matching: Fine-tuning flow and diffusion generative models with memoryless stochastic optimal control. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=xQBRrtQM8u.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 12606–12633. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/esser24a.html.

Shansan Gong, Ruixiang Zhang, Huangjie Zheng, Jiatao Gu, Navdeep Jaitly, Lingpeng Kong, and Yizhe Zhang. DiffuCoder: Understanding and improving masked diffusion models for code generation. *arXiv preprint arXiv:2506.20639*, 2025.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL https://proceedings.neurips.cc/paper_files/paper/2014/file/f033ed80deb0234979a61f95710dbe25-Paper.pdf.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025a.

Wei Guo, Jaemoo Choi, Yuchen Zhu, Molei Tao, and Yongxin Chen. Proximal diffusion neural sampler. *arXiv preprint arXiv:2510.03824*, 2025b.

Thomas Hayes, Roshan Rao, Halil Akin, Nicholas J Sofroniew, Deniz Oktay, Zeming Lin, Robert Verkuil, Vincent Q Tran, Jonathan Deaton, Marius Wiggert, et al. Simulating 500 million years of evolution with a language model. *Science*, pp. 850–858, 2025. doi: 10.1126/science.ads0018.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In J. Vanschoren and S. Yeung (eds.), *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper_files/paper/2021/file/be83ab3ecd0db773eb2dc1b0a17836a1-Paper-round2.pdf.

Feng Hong, Geng Yu, Yushi Ye, Haicheng Huang, Huangjie Zheng, Ya Zhang, Yanfeng Wang, and Jiangchao Yao. Wide-in, narrow-out: Revokable decoding for efficient and effective DLLMs. *arXiv preprint arXiv:2507.18578*, 2025.

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. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Zhanqiu Hu, Jian Meng, Yash Akhauri, Mohamed S Abdelfattah, Jae-sun Seo, Zhiru Zhang, and Udit Gupta. Accelerating diffusion language model inference via efficient KV caching and guided diffusion. *arXiv preprint arXiv:2505.21467*, 2025.

Inception Labs, Samar Khanna, Siddhant Kharbanda, Shufan Li, Harshit Varma, Eric Wang, Sawyer Birnbaum, Ziyang Luo, Yanis Miraoui, Akash Palrecha, et al. Mercury: Ultra-fast language models based on diffusion. *arXiv preprint arXiv:2506.17298*, 2025.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. OpenAI o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

Amin Karimi Monsefi, Nikhil Bhendawade, Manuel Rafael Ciosici, Dominic Culver, Yizhe Zhang, and Irina Belousova. FS-DFM: Fast and accurate long text generation with few-step diffusion language models. *arXiv preprint arXiv:2509.20624*, 2025.

Jaeyeon Kim, Kulin Shah, Vasilis Kontonis, Sham M. Kakade, and Sitan Chen. Train for the worst, plan for the best: Understanding token ordering in masked diffusions. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=DjJmre5IkP.

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, et al. Kimi k1. 5: Scaling reinforcement learning with LLMs. *arXiv preprint arXiv:2501.12599*, 2025.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2023.

Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di ZHANG, and Wanli Ouyang. Flow-GRPO: Training flow matching models via online RL. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025a. URL https://openreview.net/forum?id=oCBKGw5HNf.

Zhiyuan Liu, Yicun Yang, Yaojie Zhang, Junjie Chen, Chang Zou, Qingyuan Wei, Shaobo Wang, and Linfeng Zhang. dLLM-Cache: Accelerating diffusion large language models with adaptive caching. *arXiv preprint arXiv:2506.06295*, 2025b.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding R1-Zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025c.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32819–32848. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/lou24a.html.

Xinyin Ma, Runpeng Yu, Gongfan Fang, and Xinchao Wang. dKV-Cache: The cache for diffusion language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=Gppo2JImHs.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393*, 2025.

Shen Nie, Fengqi Zhu, Chao Du, Tianyu Pang, Qian Liu, Guangtao Zeng, Min Lin, and Chongxuan Li. Scaling up masked diffusion models on text. 2025a. URL https://openreview.net/forum?id=WNvvwK0tut.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025b. URL https://openreview.net/forum?id=KnqiC0znVF.

Hunter Nisonoff, Junhao Xiong, Stephan Allenspach, and Jennifer Listgarten. Unlocking guidance for discrete state-space diffusion and flow models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=XsgHl54yO7.

Alexander Novikov, Ngân Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco JR Ruiz, Abbas Mehrabian, et al. AlphaEvolv: A coding agent for scientific and algorithmic discovery. *arXiv preprint arXiv:2506.13131*, 2025.

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. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=sMyXP8Tanm.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/b1efde53be364a73914f58805a001731-Paper-Conference.pdf.

Jiayi Pan, Junjie Zhang, Xingyao Wang, Lifan Yuan, Hao Peng, and Alane Suhr. Tinyzero. https://github.com/Jiayi-Pan/TinyZero, 2025. Accessed: 2025-01-24.

Jarrid Rector-Brooks, Mohsin Hasan, Zhangzhi Peng, Cheng-Hao Liu, Sarthak Mittal, Nouha Dziri, Michael M. Bronstein, Pranam Chatterjee, Alexander Tong, and Joey Bose. Steering masked discrete diffusion models via discrete denoising posterior prediction. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=Ombm8S40zN.

Yi Ren and Danica J. Sutherland. Learning dynamics of LLM finetuning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=tPNHOoZFl9.

Yinuo Ren, Haoxuan Chen, Grant M. Rotskoff, and Lexing Ying. How discrete and continuous diffusion meet: Comprehensive analysis of discrete diffusion models via a stochastic integral framework. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=6awxwQEI82.

Yinuo Ren, Haoxuan Chen, Yuchen Zhu, Wei Guo, Yongxin Chen, Grant M. Rotskoff, Molei Tao, and Lexing Ying. Fast solvers for discrete diffusion models: Theory and applications of high-order algorithms. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025b. URL https://openreview.net/forum?id=OuklL6Q3sO.

Yinuo Ren, Wenhao Gao, Lexing Ying, Grant M Rotskoff, and Jiequn Han. DriftLite: Lightweight drift control for inference-time scaling of diffusion models. *arXiv preprint arXiv:2509.21655*, 2025c.

Kevin Rojas, Ye He, Chieh-Hsin Lai, Yuta Takida, Yuki Mitsufuji, and Molei Tao. Theory-informed improvements to classifier-free guidance for discrete diffusion models. *arXiv preprint arXiv:2507.08965*, 2025a.

Kevin Rojas, Yuchen Zhu, Sichen Zhu, Felix X-F. Ye, and Molei Tao. Diffuse everything: Multimodal diffusion models on arbitrary state spaces. In *Forty-second International Conference on Machine Learning*, 2025b. URL https://openreview.net/forum?id=AjbiIcRt6q.

Subham Sekhar Sahoo, Marianne Arriola, Aaron Gokaslan, Edgar Mariano Marroquin, Alexander M Rush, Yair Schiff, Justin T Chiu, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=L4uaAR4ArM.

Subham Sekhar Sahoo, Zhihan Yang, Yash Akhauri, Johnna Liu, Deepansha Singh, Zhoujun Cheng, Zhengzhong Liu, Eric Xing, John Thickstun, and Arash Vahdat. Esoteric language models. *arXiv preprint arXiv:2506.01928*, 2025.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1889–1897, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/schulman15.html.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis Titsias. Simplified and generalized masked diffusion for discrete data. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=xcqSOfHt4g.

Qingyu Shi, Jinbin Bai, Zhuoran Zhao, Wenhao Chai, Kaidong Yu, Jianzong Wu, Shuangyong Song, Yunhai Tong, Xiangtai Li, Xuelong Li, et al. Muddit: Liberating generation beyond text-to-image with a unified discrete diffusion model. *arXiv preprint arXiv:2505.23606*, 2025.

Yuxuan Song, Zheng Zhang, Cheng Luo, Pengyang Gao, Fan Xia, Hao Luo, Zheng Li, Yuehang Yang, Hongli Yu, Xingwei Qu, et al. Seed diffusion: A large-scale diffusion language model with high-speed inference. *arXiv preprint arXiv:2508.02193*, 2025.

Sophia Tang, Yinuo Zhang, and Pranam Chatterjee. PepTune: De novo generation of therapeutic peptides with multi-objective-guided discrete diffusion. In *Forty-second International Conference on Machine Learning*, 2025a. URL https://openreview.net/forum?id=FQoy1Y1Hd8.

Sophia Tang, Yuchen Zhu, Molei Tao, and Pranam Chatterjee. TR2-D2: Tree search guided trajectory-aware fine-tuning for discrete diffusion. *arXiv preprint arXiv:2509.25171*, 2025b.

Xiaohang Tang, Rares Dolga, Sangwoong Yoon, and Ilija Bogunovic. wd1: Weighted policy optimization for reasoning in diffusion language models. *arXiv preprint arXiv:2507.08838*, 2025c.

Benigno Uria, Marc-Alexandre Côté, Karol Gregor, Iain Murray, and Hugo Larochelle. Neural autoregressive distribution estimation. *Journal of Machine Learning Research*, 17(205):1–37, 2016. URL http://jmlr.org/papers/v17/16-272.html.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. TRL: Transformer reinforcement learning. https://github.com/huggingface/trl, 2020.

Chenyu Wang, Masatoshi Uehara, Yichun He, Amy Wang, Avantika Lal, Tommi Jaakkola, Sergey Levine, Aviv Regev, Hanchen, and Tommaso Biancalani. Fine-tuning discrete diffusion models via reward optimization with applications to DNA and protein design. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=G328D1xt4W.

Yinjie Wang, Ling Yang, Bowen Li, Ye Tian, Ke Shen, and Mengdi Wang. Revolutionizing reinforcement learning framework for diffusion large language models. *arXiv preprint arXiv:2509.06949*, 2025b.

Lilian Weng. Reward hacking in reinforcement learning. *lilianweng.github.io*, Nov 2024. URL https://lilianweng.github.io/posts/2024-11-28-reward-hacking/.

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

Zeyue Xue, Jie Wu, Yu Gao, Fangyuan Kong, Lingting Zhu, Mengzhao Chen, Zhiheng Liu, Wei Liu, Qiushan Guo, Weilin Huang, et al. DanceGRPO: Unleashing GRPO on visual generation. *arXiv preprint arXiv:2505.07818*, 2025.

Ling Yang, Ye Tian, Bowen Li, Xinchen Zhang, Ke Shen, Yunhai Tong, and Mengdi Wang. MMaDA: Multimodal large diffusion language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=wczmXLuLGd.

Feng Yao, Liyuan Liu, Dinghuai Zhang, Chengyu Dong, Jingbo Shang, and Jianfeng Gao. Your efficient RL framework secretly brings you off-policy RL training, August 2025. URL https://fengyao.notion.site/off-policy-rl.

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.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, YuYue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, LingJun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Ru Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Yonghui Wu, and Mingxuan Wang. DAPO: An open-source LLM reinforcement learning system at scale. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=2a36EMSSTp.

Oussama Zekri and Nicolas Boullé. Fine-tuning discrete diffusion models with policy gradient methods. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=rXFzVRZsbt.

Ruixiang Zhang, Shuangfei Zhai, Jiatao Gu, Yizhe Zhang, Huangjie Zheng, Tianrong Chen, Miguel Ángel Bautista, Joshua M. Susskind, and Navdeep Jaitly. Flexible language modeling in continuous space with transformer-based autoregressive flows. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025a. URL https://openreview.net/forum?id=MR7Fn23hSE.

Ruixiang Zhang, Shuangfei Zhai, Yizhe Zhang, James Thornton, Zijing Ou, Joshua M. Susskind, and Navdeep Jaitly. Target concrete score matching: A holistic framework for discrete diffusion. In *Forty-second International Conference on Machine Learning*, 2025b. URL https://openreview.net/forum?id=ZMrdvSm7xi.

Siyan Zhao, Devaansh Gupta, Qinqing Zheng, and Aditya Grover. d1: Scaling reasoning in diffusion large language models via reinforcement learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025a. URL https://openreview.net/forum?id=7ZVRlBFuEv.

Yuzhong Zhao, Yue Liu, Junpeng Liu, Jingye Chen, Xun Wu, Yaru Hao, Tengchao Lv, Shaohan Huang, Lei Cui, Qixiang Ye, et al. Geometric-mean policy optimization. *arXiv preprint arXiv:2507.20673*, 2025b.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025a.

Haoyang Zheng, Xinyang Liu, Cindy Xiangrui Kong, Nan Jiang, Zheyuan Hu, Weijian Luo, Wei Deng, and Guang Lin. Ultra-fast language generation via discrete diffusion divergence instruct. *arXiv preprint arXiv:2509.25035*, 2025b.

Huangjie Zheng, Shansan Gong, Ruixiang Zhang, Tianrong Chen, Jiatao Gu, Mingyuan Zhou, Navdeep Jaitly, and Yizhe Zhang. Continuously augmented discrete diffusion model for categorical generative modeling. *arXiv preprint arXiv:2510.01329*, 2025c.

Kaiwen Zheng, Huayu Chen, Haotian Ye, Haoxiang Wang, Qinsheng Zhang, Kai Jiang, Hang Su, Stefano Ermon, Jun Zhu, and Ming-Yu Liu. DiffusionNFT: Online diffusion reinforcement with forward process. *arXiv preprint arXiv:2509.16117*, 2025d.

Kaiwen Zheng, Yongxin Chen, Huayu Chen, Guande He, Ming-Yu Liu, Jun Zhu, and Qinsheng Zhang. Direct discriminative optimization: Your likelihood-based visual generative model is secretly a GAN discriminator. In *Forty-second International Conference on Machine Learning*, 2025e. URL https://openreview.net/forum?id=OJ6WE7F8tK.

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. In *The Thirteenth International Conference on Learning Representations*, 2025f. URL https://openreview.net/forum?id=CTC7CmirNr.

Cai Zhou, Chenxiao Yang, Yi Hu, Chenyu Wang, Chubin Zhang, Muhan Zhang, Lester Mackey, Tommi Jaakkola, Stephen Bates, and Dinghuai Zhang. Coevolutionary continuous discrete diffusion: Make your diffusion language model a latent reasoner. *arXiv preprint arXiv:2510.03206*, 2025.

Fengqi Zhu, Rongzhen Wang, Shen Nie, Xiaolu Zhang, Chunwei Wu, Jun Hu, Jun Zhou, Jianfei Chen, Yankai Lin, Ji-Rong Wen, et al. LLaDA 1.5: Variance-reduced preference optimization for large language diffusion models. *arXiv preprint arXiv:2505.19223*, 2025a.

Sichen Zhu, Yuchen Zhu, Molei Tao, and Peng Qiu. Diffusion generative modeling for spatially resolved gene expression inference from histology images. In *The Thirteenth International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=FtjLUHyZAO.

Xuekai Zhu, Daixuan Cheng, Dinghuai Zhang, Hengli Li, Kaiyan Zhang, Che Jiang, Youbang Sun, Ermo Hua, Yuxin Zuo, Xingtai Lv, et al. FlowRL: Matching reward distributions for LLM reasoning. *arXiv preprint arXiv:2509.15207*, 2025c.

Yuanzhi Zhu, Xi Wang, Stéphane Lathuilière, and Vicky Kalogeiton. Di[M]O: Distilling masked diffusion models into one-step generator. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 18606–18618, October 2025d.

Yuanzhi Zhu, Xi Wang, Stéphane Lathuilière, and Vicky Kalogeiton. Soft-Di[M]O: Improving one-step discrete image generation with soft embeddings. *arXiv preprint arXiv:2509.22925*, 2025e.

Yuchen Zhu, Tianrong Chen, Lingkai Kong, Evangelos Theodorou, and Molei Tao. Trivialized momentum facilitates diffusion generative modeling on Lie groups. In *The Thirteenth International Conference on Learning Representations*, 2025f. URL https://openreview.net/forum?id=DTatjJTDl1.

Yuchen Zhu, Wei Guo, Jaemoo Choi, Guan-Horng Liu, Yongxin Chen, and Molei Tao. MDNS: Masked diffusion neural sampler via stochastic optimal control. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025g. URL https://openreview.net/forum?id=xIH95kXNR2.

Algorithm 1 Distribution Matching Policy Optimization (DMPO)

Require: Training dataset $\mathcal{D}$, number of prompts per batch B, number of rollouts per prompt N, frequency for sampling buffer F, model policy $\boldsymbol{\pi}_\theta$.

for $\texttt{step} = 0, 1, 2, ...$ **do**
  if $\texttt{step} \bmod F = 0$ **then** ▷ Prepare the buffer using the current policy, denoted $\boldsymbol{\pi}_v$.
    Sample B prompts $\{\boldsymbol{q}^{(i)}\}_{1 \le i \le B}$ from the dataset $\mathcal{D}$.
    for $1 \le i \le B$ (in parallel, with gradient computation disabled) **do**
      Sample N orders and generate N rollouts $\{\boldsymbol{o}^{(i,n)}\}_{1 \le n \le N}$ conditional on prompt $\boldsymbol{q}^{(i)}$.
      Evaluate reward and compute weights $w(\boldsymbol{o}^{(i,n)}|\boldsymbol{q}^{(i)}; \boldsymbol{\sigma}^{(i,n)})$ according to (10).
      Compute the weight baseline according to (13), (14), or (15), and obtain the real weights $w_{\text{real}}(\boldsymbol{o}^{(i,n)}|\boldsymbol{q}^{(i)}; \boldsymbol{\sigma}^{(i,n)})$ according to (12).
  For each $\boldsymbol{o}^{(i,n)}$, sample a mask assignment and obtain $\widetilde{\boldsymbol{o}}^{(i,n)}$.
  Feed all pairs of $(\boldsymbol{q}^{(i)}, \widetilde{\boldsymbol{o}}^{(i,n)})$ into $\boldsymbol{\pi}_\theta$ and compute the WDCE loss (11), then update θ.
return $\boldsymbol{\pi}_\theta$

A Related Work

Here, we focus on the literature for discrete diffusion models, as well as the methods for fine-tuning MDMs, dLLMs, and LLMs. We also briefly review several GRPO-style algorithms for domains outside of LLMs.

Discrete Diffusion Models. Diffusion models have been top-performing approaches for generating various data modalities (Zhu et al., 2025f; Esser et al., 2024; Zhu et al., 2025b; Rojas et al., 2025b; Zheng et al., 2025e; Chen et al., 2025a; Ren et al., 2025c). Discrete diffusion models (Austin et al., 2021; Campbell et al., 2022; Lou et al., 2024; Zhang et al., 2025b), a natural extension of diffusion models to finite state spaces, have emerged as powerful approaches for generating categorical, sequence data, with applications to text (Nie et al., 2025a;b; Ye et al., 2025), images (Chang et al., 2022; Bai et al., 2025; Shi et al., 2025), and biological sequences (Tang et al., 2025a; Chen et al., 2025b). One of the most effective variants of discrete diffusion models is masked diffusion models (MDM) (Sahoo et al., 2024; Ou et al., 2025; Shi et al., 2024) and its variants (Arriola et al., 2025; Sahoo et al., 2025; Chao et al., 2025). Recently, continuous latents have also been introduced into the modeling of discrete data (Zhang et al., 2025a; Zhou et al., 2025; Zheng et al., 2025c), resulting in improved and more appealing performance.

One particularly important line of development for discrete diffusion models centers on their inference techniques, with the aim of improving generation quality (Nisonoff et al., 2025; Rojas et al., 2025a; Kim et al., 2025; Besnier et al., 2025) and accelerating sampling speed (Ren et al., 2025b; Ben-Hamu et al., 2025; Wu et al., 2025; Hong et al., 2025). Besides these training-free approaches, learning-based approaches, such as few-step distillation, have also achieved decent success for discrete diffusion models (Deschenaux & Gulcehre, 2025; Karimi Monsefi et al., 2025; Zheng et al., 2025b; Zhu et al., 2025d;e). DMPO is closely tied to the literature on fast inference, as it can benefit from it by enjoying a similar training speed acceleration due to its forward nature.

Fine-tuning general discrete diffusion models. Earlier works on fine-tuning discrete diffusion models primarily focus on applications in biological and chemical domains, e.g., SVDD (**?**), DDPP (Rector-Brooks et al., 2025), DRAKES (Wang et al., 2025a), SEPO (Zekri & Boullé, 2025), and TR2-D2 (Tang et al., 2025b). Although these methods work well for their respective tasks, they are not directly applicable to dLLMs due to the unique challenges posed by the language domain, such as large model size, high dimensionality, and the need to maintain linguistic coherence and diversity.

Fine-tuning diffusion LLMs. Recently, numerous works have proposed RL algorithms for fine-tuning dLLMs, with most existing works being adaptations of the GRPO algorithm (Shao et al., 2024) for AR LLMs. For example, Zhao et al. (2025a) proposed Diffu-GRPO that estimates the per-token response log probabilities via masking all except the required response positions, and partially masking the prompt to get the model output, while their sequence log probability is estimated by mean-field approximation. Gong et al. (2025) introduced Coupled GRPO that modified the Diffu-GRPO method by not partially masking the prompt, and using complementary pairs of masks to

mask the same response that fully uses the model output, which we also adopt in our experiments. Yang et al. (2025) proposed UniGRPO, which involves a structured noise strategy and a modified log-likelihood approximation (both per-token and sequence). Concurrent with our work, TraceRL (Wang et al., 2025b) improves dLLM RL training by minimizing a training-inference gap. wd1 (Tang et al., 2025c) introduces additional regularization to the old policy, alongside the regularization applied to the reference model policy, which resembles the case discussed in App. B.2. We highlight that all these methods are GRPO-style algorithms that require estimating per-token response log probabilities, which are typically intractable and challenging for dLLMs. In contrast, our method offers the advantage of being a forward one, with greater efficiency and accuracy.

Fine-tuning LLMs. For fine-tuning LLMs, pre-LLM era works such as Trust Region Policy Optimization (TRPO, Schulman et al. (2015)) and Proximal Policy Optimization (PPO, Schulman et al. (2017)) have been widely used for RLHF (Ouyang et al., 2022). Since the huge success of GRPO (Shao et al., 2024) on DeepSeek-R1 (Guo et al., 2025a), there have been many follow-up works that improve GRPO in various ways, for instance: GRPO Done Right (Dr-GRPO, Liu et al. (2025c)), Decoupled clip and Dynamic sAmpling Policy Optimization (DAPO, Yu et al. (2025)), Group Policy Gradient (GPG, Chu et al. (2025)), Group Sequence Policy Optimization (GSPO, Zheng et al. (2025a)), Geometric-Mean Policy Optimization (GMPO, Zhao et al. (2025b)), etc.

Apart from the aforementioned policy gradient-based methods, GFlowNet (Bengio et al., 2021) has also been applied to finetuning LLMs, with successful applications seen in Kimi 1.5 (Kimi Team et al., 2025) and FlowRL (Zhu et al., 2025c). Notably, concurrent with our work, FlowRL shares the same high-level goal as our DMPO, targeting also policy distribution matching rather than merely reward maximization for AR-LLMs. However, distinct from DMPO, FlowRL derives its objectives from reverse KL and utilizes GFlowNet objectives. In contrast, our approach considers forward KL, which is known to be mass-covering, and implements it using importance sampling and weighted denoising cross-entropy.

GRPO-style algorithms for fine-tuning diffusion and flow-based models. GRPO-type algorithms have also been adapted to diffusion and flow-based models, such as flow-GRPO (Liu et al., 2025a) and DanceGRPO (Xue et al., 2025). Aside from that, there are also SOC-based fine-tuning algorithms for diffusion models, such as adjoint matching (Domingo-Enrich et al., 2025), with which our work shares similarity at a high level. Concurrent with our work, DiffusionNFT (Zheng et al., 2025d) has been proposed to finetune continuous diffusion models for text-to-image generation tasks. While formulated in drastically different ways, DiffusionNFT shares a similarity with our DMPO in that it is also an algorithm that primarily depends on model forward passes rather than backward trajectories.

B Theory of Distribution Matching Policy Optimization

B.1 Distribution Matching Policy Optimization from the Stochastic Optimal Control Perspective

This section aims at providing an alternative derivation of DMPO from the perspective of stochastic optimal control (SOC), which is inspired by DRAKES (Wang et al., 2025a) and MDNS (Zhu et al., 2025g). We will first introduce the necessary background on continuous-time Markov chains (CTMCs), then show how MDM sampling can be viewed as a CTMC. Finally, we derive the DMPO framework from the SOC perspective.

Introduction to Continuous-time Markov Chains. To derive the SOC framework for fine-tuning, we view the sampling of an MDM as a time-indexed stochastic process, and the proper mathematical tool is the **continuous-time Markov chain (CTMC)**. A CTMC $X = (X_t)_{t\in[0,1]}$ is a stochastic process taking value in a discrete state space $\mathcal{X}$. Its law is characterized by the **rate matrix** $Q = (Q_t)_{t\in[0,1]}$, defined as

$$Q_t(x, y) = \lim_{\Delta t\to 0} \frac{\Pr(X_{t+\Delta t} = y|X_t = x) - 1_{x=y}}{\Delta t}, \ \forall x, y \in \mathcal{X}. \tag{17}$$

By definition, the off-diagonal entries of Q_t are non-negative, and each row sums to zero.

The **path** of X, i.e., $t \mapsto X_t(\omega)$, is piecewise constant with discontinuous jumps, and one typically assumes that the path is right continuous with left limits. The **path measure** a CTMC X is a probability measure on the space of paths defined as $\mathbb{P}^X(A) := \Pr(X \in A)$, which is the distribution of X. The following lemma shows how to compute the **Radon-Nikodým (RN) derivative** between two path measures driven by CTMCs with different rate matrices and initial distributions:

Lemma 1. *Given two CTMCs with rate matrices Q^1, Q^2 and initial distributions μ_1, μ_2 on $\mathcal{X}$, let $\mathbb{P}^1, \mathbb{P}^2$ be the associated path measures. Then, for any path $\xi = (\xi_t)_{t\in[0,1]}$,*

$$\log \frac{\mathrm{d}\mathbb{P}^1}{\mathrm{d}\mathbb{P}^2}(\xi) = \log \frac{\mathrm{d}\mu_1}{\mathrm{d}\mu_2}(\xi_0) + \sum_{t:\xi_{t-}\neq\xi_t} \log \frac{Q^1_t(\xi_{t-},\xi_t)}{Q^2_t(\xi_{t-},\xi_t)} + \int_0^1 (Q^1_t(\xi_t,\xi_t) - Q^2_t(\xi_t,\xi_t))\mathrm{d}t. \quad (18)$$

For the proof, see Campbell et al. (2024, App. C.1), Ren et al. (2025a, Thm. 3.3), or Zhu et al. (2025g, Lem. 1). An intuitive interpretation of (18) is to view the RN derivative as the limit of density ratios between finite-dimensional joint distributions, and approximate the transition probability by (17).

Masked Diffusion Models as Continuous-Time Markov Chains. We will now delve into the CTMC formulation of sampling from an MDM. To avoid notational clutter, we use superscript to denote the position index, and subscript to denote the time index (e.g., $\boldsymbol{\xi}_t = (\xi_t^1, ..., \xi_t^D)$). We present the theory only in the case of *unconditional generation* with sequence length D for simplicity of notations, but it can be easily easily generalized to the case of conditional generation of $\boldsymbol{o}$ given a prompt $\boldsymbol{q}$.

As shown in Ou et al. (2025), by introducing a noise schedule $\gamma(t) = \frac{1}{1-t}$, [2] the random order autoregressive sampling of an MDM $\boldsymbol{\pi}_\theta$ can be viewed as a CTMC with the rate matrix $Q^\theta = (Q^\theta_t)_{t\in[0,1]}$ such that for $\boldsymbol{x} \neq \boldsymbol{y} \in \overline{\mathcal{V}}^D$,

$$Q^\theta_t(\boldsymbol{x}, \boldsymbol{y}) = \gamma(t)\boldsymbol{\pi}_\theta(\boldsymbol{x})_{d,n}, \text{ if } \boldsymbol{x}^d = \mathsf{M} \text{ and } \boldsymbol{y} = \boldsymbol{x}^{d\leftarrow n},$$

and 0 if otherwise, where $\boldsymbol{x}^{d\leftarrow n}$ means the sequence obtained by replacing the d-th position of $\boldsymbol{x}$ by n. The diagonal terms of Q^θ_t can be computed as

$$\begin{aligned} Q^\theta_t(\boldsymbol{x},\boldsymbol{x}) &= -\sum_{\boldsymbol{y}\neq\boldsymbol{x}} Q^\theta_t(\boldsymbol{x},\boldsymbol{y}) = -\sum_{d:\boldsymbol{x}^d=\mathsf{M}}\sum_n Q^\theta_t(\boldsymbol{x},\boldsymbol{x}^{d\leftarrow n}) \\ &= -\sum_{d:\boldsymbol{x}^d=\mathsf{M}}\sum_n \gamma(t)\boldsymbol{\pi}_\theta(\boldsymbol{x})_{d,n} = -\gamma(t)\cdot|\{d:\boldsymbol{x}^d=\mathsf{M}\}|. \end{aligned} \quad (19)$$

Therefore, if $\mathbb{P}^\theta$, $\mathbb{P}^{\theta'}$ are the path measures of the sampling processes of two MDMs parameterized by θ and θ', respectively, then by (18), assuming that the jump from $\boldsymbol{\xi}_{t-}$ to $\boldsymbol{\xi}_t$ is at the $d(t)$-th position, we have

$$\log \frac{\mathrm{d}\mathbb{P}^{\theta'}}{\mathrm{d}\mathbb{P}^{\theta}}(\boldsymbol{\xi}) = \sum_{t:\boldsymbol{\xi}_{t-}\neq\boldsymbol{\xi}_t} \log \frac{\boldsymbol{\pi}_{\theta'}(\boldsymbol{\xi}_{t-})_{d(t),\xi_t^{d(t)}}}{\boldsymbol{\pi}_{\theta}(\boldsymbol{\xi}_{t-})_{d(t),\xi_t^{d(t)}}},\ \forall \boldsymbol{\xi} = (\boldsymbol{\xi}_t)_{t\in[0,1]}, \quad (20)$$

as the first term in (18) is always zero (both initial distributions are the point mass on the fully masked sequence), and the diagonal terms in the third term cancel out due to (19).

Moreover, as proved in Ou et al. (2025), the training of an MDM $\boldsymbol{\pi}_\theta$ given i.i.d. samples from the target distribution p_{data} can be interpreted as minimizing the KL divergence between the target path measure $\mathbb{P}^*$ and the parameterized path measure $\mathbb{P}^\theta$, where $\mathbb{P}^*$ is defined as the path measure of the CTMC with rate matrix $Q^*_t(\boldsymbol{x}, \boldsymbol{x}^{d\leftarrow n}) = \gamma(t)\Pr_{\boldsymbol{X}\sim p_{\mathrm{data}}}(X^d = n|\boldsymbol{X}^{\mathrm{UM}} = \boldsymbol{x}^{\mathrm{UM}})1_{x^d=\mathsf{M}}$, i.e., with the ground-truth conditional distribution. Moreover, one can derive

$$\mathrm{KL}(\mathbb{P}^*\|\mathbb{P}^\theta) = \mathbb{E}_{p_{\mathrm{data}}(\boldsymbol{x})}\,\mathbb{E}_{m\sim\mathrm{Unif}\{1,\ldots,D\}}\left[\frac{D}{m}\,\mathbb{E}_{\mu_m(\widetilde{\boldsymbol{x}}|\boldsymbol{x})}\sum_{d:\widetilde{\boldsymbol{x}}^d=\mathsf{M}} -\log\boldsymbol{\pi}_\theta(\widetilde{\boldsymbol{x}})_{d,x^d}\right] + \mathrm{const},$$

[2] The choice of noise schedule is essentially not important for MDM. In fact, γ can be any positive function with $\int_0^1 \gamma(t)\mathrm{d}t = \infty$. Here, we follow the convention in most of the literature on MDM and choose this specific γ such that the conditional distribution of $\boldsymbol{\xi}_t \in \overline{\mathcal{V}}^D$ given $\boldsymbol{\xi}_1 \in \mathcal{V}^D$ is obtained by independently masking each position in $\boldsymbol{\xi}_1$ with probability $1-t$.

where const does not depend on θ, and $\mu_m(\cdot|\boldsymbol{x})$ means to sample a uniformly random subset of $\{1, ..., D\}$ of size m and mask the corresponding positions in $\boldsymbol{x}$. Note that this is exactly the denoising cross-entropy loss $\mathbb{E}_{p_{\text{data}}(\boldsymbol{x})} \mathcal{L}_\theta(\boldsymbol{x})$ as presented in (2). In other words, minimizing the KL divergence between sequence-level probabilities $p_*(\boldsymbol{\xi}_1)$ and $p_\theta(\boldsymbol{\xi}_1) \approx \mathrm{e}^{-\mathcal{L}_\theta(\boldsymbol{\xi}_1)}$ in (11) can be interpreted as precisely minimizing the KL divergence between path-level probabilities $\mathbb{P}^*(\boldsymbol{\xi})$ and $\mathbb{P}^\theta(\boldsymbol{\xi})$.

Fine-tuning MDMs as a Stochastic Optimal Control Problem on Path Measures. The task of fine-tuning a pretrained MDM can be viewed as a stochastic optimal control (SOC) problem on the space of path measures: given a pretrained MDM $\boldsymbol{\pi}_{\text{ref}}$ which generates a distribution p_{ref}, we define its induced **reference path measure** as $\mathbb{P}^{\text{ref}}$, with rate matrix $Q_t^{\text{ref}}(\boldsymbol{x}, \boldsymbol{x}^{d\leftarrow n}) = \gamma(t)\boldsymbol{\pi}_{\text{ref}}(\boldsymbol{x})_{d,n} 1_{x^d=\mathsf{M}}$, and has terminal distribution $\mathbb{P}_1^{\text{ref}} = p_{\text{ref}}$. We aim at finding a target rate matrix Q^* such that the associated target path measure $\mathbb{P}^*$ has a terminal distribution p_* defined in the following way of tilting by reward:

$$p_*(\boldsymbol{x}) = \frac{1}{Z} p_{\text{ref}}(\boldsymbol{x}) \mathrm{e}^{r(\boldsymbol{x})/\alpha},\ \boldsymbol{x} \in \mathcal{V}^D, \qquad \text{where } Z = \sum_{\boldsymbol{x}} p_{\text{ref}}(\boldsymbol{x}) \mathrm{e}^{r(\boldsymbol{x})/\alpha}.$$

This can be achieved by defining the target path measure $\mathbb{P}^*$ as

$$\mathbb{P}^*(\boldsymbol{\xi}) = \mathbb{P}^{\text{ref}}(\boldsymbol{\xi}_{[0,1)}|\boldsymbol{\xi}_1) p_*(\boldsymbol{\xi}_1) = \mathbb{P}^{\text{ref}}(\boldsymbol{\xi}) \frac{p_*(\boldsymbol{\xi}_1)}{p_{\text{ref}}(\boldsymbol{\xi}_1)} = \frac{1}{Z} \mathbb{P}^{\text{ref}}(\boldsymbol{\xi}) \mathrm{e}^{r(\boldsymbol{\xi}_1)/\alpha},\ \forall \boldsymbol{\xi} = (\boldsymbol{\xi}_t)_{t\in[0,1]}. \quad (21)$$

We use a network $\boldsymbol{\pi}_\theta$ to parameterize the new rate matrix, initialized at $\boldsymbol{\pi}_{\text{ref}}$. Given a current path measure $\mathbb{P}^\theta$ induced by a CTMC with rate matrix $Q_t^\theta(\boldsymbol{x}, \boldsymbol{x}^{d\leftarrow n}) = \gamma(t)\boldsymbol{\pi}_\theta(\boldsymbol{x})_{d,n} 1_{x^d=\mathsf{M}}$, we can first derive the RN derivative between the path measures by (20):

$$\begin{aligned}
\log \frac{\mathrm{d}\mathbb{P}^*}{\mathrm{d}\mathbb{P}^\theta}(\boldsymbol{\xi}) &= \log \frac{\mathrm{d}\mathbb{P}^*}{\mathrm{d}\mathbb{P}^{\text{ref}}}(\boldsymbol{\xi}) + \log \frac{\mathrm{d}\mathbb{P}^0}{\mathrm{d}\mathbb{P}^\theta}(\boldsymbol{\xi}) \\
&= \frac{r(\boldsymbol{\xi}_1)}{\alpha} - \log Z + \sum_{t:\boldsymbol{\xi}_{t-}\neq\boldsymbol{\xi}_t} \log \frac{Q_t^0}{Q_t^\theta}(\boldsymbol{\xi}_{t-}, \boldsymbol{\xi}_t) + \int_0^1 \sum_{y\neq\boldsymbol{\xi}_t} (Q_t^\theta - Q_t^0)(\boldsymbol{\xi}_t, y)\mathrm{d}t \\
&= \frac{r(\boldsymbol{\xi}_1)}{\alpha} + \sum_{t:\boldsymbol{\xi}_{t-}\neq\boldsymbol{\xi}_t} \log \frac{\boldsymbol{\pi}_{\text{ref}}(\boldsymbol{\xi}_{t-})_{d(t),\xi_t^{d(t)}}}{\boldsymbol{\pi}_\theta(\boldsymbol{\xi}_{t-})_{d(t),\xi_t^{d(t)}}} - \log Z =: W^\theta(\boldsymbol{\xi}) - \log Z, \qquad (22)
\end{aligned}$$

where we assume that the jump from $\boldsymbol{\xi}_{t-}$ to $\boldsymbol{\xi}_t$ is at the $d(t)$-th position. The idea of the weighted denoising cross-entropy (WDCE) loss is essentially to treat i.i.d. samples from the current policy $\mathbb{P}^\theta$ as weighted samples from $\mathbb{P}^*$, and minimizing the following loss:

$$\begin{aligned}
\mathrm{KL}(\mathbb{P}^*\|\mathbb{P}^\theta) + \text{const} &= \mathbb{E}_{p_*(\boldsymbol{x})} \mathcal{L}_\theta(\boldsymbol{x}) = \mathbb{E}_{\mathbb{P}^*(\boldsymbol{\xi})} \mathcal{L}_\theta(\boldsymbol{\xi}_1) \\
&= \mathbb{E}_{\mathbb{P}^v(\boldsymbol{\xi})} \frac{\mathrm{d}\mathbb{P}^*}{\mathrm{d}\mathbb{P}^v}(\boldsymbol{\xi}) \mathcal{L}_\theta(\boldsymbol{\xi}_1) = \mathbb{E}_{\mathbb{P}^v(\boldsymbol{\xi})} \frac{1}{Z} \mathrm{e}^{W^v(\boldsymbol{\xi})} \mathcal{L}_\theta(\boldsymbol{\xi}_1),
\end{aligned}$$

where $\mathbb{P}^v$ is the path measure induced by a CTMC with rate matrix Q^v where the network is parameterized by v (e.g., the old parameters θ_{old}), whose parameters do not involve gradient calculation. For instance, we can set $v = \theta_{\text{old}}$. Note that $Z = \mathbb{E}_{\mathbb{P}^v(\boldsymbol{\xi})} \mathrm{e}^{W^v(\boldsymbol{\xi})}$, which, if estimated via samples, is equivalent to doing softmax normalization on the logits $W^v(\boldsymbol{\xi})$ in the batch. Comparing with the WDCE loss (11) presented in Sec. 3.2, we conclude that they are essentially the same.

B.2 Generalizing WDCE to Zero Temperature with Proximal Descent

Recall that our target distribution is (5), which is under a temperature $\alpha > 0$. We propose to generalize the WDCE loss (11) to incorporate the limiting case $\alpha \to 0$ from the viewpoint of **proximal descent** (Guo et al., 2025b).

The reward maximization problem (4) provides a variational characterization of the target distribution $p_*(\boldsymbol{o}|\boldsymbol{q})$. Suppose now we have a dLLM policy $\boldsymbol{\pi}_{\theta_{\text{old}}}(\boldsymbol{o}|\boldsymbol{q})$ that outputs a distribution $p_{\theta_{\text{old}}}(\boldsymbol{o}|\boldsymbol{q})$. We define the next target distribution $p_{\text{tar}}(\boldsymbol{o}|\boldsymbol{q})$ as

$$p_{\text{tar}}(\boldsymbol{o}|\boldsymbol{q}) = \underset{p_\theta(\boldsymbol{o}|\boldsymbol{q})}{\operatorname{argmax}} \left\{ \mathbb{E}_{p_\theta(\boldsymbol{o}|\boldsymbol{q})}[r(\boldsymbol{q}, \boldsymbol{o})] - \alpha\, \mathrm{KL}(p_\theta(\cdot|\boldsymbol{q})\|p_{\text{ref}}(\cdot|\boldsymbol{q})) - \frac{1}{\eta'} \mathrm{KL}(p_\theta(\cdot|\boldsymbol{q})\|p_{\theta_{\text{old}}}(\cdot|\boldsymbol{q})) \right\}, \quad (23)$$

where $\eta' > 0$ is the step size. Let $\eta = \frac{\eta'}{1+\eta'\alpha} \in \left(0, \frac{1}{\alpha}\right)$. It is easy to see that the solution is given by

$$\begin{aligned} p_{\mathrm{tar}}(\boldsymbol{o}|\boldsymbol{q}) &\propto_{\boldsymbol{o}} p_{\theta_{\mathrm{old}}}(\boldsymbol{o}|\boldsymbol{q})^{1-\eta\alpha} p_{\mathrm{ref}}(\boldsymbol{o}|\boldsymbol{q})^{\eta\alpha} \mathrm{e}^{\eta r(\boldsymbol{q},\boldsymbol{o})}, \\ &\propto_{\boldsymbol{o}} p_{\theta_{\mathrm{old}}}(\boldsymbol{o}|\boldsymbol{q})^{1-\eta\alpha} p_*(\boldsymbol{o}|\boldsymbol{q})^{\eta\alpha}. \end{aligned} \tag{24}$$

In fact, the term inside the brackets in (23) is $-\frac{1}{\eta}\,\mathrm{KL}(p_\theta(\cdot|\boldsymbol{q})\|p_{\mathrm{tar}}(\cdot|\boldsymbol{q})) + \mathrm{const}$. This means the next target distribution is a geometric interpolation between the current model distribution $p_{\theta_{\mathrm{old}}}$ and the optimal distribution p_*, with $\eta > 0$ being a step size parameter. (24) is well-defined even when $\alpha = 0$, although in this case, the target distribution concentrates on the set of maximizers of $r(\boldsymbol{q}, \boldsymbol{o})$ (e.g., all correct question-response pairs) without regularization from the base model $p_{\mathrm{ref}}(\boldsymbol{o}|\boldsymbol{q})$.

For $\alpha = 0$, $p_{\mathrm{tar}}(\boldsymbol{o}|\boldsymbol{q}) \propto_{\boldsymbol{o}} p_{\theta_{\mathrm{old}}}(\boldsymbol{o}|\boldsymbol{q}) \mathrm{e}^{\eta r(\boldsymbol{q},\boldsymbol{o})}$. We can similarly solve the distribution matching problem via the WDCE loss:

$$\begin{aligned} \mathrm{KL}(p_{\mathrm{tar}}(\cdot|\boldsymbol{q})\|p_\theta(\cdot|\boldsymbol{q})) &= \mathbb{E}_{p_{\mathrm{tar}}(\boldsymbol{o}|\boldsymbol{q})}[-\log p_\theta(\boldsymbol{o}|\boldsymbol{q})] + \mathrm{const} \\ &= \mathbb{E}_{p_v(\boldsymbol{o}|\boldsymbol{q})} \underbrace{\frac{p_{\mathrm{tar}}(\boldsymbol{o}|\boldsymbol{q})}{p_v(\boldsymbol{o}|\boldsymbol{q})}}_{=:w(\boldsymbol{o}|\boldsymbol{q})} [-\log p_\theta(\boldsymbol{o}|\boldsymbol{q})] + \mathrm{const} \\ &\le \mathbb{E}_{p_v(\boldsymbol{o}|\boldsymbol{q})}\, w(\boldsymbol{o}|\boldsymbol{q}) \mathcal{L}_\theta(\boldsymbol{o}|\boldsymbol{q}) + \mathrm{const}, \end{aligned}$$

where the importance weight $w(\boldsymbol{o}|\boldsymbol{q}) \propto_{\boldsymbol{o}} \exp\left(\eta r(\boldsymbol{q}, \boldsymbol{o}) + \log \frac{p_{\theta_{\mathrm{old}}}(\boldsymbol{o}|\boldsymbol{q})}{p_{\theta_v}(\boldsymbol{o}|\boldsymbol{q})}\right)$. For $v \leftarrow \theta_{\mathrm{old}}$, the weight simplifies to the softmax of $\eta r(\boldsymbol{q}, \boldsymbol{o})$ over all responses for the same prompt $\boldsymbol{q}$. The weight baseline subtraction tricks also apply here.

We remark that when picking $\alpha = 0$, through the proximal gradient descent formulation, DMPO becomes completely *forward-only*, as it eliminates the need for estimating the sequence log probability ratio of the form $\log \frac{p_{\mathrm{ref}}(\boldsymbol{o}|\boldsymbol{q})}{p_v(\boldsymbol{o}|\boldsymbol{q})}$, making it the best option to incorporate fast dLLM inference techniques for RL training speed-up. However, in this case, we can no longer guarantee the diversity in the target optimal distribution, and thus, we save this direction for future investigation.

B.3 Insights for Weight Baselines: Approximate Variance Reduction

We first recall a classical equality in statistics regarding the **score function**: if $p_\theta(x)$ is a probability density or probability mass function parameterized by a continuous parameter θ, then under certain weak regularity conditions, we have $\mathbb{E}_{p_\theta(x)} \nabla_\theta \log p_\theta(x) = 0$.

Therefore,

$$\begin{aligned} 0 &= \mathbb{E}_{p_\theta(\boldsymbol{o}|\boldsymbol{q})} \nabla_\theta \log p_\theta(\boldsymbol{o}|\boldsymbol{q}) = \nabla_\theta\, \mathbb{E}_{p_{\bar\theta}(\boldsymbol{o}|\boldsymbol{q})} \log p_\theta(\boldsymbol{o}|\boldsymbol{q}) \\ &= \nabla_\theta\, \mathbb{E}_{\boldsymbol{\sigma}}\, \mathbb{E}_{p_{\bar\theta}(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})} \log p_\theta(\boldsymbol{o}|\boldsymbol{q}) \\ &= \nabla_\theta\, \mathbb{E}_{\boldsymbol{\sigma}}\, \mathbb{E}_{p_v(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})} \frac{p_{\bar\theta}(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})}{p_v(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})} \log p_\theta(\boldsymbol{o}|\boldsymbol{q}). \end{aligned}$$

Combined with (9), we can see that subtracting $\frac{p_{\bar\theta}(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})}{p_v(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})}$ from the weight does not change the gradient of the CE loss, i.e.,

$$\nabla_\theta\, \mathrm{KL}(p_*(\cdot|q)\|p_\theta(\cdot|q)) = \nabla_\theta\, \mathbb{E}_{\boldsymbol{\sigma}}\, \mathbb{E}_{p_v(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})} \left(\frac{p_*(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})}{p_v(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})} - \lambda \frac{p_{\bar\theta}(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})}{p_v(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})}\right) [-\log p_\theta(\boldsymbol{o}|\boldsymbol{q})],\ \forall \lambda \in \mathbb{R}.$$

Theoretically, there is an optimal choice of λ that minimizes the variance. The natural choice of $\lambda = 1$ means implicitly matching the probability $p_\theta(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})$ to fit $p_*(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})$, which corresponds to our model weight baseline (15). When the frequency for sampling buffer F is small, we can assume $p_\theta(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})$ does not deviate too much from $p_v(\boldsymbol{o}|\boldsymbol{q};\boldsymbol{\sigma})$, thus this ratio should be close to 1, which corresponds to our group weight baseline (13). Finally, as we actually use the negative ELBO $\mathcal{L}_\theta(\boldsymbol{o}|\boldsymbol{q})$ instead of $-\log p_\theta(\boldsymbol{o}|\boldsymbol{q})$ in computing the loss, the variance reduction only holds *approximately*.

B.4 Proofs for the Weighted Direct Discriminative Optimization Objective

For notational simplicity, we ignore the conditional dependence on $\boldsymbol{q}$. Write

$$\mathcal{F}(p_\theta) = -\,\mathbb{E}_{p_*} \log \frac{p_\theta}{p_\theta + p_v} - \mathbb{E}_{p_v} \log \frac{p_v}{p_\theta + p_v}.$$

For any fixed $\boldsymbol{o}$, consider the function

$$p_\theta(\boldsymbol{o}) \mapsto -p_*(\boldsymbol{o}) \log \frac{p_\theta(\boldsymbol{o})}{p_\theta(\boldsymbol{o}) + p_v(\boldsymbol{o})} - p_v(\boldsymbol{o}) \log \frac{p_v(\boldsymbol{o})}{p_\theta(\boldsymbol{o}) + p_v(\boldsymbol{o})}.$$

The derivative with respect to $p_\theta(\boldsymbol{o})$ is $-\frac{p_*(\boldsymbol{o})}{p_\theta(\boldsymbol{o})} + \frac{p_*(\boldsymbol{o})+p_v(\boldsymbol{o})}{p_\theta(\boldsymbol{o})+p_v(\boldsymbol{o})}$, which is > 0 if $p_\theta(\boldsymbol{o}) > p_*(\boldsymbol{o})$ and < 0 if $p_\theta(\boldsymbol{o}) < p_*(\boldsymbol{o})$. Therefore, this function is minimized at $p_\theta(\boldsymbol{o}) \leftarrow p_*(\boldsymbol{o})$, which completes the proof.

C Details of Experiments and Further Results

C.1 Introduction of Datasets and Rewards used

Experimental setups. We perform experiments on 4 different reasoning benchmarks: GSM8k, MATH500, Sudoku, and Countdown. For all pretrained dLLM models, we evaluate the latest available checkpoints for each task. For d1 and cGRPO, we reproduce their results exactly following the provided guidelines. To ensure a fair comparison, we train DMPO-LLaDA on the same datasets as d1 for each task with rollouts generated using a fixed sequence length of 256. All evaluations are conducted with zero-shot prompting using three different generation lengths: 128, 256, and 512, following a similar practice as in Zhao et al. (2025a). For a self-contained presentation, we list the datasets and the rewards below.

GSM8K. GSM8k (Cobbe et al., 2021) is a mathematical reasoning dataset featuring multi-step grade school math problems. We conduct fine-tuning on the train split and evaluate on the test split.[3]

The reward is decomposed as follows:

1. *XML Structure Reward*: $+0.125$ for each correctly placed opening and closing tag (`<reasoning>`, `</reasoning>`, `<answer>`, `</answer>`) and -0.001 for each extra token after the closing tag `</answer>`.
2. *Soft Format Reward*: $+0.5$ for responses matching the pattern `<reasoning>...</reasoning><answer>...</answer>`.
3. *Strict Format Reward*: $+0.5$ for matching the specified format precisely with correct line breaks.
4. *Integer Answer Reward*: $+0.5$ if the retrieved answer parses as an integer.
5. *Correctness Reward*: $+2$ if the returned answer equals the ground truth exactly.

MATH500. MATH500 (Lightman et al., 2023) is a mathematical reasoning dataset, as well as a curated collection of 500 high-school-level problems sampled from the MATH (Hendrycks et al., 2021) dataset. We conduct fine-tuning on the train split and evaluate on the test split.[4]

The reward comprises

1. *Format Reward*: 1 when answer tags are present and `\boxed` appears inside them; 0.75 when the tags are present but `\boxed` is absent; 0.50 when the tags are missing but `\boxed` is present; 0.25 when neither the tags nor `\boxed` appear.
2. *Correctness Reward*: $+2$ when the correct answer is enclosed in `\boxed{}`.

[3] `https://huggingface.co/datasets/openai/gsm8k`

[4] `https://huggingface.co/datasets/ankner/math-500`

Countdown. Countdown (Pan et al., 2025) is a planning task that requires solving a combinatorial arithmetic challenge, which is to form a target number using basic arithmetic operations with a provided set of 3 numbers, where each number can only be used once. We train on the training split of the dataset from the TinyZero project (Pan et al., 2025), restricting to instances that use only three numbers, and evaluate on 256 synthetically generated countdown questions with three numbers.

The reward checks if an arithmetic expression constructed from given numbers reaches a target value. More specifically, it is 1 when the equation equals the target and uses exactly the available numbers, 0.1 when the equation uses the right numbers but does not reach the target, and 0 if otherwise.

Sudoku. Sudoku is a planning task that requires solving 4×4 Sudoku puzzles, which demand constraint satisfaction and logical elimination to correctly fill the grid. We use the training dataset from `https://github.com/Black-Phoenix/4x4-Sudoku-Dataset`, in particular, the subset containing one million unique puzzles, which was synthetically generated using code from Arel (2025). For evaluation purposes, we randomly generate 256 Sudoku puzzles using this generator. The reward equals the fraction of originally blank cells that the model fills correctly.

C.2 Training Hyperparameters and Evaluation

We choose the training hyperparameters following Zhao et al. (2025a) for a fair comparison. We also use the Transformer Reinforcement Learning library (TRL, von Werra et al. (2020) to implement DMPO. During training, we also employed the same Low-Rank Adaptation (LoRA, Hu et al. (2022)) with a rank of $r = 128$ and scaling factor $\alpha = 64$. For all tasks, the training was conducted on 8 NVIDIA H100 or H200 GPUs with the hyperparameters described below.

We use a maximum generation length 256 tokens, a batch size of 8 per GPU, and gradient accumulation steps of 2, and 16 generated rollouts per prompt. We optimized the model using the AdamW optimizer (Loshchilov & Hutter, 2019) with parameters $\beta_1 = 0.9, \beta_2 = 0.99$, weight decay of 0.1, learning rate of 3×10^{-6}, and gradient clipping at 0.2. For each clean sequence, we sampled 4 partially masked tokens to compute the WDCE/WDDO loss. For rollouts generation during training, we use a semi-autoregressive random order sampler (with temperature 0) and fast-dLLM (with temperature 0.2) with a block size of 32 to generate diverse responses, which is the recommended practice for using LLaDA series models as is described in Nie et al. (2025b). We train $4,000$ steps (number of gradient updates) for GSM8K and MATH500, Countdown, and Sudoku, respectively. In the main result table (Tab. 1), DMPO-LLaDA uses the group weight baseline (13) on GSM8K, MATH500, and Sudoku, and the individual weight baseline (14) on Countdown. DMPO-LLaDA-SFT and DMPO-LLaDA-1.5 adopt the individual weight baseline (14) in all cases.

For the reproduction of the d1 results, we follow the guidelines listed in Zhao et al. (2025a) and first perform SFT on s1k (Muennighoff et al., 2025) before applying diffu-GRPO. We use the recommended hyperparameter setups and train for up to $13,000$ iterations on each dataset before evaluating the results.

For computational efficiency, we use Flash Attention 2 (Dao, 2024) and 4-bit quantization. All experiments on DMPO share these hyperparameters. The main result reported in Tab. 1 used the group weight baseline defined in (13). The ablation study in Fig. 4 also follows the same set of hyperparameters above, except for using different choices of weight baselines.

For the evaluation of all model checkpoints, we consider three different generation lengths: 128, 256, and 512. We correspondingly use 128, 256, and 512 steps for generation. For the LLaDA series of models, such as LLaDA-Instruct, LLaDA-1.5, d1-LLaDA, and our own DMPO-LLaDA, we employ the semi-autoregressive sampler with a block size of 32, a greedy decoding scheme with a temperature of 0, and the top-k remasking scheme to achieve the best inference results. For the Dream model, we also employ the recommended practice and perform inference with temperature 0.95 and the top-k remasking scheme.

C.3 Further Experimental Results

DMPO consistently achieves higher rewards. In Fig. 6, we present the reward dynamics of DMPO across training steps and compare with that of d1. DMPO consistently achieves higher reward values after an initial warm-up phase and ultimately discovers responses with higher rewards than d1,

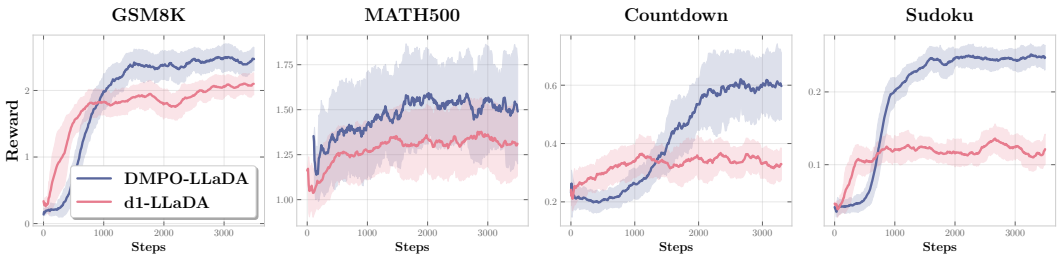


Figure 6: Reward dynamics during training. DMPO consistently produces higher rewards than d1.

possibly because it continuously explores the reward distribution landscape throughout training. In the first $1,000$ steps, DMPO often produces lower reward values than d1, potentially due to the lack of an SFT phase before RL scaling. Moreover, we observe that the performance of DMPO does not saturate after $4,000$ gradient steps, suggesting its greater potential than GRPO-type algorithms.

DMPO benefits from other means of post-training techniques. To showcase the robustness and general efficacy of DMPO, we apply it to LLaDA-SFT and LLaDA-1.5. LLaDA-SFT is obtained by performing SFT of LLaDA-Instruct (8B) on s1k (Muennighoff et al., 2025), a dataset of 1k examples of high-quality reasoning questions with distilled reasoning traces from Gemini Thinking. LLaDA-1.5 is obtained by performing DPO on 350K preference pairs covering a wide range of topics such as writing and reasoning. We then apply DMPO to these base models to obtain DMPO-LLaDA-SFT and DMPO-LLaDA-1.5, with performance reported in Tab. 1. DMPO continues to deliver performance gains for post-trained models, with consistent and significant accuracy improvements over base models, especially at generation lengths of 128 and 256 for the math reasoning datasets, with $+\mathbf{4.78}\%$ and $+\mathbf{2.60}\%$ on GSM8K, $+\mathbf{1.80}\%$ and $+\mathbf{3.40}\%$ on MATH500 compared with d1. This underscores that DMPO is a powerful method that integrates smoothly with existing solutions.

Ablation studies on the hyperparameter dependence. We provide an ablation study on two of the main hyperparameters in Alg. 1, namely the number of rollouts N and the frequency for sampling buffer F, in Figs. 7 and 8, respectively. For each run shown in Fig. 7, we train for 6 hours using 8 NVIDIA H200 GPUs. For each run shown in Fig. 8, we train for 8 hours with 8 NVIDIA H200 GPUs. We only vary the resampling buffer frequency F and the number of rollouts sampled per prompt N, while fixing other hyperparameters, such as the total effective batch size, to maintain a fair comparison.

For the number of rollouts per question N, we observe that a larger number of N does not necessarily lead to longer training time, even with the same number of steps, due to the parallelism of the generation process, since we kept the total batch size fixed while varying the hyperparameter N. The algorithm is robust across various values of N ranging from 4 to 32 thanks to the mechanism for inserting negative gradients.

For the buffer sampling frequency F, we observe that it significantly affects training speed. The figure clearly demonstrates the advantage of DMPO due to its *off-policy* nature, whereas a purely on-policy realization of WDCE loss (with $F = 1$) is not only extremely slow but also does not show a significant boost in per-step reward gains. The figure also underscores the unique benefit of WDCE being a *forward* loss: given the generated rollouts and their weights, one can train using the simple forward process via random masking. Our algorithm is robust to choices of F up to 24, whereas an even larger F may cause slight instability later in training when the reward is high.

Visualizing rollout entropy of DMPO In Fig. 9, we compare the reward and entropy of the generated rollouts during training for both the relative-entropy-based (diffu-GRPO) RL algorithm and the cross-entropy-based (DMPO) RL algorithm. Here, in both experiments, we fix $N = 16$ and $F = 8$ and evaluate the entropy of generated samples every 10 generations. The evaluation of entropy is as follows: we use random-order autoregressive generation with block length 32, and at the d-th step of unmasking (where d ranges from 1 to $D = |\boldsymbol{o}|$), we compute the entropy of the predicted logits at the d-th position, and take average of all the D entropy values as the final value of sequential entropy. From the figure, the trend of consistently higher sample entropy for WDCE loss than for diffu-GRPO agrees with our expectation that cross-entropy-based methods are less prone to mode-seeking and maintain a higher level of diversity throughout training.

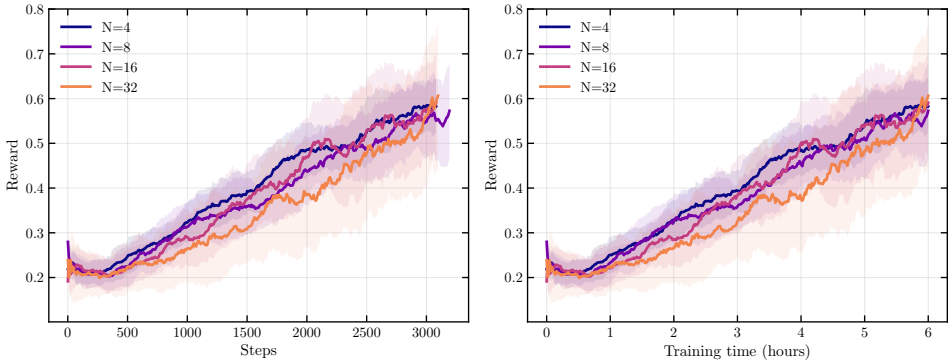


Figure 7: Ablation study of the number of rollouts per prompt N on Countdown dataset under the same training time and compute. The performance is robust to this hyperparameter.

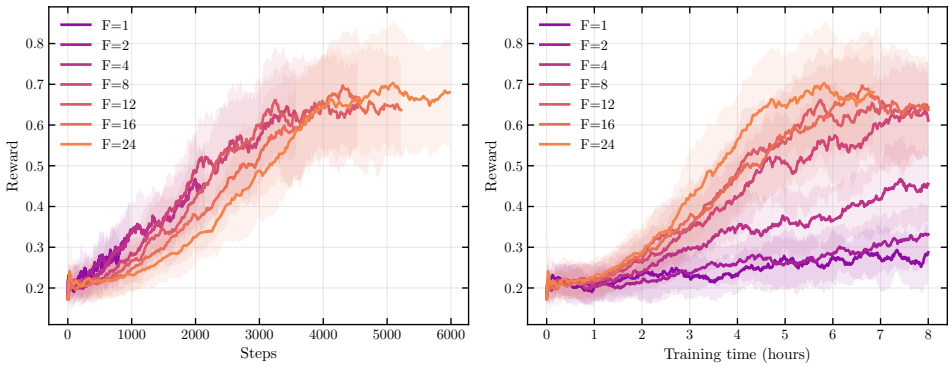


Figure 8: Ablation study of the resampling frequency F on Countdown dataset. A larger F is generally more time-efficient though may cause instability when the reward is high.

C.4 Example Outputs of the Model after Fine-tuning

We present two example outputs of the DMPO-LLaDA model in the following.

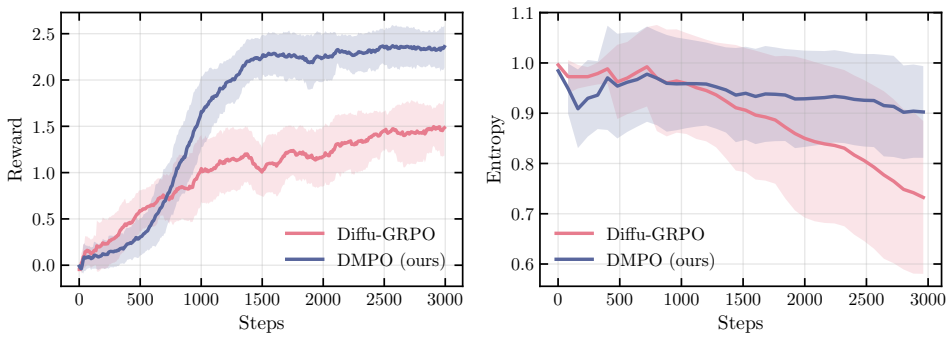


Figure 9: Reward (left) and entropy (right) of generated rollouts during training for both diffu-GRPO (Zhao et al., 2025a) and DMPO.

Model: DMPO-LLaDA
Dataset: MATH500
Generation Sequence Length: 512
Question: The product of $3t^2 + 5t + a$ and $4t^2 + bt - 2$ is $12t^4 + 26t^3 - 8t^2 - 16t + 6$. What is $a + b$?

To find the values of a and b such that the product of $3t^2 + 5t + a$ and $4t^2 + bt - 2$ equals $12t^4 + 26t^3 - 8t^2 - 16t + 6$, we need to expand the left-hand side and compare coefficients with the right-hand side.
First, expand the product:

$$(3t^2 + 5t + a)(4t^2 + bt - 2)$$

Using the distributive property (FOIL method for binomials):

$$3t^2 \cdot 4t^2 + 3t^2 \cdot bt + 3t^2 \cdot (-2) + 5t \cdot 4t^2 + 5t \cdot bt + 5t \cdot (-2) + a \cdot 4t^2 + a \cdot bt + a \cdot (-2)$$

Simplify each term:

$$12t^4 + 3bt^3 - 6t^2 + 20t^3 + 5bt^2 - 10t + 4at^2 + abt - 2a$$

Combine like terms:

$$12t^4 + (3b + 20)t^3 + (-6 + 5b + 4a)t^2 + (-10 + ab)t - 2a$$

We need this to equal:

$$12t^4 + 26t^3 - 8t^2 - 16t + 6$$

By comparing coefficients, we get the following system of equations:

$$3b + 20 = 26 \implies 3b = 6 \implies b = 2$$

$$-6 + 5(2) + 4a = -8 \implies 4 + 4a = -8 \implies 4a = -12 \implies a = -3$$

Thus, $a = -3$ and $b = 2$.

$$a + b = -3 + 2 = -1$$

</reasoning><answer>$\boxed{-1}$</answer>

Model: DMPO-LLaDA
Dataset: GSM8K
Generation Sequence Length: 512
Question: Katarina has 5 less cookies than Max has. Max has 12 more cookies than the Cookie Monster, and Summer has 23 more cookies than Max. If Katarina has 68 cookies, how many cookies do they have in total?

To determine the total number of cookies Katarina, Max, and Summer have, we need to find out how many cookies each person has and then sum them up.
1. We know that Katarina has 68 cookies.
2. Katarina has 5 less cookies than Max.
3. Max has 12 more cookies than the Cookie Monster.
4. Summer has 23 more cookies than Max.
Let's denote the number of cookies each person has as follows:
- K for Katarina's cookies
- M for Max's cookies
- C for the Cookie Monster's cookies
- S for Summer's cookies
From the problem, we have:
- $K = 68$
- $K = M - 5$
- $M = C + 12$
- $S = M + 23$
First, we substitute the known value of K into the equations:
- $68 = M - 5$
- $M = C + 12$
- $S = M + 23$
First, solve for M:

$$68 = M - 5$$
$$M = 68 + 5$$
$$M = 73$$

Next, solve for C:

$$73 = C + 12$$
$$C = 73 - 12$$
$$C = 61$$

Finally, solve for S:

$$73 = S - 23$$
$$S = 73 + 23$$
$$S = 96$$

Now, we have the number of cookies each person has:
- Katarina: 68 cookies
- Max: 73 cookies
- The Cookie Monster: 61 cookies
- Summer: 96 cookies
To find the total number of cookies, we sum these amounts:

$$68 + 73 + 61 + 96 = 298$$

Thus, the total number of cookies is $\boxed{298}$.
</reasoning><answer>$\boxed{298}$</answer>