DiFFPO: Training Diffusion LLMs to Reason Fast and Furious via Reinforcement Learning

Hanyang Zhao
Columbia University
hz2684@columbia.edu

Dawen Liang
Netflix
dliang@netflix.com

Wenpin Tang
Columbia University
wt2319@columbia.edu

David D. Yao
Columbia University
yao@columbia.edu

Nathan Kallus
Netflix
nkallus@netflix.com

Abstract

We propose **DiFFPO**, **Di**ffusion **F**ast and **F**urious Policy Optimization, a unified framework for training masked diffusion large language models (dLLMs) to reason not only *better (furious)*, but also *faster* via reinforcement learning (RL). We first unify the existing baseline approach such as `d1` [50] by proposing to train surrogate policies via off-policy RL, whose likelihood is much more tractable as an approximation to the true dLLM policy. This naturally motivates a more accurate and informative two-stage likelihood approximation combined with importance sampling correction, which leads to generalized RL algorithms with better sample efficiency and superior task performance. Second, we propose a new direction of joint training efficient samplers/controllers of dLLMs policy. Via RL, we incentivize dLLMs' natural multi-token prediction capabilities by letting the model learn to adaptively allocate an inference threshold for each prompt. By jointly training the sampler, we yield better accuracies with lower number of function evaluations (NFEs) compared to training the model only, obtaining the best performance in improving the Pareto frontier of the inference-time compute of dLLMs. We showcase the effectiveness of our pipeline by training open source large diffusion language models over benchmark math and planning tasks.

1 Introduction

Reinforcement Learning from Verifiable Rewards (RLVR, [16]) has achieved remarkable success in enhancing the reasoning capabilities of Large Language Models (LLMs) [8, 13]. The fine-tuned Large Reasoning Models (LRMs) show drastic improvement in solving tasks which require strong reasoning capabilities, such as Math and Coding [17, 30]. These LRMs match, and even surpass the performance of the best human players in math contests [4]. Despite these successes, LRMs are notorious for long inference-time, and overthinking for easy questions [5, 34], which limit their applicability to scenarios that have less tolerance on latencies or inference budgets. An active line of recent research on efficient reasoning [35] proposed either training-free early stopping mechanisms, or generation-length-aware RL (Reinforcement Learning) objectives [15, 44]. However, the resulting RL fine-tuned models are still fundamentally bottle-necked by the left-to-right autoregressive (AR) decoding in decoder-only transformers [38], which will inevitably suffer from quadratic inference costs with respect to the length of reasoning traces.

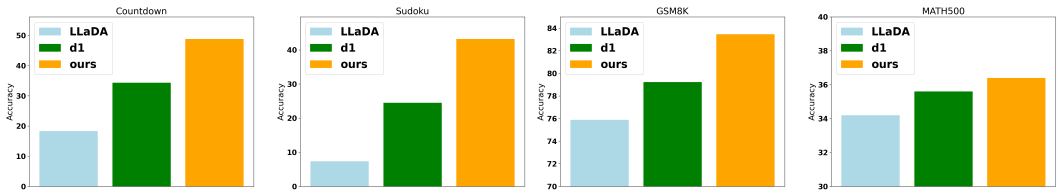
Diffusion LLMs (dLLMs) [18, 19, 26, 32, 47], an emerging family of LLMs based on discrete-space diffusion models [1, 10], have the natural premise of going beyond *left-to-right generations* to *any-order generations* and *multi-token predictions*. Properiatary dLLMs like `Mercury` [14], `Gemini`

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Efficient Reasoning.

Diffusion and Seed Diffusion [33] maintain comparable quality to the state-of-the-art AR models while achieving up to 10 times of token throughput. In contrast with the extensive literature on post-training AR models, the relevant study for post-training dLLMs through RL in the literature so far remains rather limited. Whether RL can be used to enhance the reasoning capabilities of dLLMs still remains largely open, and how to design RL algorithms catered for dLLMs to incentives the capability of base model remains unexplored yet of crucial importance to improve frontier dLLMs.

To narrow this gap, in this work, we design scalable and effective RL algorithms for incentiving reasoning capability of dLLMs. Our contributions in this paper can be summarized as follows:

(1) We first propose an efficient RL post-training paradigm for fine-tuning dLLMs from an off-policy RL perspective. Concretely, we generalize recently proposed algorithms like d1 [50] by proposing to train efficient surrogate policies with more tractable likelihood, as opposed to directly training base dLLMs policies, which requires extensive GPU memory to estimate and is inefficient to compute dLLM's true likelihood. Based on our framework, we propose a new RL algorithm utilizing new surrogate policies by conditioning on additional latents at the response-generation level for better approximation, instead of only conditioning on prompts as in d1. Inspired by classical off-policy RL, we also introduce an importance-sampling correction term to address the distribution mismatch between the surrogate policies and the true dLLMs behavior policies. We provide both theoretical guarantees and strong empirical evidence, achieving evident performance gains over the baseline method, especially for planning tasks as in Figure 1.



(a) Countdown (b) Sudoku (c) GSM8K (d) Math

Figure 1: Benchmark results of RL post-trained models across different math and planning tasks.

(2) Secondly, we innovatively propose to adopt efficient dLLMs samplers in RL post-training. Unlike prior work which post-trains the models based on a fixed (often inefficient) sampler, we directly train the model upon efficiently dLLMs samplers to inventive the base model to reason better and faster. To avoid overfitting to the utilized samplers, we propose to train a prompt-aware inference threshold motivated by the Entropy-Bounded (EB) sampler proposed in [2] via RL as opposed to a fixed threshold across different prompts. This approach naturally leverages the structural property of masked dLLMs for multi-token predictions, and lets the model perform inference based on a predicted "inference threshold" for the prompts. We enable efficient joint RL train of the model and sampler by treating the inference threshold as an additional token, which is thus compatible with our earlier proposed RL framework. We showcase that jointly training the model and sampler can yield improved post-trained models with better performance/accuracy under the same or even less compute than training the model only, unlocking a novel new direction for further research to investigate mechanisms between model and samplers in RL post training.

We term our pipeline as **DiFFPO** - **Di**ffusion **F**ast and **F**urious Policy Optimization, for training masked dLLMs to reason not only *better (furious)*, but also *faster* via RL. We demonstrate the effectiveness of DiFFPO in enhancing the inference-time performance of diffusion LLMs on benchmarking math and planning tasks, and clearly push the boundaries of designing scalable RL algorithms towards training efficient and capable LRMs.

The rest of the paper is organized as follows: In Section 2, we review the preliminaries of masked dLLMs, dLLMs samplers, and RL for discrete diffusion models. Section 3 presents our detailed algorithm of DiFFPO, with experimental results provided in Section 4. In Section 5 we discuss other relevant references and conclude with Section 6.

2 Preliminaries

In this work, we focus on *masked* discrete diffusion language models for simplicity, which has been shown to achieve the best performance among different formulations of dLLMs. Our framework is general and can be modified to other discrete diffusion models as well. For an introduction to dLLMs, we follow the presentations in MDLM [26].

Masked Diffusion Models. Let `[mask]` be an extra special token additional to the token vocabulary $\mathcal{V}$, and denote by $\boldsymbol{m}$ the one-hot representation of this mask token. The forward processes of MDLM [1, 26] interpolate between the clean data distribution $\boldsymbol{x} \sim p_{\text{data}}(\cdot)$ and a target non-informative distribution $\text{Cat}(\cdot; \boldsymbol{m})$ (the categorical distribution). The latent variable $\boldsymbol{z}_t$ with $t \in [0, 1]$ in the forward process is:

$$q\left(\boldsymbol{z}_t \mid \boldsymbol{x}\right) = \text{Cat}\left(\mathbf{z}_t; \alpha_t \boldsymbol{x} + (1 - \alpha_t)\, \boldsymbol{m}\right), \tag{1}$$

where $\alpha_t \in [0, 1]$ is a strictly decreasing function in t, with $\alpha_0 \approx 1$ and $\alpha_1 \approx 0$. This process can be viewed as a masking process, because $q\left(\boldsymbol{z}_1 \mid \boldsymbol{x}\right) \approx \boldsymbol{m}$. The posterior of the masking process $(t > s)$ can be explicitly computed as (see e.g., [26] for a derivation):

$$q\left(\boldsymbol{z}_s \mid \boldsymbol{z}_t, \boldsymbol{x}\right) = \begin{cases} \text{Cat}\left(\boldsymbol{z}_s; \boldsymbol{z}_t\right) & \boldsymbol{z}_t \neq \boldsymbol{m}, \\ \text{Cat}\left(\mathbf{z}_s; \frac{1-\alpha_s}{1-\alpha_t}\boldsymbol{m} + \frac{\alpha_s - \alpha_t}{1-\alpha_t}\boldsymbol{x}\right) & \boldsymbol{z}_t = \boldsymbol{m}. \end{cases} \tag{2}$$

Since $\boldsymbol{x}$ is unknown during the forward process, MDLMs learn a function approximation $\boldsymbol{x}_\theta(\boldsymbol{z}_t, t)$, and approximate the true posterior with $p_\theta\left(\boldsymbol{z}_s \mid \boldsymbol{z}_t\right)$ by replacing $\boldsymbol{x}$ with approximated $\boldsymbol{x}_\theta$ in Equation 2 when $\boldsymbol{z}_t = \boldsymbol{m}$.

dLLMs efficient samplers. For masked dLLMs, the backward process needs to choose among unmasked positions which to unmask first. This leaves an extra degree of freedom for designing efficient dLLMs samplers. Most existing works focus on a fixed number of multi-token predictions, such as random sampling or Top-k sampling [19] by choosing the position based on some pre-defined scores. The scores are generally chosen from proxy metrics like confidence $s_l = \max_{v \in \mathcal{V}} p_l(v)$ or negative entropy $-\mathcal{H}(p_l)$ of the predictive distribution of dLLMs $p_l \in \Delta^{|\mathcal{V}|-1}$ on the position l. These top-k based samplers are shown to be able to outperform randomly choosing k positions [15].

In this work, our sampling scheme mainly follows the Entropy-Bounded (EB) sampler proposed in [2], which yields better accuracy and efficiency tradeoff than the Top-k sampler that sweeps over different k's. When jointly unmasking a sequence of variables $X = (\boldsymbol{x}^1, \cdots, \boldsymbol{x}^L)$, the EB-sampler computes the entropy of the predictive distribution of dLLMs at each unmasked position as a cost of that position $c(l) = H(p^\theta(\boldsymbol{x}^l \mid X))$, and then unmask all positions in U with the maximum cardinality: $\sum_{l \in U} c(\ell) \leq \gamma$, where γ is a predetermined threshold hyperparameter. If $\min_{l \in U} c(\ell) > \gamma$, EB sampler will only unmask the position with the smallest cost. Intuitively, small γ will lead to behaviours similar to top-1 sampling, while large γ will allow unmasking of several positions at the same time. Different γ yields different samplers, which forms an inference-time compute frontier for dLLMs.

RL for dLLMs. RLVR aims at training models to maximize the expected reward of the policy generations plus an KL-regularization term via Reinforcement Learning [36], i.e.,

$$\mathbb{E}_{c \sim \mathcal{D}, o \sim \pi_\theta(\cdot|c)}\left[r(c, o) - \beta\, \text{KL}(\pi_\theta(\cdot \mid c), \pi_{\theta_{\text{ref}}}(\cdot \mid c))\right], \tag{3}$$

where $r(c, o)$ is the verifiable reward of generation o with respect to the prompt c (sampled from the population $\mathcal{D}$), such as correctness or unit test success rates, $\text{KL}(p, q)$ denotes the Kullback–Leibler divergence between two distributions p and q, and $\beta > 0$ is the KL penalty constant. GRPO [30], as a variant of REINFORCE [37] and PPO [29], first samples a group of outputs $\{o_i\}_{i=1}^G$ from the behaviour (old) policy $\pi_{\theta_{\text{old}}}$ under the same prompt c, and computes the normalized advantage function $A_i = \left(r\left(c, o_i\right) - \frac{1}{G}\sum_{j=1}^G r\left(c, o_j\right)\right)/\sigma$, in which σ is the standard deviation of the group rewards. Denote $o^{<k}$ as tokens already generated before the kth token is decoded within completion o, GRPO maximizes the following token level objective:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{c \sim \mathcal{D}, o_i \sim \pi_{\theta_{\text{old}}}(\cdot|c)} \sum_{i=1}^{G} \frac{1}{|o_i|} \sum_{k=1}^{|o_i|} \min\left(\rho_i^k A_i, \text{clip}\left(\rho_i^k, 1-\varepsilon, 1+\varepsilon\right) A_i\right), \tag{4}$$

where $\rho_i^k = \pi_\theta\left(o_i^k \mid c, o_i^{<k}\right) / \pi_{\text{old}}\left(o_i^k \mid c, o_i^{<k}\right)$ is the token-level likelihood ratio. Notably we omit the KL divergence term in Equation 4 onwards for simplicity, since it has been found to be less impactful in the reasoning performance, see e.g., Magistral [24]. Since ρ_i^k is inefficient to compute for every token in practice, d1 [50] proposed a *mean-field approximation* $\hat{\rho}_i^k = \pi_\theta\left(o_i^k \mid c\right) / \pi_{\text{old}}\left(o_i^k \mid c\right)$ to replace ρ_i^k in Equation 4, which is easily computable from dLLMs. In essence, the likelihood ratio used in `d1` only conditions on the prompts c, omitting the already unmasked tokens $o_i^{<k}$ which dLLMs actually condition on during generation.

3 DiFFPO

In this section, we present the derivation of our Reinforcement Learning algorithm including the optimization loss function and the update procedures.

3.1 Improved techniques in efficient model post-training

We first consider a fixed top-k confidence score sampler, which is how the completions of prompts/questions are generated. We use $k = 1$ for simplicity here to derive the loss objectives.

Motivation. We first revisit the objective of `d1` [50] from a likelihood approximation perspective. The loss objective of diffu-GRPO algorithm is:

$$\mathcal{J}_{\text{d1}}(\theta) = \mathbb{E}_{c\sim\mathcal{D}, o_i\sim\pi_{\theta_{\text{old}}}} \sum_{i=1}^{G} \frac{1}{|o_i|} \sum_{k=1}^{|o_i|} \min\left(\frac{\pi_\theta\left(o_i^k \mid c\right)}{\pi_{\text{old}}\left(o_i^k \mid c\right)} A_i, \text{clip}(\frac{\pi_\theta\left(o_i^k \mid c\right)}{\pi_{\text{old}}\left(o_i^k \mid c\right)}, 1-\varepsilon, 1+\varepsilon) A_i\right), \tag{5}$$

in which $\pi_\theta\left(o_i^k \mid c\right)$ is the approximation to true conditional token likelihood, and referred by [50] as *mean-field approximation*. This approximation has its own advantage of being simple and memory-efficient to compute, making it appealing to be utilized for update at loss optimization, especially when fine-tuning large language models with massive scale.

However, there are two shortcomings in this approximation. **Firstly**, there is a clear mismatch between this and the true likelihood, since the dLLMs need to condition on already unmasked tokens, which is omitted in the mean-field approximation. **Secondly**, hardly any theoretical performance guarantees can be made for the policy obtained from minimizing the loss in Equation 5.

To illustrate these limitations, we first change the index from token number to time index to express our loss in a standard way in the discrete diffusion literature, which eliminates the ambiguity in the generative process.[1] For notations, we denote t_i^k to be the actual timestep when the the k^{th} token in completion o_i is unmasked (thus $1 \le t_i^k \le |o_i|$), and we use z_i^t to denote the sequence latents at time t. For a discrete diffusion model / policy, we can write an equivalent loss to the GRPO objective [30] for dLLM as $\mathcal{J}_{\text{dLLM}-\text{GRPO}}(\theta) =:$

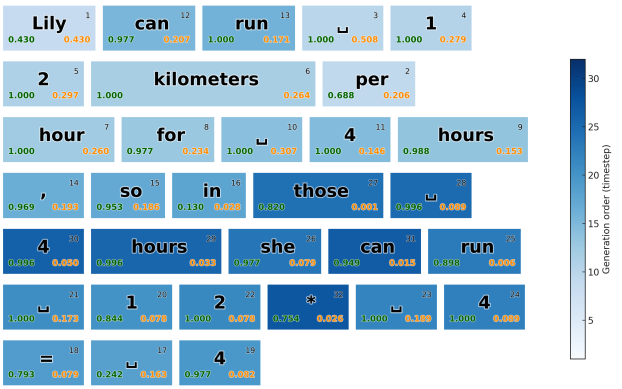
$$\mathbb{E} \sum_{i=1}^{G} \frac{1}{|o_i|} \sum_{k=1}^{|o_i|} \min\left(\frac{\pi_\theta\left(o_i^k \mid c, z_i^{t_i^k-1}\right)}{\pi_{\text{old}}\left(o_i^k \mid c, z_i^{t_i^k-1}\right)} A_i, \text{clip}(\frac{\pi_\theta\left(o_i^k \mid c, z_i^{t_i^k-1}\right)}{\pi_{\text{old}}\left(o_i^k \mid c, z_i^{t_i^k-1}\right)}, 1-\varepsilon, 1+\varepsilon) A_i\right), \tag{6}$$

which is equivalent to (we now denote $\hat{o}_i^t$ as the difference between z_i^t and z_i^{t-1}):

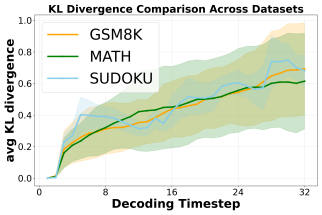
$$\mathbb{E} \sum_{i=1}^{G} \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \min\left(\frac{\pi_\theta\left(\hat{o}_i^t \mid c, z_i^{t-1}\right)}{\pi_{\text{old}}\left(\hat{o}_i^t \mid c, z_i^{t-1}\right)} A_i, \text{clip}(\frac{\pi_\theta\left(\hat{o}_i^t \mid c, z_i^{t-1}\right)}{\pi_{\text{old}}\left(\hat{o}_i^t \mid c, z_i^{t-1}\right)}, 1-\varepsilon, 1+\varepsilon) A_i\right). \tag{7}$$

The `d1` objective can be interpreted as removing all z_i^{t-1} terms in the loss objective Equation 7. However, as shown through generated sample example in Figure 2a and average KL divergence plot in Figure 2b, when the timesteps increase, there is growing mismatch between the true conditional likelihood $\pi_\theta(\hat{o}_i^t \mid c, z_i^{t-1})$ and mean-field approximation $\pi_\theta(\hat{o}_i^t \mid c)$. The average KL divergence between these two distributions grows monotonicly with respect to t, likely due to the accumulated effect of neglecting the latents.

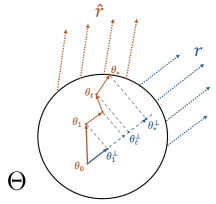
[1] There is no such ambiguity for AR LLMs, since the generation will be always from left to right.



(a) Sample Generation and Computed Likelihoods: For each block, for example the first one, black part "Lily" represents the unmasked token, green number (0.430) represents the true likelihood when the token gets unmasked, and the orange number (0.430) denotes the mean-field approximation likelihood



(b) Average KL between the true likelihoods and approximated likelihoods at each timestep.



(c) Visualization of off-policy RL for dLLM through optimizing the surrogate policy

Figure 2: Generation sample and average KL divergence error at each decoding timestep (100 prompts per dataset).

Rethinking via off-policy RL. Given the crude approximations utilized in `d1` and the lack of any theoretical guarantee, we propose to formulate the RL task for dLLMs from an off-policy RL perspective by noticing the unique structure of the dLLMs. Since the true likelihood of dLLMs is inefficient to compute, our aim is to bypass it by: (1) find a surrogate policy that is close to the actual dLLM policy, while its conditional likelihood is tractable to compute; (2) optimize the surrogate policy via off-policy RL. A visualization of our pipeline can be found in Figure 2c. Before we discuss how we choose the surrogate policies, we highlight that our current pipeline could provide a worst-case performance guarantee on the obtained fine-tuned models:

Theorem 1. *Assume two parameterized policies by different model family yet share the same parameters being $\pi_\theta, \hat{\pi}_\theta$ respectively, with $\theta \in \Theta$. Further assuming that reward function is postive and upper bounded, i.e. there exists $M > 0$, such that $0 \le r(c, o) \le M$ for any c and o. Then given that $\max_{\theta\in\Theta} \mathrm{KL}(\hat{\pi}_\theta \| \pi_\theta) \le \epsilon^2$, we have that for any policy $\theta \in \Theta$:*

$$\mathbb{E}_{o\sim\pi_\theta(\cdot|c)} r(c, o) \ge \mathbb{E}_{o\sim\hat{\pi}_\theta(\cdot|c)} r(c, o) - \sqrt{2}M\epsilon. \tag{8}$$

The proof is a straightforward application of Pinsker's inequality. Yet, a direct corollary of Theorem 1 implies that, the worst-case of the performance of π^θ can still be guaranteed if $\mathbb{E}_{o\sim\hat{\pi}_\theta(\cdot|c)} r(c, o)$ is large (e.g. after RL optimization) and two policies are close. Back to the dLLM setup, if we take a policy $\hat{\pi}_\theta$ which generates tokens on every position by conditioning only on the prompt, then this is the exact policy `d1` utilized for likelihood approximation, which we have shown to be too coarse with a large KL-divergence. Thus we first construct better surrogate policies to make it closer to the true dLLM policy.

Additional conditioning latents. We propose to condition on one additional latent at a randomly sampled timestep $\tau \in [0, T)$ at each optimization step, and consider a new surrogate policy which samples tokens according to the `d1` likelihoods until $t = \tau$. When $t > \tau$, we sample tokens by conditioning on both prompts and latents z^τ. Conditioning on the randomly drawn timestep τ, we define the likelihood of the completion as $\pi_\theta(\hat{o}_i^t \mid c, z_i^{s_\tau(t)})$ where we define a step function s_τ such that $s(t) = \tau$ if $t > \tau$, and 0 otherwise. We can interpret this as an alternative approximation to the true dLLM likelihood. However, instead of only conditioning on prompts, when $t > \tau$ the generation will also condition on a randomly drawn latent z^τ. We refer to this as Two Times Mean-Field Approximation (abbr. as 2-MF). Compared to `d1` [50], two times mean-field yields a strictly better likelihood approximation. We theoretically characterize its benefits by the following assumptions and theorems.

Assumption 2. *Assuming that for any trajectory $z_i^t (t = 0, \cdots, T)$ and three timesteps $s < \ell < t$,*

$$f(s;t) := \mathrm{KL}(\pi_\theta(\hat{o}_i^t \mid c, z_i^{t-1}) || \pi_\theta(\hat{o}_i^t \mid c, z_i^s)) \geq \mathrm{KL}(\pi_\theta(\hat{o}_i^t \mid c, z_i^{t-1}) || \pi_\theta(\hat{o}_i^t \mid c, z_i^\ell)) = f(\ell;t), \tag{9}$$

i.e. $f(\tau;t)$ is a monotonously decreasing function with respect to $\tau \in [0,t]$.

Assumption 2 assumes that a more recent latent on the same generation trajectory will be more informative than older ones. It is reasonable since the older latents are closer to the non-informative priors. Based on this assumption, we characterize the theoretical benefits of our approximation:

Theorem 3. *Under Assumption 2, we have that two times mean-field approximation provides better approximation than only conditioning on prompts, i.e. for any t, we have:*

$$\mathbb{E}_{\tau,t}\, \mathrm{KL}(\pi_\theta(\hat{o}_i^t \mid c, z_i^{t-1}) || \pi_\theta(\hat{o}_i^t \mid c, z_i^{s_\tau(t)})) \leq \mathbb{E}_t\, \mathrm{KL}(\pi_\theta(\hat{o}_i^t \mid c, z_i^{t-1}) || \pi_\theta(\hat{o}_i^t \mid c)). \tag{10}$$

Proof. The inequality could be directly obtained by tower's rule of expectation and proving that

$$\mathbb{E}_t\, \mathrm{KL}(\pi_\theta(\hat{o}_i^t \mid c, z_i^{t-1}) || \pi_\theta(\hat{o}_i^t \mid c, z_i^{s_\tau(t)})) \leq \mathbb{E}_t\, \mathrm{KL}(\pi_\theta(\hat{o}_i^t \mid c, z_i^{t-1}) || \pi_\theta(\hat{o}_i^t \mid c))$$

holds for any fixed τ. This can be easily obtained from Assumption 2. □

Off-policy RL via Importance Sampling. With a better surrogate policy, we now seek to optimize such policy using off-policy RL. Note that the generations are still sampled from the base policy. We can thus utilize these samples to optimize $\hat{\pi}^\theta$ using importance sampling, as for any function f, we have:

$$\mathbb{E}_{o\sim\hat{\pi}_\theta(\cdot|c)} f(c,o) = \mathbb{E}_{o\sim\pi_\theta(\cdot|c)} \frac{\hat{\pi}_\theta(\cdot \mid c)}{\pi_\theta(\cdot \mid c)} f(c,o). \tag{11}$$

Then replacing f with the GRPO loss, our DiFFPO loss objective for training the model only is thus:

$$\mathcal{J}_{\text{DiFFPO-model}} =$$

$$\mathbb{E}_{o_i\sim\pi_{\theta_{\text{old}}},\tau\sim\mathcal{U}[0,T]} \sum_{i=1}^{G} \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \underbrace{\min(C, \frac{\pi_{\text{old}}(\hat{o}_i^t \mid c, z_i^{s_\tau(t)})}{\pi_{\text{old}}(\hat{o}_i^t \mid c, z_i^{t-1})})}_{\text{IS term}} \min\left(\bar{\rho_i}^t A_i, \text{clip}\left(\bar{\rho_i}^t, 1-\varepsilon, 1+\varepsilon\right) A_i\right), \tag{12}$$

where $\bar{\rho}_i^t = \frac{\pi_\theta\left(\hat{o}_i^t | c, z_i^{s_\tau(t)}\right)}{\pi_{\text{old}}\left(\hat{o}_i^t | c, z_i^{s_\tau(t)}\right)}$ is the likelihood ratio of the surrogate policy and we also impose a maximum threshold C over the importance sampling ratio to ensure the numerical stablibity.

3.2 Unlocking better adaptive multi-token prediction via sampler post-training

Common practice of RL post training considers a fixed generative model sampler in optimization. In this work, we however investigate on joint training the sampler together with the model weights for achieving the best performance in enhancing the inference time compute frontier.

Unlike the EB-sampler [2] that uses a fixed threshold to decide which token(s) to unmask for all prompts, we propose to use a parameterized function to encode prompt features and output a predicted "inference threshold". We obtain the dLLMs embeddings of the prompts by averaging over the hidden dimension features $f_\theta^l(c)$ of each position l, and train a linear mapping $\boldsymbol{w}$ on top of the embeddings (we omit the bias term here for clarity). We also introduce an upper threshold $\gamma_{\max}$, and map each feature to the inference threshold by

$$\gamma_{\boldsymbol{w}}(c) = \gamma_{\max}\ \text{sigmoid}\left(\boldsymbol{w}^\top\left(\frac{1}{L}\sum_{l=1}^{L} f_\theta^l(c)\right) + \beta\right), \quad \text{with } \beta = \log\frac{\gamma}{\gamma_{\max}-\gamma}, \tag{13}$$

where $\gamma \in (0, \gamma_{\max})$ is the initial global threshold, since $\gamma_{\boldsymbol{w}_0}(c) = \gamma$ when the initial linear layer weights $\boldsymbol{w}_0$ are set to be all 0. For RL training, we fix a noise level σ for Gaussian exploration $\epsilon \sim \mathcal{N}(0, I)$ before applying sigmoid activation and use the perturbed threshold for inference:

$$\gamma(c) = \gamma_{\max}\ \text{sigmoid}\left(\boldsymbol{w}^\top\left(\frac{1}{L}\sum_{l=1}^{L} f_\theta^l(c)\right) + \beta + \sigma\epsilon\right). \tag{14}$$

We use π_θ^w to represent the sampler threshold policy which is dependent on both model weights θ and header weights w. To joint optimize the model and sampler, we apply a trick by treating the predicted threshold as an additional token to unmask at time step 0. Thus, given the group completion (γ_i, o_i), we have yield our final DiFFPO loss as:

$$\mathcal{J}_{\text{DiffPO}} = \mathcal{J}_{\text{DiffPO-model}} + \mathbb{E}_{c\sim\mathcal{D}} \sum_{i=1}^{G} \frac{1}{|o_i|} \min \left(\frac{\pi_{\theta_-}^{w} (\gamma_i \mid c)}{\pi_{\theta_-}^{w_{\text{old}}} (\gamma_i \mid c)} A_i, \text{clip}\left(\frac{\pi_{\theta_-}^{w} (\gamma_i \mid c)}{\pi_{\theta_-}^{w_{\text{old}}} (\gamma_i \mid c)}, 1-\varepsilon, 1+\varepsilon \right) A_i \right), \tag{15}$$

We found that the model learns to generate shorter sequences even without the length penalty involved as in the experiments section, which prevents sensitive hyperparameter tunings. In addition, fixing model weights (i.e. `stopgrad` on θ_-) for the sampler loss in Equation 15 helps both stable training and better performance.

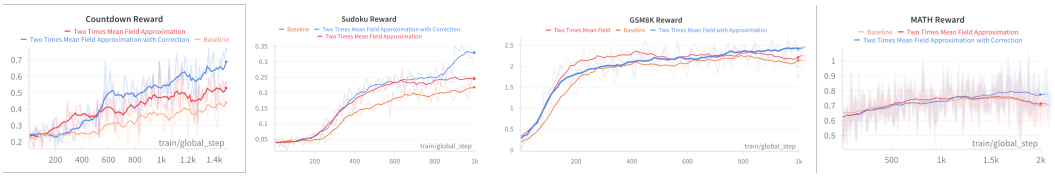
4 Experiments

To evaluate the performance of the our proposed RL algorithms, we conducted extensive experiments on training base dLLMs with various reasoning tasks and compared their reasoning capabilities in terms of both accuracy and efficiency after RL post-training.

Experimental Setup. We choose LLaDA-8B-Instruct [19][2] as our base model since this is the first diffusion large language model whose performance is on par with AR LLMs (Llama3-8B). Improving over such a capable base model is of great interest to the whole dLLMs community. We directly start from the instruct model as opposed to the base model which has not been fine-tuned for instruction following. This avoids any potential confounding effect of us performing additional supervised fine-tuning (SFT) stage which can affect the performance and fair comparison of RL algorithms.

We evaluate all the methods on math and planning benchmark tasks including GSM8K, Math, Sudoku, and Countdown. We use LoRA [11] with rank $r = 128$ for parameter-efficient training – the same hyparameter utilized as in `d1` [50], along with other model generation setups for a fair comparison. In addition, we use the same five random reeds for the baseline `d1` and our proposed algorithm, and report the best performing model among different seeds and checkpoints. For the dLLM inference setup, our experiments are all based on block size 32 and maximum sequence length of 256.

Improved sample efficiency and task performance. We first showcase the effectiveness of our algorithm when training a DiFFPO model with Equation 12 using a fixed top-k sampler on math and planning benchmarks.



(a) Countdown (b) Sudoku (c) GSM8K (d) Math

Figure 3: Benchmark results (−: DiFFPO; −: DiFFPO without IS; −: `d1` baseline).

We plot the reward as the training progresses in Figure 3. Our algorithm shows a clear margin over `d1` in all tasks, especially in planning tasks. Concretely, using two-times mean-field approximation greatly improves the sample efficiency, and applying importance sampling correction further enhances the performance of RL training.

We report concrete benchmark statistics in Table 1. Evidently, our proposed DiFFPO, combining both two times mean-field approximation and importance sampling, yields the most significant performance gain, which both components contribute to. We also compare the performance difference for the same model checkpoint utilizing different maximum length as in Table 2 (in which we denote '2MF' as an abbreviation for Two Times Mean-Field Approximation, and 'IS' for importance sampling). We found that our DiFFPO trained models stably perform the best.

[2] `https://huggingface.co/GSAI-ML/LLaDA-8B-Instruct`.

Model	Countdown			Sudoku			GSM8K			MATH500		
	Acc. ↑	Δ	ETs	Acc. ↑	Δ	ETs	Acc. ↑	Δ	ETs	Acc. ↑	Δ	ETs
LLaDA-8B-Instruct	18.36	–	214.18	7.37	–	220.45	75.89	–	211.63	34.20	–	234.66
+ d1	34.38	+16.02	176.04	24.51	+17.14	233.42	79.23	+3.34	143.12	35.60	+1.40	228.31
+ Two Times Mean-Field	44.53	+26.17	78.94	29.74	+22.37	235.39	81.05	+5.16	127.33	35.80	+1.60	232.36
+ Importance Sampling	**48.83**	**+30.47**	74.98	**43.21**	**+35.84**	244.22	**83.47**	**+7.58**	128.95	**36.40**	**+2.20**	232.98

Table 1: Benchmark results with statistics **(Accuracy, Δ, Effective Tokens (ETs))** under Top k sampler. Δ indicates the accuracy improvement over **LLaDA-8B-Instruct**; positives are highlighted in green.

Model \Acc. ↑ \Seq Len	Sudoku			Countdown			GSM8K			MATH500		
	128	256	512	128	256	512	128	256	512	128	256	512
LLaDA-8B-Instruct	10.01	7.37	6.40	17.97	18.36	21.09	68.16	75.89	79.83	26.20	34.20	35.20
d1	23.88	24.51	25.98	27.34	34.38	37.89	72.86	79.23	78.92	33.20	35.60	37.20
2MF+IS	**36.43**	**43.21**	**45.70**	**58.98**	**48.83**	**56.64**	**75.74**	**83.47**	**82.26**	**34.80**	**36.40**	**38.60**

Table 2: Ablations on the maximum sequence length of the model.

Training DiFFPO with efficient samplers. We also investigate the performance of RL-training on existing efficient samplers to gain the best performance. We tested EB samplers [2], which yields a greater trade-off between accuracy and NFEs compared to the top-k sampler that we used above. We found that the inference threshold $\gamma = 0.1$ yields the best results in achieving the highest reward, on par with smaller γ's and top-k (Table 1, $k = 2$), and better than larger γ's which will typically yield a smaller average reward even after RL training.

Model	Sudoku		Countdown		GSM8K	
	Accuracy ↑	NFEs ↓	Accuracy ↑	NFEs ↓	Accuracy ↑	NFEs ↓
LLaDA-8B-Instruct	10.45	201.23	22.66	212.33	80.74	98.14
+ d1	25.88	105.39	35.16	197.05	82.34	65.71
+ **DiFFPO** w/o sampler joint training	32.23	141.14	47.27	38.80	83.47	57.68
+ **DiFFPO** w/ sampler joint training	**37.16**	**56.80**	**51.17**	**30.45**	**84.00**	**44.00**

Table 3: Benchmarking planning and math tasks results on both accuracies and NFEs under EB.

Joint training of the model and the sampler. Finally we consider joint training of the model and sampler by adding up the model and the sampler losses (Equation 15), to improve the inference-time frontier. In Table 3, we report the optimal checkpoint performance alongside the baseline algorithms. We observe that joint training not only improves the (optimal) correctness, but also uses even less NFEs compared to training the model only.

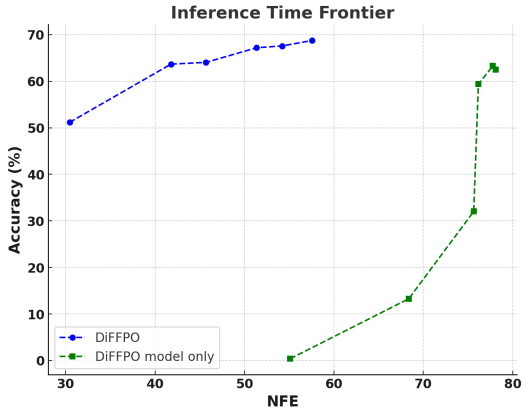


Figure 4: Inference-time frontier obtained by models by DiFFPO with or without sampler training and EB sampler with different inference threshold.

For a more detailed comparison, we draw the inference-time compute frontier for Countdown for two settings: 1) post-trained DiFFPO model with a fixed top-k sampler (Equation 12); and 2) the model and sampler jointly trained by DiFFPO (Equation 15), under the same 256 maximum generation length in Figure 4. The dots are obtained by utilizing different base inference threshold in EB sampler on two models. The clear advantage of DiFFPO trained models (i.e. with sampler joint training) showcases the importance of choosing proper performant samplers when training the models via RL which is usually overloooked, and the effectiveness of our joint training mechanism in enhancing the inference-time compute frontier.

5 Related Works

We discuss other related works on dLLMs, inference, and RL.

Discrete Diffusion Models. In our paper, we mainly focus on masked diffusion models with a discrete-time Markov chain setup [1, 26]. Other works have pursued different formulations, e.g., through continuous-time Markov chains [3] or flows [6, 31]. Despite our focus on masked diffusion models, our methodology developed in this paper is general and could benefit RL training for these other formulations. Several works have developed unified viewpoints for diffusion models in both discrete and continuous spaces, including [9, 25, 27]. The dLLMs family has also been expanded to the multimodal domain, e.g., MMaDA [45] and Fudoki [40].

dLLMs inference. Most existing works on dLLMs inference have focused on developing more performant samplers, using ideas around remasking/predictor-corrector sampling [6, 39] for inference-time scaling and/or planning [12, 22, 46, 51]. EB-sampler [2] focuses on efficiency, which is less studied for discrete diffusion models. Prior work like [21] also studied accelerating diffusion LLMs via a joint dependence perspective. There are also several recent works [28, 43] on enabling KV caches for discrete diffusion models, which will speed up both the inference and RL training processes. Most of these ideas are orthogonal to our contributions and can be easily integrated. We leave them as future work.

RL for dLLMs. Comparing to the extensive study on RLHF [20, 42] and RLVR [16, 49] for LLMs, there is noticably less study on dLLMs. `d1` [50] proposes diffu-GRPO, which is the first work on training base dLLMs for reasoning with RL. Earlier work like [48] has explored policy gradient methods on smaller scale models. [52] proposed a DPO-style [23] preference optimization method underpinning the development of LLaDA 1.5. [45] proposed UniGRPO which utilizes random partial masking on completions instead of full masking. DiffuCoder [7] used coupled-GRPO to improve the base coding dLLMs. LLaDOU [12] trained a token-position aware sampler for enhanced dLLMs reasoning performance. Concurrent to our work, TraceRL [41] also proposes to utilize online trajectory token latents, which is similar to our two times mean-field approximation, yet our parameterization based on time steps is more generalizable and flexible for different dLLM samplers.

6 Discussion and Future Works

In this work, we propose DiFFPO – an RL pipeline for post-training dLLMs towards efficient reasoning. We showcase the effectiveness of our proposed algorithms in terms of better likelihood approximation, a unified off-policy RL perspective, and sampler joint training, which achieves better sample efficiency, superior performance, and enhancing the accuracy/efficiency tradeoff. We believe our contributions in the paper help push the boundary of RL for dLLMs post training research which is less explored.

There are several directions that can be pursued to improve or extend our work. It is interesting to understand the generalization behavior of the trained inference threshold; it is also possible to adapt our RL framework to a state (or group)-dependent inference threshold instead of being fixed for each prompt. In addition, we only test our model on LLaDA [19], it will be complimentary to see the performance on other open source dLLMs like Dream [47]. We leave these investigations for future work.

Acknowledgment

The works of Hanyang Zhao and Wenpin Tang are supported by NSF grant DMS-2206038. Wenpin Tang is supported by the Columbia Innovation Hub grant, and the Tang Family Assistant Professorship. The works of Hanyang Zhao, and David Yao are part of a ColumbiaCityU/HK collaborative project that is supported by InnoHK Initiative, The Government of the HKSAR and the AIFT Lab.

References

[1] J. Austin, D. D. Johnson, J. Ho, D. Tarlow, and R. Van Den Berg. Structured denoising diffusion models in discrete state-spaces. In *Neurips*, volume 34, pages 17981–17993, 2021.

[2] H. Ben-Hamu, I. Gat, D. Severo, N. Nolte, and B. Karrer. Accelerated sampling from masked diffusion models via entropy bounded unmasking. 2025. arXiv:2505.24857.

[3] A. Campbell, J. Benton, V. De Bortoli, T. Rainforth, G. Deligiannidis, and A. Doucet. A continuous time framework for discrete denoising models. *Advances in Neural Information Processing Systems*, 35:28266–28279, 2022.

[4] L. Chen, J. Gu, L. Huang, W. Huang, Z. Jiang, A. Jie, X. Jin, X. Jin, C. Li, and K. Ma. Seed-prover: Deep and broad reasoning for automated theorem proving. 2025. arXiv:2507.23726.

[5] X. Chen, J. Xu, T. Liang, Z. He, J. Pang, D. Yu, L. Song, Q. Liu, M. Zhou, and Z. Zhang. Do not think that much for 2+ 3=? on the overthinking of o1-like llms. 2024. arXiv:2412.21187.

[6] I. Gat, T. Remez, N. Shaul, F. Kreuk, R. T. Chen, G. Synnaeve, Y. Adi, and Y. Lipman. Discrete flow matching. *Advances in Neural Information Processing Systems*, 37:133345–133385, 2024.

[7] S. Gong, R. Zhang, H. Zheng, J. Gu, N. Jaitly, L. Kong, and Y. Zhang. Diffucoder: Understanding and improving masked diffusion models for code generation. *arXiv preprint arXiv:2506.20639*, 2025.

[8] D. Guo, D. Yang, H. Zhang, J. Song, R. Zhang, R. Xu, Q. Zhu, S. Ma, P. Wang, and X. Bi. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. 2025. arXiv:2501.12948.

[9] P. Holderrieth, M. Havasi, J. Yim, N. Shaul, I. Gat, T. Jaakkola, B. Karrer, R. T. Chen, and Y. Lipman. Generator matching: Generative modeling with arbitrary markov processes. *arXiv preprint arXiv:2410.20587*, 2024.

[10] E. Hoogeboom, D. Nielsen, P. Jaini, P. Forré, and M. Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. In *Neurips*, volume 34, pages 12454–12465, 2021.

[11] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. Lora: Low-rank adaptation of large language models. In *ICLR*, 2022.

[12] Z. Huang, Z. Chen, Z. Wang, T. Li, and G.-J. Qi. Reinforcing the diffusion chain of lateral thought with diffusion language models. 2025. arXiv:2505.10446.

[13] A. Jaech, A. Kalai, A. Lerer, A. Richardson, A. El-Kishky, A. Low, A. Helyar, A. Madry, A. Beutel, and A. Carney. Openai o1 system card. 2024. arXiv:2412.16720.

[14] S. Khanna, S. Kharbanda, S. Li, H. Varma, E. Wang, S. Birnbaum, Z. Luo, Y. Miraoui, A. Palrecha, S. Ermon, et al. Mercury: Ultra-fast language models based on diffusion. *arXiv preprint arXiv:2506.17298*, 2025.

[15] J. Kim, K. Shah, V. Kontonis, S. Kakade, and S. Chen. Train for the worst, plan for the best: Understanding token ordering in masked diffusions. 2025. arXiv:2502.06768.

[16] N. Lambert, J. Morrison, V. Pyatkin, S. Huang, H. Ivison, F. Brahman, L. J. V. Miranda, A. Liu, N. Dziri, and S. Lyu. Tulu 3: Pushing frontiers in open language model post-training. 2024. arXiv:2411.15124.

[17] H. Le, Y. Wang, A. D. Gotmare, S. Savarese, and S. C. H. Hoi. Coderl: Mastering code generation through pretrained models and deep reinforcement learning. In *Neurips*, volume 35, pages 21314–21328, 2022.

[18] A. Lou, C. Meng, and S. Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. 2023. arXiv:2310.16834.

[19] S. Nie, F. Zhu, Z. You, X. Zhang, J. Ou, J. Hu, J. Zhou, Y. Lin, J.-R. Wen, and C. Li. Large language diffusion models. 2025. arXiv:2502.09992.

[20] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[21] Y.-H. Park, C.-H. Lai, S. Hayakawa, Y. Takida, and Y. Mitsufuji. Jump your steps: Optimizing sampling schedule of discrete diffusion models. In *The Thirteenth International Conference on Learning Representations*, 2024.

[22] F. Z. Peng, Z. Bezemek, S. Patel, J. Rector-Brooks, S. Yao, A. J. Bose, A. Tong, and P. Chatterjee. Path planning for masked diffusion model sampling. *arXiv preprint arXiv:2502.03540*, 2025.

[23] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

[24] A. Rastogi, A. Q. Jiang, A. Lo, G. Berrada, G. Lample, J. Rute, J. Barmentlo, K. Yadav, K. Khandelwal, K. R. Chandu, et al. Magistral. *arXiv preprint arXiv:2506.10910*, 2025.

[25] Y. Ren, H. Chen, G. M. Rotskoff, and L. Ying. How discrete and continuous diffusion meet: Comprehensive analysis of discrete diffusion models via a stochastic integral framework. *arXiv preprint arXiv:2410.03601*, 2024.

[26] S. Sahoo, M. Arriola, Y. Schiff, A. Gokaslan, E. Marroquin, J. Chiu, A. Rush, and V. Kuleshov. Simple and effective masked diffusion language models. In *Neurips*, volume 37, pages 130136–130184, 2024.

[27] S. S. Sahoo, J. Deschenaux, A. Gokaslan, G. Wang, J. Chiu, and V. Kuleshov. The diffusion duality. *arXiv preprint arXiv:2506.10892*, 2025.

[28] S. S. Sahoo, Z. Yang, Y. Akhauri, J. Liu, D. Singh, Z. Cheng, Z. Liu, E. Xing, J. Thickstun, and A. Vahdat. Esoteric language models. *arXiv preprint arXiv:2506.01928*, 2025.

[29] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[30] Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, X. Bi, H. Zhang, M. Zhang, Y. Li, and Y. Wu. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. 2024. arXiv:2402.03300.

[31] N. Shaul, I. Gat, M. Havasi, D. Severo, A. Sriram, P. Holderrieth, B. Karrer, Y. Lipman, and R. T. Chen. Flow matching with general discrete paths: A kinetic-optimal perspective. *arXiv preprint arXiv:2412.03487*, 2024.

[32] J. Shi, K. Han, Z. Wang, A. Doucet, and M. Titsias. Simplified and generalized masked diffusion for discrete data. In *Neurips*, volume 37, pages 103131–103167, 2024.

[33] Y. Song, Z. Zhang, C. Luo, P. Gao, F. Xia, H. Luo, Z. Li, Y. Yang, H. Yu, X. Qu, et al. Seed diffusion: A large-scale diffusion language model with high-speed inference. *arXiv preprint arXiv:2508.02193*, 2025.

[34] J. Su, J. Healey, P. Nakov, and C. Cardie. Between underthinking and overthinking: An empirical study of reasoning length and correctness in llms. 2025. arXiv:2505.00127.

[35] Y. Sui, Y.-N. Chuang, G. Wang, J. Zhang, T. Zhang, J. Yuan, H. Liu, A. Wen, S. Zhong, and H. Chen. Stop overthinking: A survey on efficient reasoning for large language models. 2025. arXiv:2503.16419.

[36] R. S. Sutton, A. G. Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[37] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999.

[38] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Neurips*, volume 30, 2017.

[39] G. Wang, Y. Schiff, S. S. Sahoo, and V. Kuleshov. Remasking discrete diffusion models with inference-time scaling. *arXiv preprint arXiv:2503.00307*, 2025.

[40] J. Wang, Y. Lai, A. Li, S. Zhang, J. Sun, N. Kang, C. Wu, Z. Li, and P. Luo. Fudoki: Discrete flow-based unified understanding and generation via kinetic-optimal velocities. *arXiv preprint arXiv:2505.20147*, 2025.

[41] Y. Wang, L. Yang, B. Li, Y. Tian, K. Shen, and M. Wang. Revolutionizing reinforcement learning framework for diffusion large language models. *arXiv preprint arXiv:2509.06949*, 2025.

[42] G. I. Winata, H. Zhao, A. Das, W. Tang, D. D. Yao, S.-X. Zhang, and S. Sahu. Preference tuning with human feedback on language, speech, and vision tasks: A survey. *Journal of Artificial Intelligence Research*, 82:2595–2661, 2025.

[43] C. Wu, H. Zhang, S. Xue, Z. Liu, S. Diao, L. Zhu, P. Luo, S. Han, and E. Xie. Fast-dllm: Training-free acceleration of diffusion llm by enabling kv cache and parallel decoding. *arXiv preprint arXiv:2505.22618*, 2025.

[44] Y. Xu, H. Dong, L. Wang, D. Sahoo, J. Li, and C. Xiong. Scalable chain of thoughts via elastic reasoning. 2025. arXiv:2505.05315.

[45] L. Yang, Y. Tian, B. Li, X. Zhang, K. Shen, Y. Tong, and M. Wang. Mmada: Multimodal large diffusion language models. *arXiv preprint arXiv:2505.15809*, 2025.

[46] J. Ye, S. Gong, L. Chen, L. Zheng, J. Gao, H. Shi, C. Wu, X. Jiang, Z. Li, W. Bi, et al. Diffusion of thought: Chain-of-thought reasoning in diffusion language models. *Advances in Neural Information Processing Systems*, 37:105345–105374, 2024.

[47] J. Ye, Z. Xie, L. Zheng, J. Gao, Z. Wu, X. Jiang, Z. Li, and L. Kong. Dream 7b: Diffusion large language models. 2025. arXiv:2508.15487.

[48] O. Zekri and N. Boullé. Fine-tuning discrete diffusion models with policy gradient methods. *arXiv preprint arXiv:2502.01384*, 2025.

[49] K. Zhang, Y. Zuo, B. He, Y. Sun, R. Liu, C. Jiang, Y. Fan, K. Tian, G. Jia, P. Li, et al. A survey of reinforcement learning for large reasoning models. *arXiv preprint arXiv:2509.08827*, 2025.

[50] S. Zhao, D. Gupta, Q. Zheng, and A. Grover. d1: Scaling reasoning in diffusion large language models via reinforcement learning. 2025. arXiv:2504.12216.

[51] L. Zheng, J. Yuan, L. Yu, and L. Kong. A reparameterized discrete diffusion model for text generation. *arXiv preprint arXiv:2302.05737*, 2023.

[52] F. Zhu, R. Wang, S. Nie, X. Zhang, C. Wu, J. Hu, J. Zhou, J. Chen, Y. Lin, J.-R. Wen, et al. Llada 1.5: Variance-reduced preference optimization for large language diffusion models. *arXiv preprint arXiv:2505.19223*, 2025.