DiRL: An Efficient Post-Training Framework for Diffusion Language Models

Ying Zhu[1,2,3], **Jiaxin Wan**[2], **Xiaoran Liu**[1,2,3], **Siyanag He**[1,2,3], **Qiqi Wang**[1,2,3], **Xu Guo**[1,2], **Tianyi Liang**[2,3], **Zengfeng Huang**[1,2], **Ziwei He**[2,3†], **Xipeng Qiu**[1,2†]

[1]Fudan University, [2]Shanghai Innovation Institute, [3]OpenMoss Team

[†]Corresponding author.

Abstract

Diffusion Language Models (dLLMs) have emerged as promising alternatives to Auto-Regressive (AR) models. While recent efforts have validated their pre-training potential and accelerated inference speeds, the post-training landscape for dLLMs remains underdeveloped. Existing methods suffer from computational inefficiency and objective mismatches between training and inference, severely limiting performance on complex reasoning tasks such as mathematics. To address this, we introduce DiRL, an efficient post-training framework that tightly integrates FlexAttention-accelerated blockwise training with LMDeploy-optimized inference. This architecture enables a streamlined online model update loop, facilitating efficient two-stage post-training (Supervised Fine-Tuning followed by Reinforcement Learning). Building on this framework, we propose DiPO, the first unbiased Group Relative Policy Optimization (GRPO) implementation tailored for dLLMs. We validate our approach by training DiRL-8B-Instruct on high-quality math data. Our model achieves state-of-the-art math performance among dLLMs and surpasses comparable models in the Qwen2.5 series on several benchmarks.

Code: https://github.com/OpenMOSS/DiRL
Model: https://huggingface.co/OpenMOSS-Team/DiRL-8B-Instruct

1 Introduction

Large Diffusion Language Models (dLLMs) have become a hot topic in NLP (Nie et al., 2025; Inception, 2025). The emergence of dLLMs such as LLaDA (Nie et al., 2025), Dream (Ye et al., 2025), Mercury (Inception, 2025) and Gemini Diffusion (Gemini, 2025), together with blockwise hybrids like SDAR (Cheng et al., 2025) that combine diffusion with traditional auto-regression (AR), confirms the scalability of this paradigm (Nie et al., 2024; Gong et al., 2024; Ni et al., 2025). Building on these results, a growing body of work now seeks to advance their performance in multi-modality (Yang et al., 2025; You et al., 2025), long-context modeling (Liu et al., 2025b; He et al., 2025) and inference efficiency (Wu et al., 2025b;a; Song et al., 2025). Although pre-training

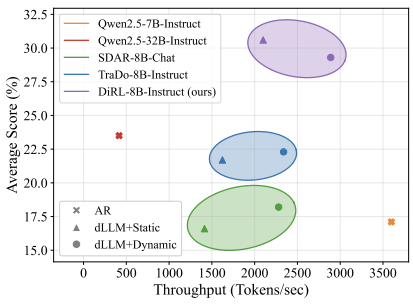


Figure 1: Performance of DiRL-8B-Instruct.

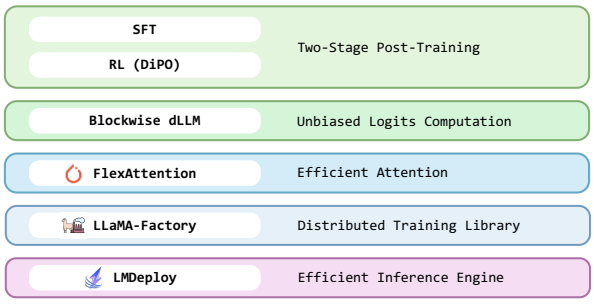


Figure 2: Features of our DiRL framework.

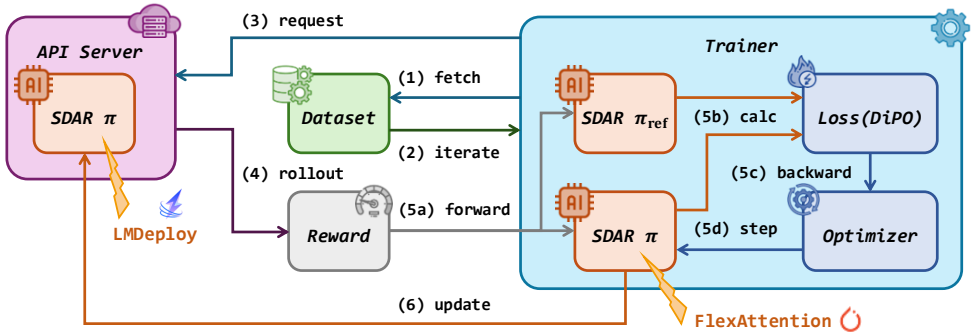


Figure 3: Overview of the RL pipeline in our DiRL framework.

of dLLMs is now proven feasible, post-training of dLLMs, especially reinforcement learning (RL), remains underdeveloped, limiting dLLMs' performance on math tasks and real-world deployment.

The difficulty of dLLM post-training, especially RL, lies in the fact that the logits and derived policy cannot be computed exactly (Zhao et al., 2025a; Zhu et al., 2025b). In original fully bidirectional dLLMs, the generation order is unconstrained, making teacher-forcing style logit acquisition during SFT infeasible. Injecting uniform random noise into the output fails to reproduce the realistic inference step map, resulting in biased logits and a large mismatch between training and inference objectives (Zhao et al., 2025a; Wang et al., 2025b). Furthermore, in the RL stage, the absence of a KV cache further increases computational overhead (Liu et al.; Ma et al., 2025; Song et al., 2025). Most existing dLLM-based RL efforts lack an inference-engine backend, efficient training–inference co-design, and fast rollouts with online model updates, which prevents the practical adoption of mature RL algorithms such as GRPO (DeepSeek-AI, 2024; Guo et al., 2025). Blockwise dLLMs partially alleviate these issues by restricting generation within blocks, enabling exact logit computation through blockwise forward passes (Cheng et al., 2025; Wang et al., 2025b). However, they do not fully resolve the train–inference mismatch in post-training, nor do they address the efficiency and algorithmic challenges of dLLM RL. How to achieve consistent training and inference while enabling scalable RL for dLLMs remains underexplored. To fill this gap, we introduce our dLLM RL algorithm ***DiPO***, together with the training framework ***DiRL***, which enforces training–inference consistency and enables efficient rollouts and policy optimization for blockwise dLLMs. We further present the state-of-the-art dLLM ***DiRL-8B-Instruct***, as shown in Figure 1, Figure 2 and Figure 3.

At the algorithm level, ***DiPO*** leverages the good property of blockwise dLLM to achieve the first unbiased GRPO implementation for dLLMs through efficient, unbiased logit computation. At the framework level, ***DiRL*** supports the two-stage (SFT-RL) post-training of dLLMs, aligning training and inference objectives while surpassing existing methods in efficiency as shown in Figure 2. Concretely, we integrate the efficient inference property of blockwise dLLMs with the efficient FlexAttention interface (Dong et al., 2024) and LMDeploy framework (InternLM, 2023) to enable fast rollout and online model updates in the API server. At the model level, based on high-quality math datasets, we train ***DiRL-8B-Instruct*** from SDAR-8B-Chat and achieve best performance on math tasks in dLLMs, even outperform Qwen2.5 Series (Qwen et al., 2024), the widely-acknowledged larger AR model, in AIME24, AIME25 (MAA, 2024; 2025) and OlympiadBench (He et al., 2024) as shown in Figure 1. Our contributions can be summarized as follows.

- ***DiRL***, an efficient post-training framework for dLLMs that replaces offline model loading with inference-server-based rollouts and online policy updates, ensuring training–inference consistency and accelerated by FlexAttention.
- ***DiPO***, the first unbiased GRPO implementation in dLLMs, leveraging the unbiased logits computation of blockwise dLLM.
- ***DiRL-8B-Instruct***, the state-of-the-art dLLMs in math tasks, based on the above algorithm and engineering improvements, as well as high-quality math data.

2 Preliminary

2.1 Block Diffusion Language Model

Blockwise Diffusion Language Models, as exemplified by BD3-LMs (Arriola et al., 2025) and SDAR (Cheng et al., 2025), primarily unify the global sequential dependency of AR models with the local parallel generation capability of dLLMs through a Semi-Autoregressive generation paradigm.

In blockwise dLLMs, given a discrete sequence $\mathbf{x}$, we partition it into K non-overlapping text blocks, denoted as $\mathbf{x} = (\mathbf{b}^1, \mathbf{b}^2, \ldots, \mathbf{b}^K)$, where each block contains B tokens. Then the joint probability distribution of the sequence $p_\theta(\mathbf{x})$ is factorized into a product of conditional probabilities, where $\mathbf{b}^{<k}$ represents the historical context preceding the current block.

$$\log p_\theta(\mathbf{x}) = \sum_{k=1}^{K} \log p_\theta(\mathbf{b}^k \mid \mathbf{b}^{<k}) \tag{1}$$

In contrast to the token-by-token generation of AR, the intra-block conditional distribution $p_\theta(\mathbf{b}^k \mid \mathbf{b}^{<k})$ is modeled by a discrete masked diffusion process conditioned on the historical information.

Forward Process (Noising Process) A Markov chain $q(\mathbf{b}_t^k \mid \mathbf{b}_0^k)$ is defined, where as the time step $t \in [0, 1]$ increases, tokens in the current block $\mathbf{b}_0^k$ are progressively replaced by the `[MASK]` token with a probability of $1 - \alpha_t$. The noising process only acts on the current block $\mathbf{b}^k$.

Reverse Process (Denoising Process) Given the historical context $\mathbf{b}^{<k}$ and the current noisy state $\mathbf{b}_t^k$, the model predicts the original state of the current block in parallel:

$$p_\theta(\mathbf{b}_0^k \mid \mathbf{b}_t^k, \mathbf{b}^{<k}) = \prod_{j=1}^{B} p_\theta((\mathbf{b}_0^k)_j \mid \mathbf{b}_t^k, \mathbf{b}^{<k}) \tag{2}$$

As for the optimization objective, the training of dLLMs typically involves maximizing the Evidence Lower Bound (ELBO). In the blockwise diffusion framework, this translates to minimizing the Conditional Negative Evidence Lower Bound (NELBO) over all blocks, where $w(t)$ is a time-dependent weighting coefficient, and CE denotes the cross-entropy loss.

$$\mathcal{L}_{\text{BD}}(\theta) = \sum_{k=1}^{K} \mathbb{E}_{t, \mathbf{b}_0^k} \left[w(t) \cdot \sum_{j=1}^{B} \mathbb{1}[(\mathbf{b}_t^k)_j = \texttt{[MASK]}] \cdot \text{CE}\left(p_\theta(\cdot \mid \mathbf{b}_t^k, \mathbf{b}^{<k}), (\mathbf{b}_0^k)_j \right) \right] \tag{3}$$

Thanks to the combination of intra-block AR and inner-block diffusion, the KV cache mechanism can be applied to blockwise dLLMs, thus enhancing the computational efficiency of traditional dLLMs and simplifying the likelihood computation (Cheng et al., 2025; Wang et al., 2025a).

2.2 RL and its application in dLLM

Reinforcement Learning (RL) is a key training paradigm in the alignment stage of LLMs (OpenAI, 2024; Guo et al., 2025; OpenR1, 2025). In particular, Group Relative Policy Optimization (GRPO) (DeepSeek-AI, 2024; Guo et al., 2025) has been the RL algorithm mostly used on AR. It replaces the critic model in PPO (Schulman et al., 2017) by using diversified sampling and within-group normalized advantages, enabling lightweight online RL and steering LLM toward outputs that are better aligned with human preferences. Specifically, its advantage calculation is as follows. First, for a prompt q, sample G outputs $\{o_1, \ldots, o_G\}$ from the old policy $\pi_{\theta_{\text{old}}}$. Let the reward of completion o_i be r_i. The group-normalized advantage is defined as $A_i = r_i - \frac{1}{G}\sum_{j=1}^{G} r_j$, and its token-level assignment is simply $A_{i,k} = A_i$, where $k = 1, \ldots, |o_i|$. Since AR models provide per-token conditional probabilities, the importance sampling ratio for each token is

$$\rho_{i,k} = \frac{\pi_\theta(o_{i,k} \mid q, o_{i,<k})}{\pi_{\theta_{\text{old}}}(o_{i,k} \mid q, o_{i,<k})}. \tag{4}$$

Then the GRPO objective with clipping and reverse KL regularization is defined as follows:

$$\mathcal{L}_{\text{GRPO}}(\theta) = \mathbb{E}_{q, o_{1:G}} \left[\frac{1}{G} \sum_{i=1}^{G} \frac{1}{|o_i|} \sum_{k=1}^{|o_i|} \min\Big(\rho_{i,k} A_{i,k}, \text{clip}(\rho_{i,k}, 1-\epsilon, 1+\epsilon) A_{i,k} \Big) \right] - \beta D_{\text{KL}}(\pi_\theta \,\|\, \pi_{\text{ref}}). \tag{5}$$

However, RL algorithms such as GRPO are difficult to transfer to dLLM, because dLLMs do not provide per-token probabilities directly, and their log-probability estimates require multi-step sampling, resulting in high computational cost and high variance. In addition, the random masking step map used in pre-training is also mismatched with the decoding trajectory in realistic inference.

Thanks to the introduction of blockwise dLLMs (Cheng et al., 2025; Wang et al., 2025a), the computational overhead of log-probability is greatly reduced. Building on this, Wang et al. (2025b) proposes TraceRL, which leverages the algorithmic efficiency of blockwise dLLMs to enable faster unbiased logit computation. However, due to the lack of attention acceleration during training and the absence of tight integration between training and inference frameworks, TraceRL still leaves significant room for engineering optimization. Moreover, it does not implement more fine-grained RL algorithms such as GRPO. To address these limitations, we introduce our efficient training framework DiRL and DiPO, the first unbiased implementation of GRPO for dLLMs.

3 DiRL Post-Training

Using the DiRL framework, we conduct two-stage post-training, SFT followed by RL based on our DIPO, achieve sota dLLM, DiRL-8B-Instruct.

3.1 SFT stage

SFT Data We train on the OpenR1-Math dataset (OpenR1, 2025) distilled from GLM-4.6 (Zeng et al., 2025a; zai org, 2025). We chose GLM-4.6 because currently, it is almost the best-performing open-source non-reasoning LLM on math tasks and yields reasoning trajectories of manageable length. As long-reasoning evaluation for diffusion models is still scarce, we cap the reasoning length at 8k token length, which is already the longest inference length reported for dLLMs.

SFT setup SFT is conducted with LLaMA-Factory (Zheng et al., 2024). 8×H200 GPUs are applied to fine-tune with SDAR-8B-Chat with DeepSpeed ZeRO1 (Rajbhandari et al., 2020). Models are fine-tuned with a maximum length of 8k tokens. We set the global batch size to 512, the maximum learning rate to 1e-5, and the weight decay to 0, and fine-tune it in 100 steps with a cosine annealing learning rate scheduler.

3.2 RL stage

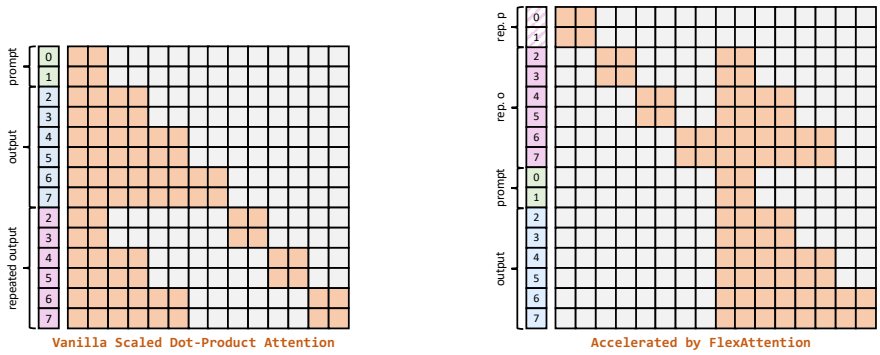
DiPO Our optimization objective is defined as:

$$J_{\text{policy}}(\theta_p) = \mathbb{E}_{Q\sim\mathcal{D}_{\text{task}},\{\tau_i\}_{i=1}^{G}\sim\pi_{\text{old}}(\cdot|Q)} \left[\sum_{i=1}^{G} \frac{1}{|\tau_i|} \sum_{t=1}^{|\tau_i|} \sum_{o_k\in\tau_i(t)} C_\epsilon\left(\frac{\pi_{\theta_p}(o_k \mid \tau_i(1{:}t-1))}{\pi_{\text{old}}(o_k \mid \tau_i(1{:}t-1))}, A_i \right) - \beta\,\mathrm{KL}\Big[\pi_{\theta_p} \,\|\, \pi_{\text{ref}}\Big] \right], \tag{6}$$

where we assume that each trajectory requires t decoding steps, and $\tau_i(1{:}t-1) \triangleq \bigcup_{j=1}^{t-1} \tau_i(j)$ denotes the prefix up to step $t-1$, namely tokens decoded by timestep $t-1$. To maintain stable learning, we use the clipping operator $C_\epsilon(r, A) \triangleq \min\left(rA,\ \mathrm{clip}(r,\,1-\epsilon,\,1+\epsilon)\,A\right)$, which limits how much the updated policy can deviate from the behavior policy at each step. The term A_i stands for the normalized advantage assigned to the i-th trajectory, while π_{old} represents the policy that produced the samples. In addition to the clipped surrogate, we also incorporate a KL penalty with respect to a fixed reference policy π_{ref}, rather than the behavior policy, so that the learned model does not drift too far away from the underlying reference model. The coefficient β controls the strength of this regularization term.

Besides, since our framework performs online updates, the behavior policy π_{old} is implemented as the detached output of the current policy π_{θ_p} in the optimization objective:

$$J_{\text{policy}}(\theta_p) = \mathbb{E}_{Q\sim\mathcal{D}_{\text{task}},\{\tau_i\}_{i=1}^{G}\sim\pi_{\text{old}}(\cdot|Q)} \left[\frac{1}{\sum_{i=1}^{G}|\tau_i|} \sum_{i=1}^{G} \sum_{t=1}^{|\tau_i|} \sum_{o_k\in\tau_i(t)} C_\epsilon\left(\frac{\pi_{\theta_p}(o_k \mid \tau_i(1{:}t-1))}{\mathrm{sg}(\pi_{\theta_p}(o_k \mid \tau_i(1{:}t-1)))}, A_i \right) - \beta\,\mathrm{KL}\Big[\pi_{\theta_p} \,\|\, \pi_{\text{ref}}\Big] \right]. \tag{7}$$



(a) Attention mask in TraceRL. **(b)** Attention mask in our DiRL.

Figure 4: The visualized comparison of the attention mask between our DiRL framework and TraceRL, where block size is 2, the length of prompt colored in green is 2, and the length of output colored in blue is 6. The loss is calculated based on the repeated output part colored in full purple.

Here $\mathrm{sg}[\cdot]$ denotes the stop-gradient operator (equivalently `.detach()` in PyTorch).

Moreover, we integrate the DAPO(Yu et al., 2025) algorithm, which uses the token-level policy gradient:

$$J_{\text{policy}}(\theta_p) = \mathbb{E}_{Q\sim\mathcal{D}_{\text{task}},\{\tau_i\}_{i=1}^{G}\sim\pi_{\text{old}}(\cdot|Q)} \left[\frac{1}{\sum_{i=1}^{G}|\tau_i|}\sum_{i=1}^{G}\sum_{t=1}^{|\tau_i|} C_\epsilon\left(\frac{\pi_{\theta_p}(\tau_i(t)\mid\tau_i(1{:}t-1))}{sg(\pi_{\theta_p}(\tau_i(t)\mid\tau_i(1{:}t-1)))}, A_i\right)\right]. \tag{8}$$

RL Data We train on the Big-Math dataset (Albalak et al., 2025), a large-scale, high-quality math dataset verfied by math-verify, and specially designed for RL.

RL setup RL is conducted after SFT with **DiRL** supported by Accelerate. 128×H200 GPUs are applied to train the model with DeepSpeed ZeRO1. Models are fine-tuned with a maximum length of 8k tokens. We set the global batch size to 128, and rollout 32 trajectories for each problem. The learning rate is set to 1e-6, and the weight decay to 0, and train it in 40 steps.

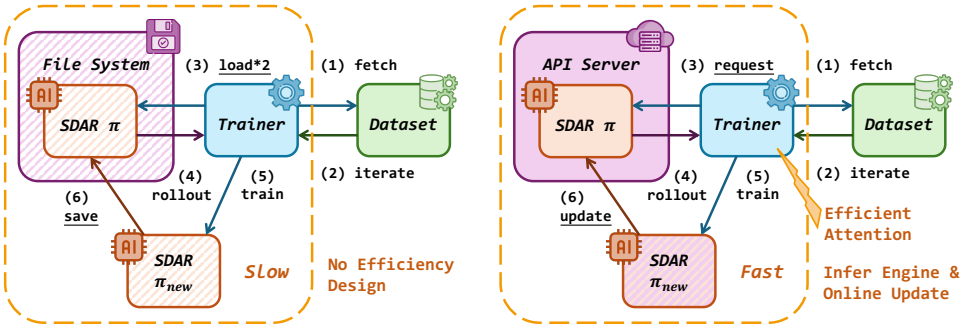
4 Engineering Optimization

As shown in Figure 2, the DiRL framework incorporates efficiency designs at both the model and system levels. At the model level, blockwise dLLM (Cheng et al., 2025) with FlexAttention (Dong et al., 2024) compute unbiased logits efficiently, and at the framework level, tight integration of the LMDeploy inference engine (InternLM, 2023) with the open-source distributed training library LLaMA-Factory (Zheng et al., 2024) enables online model updates for highly efficient training.

4.1 Blockwise dLLM with FlexAttention

At the model level, as mentioned earlier, we use blockwise dLLM, which is more efficient for logic computation than the original dLLMs in post-training. This is because it maintains an AR paradigm between blocks, performing denoising only within each block. Thanks to this design, as Figure 4a shows, TraceRL (Wang et al., 2025b) can parallelize denoising across blocks by repeating the output twice and altering the attention mask, thus enabling faster SFT training.

However, the dominant FlashAttention interface (Dao et al., 2022; Dao, 2023) cannot handle the complex attention masks required by blocksize dLLMs, limiting the practical training throughput. Fortunately, PyTorch's FlexAttention (Dong et al., 2024) fills this gap with an efficient operator that accepts fine-grained masks. DiRL does not differentiate between the prompt and output, repeats both parts blockwise, and reshapes the SFT mask, as illustrated in Figure 4b. Based on a more regular attention mask, we combines it with FlexAttention, cutting post-training latency dramatically.



(a) RL in current dLLMs. **(b)** RL in our DiRL Framework.

Figure 5: The training and inference integration of our DiRL compared to RL in current dLLMs.

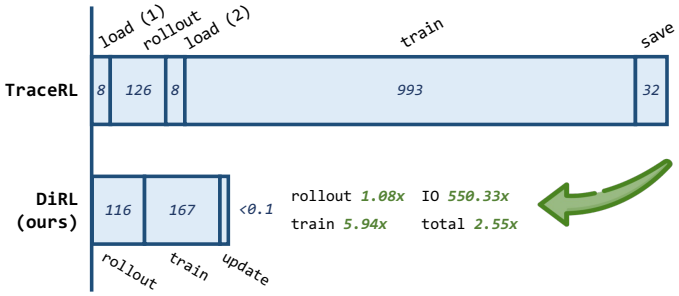


Figure 6: Comparison of time breakdown per RL training step for SDAR-8B-Chat, measured in seconds. The comparison is conducted on 8×H200 GPUs with batch size 4, input length 1024, and output length 8192.

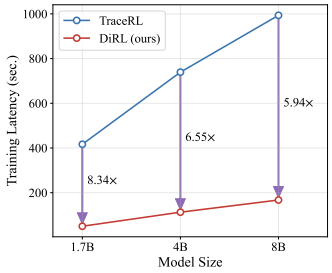


Figure 7: Reduction of training latency of our DiRL across different sizes of SDAR models.

4.2 Training and Inference Integration

Existing RL efforts in dLLMs (Wang et al., 2025b), as noted earlier, lack inference-engine support and co-design optimization. Concretely, Figure 5a illustrates the online-learning loop in existing dLLM-oriented RL. Each training step fetches new data, loads the previous checkpoint for inference, conducts rollout, loads the checkpoint twice for training, trains, and then saves the updated checkpoint back to the file system for reloading in the next step, resulting in an evident waste of IO interactions. Moreover, in earlier works (Zhao et al., 2025a; Zhu et al., 2025b), without the support of an inference engine, the inference of dLLM is slow, further amplifying the overall training inefficiency.

Compared with existing approaches, DiRL tightly couples training and inference as shown in Figure 5b. We first deploy the model in an API server through LMDeploy (InternLM, 2023), which is the only loading operation in the whole training process. Leveraging LMDeploy's in-place parameter-update API, we immediately push each training-step checkpoint into the server, eliminating IO between the file system while keeping the API server alive. As shown in Figure 6, these refinements cut per-step latency sharply and dramatically boost the efficiency of dLLM-oriented RL.

4.3 Efficiency Validation

We compare the per-step time breakdown of DiRL and TraceRL on 8×H200 GPUs with batch size 4, input length 1024 and output length 8192 in Figure 6. Since TraceRL also uses the JetEngine rollout backend (Cheng et al., 2025), the speed-up in that operation is the modest. However, FlexAttention-accelerated training reduces latency by nearly 6×. Furthermore, Figure 7 shows that this gain persists across model sizes. 8B training latency of our DiRL per step is lower than 1.7B latency of TraceRL. Moreover, replacing two model loads and one save with an almost cost-free in-place update yields an overall 2.5× throughput improvement.

4.4 Main Results

We evaluate our DiRL-8B-Instruct with other baselines on five representative math tasks, MATH500 (Hendrycks et al., 2021), GSM8k (Cobbe et al., 2021), AIME2024 (MAA, 2024),

	MATH500			GSM8k			AIME2024			AIME2025			Olympiad			Avg.
Qwen2.5-7B-Instruct	73.8	1.0	628.1	89.8	1.0	308.9	9.0	1.0	1005.5	5.6	1.0	970.5	36.6	1.0	862.0	42.9
Qwen2.5-32B-Instruct	81.1	1.0	554.6	**94.0**	1.0	291.2	12.9	1.0	831.7	11.9	1.0	839.7	45.7	1.0	742.4	49.1
SDAR-8B-Chat	71.5	1.0	603.4	89.5	1.0	712.9	5.6	1.0	1342.4	8.5	1.0	920.8	35.6	1.0	890.4	42.2
+ Dynamic	71.9	2.4	616.6	89.9	2.8	708.8	9.2	2.4	1274.2	9.4	2.1	889.8	36.0	2.4	896.4	43.3
TraDo-8B-Instruct	76.7	1.0	618.1	90.4	1.0	324.7	11.5	1.0	1036.1	13.5	1.0	988.0	40.2	1.0	864.3	46.5
+ Dynamic	75.6	2.3	618.6	91.1	2.1	315.2	11.7	2.0	1091.4	15.0	1.9	996.2	40.3	2.1	868.0	46.7
DiRL-8B-Instruct	**85.1**	1.0	1917.2	<u>93.1</u>	1.0	707.4	**21.5**	1.0	5434.2	**22.9**	1.0	5019.0	**47.3**	1.0	3556.4	**54.0**
+ Dynamic	<u>83.1</u>	2.0	2000.7	93.0	2.2	730.1	<u>20.6</u>	1.7	5468.5	<u>20.8</u>	1.7	5129.6	<u>46.4</u>	1.8	3614.5	<u>52.8</u>

Table 1: Comprehensive benchmark results of our DiRL-8B-Instruct compared with current dLLMs, SDAR-8B-Chat (Cheng et al., 2025) and TraDo-8B-Instruct (Wang et al., 2025b), as well as well-acknowledged AR model Qwen2.5 Series (Qwen et al., 2024). Each cell presents the accuracy, with best values in bold and suboptimal values underlined, as well as the average number of decoding tokens per step and the average output lengths.

AIME2025 (MAA, 2025), OlympiadBench (He et al., 2024). Baselines include the blockwise dLLMs SDAR-8B-Chat (Cheng et al., 2025) and TraDo-8B-Instruct (Wang et al., 2025b). We report results for both static and dynamic decoding. For the latter, we set a threshold of 0.9, decoding tokens whose top-1 probability exceeds 0.9 directly. As an external reference, we also list the widely used autoregressive LLM Qwen2.5 Series (Qwen et al., 2024). The results are shown in Table 1.

Our DiRL-8B-Instruct achieves the best result on the average score and the majority of tasks, outperforming blockwise dLLMs and AR models with the same model size, even exceeding the larger Qwen2.5-32B-Instruct, and establishing a new state-of-the-art dLLM. Comparing average reasoning lengths across tasks, DiRL-8B-Instruct produces the longest output, indicating that the two-stage SFT+RL post-training equips it with stronger math reasoning capability and enables more complex math derivations that yield superior performance.

5 Discussion

5.1 Ablation Study

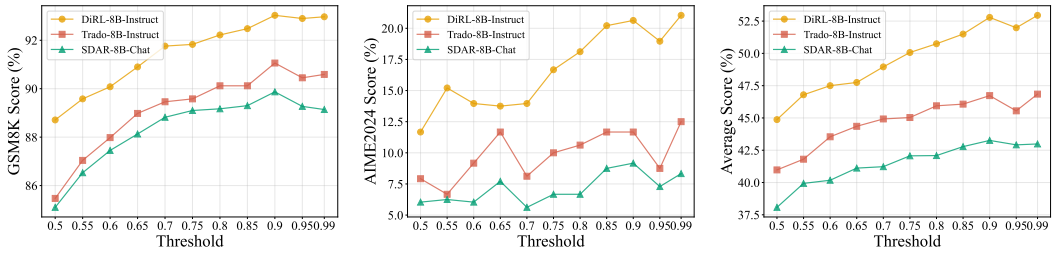
To investigate the sensitivity of the models to hyperparameters within the dynamic decoding strategy, we perform an ablation study on the dynamic sampling threshold τ, adjusting it within the range of 0.5 to 0.99, and record the accuracy and average scores of DiRL-8B-Instruct, TraDo-8B-Instruct (Wang et al., 2025b), and SDAR-8B-Chat (Cheng et al., 2025) on five mathematical benchmarks (as illustrated in Figure 8).

Experimental results indicate that regardless of the variation in τ, DiRL-8B-Instruct consistently and significantly outperforms the baseline models in all individual tasks and average performance, demonstrating strong robustness under varying degrees of aggressive decoding. Specifically, as the threshold increases (implying a more conservative decoding approach closer to greedy search), the performance of DiRL-8B-Instruct exhibits a steady upward trend. In contrast, the responses of other dLLMs are suboptimal. SDAR-8B-Chat displays performance plateau across most tasks, suggesting that mere threshold adjustment cannot compensate for its inherent reasoning deficiencies, while TraDo-8B-Instruct, despite acceptable performance on GSM8K 8a, suffers from significant fluctuations in tasks such as AIME2024 8b, indicating a lack of stability.

These findings not only confirm that the superior mathematical reasoning capacity of DiRL-8B-Instruct stems from the high-quality reasoning paths derived from the two-stage SFT+RL post-training rather than the dependency on specific hyperparameter configurations, but also reveal that $\tau = 0.9$ serves as the optimal equilibrium point for average performance across models, thus further validating the rationality of parameter selection in our main experiments.

5.2 Future work

Despite the clear gain in training efficiency and the current best-result at 8B scale, several improvements remain for our future work. First, we will scale the approach to larger dLLMs to pursue even stronger performance. Second, although our 8k inference length is already the longest reported for dLLMs, it is still short compared with AR models (Guo et al., 2025; OpenAI, 2024). We have not yet explored test-time scaling or long CoT techniques that could raise the ceiling of model itself (Liu



(a) GSM8K **(b)** AIME2024 **(c)** Average

Figure 8: Ablation study on dynamic sampling threshold

et al., 2025a; Zeng et al., 2024; 2025b). Third, we will incorporate strategies such as dynamic packing (Bai et al., 2024) from AR long-context training to further accelerate our framework. Finally, though currently focused on math tasks, we will extend evaluation to agentic and code tasks (Zeng et al., 2025a), striving to match or surpass AR models across a broader spectrum.

6 Conclusion

In this work, we introduce DiRL, an efficient framework for SFT and RL in dLLMs. By combining FlexAttention with refined attention masks, we remarkably reduce the training latency of blockwise dLLMs. By applying LMDeploy and online parameter updates, we achieve an unbiased logits computation and efficient two-stage post-training. Building upon this foundation, we propose DiPO, the first unbiased GRPO implementation for dLLMs, and train DiRL-8B-Instruct, the currently strongest dLLM on math tasks, even outperforming the Qwen2.5 series. We will continue to refine our framework for higher training efficiency and validate our approach on larger models, longer reasoning chains, and a broader downstream tasks, aiming to provide the community with a solid baseline and an open-source post-training toolkit for dLLMs.

References

Alon Albalak, Duy Phung, Nathan Lile, Rafael Rafailov, Kanishk Gandhi, Louis Castricato, Anikait Singh, Chase Blagden, Violet Xiang, Dakota Mahan, et al. Big-math: A large-scale, high-quality math dataset for reinforcement learning in language models. *arXiv preprint arXiv:2502.17387*, 2025.

Marianne Arriola, Aaron Gokaslan, Justin T Chiu, Zhihan Yang, Zhixuan Qi, Jiaqi Han, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Block diffusion: Interpolating between autoregressive and diffusion language models. *arXiv preprint arXiv:2503.09573*, 2025.

Yushi Bai, Xin Lv, Jiajie Zhang, Yuze He, Ji Qi, Lei Hou, Jie Tang, Yuxiao Dong, and Juanzi Li. Longalign: A recipe for long context alignment of large language models. *arXiv preprint arXiv:2401.18058*, 2024.

Shuang Cheng, Yihan Bian, Dawei Liu, Linfeng Zhang, Qian Yao, Zhongbo Tian, Wenhai Wang, Qipeng Guo, Kai Chen, Biqing Qi, et al. Sdar: A synergistic diffusion-autoregression paradigm for scalable sequence generation. *arXiv preprint arXiv:2510.06303*, 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. *CoRR*, abs/2307.08691, 2023. doi: 10.48550/ARXIV.2307.08691. URL https://doi.org/10.48550/arXiv.2307.08691.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in*

Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/67d57c32e20fd0a7a302cb81d36e40d5-Abstract-Conference.html.

DeepSeek-AI. Deepseek-v3 technical report, 2024. URL https://github.com/deepseek-ai/DeepSeek-V3/blob/main/DeepSeek_V3.pdf. Accessed: 2024-12-26.

Juechu Dong, Boyuan Feng, Driss Guessous, Yanbo Liang, and Horace He. Flex attention: A programming model for generating optimized attention kernels. *arXiv preprint arXiv:2412.05496*, 2024.

Gemini. Gemini diffusion, our state-of-the-art, experimental text diffusion model, 2025. URL https://deepmind.google/models/gemini-diffusion/.

Shansan Gong, Shivam Agarwal, Yizhe Zhang, Jiacheng Ye, Lin Zheng, Mukai Li, Chenxin An, Peilin Zhao, Wei Bi, Jiawei Han, et al. Scaling diffusion language models via adaptation from autoregressive models. *arXiv preprint arXiv:2410.17891*, 2024.

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, 2025. URL https://arxiv.org/abs/2506.20639.

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*, 2025.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3828–3850, 2024.

Guangxin He, Shen Nie, Fengqi Zhu, Yuankang Zhao, Tianyi Bai, Ran Yan, Jie Fu, Chongxuan Li, and Binhang Yuan. Ultrallada: Scaling the context length to 128k for diffusion large language models. *arXiv preprint arXiv:2510.10481*, 2025.

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

Inception. Introducing mercury, the world's first commercial-scale diffusion language model, 2025. URL https://www.inceptionlabs.ai/introducing-mercury.

InternLM. Lmdeploy: A toolkit for compressing, deploying, and serving llm. https://github.com/InternLM/lmdeploy, 2023.

Nianyi Lin, Jiajie Zhang, Lei Hou, and Juanzi Li. Boundary-guided policy optimization for memory-efficient rl of diffusion large language models, 2025. URL https://arxiv.org/abs/2510.11683.

Xiaoran Liu, Ruixiao Li, Mianqiu Huang, Zhigeng Liu, Yuerong Song, Qipeng Guo, Siyang He, Qiqi Wang, Linlin Li, Qun Liu, et al. Thus spake long-context large language model. *arXiv preprint arXiv:2502.17129*, 2025a.

Xiaoran Liu, Yuerong Song, Zhigeng Liu, Zengfeng Huang, Qipeng Guo, Ziwei He, and Xipeng Qiu. Longllada: Unlocking long context capabilities in diffusion llms. *arXiv preprint arXiv:2506.14429*, 2025b.

Zhiyuan Liu, Yicun Yang, Yaojie Zhang, Junjie Chen, Chang Zou, Qingyan Wei, Shaobo Wang, and Linfeng Zhang. dllm-cache: Accelerating diffusion large language models with adaptive caching.

Xinyin Ma, Runpeng Yu, Gongfan Fang, and Xinchao Wang. dkv-cache: The cache for diffusion language models. *arXiv preprint arXiv:2505.15781*, 2025.

MAA. American invitational mathematics examination-aime 2024, 2024. URL https://huggingface.co/datasets/math-ai/aime24.

MAA. American invitational mathematics examination-aime 2024, 2025. URL https://huggingface.co/datasets/math-ai/aime25.

Jinjie Ni, Qian Liu, Chao Du, Longxu Dou, Hang Yan, Zili Wang, Tianyu Pang, and Michael Qizhe Shieh. Training optimal large diffusion language models. *arXiv preprint arXiv:2510.03280*, 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. *arXiv preprint arXiv:2410.18514*, 2024.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. *arXiv preprint arXiv:2502.09992*, 2025.

OpenAI. O1: Openai's first model, 2024. URL https://openai.com/o1/. Accessed: 2024-12-25.

OpenR1. Openr1 math 220k, 2025. URL https://huggingface.co/datasets/open-r1/OpenR1-Math-220k.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024. URL https://arxiv.org/abs/2412.15115.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: memory optimizations toward training trillion parameter models. In Christine Cuicchi, Irene Qualters, and William T. Kramer (eds.), *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, SC 2020, Virtual Event / Atlanta, Georgia, USA, November 9-19, 2020*, pp. 20. IEEE/ACM, 2020. doi: 10.1109/SC41405.2020.00024. URL https://doi.org/10.1109/SC41405.2020.00024.

Kevin Rojas, Jiahe Lin, Kashif Rasul, Anderson Schneider, Yuriy Nevmyvaka, Molei Tao, and Wei Deng. Improving reasoning for diffusion language models via group diffusion policy optimization, 2025. URL https://arxiv.org/abs/2510.08554.

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

Yuerong Song, Xiaoran Liu, Ruixiao Li, Zhigeng Liu, Zengfeng Huang, Qipeng Guo, Ziwei He, and Xipeng Qiu. Sparse-dllm: Accelerating diffusion llms with dynamic cache eviction. *arXiv preprint arXiv:2508.02558*, 2025.

Xu Wang, Chenkai Xu, Yijie Jin, Jiachun Jin, Hao Zhang, and Zhijie Deng. Diffusion llms can do faster-than-ar inference via discrete diffusion forcing. *arXiv preprint arXiv:2508.09192*, 2025a.

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.

Yinjie Wang, Ling Yang, Bowen Li, Ye Tian, Ke Shen, and Mengdi Wang. Revolutionizing reinforcement learning framework for diffusion large language models, 2025c. URL https://arxiv.org/abs/2509.06949.

Chengyue Wu, Hao Zhang, Shuchen Xue, Shizhe Diao, Yonggan Fu, Zhijian Liu, Pavlo Molchanov, Ping Luo, Song Han, and Enze Xie. Fast-dllm v2: Efficient block-diffusion llm. *arXiv preprint arXiv:2509.26328*, 2025a.

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*, 2025b.

Ling Yang, Ye Tian, Bowen Li, Xinchen Zhang, Ke Shen, Yunhai Tong, and Mengdi Wang. Mmada: Multimodal large diffusion language models. *arXiv preprint arXiv:2505.15809*, 2025.

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.

Zebin You, Shen Nie, Xiaolu Zhang, Jun Hu, Jun Zhou, Zhiwu Lu, Ji-Rong Wen, and Chongxuan Li. Llada-v: Large language diffusion models with visual instruction tuning. *arXiv preprint arXiv:2505.16933*, 2025.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan 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, Mu Qiao, Yonghui Wu, and Mingxuan Wang. Dapo: An open-source llm reinforcement learning system at scale, 2025. URL https://arxiv.org/abs/2503.14476.

zai org. Glm-4.6: Advanced agentic, reasoning and coding capabilities. https://z.ai/blog/glm-4.6, 2025.

Aohan Zeng, Xin Lv, Qinkai Zheng, Zhenyu Hou, Bin Chen, Chengxing Xie, Cunxiang Wang, Da Yin, Hao Zeng, Jiajie Zhang, et al. Glm-4.5: Agentic, reasoning, and coding (arc) foundation models. *arXiv preprint arXiv:2508.06471*, 2025a.

Zhiyuan Zeng, Qinyuan Cheng, Zhangyue Yin, Bo Wang, Shimin Li, Yunhua Zhou, Qipeng Guo, Xuanjing Huang, and Xipeng Qiu. Scaling of search and learning: A roadmap to reproduce o1 from reinforcement learning perspective. *arXiv preprint arXiv:2412.14135*, 2024.

Zhiyuan Zeng, Qinyuan Cheng, Zhangyue Yin, Yunhua Zhou, and Xipeng Qiu. Revisiting the test-time scaling of o1-like models: Do they truly possess test-time scaling capabilities? *arXiv preprint arXiv:2502.12215*, 2025b.

Siyan Zhao, Devaansh Gupta, Qinqing Zheng, and Aditya Grover. d1: Scaling reasoning in diffusion large language models via reinforcement learning. *arXiv preprint arXiv:2504.12216*, 2025a.

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

Siyan Zhao, Mengchen Liu, Jing Huang, Miao Liu, Chenyu Wang, Bo Liu, Yuandong Tian, Guan Pang, Sean Bell, Aditya Grover, and Feiyu Chen. Inpainting-guided policy optimization for diffusion large language models, 2025c. URL https://arxiv.org/abs/2509.10396.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. *arXiv preprint arXiv:2403.13372*, 2024.

Dingwei Zhu, Shihan Dou, Zhiheng Xi, Senjie Jin, Guoqiang Zhang, Jiazheng Zhang, Junjie Ye, Mingxu Chai, Enyu Zhou, Ming Zhang, Caishuang Huang, Yunke Zhang, Yuran Wang, and Tao Gui. Vrpo: Rethinking value modeling for robust rl training under noisy supervision, 2025a. URL https://arxiv.org/abs/2508.03058.

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*, 2025b.

A More Related Work of RL in dLLMs

Besides TraceRL (Wang et al., 2025c) targeted on blockwise dLLMs, more works have made contributions to adapting RL to Diffusion LLMs. diffu-GRPO (Zhao et al., 2025b) uses a single forward pass to perform "one-step decoding" on masked completion tokens (with optional random prompt masking), approximating the sequence log-probability as the sum of per-token log-probabilities, which enables low-cost policy optimization but introduces substantial approximation bias. To mitigate this issue, Coupled-GRPO (Gong et al., 2025) constructs a pair of negatively correlated estimation samples by applying complementary masks at (t, T-t), thereby reducing the Monte Carlo variance of token log-probability estimates and stabilizing GRPO updates.

Similarly, when performing DPO alignment for diffusion LLMs using ELBO to approximate log-likelihood—which introduces high variance and unstable training—VRPO (Zhu et al., 2025a) systematically reduces the variance of the preference score by increasing the ELBO sampling budget, allocating the entire budget to different timesteps ($n_t = n$, $n_{y_t} = 1$), and letting the current and reference model share the same batch of (t, masked y_t) samples for antithetic sampling. However, because VRPO (Zhu et al., 2025a) must keep the computation graphs of all Monte Carlo samples in memory at once, it easily leads to GPU memory blow-ups and limits the number of samples that can be used. To resolve this, BGPO (Lin et al., 2025) reformulates VRPO's objective—"taking the exponential of the average log-likelihood difference over all time samples," which couples all samples—into a linearly separable lower bound over individual samples, enabling per-sample backpropagation and gradient accumulation and reducing memory usage from $O(n_t)$ to $O(1)$. Other variance-reduction methods optimize sampling strategies, such as GDPO (Rojas et al., 2025), which fixes the noise levels t to a small set of Gauss–quadrature points and performs one-step parallel denoising on generated answers to approximate the sequence-level ELBO, thereby enabling low-variance sequence-level importance ratios.

However, in the above approaches, random sampling and random masking do not seem to resolve the mismatch between the reinforcement learning objective and the actual reasoning trajectory. TraceRL (Wang et al., 2025c) abandons the "random mask + one-step denoise" scheme for obtaining importance sampling ratios. Instead, it segments the generated sequence into trajectories by step/block, uses a shrinkage parameter s to aggregate neighboring steps, and computes token-level ratios within each step by weighting with the old/new policy probability ratios. It then propagates learning signals step by step along the generation process, while introducing a diffusion-based value model to construct step-level GAE advantages.

In addition, to reduce the "zero-gradient" issue in GRPO, IGPO (Zhao et al., 2025c) uses ground-truth fragments as controllable hints to perform inpainting on fully incorrect samples, generating learnable correct trajectories and stabilizing their influence through entropy filtering, thereby restoring effective gradients for diffusion LLMs under sparse rewards.