

Inpainting-Guided Policy Optimization for Diffusion Large Language Models

Anonymous Author(s)

Affiliation

Address

email

Abstract

Masked diffusion large language models (dLLMs) are emerging as promising alternatives to autoregressive LLMs, offering competitive performance while supporting unique generation capabilities such as *inpainting*. We explore how *inpainting* can inform RL algorithm design for dLLMs by addressing a key challenge: sparse reward signals and sample waste when LLMs fail to discover correct solutions. We introduce IGPO (Inpainting Guided Policy Optimization), an RL framework that strategically injects partial ground-truth reasoning traces during online sampling to guide exploration toward promising trajectory spaces while preserving self-generated reasoning. Applied to group-based optimization methods like GRPO, IGPO restores meaningful gradients when exploration failures cause zero advantages. Combined with supervised fine-tuning on synthetically rewritten concise traces and entropy-based filtering, our approach achieves state-of-the-art performance on four mathematical benchmarks across full-attention based dLLMs.

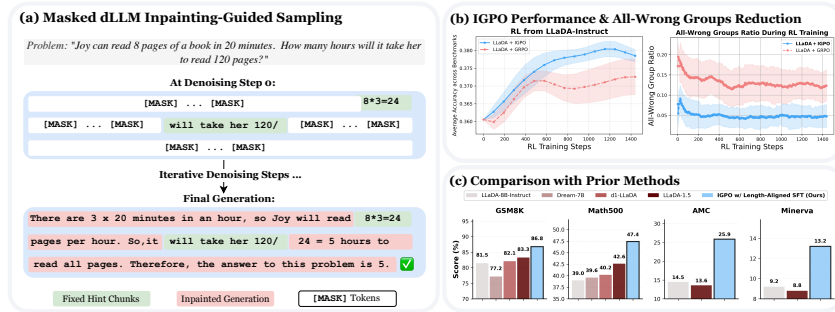


Figure 1: (a) Unlike autoregressive LLMs, diffusion LLMs can be conditioned on future reasoning hints during generation through *inpainting* via bidirectional attention, enabling guided exploration toward correct solutions. (b) Applying inpainting-guided exploration in policy optimization outperforms standard GRPO sampling and reduces all-wrong groups occurrences. (c) Our full training recipe combining *Length-Aligned* SFT on concise reasoning traces with IGPO achieves SoTA performance among full-attention masked dLLMs across four mathematical reasoning benchmarks.

1 Introduction

Recent works have shown that masked diffusion large language models (dLLMs) [Austin et al., 2021, Shi et al., 2024, Nie et al., 2025, Ye et al., 2025] can achieve performance competitive with autoregressive LLMs of similar size. Their capabilities can be further enhanced via RL post-training [Zhao et al., 2025, Gong et al., 2025b, Yang et al., 2025]. Unlike autoregressive LLMs, which decode left-to-right, dLLMs iteratively unmask tokens in parallel, enabling faster inference as shown in Mercury [Inception Labs et al., 2025] and Gemini Diffusion [DeepMind, 2025], along with flexible operations such as *inpainting*—the ability to fill missing content within existing text. Recent

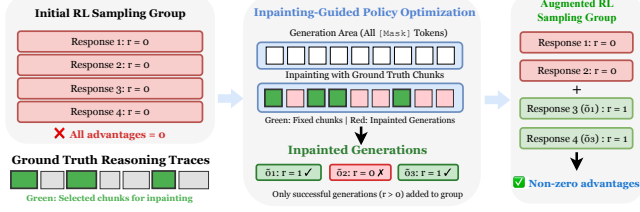


Figure 2: **Overview of IGPO:** When sampled responses yield identical incorrect rewards, we generate additional responses using ground truth hints via inpainting and replace some with correct inpainted ones to create non-zero advantage.

post-training work for dLLMs has adopted approaches similar to autoregressive LLMs, applying RLVR methods [Zhao et al., 2025, Yang et al., 2025, Gong et al., 2025b]. In RLVR, a fundamental exploration challenge persists: for difficult tasks, policies struggle to discover correct solutions and binary rewards provide minimal learning signal when most generated solutions are incorrect, leading to substantial sample waste and poor training efficiency.

The bidirectional structure of dLLMs provides a unique mechanism to address this challenge. Since dLLMs are trained through stochastic masking, they possess inherent capability for *inpainting* with externally provided partial hints. We leverage this to introduce IGPO (Inpainting Guided Policy Optimization), a novel RL framework that strategically guides exploration by injecting reasoning hints for difficult problems. When the policy is unlikely to generate correct solutions, partial reasoning traces are injected into the generation region, and the dLLM completes the remaining sequence. Only successful inpainting completions are used for policy optimization. This approach is particularly effective for group-based methods like GRPO [Shao et al., 2024], where all-wrong groups collapse advantage to zero. By reducing all-wrong groups, IGPO restores gradient signals and enables more effective RL. In summary, our work makes the following key contributions:

- We propose IGPO, the **first work to utilize the unique inpainting capabilities of diffusion LLMs** for RL. By strategically injecting partial reasoning traces during exploration, IGPO alleviates sparse reward inefficiency and mitigates the zero-advantage dilemma in group-based policy optimization methods, reducing all-wrong groups by approximately 60% (Figure 1 (b)).
- We propose *Length-Aligned* SFT for full-attention based dLLMs using synthetically rewritten, concise reasoning traces. This design better aligns SFT data length with RL sampling and evaluation length, avoiding verbose trace limitations and providing stronger initialization for RL.
- Our training recipe achieves substantial improvements on mathematical benchmarks: **+5.3% on GSM8K, +8.4% on Math500, +11.4% on AMC, and +4.0% on Minerva** relative to LLaDA-Instruct, achieving **SoTA performance among full-attention based dLLMs**.
- We conduct comprehensive ablation studies disentangling IGPO mechanisms. We show that partial inpainting consistently outperforms full ground-truth inpainting by staying closer to the policy distribution in online RL, and propose entropy-based gradient filtering that stabilizes training.

2 Methods

IGPO: Inpainting Guided Policy Optimization In GRPO, when all G responses $\{o_1, \dots, o_G\}$ for prompt q receive identical rewards, advantages become zero: $A_i = r(o_i) - \frac{1}{G} \sum_{j=1}^G r(o_j) = 0$, making the policy gradient degenerate. To address this *all-wrong* case, we introduce IGPO, which modifies masked dLLM generation. In full-attention masked dLLM generation such as LLaDA [Zhu et al., 2025], the model input at denoising step 0 is the concatenation $[q; z_{\text{mask}}]$, where q represents the prompt and z_{mask} denotes a fully masked completion sequence of predetermined length. The generation process progressively unmask these positions through iterative denoising until producing the final output. IGPO fixes selected positions of z_{mask} to ground-truth tokens from reasoning trace y^* . As shown in Figure 2, IGPO triggers hint injection only when all responses yield incorrect rewards, segmenting y^* into variable-length chunks with sizes from $\mathcal{U}[s_{\min}, s_{\max}]$ and randomly selecting $\lfloor \eta \cdot N \rfloor$ chunks for injection with ratio $\eta \sim \mathcal{U}[\eta_{\text{low}}, \eta_{\text{high}}]$. We generate additional responses $\{\tilde{o}_1, \dots, \tilde{o}_G\}$ through inpainting, then replace $K = \min(|\{\tilde{o}_i : r(\tilde{o}_i) = 1\}|, \lfloor \lambda G \rfloor)$ original responses with correct inpainted ones. The IGPO objective modifies GRPO by including verified correct inpainted responses. We also design an entropy-based gradient filtering technique for learning stability, where we only apply gradient updates on hint tokens where the model exhibits sufficient uncertainty, updating only the top τ percentile of hint positions with highest entropy. A detailed method description is in Appendix A.1.

Length-Aligned SFT via Concise Reasoning Trace Rewriting Full-attention masked dLLMs lack KV cache optimization by default, requiring full-sequence attention at every denoising step. We restrict RL rollouts to 256 tokens, but popular reasoning SFT corpora contain verbose traces often exceeding 10k tokens and evaluation of recent dLLMs often only requires less than 1024 tokens on mathematical benchmarks [Zhu et al., 2025], creating distribution mismatch. We systematically rewrite verbose traces into concise forms using LLaMA-4-Maverick [Meta, 2025], removing redundant reflections while preserving essential reasoning. Our *Length-Aligned SFT* trains exclusively on rewritten traces, providing better RL initialization within fixed computational bounds.

3 Experiments

To investigate how the inpainting capabilities of masked dLLMs can address exploration challenges in RL and how *Length-Aligned SFT* improves performance, we conduct comprehensive experiments using a two-stage training pipeline. **Stage 1: Supervised Fine-Tuning with Rewritten Traces.** We begin with *Length-Aligned SFT* on the LLaDA-8B-Instruct model using the OpenR1-Math-220K dataset [Cobbe et al., 2021] with all reasoning traces rewritten to ensure consistency between training distribution and downstream RL/evaluation phases. **Stage 2: Reinforcement Learning with IGPO.** Following *Length-aligned SFT*, we apply IGPO using reasoning traces from the MetaMathQA dataset [Yu et al., 2023] for strategic inpainting-guided policy optimization. We evaluate our approach on four mathematics benchmarks: GSM8K [Cobbe et al., 2021], MATH500 [Hendrycks et al., 2021], AMC [LI et al., 2024] and Minerva Math [Lewkowycz et al., 2022]. Experiments are conducted using LLaDA-8B-Instruct as the base model with sampling temperature of 1.2 for RL online generation, with detailed hyperparameters and experimental setup provided in Appendix J.

3.1 Main Results and Ablation Studies

Table 1: Performance across multiple mathematics tasks. Underlined scores indicate the best *within each initialization group*. Parenthesized deltas typeset via (+) denote absolute percentage-point improvements *relative to the LLaDA-8B-Instruct baseline*.

Model	GSM8K (pass @ 1)	MATH500 (pass @ 1)	AMC (avg @ 16)	Minerva (pass @ 1)	Average
<i>Similar-sized autoregressive LLMs</i>					
LLaMA3-8B [AI@Meta, 2024]	79.6	30.0	–	–	–
Qwen2.5-7B [Team, 2024]	85.4	49.8	–	–	–
<i>Prior masked dLLM baselines</i>					
Dream-7B [Ye et al., 2025]	77.2	39.6	–	–	–
d1-LLaDA [Zhao et al., 2025]	82.1	40.2	–	–	–
wd1 [Tang et al., 2025]	82.3	39.0	–	–	–
LLaDA-1.5 [Zhu et al., 2025]	83.3	42.6	13.6	8.8	37.1
LLaDA-Instruct [Nie et al., 2025]	81.5 (+0)	39.0 (+0)	14.5 (+0)	9.2 (+0)	36.0 (+0)
<i>RL from LLaDA-Instruct</i>					
LLaDA-Instruct + UniGRPO [Yang et al., 2025]	82.2 (+0.7)	39.2 (+0.2)	15.0 (+0.5)	11.0 (+1.8)	36.9 (+0.9)
LLaDA-Instruct + DiffuGRPO [Zhao et al., 2025]	82.4 (+0.9)	40.2 (+1.2)	15.5 (+1.0)	10.3 (+1.1)	37.1 (+1.1)
LLaDA-Instruct + IGPO (ours)	<u>83.1</u> (+1.6)	<u>42.8</u> (+3.8)	<u>17.5</u> (+3.0)	<u>12.1</u> (+2.9)	<u>38.9</u> (+2.9)
<i>Length-aligned SFT on LLaDA-Instruct and RL on the SFT checkpoint</i>					
LLaDA-Instruct + Length-aligned SFT (ours)	83.6 (+2.1)	45.2 (+6.2)	22.3 (+7.8)	10.3 (+1.1)	40.4 (+4.4)
LLaDA-Instruct + Length-aligned SFT + IGPO (ours)	<u>86.8</u> (+5.3)	<u>47.4</u> (+8.4)	<u>25.9</u> (+11.4)	<u>13.2</u> (+4.0)	<u>43.3</u> (+7.3)

Main results. As shown in Table 1, our training recipe demonstrates consistent improvements across all mathematical reasoning benchmarks. Length-Aligned SFT on rewritten traces provides substantial gains over the base LLaDA-8B-Instruct model, with IGPO delivering additional improvements when applied on top of SFT. As shown in Figure 5, IGPO exhibits superior training dynamics compared to standard GRPO sampling regardless of initialization point. IGPO effectively reduces the all-wrong group ratio, as shown in Figure 1(b). Our final model (LLaDA + Length-Aligned SFT + IGPO) outperforms all baseline approaches including the recent LLaDA-1.5 model across all evaluated benchmarks. Notably, even without SFT, applying IGPO directly on LLaDA achieves better performance than previous LLaDA-1.5 and other RL methods for full-attention dLLMs, establishing a new state-of-the-art recipe for mathematical reasoning in masked diffusion language models.

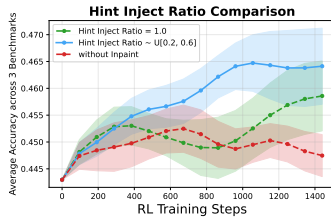


Figure 3: **Impact of hint injection ratio** across 3 datasets (GSM8K, MATH500, AMC) and 3 seeds with standard error as shaded areas. Partial hint injection ($\eta \sim \mathcal{U}[0.2, 0.6]$) consistently outperforms full hint injection ($\eta = 1.0$), demonstrating benefits of self-generated reasoning. Both variants outperform baseline without hint injection.

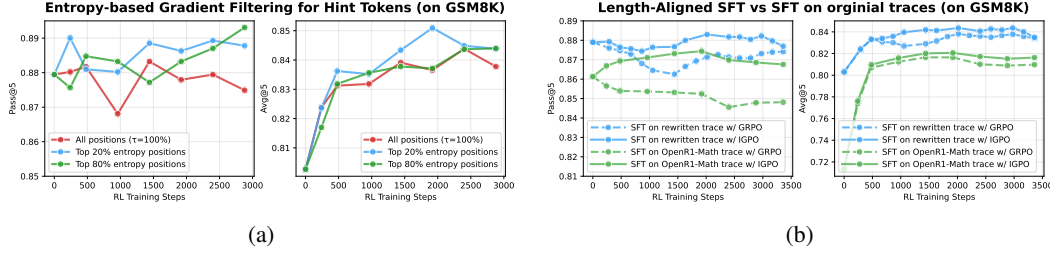


Figure 4: (a) **Impact of entropy clipping threshold on hint tokens.** Performance comparison across different entropy clipping thresholds in IGPO, where $\tau = 0.2$ learns from top 20% highest-entropy hint positions while $\tau = 1.0$ learns from all positions. (b) **SFT and RL dynamics with rewritten vs. original traces.** Models fine-tuned on concise rewritten traces (< 1024 tokens) vs original OpenR1-Math traces (truncated at 4096 tokens). Rewritten traces yield stronger SFT and RL performance.

Self-generated inpainted traces provide better learning signal than ground truth traces. Figure 3 shows partial hint injection achieves higher performance than full hint injection. With lower hint injection ratios, the model generates self-rationalized inpainting traces (Section I), adding only those leading to correct solutions for gradient updates. Through inpainting, the model coherently connects provided hint chunks with its own reasoning steps. Inpainted generation produces learning signals bridging the gap between current capabilities and target behavior. Self-generated portions reflect current reasoning patterns and are more "on-policy" while incorporating structural guidance from ground truth chunks, resulting in more effective policy optimization than pure supervised learning by reducing distributional mismatch. **This bridging of SFT and online RL through partial self-generation enables more effective policy optimization.**

Entropy clipping prevents training instability from off-policy tokens. Figure 4a shows learning from only the top 20% highest-entropy hint token positions ($\tau = 0.2$) achieves the best performance and exhibits the most stable training dynamics. In contrast, learning from all hint token positions ($\tau = 1.0$) or a large fraction ($\tau = 0.8$) leads to more unstable training with performance fluctuations compared to lower values like 0.2. This supports our motivation that restricting gradient updates to high-entropy positions prevents the destabilizing effects of large gradients on high-entropy positions on hints tokens, since these injected hint tokens are from ground-truth dataset and are "off-policy" to the current learning policy.

Effect of reasoning trace rewriting for SFT and subsequent RL training. Figure 4b shows two key findings. First, SFT on rewritten traces produces substantially stronger checkpoints than original traces by eliminating verbose reflection and compressing reasoning into concise trajectories aligned with LLaDA's generation budget. Second, while RL training partially compensates for weaker SFT checkpoints, stronger rewritten SFT initialization leads to consistently higher final performance. IGPO outperforms standard RL across both settings while preserving output diversity and stabilizing pass@5 performance, whereas standard GRPO exhibits pass@k decline indicating mode collapse.

Elastic inpainting outperforms sequential SFT and GRPO We further validate the effectiveness of our elastic inpainting approach by comparing it against sequentially performing SFT on the RL dataset's reasoning traces followed by standard GRPO (see Section G for details). This ablation confirms that IGPO's elastic hint injection during zero-advantage scenarios is superior to uniformly applying SFT on concise reasoning traces across all prompts before applying GRPO. The uniform SFT approach can degrade initial performance due to distribution shift in reasoning patterns, whereas injecting partial hints allows dLLMs to inpaint longer, more "on-policy" reasoning traces.

4 Conclusion

We introduced IGPO, a reinforcement learning algorithm that leverages masked diffusion language models' inpainting capabilities to address exploration bottlenecks in RL. By injecting ground-truth reasoning hints during denoising, IGPO resolves the zero-advantage dilemma and induces reward variance for effective policy gradient updates. Combined with Length-Aligned SFT and entropy-based gradient filtering, our approach achieves state-of-the-art performance among full-attention masked dLLMs on mathematical reasoning benchmarks, demonstrating how architectural properties can be systematically exploited for RL optimization.

References

- Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce style optimization for learning from human feedback in llms. *arXiv preprint arXiv:2402.14740*, 2024.
- AI@Meta. Llama 3 model card. 2024. URL https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md.
- Chenxin An, Zihui Xie, Xiaonan Li, Lei Li, Jun Zhang, Shansan Gong, Ming Zhong, Jingjing Xu, Xipeng Qiu, Mingxuan Wang, and Lingpeng Kong. Polaris: A post-training recipe for scaling reinforcement learning on advanced reasoning models, 2025. URL <https://hkunlp.github.io/blog/2025/Polaris>.
- 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. In *The Thirteenth International Conference on Learning Representations*, 2025.
- Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne Van Den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in neural information processing systems*, 34:17981–17993, 2021.
- Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.
- Ting Chen, Ruixiang Zhang, and Geoffrey E. Hinton. Analog bits: Generating discrete data using diffusion models with self-conditioning. *ArXiv*, abs/2208.04202, 2022.
- 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.
- DeepMind. Gemini diffusion, 2025. URL <https://deepmind.google/models/gemini-diffusion/>.
- Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and Lingpeng Kong. DiffuSeq: Sequence to sequence text generation with diffusion models. In *International Conference on Learning Representations, ICLR*, 2023.
- Shansan Gong, Shivam Agarwal, Yizhe Zhang, Jiacheng Ye, Lin Zheng, Mukai Li, Chenxin An, Peilin Zhao, Wei Bi, Jiawei Han, Hao Peng, and Lingpeng Kong. Scaling diffusion language models via adaptation from autoregressive models. In *The Thirteenth International Conference on Learning Representations*, 2025a.
- Shansan Gong, Huangjie Zheng, Ruixiang Zhang, Jiatao Gu, Navdeep Jaitly, Lingpeng Kong, and Yizhe Zhang. Diffucoder: Understanding and improving masked diffusion models for code generation. 2025b. 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.
- Jiaqi Han, Austin Wang, Minkai Xu, Wenda Chu, Meihua Dang, Yisong Yue, and Stefano Ermon. Discrete diffusion trajectory alignment via stepwise decomposition, 2025. URL <https://arxiv.org/abs/2507.04832>.
- 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.
- Zhanqiu Hu, Jian Meng, Yash Akhauri, Mohamed S. Abdelfattah, Jae sun Seo, Zhiru Zhang, and Udit Gupta. Accelerating diffusion language model inference via efficient kv caching and guided diffusion, 2025. URL <https://arxiv.org/abs/2505.21467>.

189 Zeyu Huang, Tianhao Cheng, Zihan Qiu, Zili Wang, Yinghui Xu, Edoardo M. Ponti, and Ivan
190 Titov. Blending supervised and reinforcement fine-tuning with prefix sampling, 2025. URL
191 <https://arxiv.org/abs/2507.01679>.

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

196 Daniel Israel, Guy Van den Broeck, and Aditya Grover. Accelerating diffusion llms via adaptive
197 parallel decoding, 2025. URL <https://arxiv.org/abs/2506.00413>.

198 Aitor Lewkowycz, Anders Johan Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski,
199 Vinay Venkatesh Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai
200 Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems
201 with language models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho,
202 editors, *Advances in Neural Information Processing Systems*, 2022. URL [https://openreview](https://openreview.net/forum?id=IFXTZERXdm7)
203 [.net/forum?id=IFXTZERXdm7](https://openreview.net/forum?id=IFXTZERXdm7).

204 Jia LI, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang,
205 Kashif Rasul, Longhui Yu, Albert Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau,
206 Guillaume Lample, and Stanislas Polu. Numinamath. [https://github.com/project-numina](https://github.com/project-numina/aimo-progress-prize/blob/main/report/numina_dataset.pdf)
207 [a/aimo-progress-prize/blob/main/report/numina_dataset.pdf](https://github.com/project-numina/aimo-progress-prize/blob/main/report/numina_dataset.pdf), 2024.

208 Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori Hashimoto. Diffusion-lm
209 improves controllable text generation. *ArXiv*, abs/2205.14217, 2022.

210 Ziniu Li, Tian Xu, Yushun Zhang, Zhihang Lin, Yang Yu, Ruoyu Sun, and Zhi-Quan Luo. Remax: A
211 simple, effective, and efficient reinforcement learning method for aligning large language models.
212 *arXiv preprint arXiv:2310.10505*, 2023.

213 Zhiyuan Liu, Yicun Yang, Yaojie Zhang, Junjie Chen, Chang Zou, Qingyuan Wei, Shaobo Wang, and
214 Linfeng Zhang. dllm-cache: Accelerating diffusion large language models with adaptive caching.
215 *arXiv preprint arXiv:2506.06295*, 2025a.

216 Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min
217 Lin. Understanding rl-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*,
218 2025b.

219 Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios
220 of the data distribution. In *Forty-first International Conference on Machine Learning*, 2024.

221 Xinyin Ma, Runpeng Yu, Gongfan Fang, and Xinchao Wang. dkv-cache: The cache for diffusion
222 language models, 2025. URL <https://arxiv.org/abs/2505.15781>.

223 AI Meta. The llama 4 herd: The beginning of a new era of natively multimodal ai innovation.
224 <https://ai.meta.com/blog/llama-4-multimodal-intelligence/>, 2025.

225 Jinjie Ni and the team. Diffusion language models are super data learners. [https://jinjieni.n](https://jinjieni.notion.site/Diffusion-Language-Models-are-Super-Data-Learners-239d8f03a866800ab196e49928c019ac)
226 [otion.site/Diffusion-Language-Models-are-Super-Data-Learners-239d8f03a86](https://jinjieni.notion.site/Diffusion-Language-Models-are-Super-Data-Learners-239d8f03a866800ab196e49928c019ac)
227 [6800ab196e49928c019ac](https://jinjieni.notion.site/Diffusion-Language-Models-are-Super-Data-Learners-239d8f03a866800ab196e49928c019ac), 2025. Notion Blog.

228 Shen Nie, Fengqi Zhu, Chao Du, Tianyu Pang, Qian Liu, Guangtao Zeng, Min Lin, and Chongxuan
229 Li. Scaling up masked diffusion models on text. *arXiv preprint arXiv:2410.18514*, 2024.

230 Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin,
231 Ji-Rong Wen, and Chongxuan Li. Large language diffusion models, 2025. URL [https://arxiv.](https://arxiv.org/abs/2502.09992)
232 [org/abs/2502.09992](https://arxiv.org/abs/2502.09992).

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

236 Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong
237 Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow
238 instructions with human feedback. *Advances in neural information processing systems*, 35:27730–
239 27744, 2022.

240 Mihir Prabhudesai, Mengning Wu, Amir Zadeh, Katerina Fragkiadaki, and Deepak Pathak. Diffusion
241 beats autoregressive in data-constrained settings, 2025. URL [https://arxiv.org/abs/2507.1](https://arxiv.org/abs/2507.15857)
242 [5857](https://arxiv.org/abs/2507.15857).

243 Subham S. Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T. Chiu,
244 Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models.
245 In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M.
246 Tomczak, and Cheng Zhang, editors, *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL [http://papers.nips.cc/paper_files/p](http://papers.nips.cc/paper_files/paper/2024/hash/eb0b13cc515724ab8015bc978fdde0ad-Abstract-Conference.html)
247 [aper/2024/hash/eb0b13cc515724ab8015bc978fdde0ad-Abstract-Conference.html](http://papers.nips.cc/paper_files/paper/2024/hash/eb0b13cc515724ab8015bc978fdde0ad-Abstract-Conference.html).
248
249

250 Subham Sekhar Sahoo, Zhihan Yang, Yash Akhauri, Johnna Liu, Deepansha Singh, Zhoujun Cheng,
251 Zhengzhong Liu, Eric Xing, John Thickstun, and Arash Vahdat. Esoteric language models, 2025.
252 URL <https://arxiv.org/abs/2506.01928>.

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

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

258 Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis K. Titsias. Simplified and
259 generalized masked diffusion for discrete data. In Amir Globersons, Lester Mackey, Danielle
260 Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang, editors, *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL [http://papers.nips.cc/paper_files/paper/2024/hash/bad233b9849f019aead5ec](http://papers.nips.cc/paper_files/paper/2024/hash/bad233b9849f019aead5ec60cef70f-Abstract-Conference.html)
261 [c60cef70f-Abstract-Conference.html](http://papers.nips.cc/paper_files/paper/2024/hash/bad233b9849f019aead5ec60cef70f-Abstract-Conference.html).
262
263
264

265 Xiaohang Tang, Rares Dolga, Sangwoong Yoon, and Ilija Bogunovic. wd1: Weighted policy
266 optimization for reasoning in diffusion language models, 2025. URL [https://arxiv.org/abs/](https://arxiv.org/abs/2507.08838)
267 [2507.08838](https://arxiv.org/abs/2507.08838).

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

271 Qwen Team. Qwen2.5: A party of foundation models, September 2024. URL [https://qwenlm.g](https://qwenlm.github.io/blog/qwen2.5/)
272 [ithub.io/blog/qwen2.5/](https://qwenlm.github.io/blog/qwen2.5/).

273 Chengyue Wu, Hao Zhang, Shuchen Xue, Zhijian Liu, Shizhe Diao, Ligeng Zhu, Ping Luo, Song
274 Han, and Enze Xie. Fast-dllm: Training-free acceleration of diffusion llm by enabling kv cache
275 and parallel decoding, 2025. URL <https://arxiv.org/abs/2505.22618>.

276 Ling Yang, Ye Tian, Bowen Li, Xincheng Zhang, Ke Shen, Yunhai Tong, and Mengdi Wang. Mmada:
277 Multimodal large diffusion language models. *ArXiv preprint*, abs/2505.15809, 2025. URL
278 <https://arxiv.org/abs/2505.15809>.

279 Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng
280 Kong. Dream 7b, 2025. URL <https://hkunlp.github.io/blog/2025/dream>.

281 Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo
282 Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions for
283 large language models. *arXiv preprint arXiv:2309.12284*, 2023.

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

- 286 Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang,
287 Yuqiong Liu, Rui Men, An Yang, et al. Group sequence policy optimization. *arXiv preprint*
288 *arXiv:2507.18071*, 2025.
- 289 Fengqi Zhu, Rongzhen Wang, Shen Nie, Xiaolu Zhang, Chunwei Wu, Jun Hu, Jun Zhou, Jianfei
290 Chen, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Llada 1.5: Variance-reduced preference
291 optimization for large language diffusion models. *ArXiv preprint*, abs/2505.19223, 2025. URL
292 <https://arxiv.org/abs/2505.19223>.

A Detailed Methods

A.1 IGPO: Inpainting Guided Policy Optimization

Zero-Advantage Dilemma. In the GRPO framework, when sampling G responses $\{o_1, o_2, \dots, o_G\}$ for a given prompt q , the advantage computation relies on reward variance across the group. However, when all responses receive identical rewards—either all correct or all incorrect—the advantages become zero: $A_i = r(o_i) - \frac{1}{G} \sum_{j=1}^G r(o_j) = 0$. This zero-advantage scenario makes the policy gradient component degenerate. Specifically, the clipped surrogate objective collapses to zero regardless of whether the update lies in the clipped or unclipped region, since both terms contain $A_i = 0$. The policy gradient for this prompt q therefore becomes:

$$\frac{1}{G} \sum_{i=1}^G \frac{1}{|o_i|} \sum_{k=1}^{|o_i|} A_i \rho_i^k \nabla_{\theta} \log \pi_{\theta}(o_i^k | q) = 0 \quad \text{since } A_i = 0 \forall i.$$

As a result, no meaningful policy update can be extracted from the reward signal, wasting compute sampling these responses. **In this work, we specifically focus on mitigating the *all-wrong* case.**

Masked dLLM Generation and Inpainting. In full-attention masked dLLM generation, the model input at denoising step 0 is the concatenation $[q; z_{\text{mask}}]$, where q represents the prompt and $z_{\text{mask}} = [\text{mask}, \text{mask}, \dots, \text{mask}]$ denotes a fully masked completion sequence of predetermined length L . The generation process progressively unmask these positions through iterative denoising until producing the final output.

Hint injection modifies this formulation by fixing selected positions of z_{mask} to ground-truth tokens. During RL training, we assume access to ground-truth reasoning trace $y^* = [y_1^*, y_2^*, \dots, y_{|y^*|}^*]$ for every question q . For injection, we create a binary mask $m \in \{0, 1\}^L$ indicating which positions to inject as fixed hints, we construct the hint-injected initialization:

$$z^{\text{hint}}[i] = \begin{cases} y_i^* & \text{if } m[i] = 1 \text{ and } i \leq |y^*|, \\ \text{mask} & \text{otherwise.} \end{cases} \quad (1)$$

The masked dLLM then performs bidirectional denoising on $[q; z^{\text{hint}}]$ through the inpainting process, leveraging both the prompt and injected hint tokens to generate coherent responses. The injected hint tokens remain fixed throughout the iterative denoising steps.

Algorithm 1 IGPO: Inpainting-Guided Policy Optimization for Masked dLLMs

Require: Reference model π_{ref} , prompt distribution $\mathcal{D}$, ground-truth reasoning traces $\{y^*\}$, number of completions per prompt G , number of inner updates μ , hint injection ratio range $[\eta_{\text{low}}, \eta_{\text{high}}]$, replacement fraction λ , entropy filter threshold τ , chunk size range $[s_{\text{min}}, s_{\text{max}}]$

```

1: Initialize  $\pi_{\theta} \leftarrow \pi_{\text{ref}}$ 
2: while not converged do
3:    $\pi_{\text{old}} \leftarrow \pi_{\theta}$ ; sample prompt  $q \sim \mathcal{D}$  and responses  $o_{1:G} \sim \pi_{\text{old}}(\cdot | q)$ ; compute rewards  $r_{1:G}$ 
4:   if all  $r_i = 0$  (zero-advantage case) then
5:     Segment ground-truth reasoning  $y^*$  into chunks  $\{c_1, \dots, c_N\}$  with  $|c_j| \sim \mathcal{U}[s_{\text{min}}, s_{\text{max}}]$ 
6:     for  $i = 1, \dots, G$  do
7:       Sample hint injection ratio  $\eta \sim \mathcal{U}[\eta_{\text{low}}, \eta_{\text{high}}]$  and select  $\lfloor \eta N \rfloor$  chunks from  $\{c_1, \dots, c_N\}$  randomly
8:       Inject selected chunk tokens as fixed hints at corresponding positions
9:       Generate  $\tilde{o}_i$  via inpainting: denoise only masked positions, keep hint tokens fixed
10:      Evaluate rewards  $r(\tilde{o}_i)$  and replace up to  $\lfloor \lambda G \rfloor$  incorrect  $o_i$  with correct  $\tilde{o}_i$ 
11:    Compute advantages  $A_i$  on the updated response set
12:    for  $n = 1, \dots, \mu$  do
13:      Estimate  $\log \pi_{\theta}, \log \pi_{\text{old}}, \log \pi_{\text{ref}}$ ; apply top- $\tau$  entropy filter on hint positions
14:      Update  $\pi_{\theta}$  via  $\mathcal{L}_{\text{IGPO}}(\theta)$  (Eq. 2)
15: return  $\pi_{\theta}$ 

```

Constructing Hint Patterns for Inpainting. To construct meaningful hint patterns for the inpainting process, we segment the ground truth reasoning trace y^* into variable-length contiguous chunks $\mathcal{C} = \{c_1, c_2, \dots, c_N\}$, where each chunk length $|c_j|$ is sampled from $\mathcal{U}[s_{\min}, s_{\max}]$. We explicitly exclude the final answer tokens from chunking to prevent reward hacking behaviors where the model ignores reasoning and collapses. For a given hint injection ratio $\eta \in [0, 1]$, we randomly select $\lfloor \eta \cdot N \rfloor$ chunks and set their corresponding positions in the binary mask m to 1 for hint injection.

Elastic Inpainting-Triggered Sampling. With the above inpainting setup, we design IGPO (as in Algorithm 1) to be **elastic**: hint injection is only triggered when all sampled responses in a group yield incorrect rewards (the zero-advantage case), and when activated, both the hint injection ratio η and chunk sizes ($\mathcal{U}[s_{\min}, s_{\max}]$) are randomized to provide diverse training signals. Concretely, when detecting that all sampled responses $\{o_1, \dots, o_G\}$ for query q yield identical rewards $r(o_i) = 0$, we generate an additional set of responses $\{\tilde{o}_1, \dots, \tilde{o}_G\}$ through the inpainting process. Each response $\tilde{o}_i$ is generated via inpainting with a distinct hint injection ratio $\eta_i \sim \mathcal{U}[\eta_{\text{low}}, \eta_{\text{high}}]$ to ensure diverse hint densities. Following inpainting generation, we evaluate the correctness of $\{\tilde{o}_i\}$ and only use the correct ones for replacement. Specifically, we replace $K = \min(|\{\tilde{o}_i : r(\tilde{o}_i) = 1\}|, \lfloor \lambda G \rfloor)$ of the original incorrect responses with correct responses generated through inpainting, where $\lambda \in (0, 1)$ controls the replacement fraction.

The complete IGPO objective modifies the GRPO formulation by incorporating the augmented sampling procedure:

$$\mathcal{L}_{\text{IGPO}}(\theta) = \mathbb{E}_{\substack{q \sim \mathcal{D} \\ \{o_1, \dots, o_{G-K}, \tilde{o}_1, \dots, \tilde{o}_K\} \sim \text{IGPO-Sample}(\pi_\theta, q, y^*)}} \left[\left(\frac{1}{G} \sum_{i=1}^G \frac{1}{L_i} \sum_{k=1}^{L_i} \min(\rho_i^k A_i^k, \text{clip}(\rho_i^k, 1 - \varepsilon, 1 + \varepsilon) A_i^k) \right) - \beta D_{\text{KL}}[\pi_\theta(\cdot|q) \parallel \pi_{\text{ref}}(\cdot|q)] \right], \quad (2)$$

where $\text{IGPO-Sample}(\pi_\theta, q, y^*)$ denotes the augmented sampling procedure that applies inpainting-based augmentation when zero-advantage scenarios are detected, producing the augmented RL sampling group $\{o_1, \dots, o_{G-K}, \tilde{o}_1, \dots, \tilde{o}_K\}$ containing $(G - K)$ original responses and K verified correct inpainted responses $\{\tilde{o}_i\}$ after replacement. L_i denotes the length of the i -th response (whether o_i or $\tilde{o}_i$). Crucially, only inpainted responses that pass correctness verification are included in the augmented group, satisfying $r(\tilde{o}_i) = 1$. Advantages A_i are computed normally. We built IGPO with DiffuGRPO [Zhao et al., 2025]’s log probability estimation methods, where all completion tokens are masked during estimation and we remove the random masking applied to prompt tokens as done in DiffuGRPO. Since we use a small number of policy iterations (i.e. $\mu = 4$), this alleviates the need for random prompt masking to reduce overfitting. Inspired by Zheng et al. [2025], we compute sequence-level importance-ratio through mean-field approximation for stability purposes.

Entropy-based Gradient Filtering for Hint Tokens. When applying IGPO to zero-advantage scenarios, the responses generated through inpainting contain ground truth reasoning chunks that originate from a different distribution than the current policy π_θ . This creates an off-policy learning scenario where gradient updates from ground truth tokens can conflict with the model’s current beliefs, particularly at positions where the model has high confidence (low entropy). To mitigate potential training instability from this distribution mismatch, we implement an entropy-based filtering approach that restricts learning to hint token positions where the model exhibits sufficient uncertainty, as inspired by Huang et al. [2025]. Specifically, for each hint token position (i.e., positions with injected ground-truth tokens) we compute the entropy. We then apply gradient updates only to the top τ percentile of hint token positions with highest entropy values. This selective learning strategy serves two purposes: high-entropy positions represent genuine decision boundaries where the model is naturally uncertain and thus more receptive to external guidance, and they correspond to flatter probability distributions that yield more stable gradient updates when incorporating ground truth information. This approach controls the policy shift by focusing learning on positions where the model is already open to change, rather than forcing updates against strong existing beliefs at low-entropy positions.

A.2 Length-Aligned SFT via Concise Reasoning Trace Rewriting

To further strengthen our training recipe, we seek better RL initialization via SFT but identified generation length mismatches across SFT, RL sampling, and evaluation phases. Full-attention masked dLLMs like LLaDA lack KV cache optimization [Wu et al., 2025] by default, requiring full-sequence attention at every denoising step, which dominates online RL training cost. As a result, we restrict RL rollouts to 256 tokens for faster convergence within a reduced exploration space, and evaluation setups

in recent work [Zhao et al., 2025, Zhu et al., 2025, Nie et al., 2025] typically use 256–1024 tokens. In contrast, popular reasoning SFT corpora (e.g., OpenR1) contain verbose traces often exceeding 10k tokens, creating distribution mismatch across SFT, RL, and evaluation, and include repeated reflective behaviors unsuited for limited context. To resolve this, we systematically rewrite verbose traces into concise, structured forms that preserve logical flow while respecting dLLM computational limits. Using LLaMA-4-Maverick [Meta, 2025] with prompts detailed in Section K, we remove redundant reflections, condense multi-sentence elaborations into precise, mathematically rigorous statements, and retain essential reasoning. Examples of revision length distributions and before/after traces are in Sections E and K. Our *Length-Aligned SFT* trains LLaDA solely on rewritten traces, improving RL initialization by avoiding implicit length compression and focusing learning on reasoning quality within fixed compute budgets. Empirical results show clear gains over training on verbose traces, and we further observe that masked dLLMs benefit from extended training (e.g., 100 epochs) relative to AR LLMs, consistent with recent works [Ni and the team, 2025, Prabhudesai et al., 2025].

B Preliminaries

B.1 Masked Diffusion Large Language Models

Masked diffusion LLMs [Austin et al., 2021, Sahoo et al., 2024, Shi et al., 2024, Ou et al., 2024, Lou et al., 2024] employ a forward diffusion process that progressively corrupts token sequences x_0 through introduction of mask tokens. This corruption process is parameterized by time $t \in [0, 1]$. At any given timestep t , the resulting sequence x_t contains partial masking, where each token maintains a probability α_t of remaining unmasked. The noise schedule α_t exhibits strict monotonic decrease with respect to t . Complete masking occurs at $t = 1$, where all tokens in x_1 become masked. The training procedure for masked dLLMs follows a forward process through definition of α_t and a bidirectional unmasking predictor f_θ with learnable parameters. During each training step, we stochastically sample timestep $t \in [0, 1)$ and apply token masking according to the designated forward process. Given these corrupted sequences, the training objective seeks to recover the original tokens. The standard optimization criterion employs the negative evidence lower bound (NELBO), which provides an upper bound for the negative log-likelihood (NLL) of the training data. For masked dLLMs, this NELBO reduces to a weighted NLL formulation, with weighting coefficients derived from transformations of α_t [Sahoo et al., 2024, Equation (10)]. For example, LLaDA [Nie et al., 2025] specifies the forward process through $\alpha_t = 1 - t$, yielding the following NELBO formulation:

$$-\mathbb{E}_{t \sim \mathcal{U}[0,1], x_0 \sim p_{\text{data}}, x_t \sim q_{t|0}(x_t|x_0)} \left[\frac{1}{t} \sum_{k=1}^{|x_t|} \mathbb{1}[x_t^k = \text{mask}] \log f_\theta(x_0^k | x_t) \right], \quad (3)$$

where $|x_t|$ denotes the sequence length of x , and x^k represents the k -th token position. The loss computation is restricted to tokens masked at timestep t .

During prompt conditional generation, the model starts with a sequence where prompt tokens remain unmasked and continuation tokens are initially masked, then progressively unmask the continuation tokens through ancestral sampling from the reverse process $p_\theta(x_s | x_t)$ for timesteps $t > s$, where the model f_θ provides the denoising predictions for masked positions. The reverse process maintains the property that unmasked tokens are carried over unchanged throughout all denoising steps.

B.2 Policy Optimization for Masked Diffusion Large Language Models

Policy-gradient methods have gained widespread adoption for post-training LLMs [Ouyang et al., 2022, Bai et al., 2022, Li et al., 2023, Ahmadian et al., 2024]. Online RL—particularly Group Relative Policy Optimization (GRPO)—has proved effective for improving language models [Shao et al., 2024, Guo et al., 2025, Team et al., 2025]. GRPO [Shao et al., 2024] offers a computationally efficient alternative to PPO [Schulman et al., 2017] by using group-based statistics for advantage estimation, avoiding separate value-function training.

The GRPO objective integrates clipping for stability and reverse KL regularization:

$$\mathcal{L}_{\text{GRPO}}(\theta) = \mathbb{E}_{\substack{q \sim \mathcal{D} \\ o_1, \dots, o_G \sim \pi_{\theta_{\text{old}}}(\cdot|q)}} \left[\frac{1}{G} \sum_{i=1}^G \frac{1}{|o_i|} \sum_{k=1}^{|o_i|} \min(\rho_i^k A_i, \text{clip}(\rho_i^k, 1 - \varepsilon, 1 + \varepsilon) A_i) - \beta D_{\text{KL}}[\pi_\theta(\cdot|q) \parallel \pi_{\text{ref}}(\cdot|q)] \right], \quad (4)$$

413 where $\rho_i^k = \frac{\pi_\theta(o_i^k | q, o_i^{<k})}{\pi_{\theta_{\text{old}}}(o_i^k | q, o_i^{<k})}$ is the likelihood ratio.

414 For a query q , GRPO samples G responses $\{o_1, \dots, o_G\}$ from the behavior policy $\pi_{\theta_{\text{old}}}$ and assigns a
 415 *single sequence-level* advantage per response. Following Liu et al. [2025b], we use the unnormalized
 416 group-relative advantage $A_i = r(o_i) - \frac{1}{G} \sum_{j=1}^G r(o_j)$, where r is the reward function. This scalar
 417 A_i is shared by *all tokens* in o_i when forming the tokenwise objective.

418 **Applying Policy Gradient Methods to Diffusion LLMs** Applying GRPO to dLLMs is nontriv-
 419 ial. The objective in Equation (4) requires (i) *token-level* probabilities for importance ratios and
 420 (ii) *sequence-level* probabilities for KL regularization. Autoregressive models provide per-token
 421 conditionals via sequential factorization, enabling one-pass sequence scoring by the chain rule:
 422 $\log \pi_{\text{AR}}(o | q) = \sum_{k=1}^{|o|} \log \pi_{\text{AR}}(o^k | q, o^{<k})$. Accordingly, the reverse-KL decomposes as

$$D_{\text{KL}}[\pi_\theta(\cdot | q) \parallel \pi_{\text{ref}}(\cdot | q)] = \mathbb{E}_{o \sim \pi_\theta(\cdot | q)} \left[\sum_{k=1}^{|o|} \log \frac{\pi_\theta(o^k | q, o^{<k})}{\pi_{\text{ref}}(o^k | q, o^{<k})} \right]. \quad (5)$$

423 In contrast, dLLMs do not admit a sequential factorization of $\pi(o | q)$. dLLM’s generation invokes the
 424 unmasking predictor f_θ across M denoising steps, making π_θ a composition of M mappings. Exact
 425 tokenwise probabilities would require marginalization over denoising trajectories and maintaining
 426 (and differentiating through) full denoising trajectories, which is computationally prohibitive. To
 427 address this, recent work develops efficient approximations for policy optimization in masked
 428 diffusion LLMs. DiffuGRPO [Zhao et al., 2025] employs a mean-field approximation that yields
 429 *single-pass* estimates of both token-level and sequence-level terms, replacing explicit multi-step
 430 unrolling with a single-sample Monte Carlo estimate. While this introduces bias relative to the
 431 exact diffusion policy, it provides a practical framework for GRPO-style optimization on dLLMs.
 432 In our method, we adopt the mean-field estimators of Zhao et al. [2025] to compute the token-level
 433 importance ratios ρ_i^k and the reverse-KL term with one forward pass per policy.

434 C Related Work

435 C.1 Diffusion Language Models

436 Diffusion language models was first explored through continuous approaches that map discrete text
 437 to continuous representations, including learned embeddings, sequence-to-sequence conditioning,
 438 and binary bit representations [Chen et al., 2022, Li et al., 2022, Gong et al., 2023]. Recently, discrete
 439 diffusion language models have been scaled up significantly, with masked diffusion established as a
 440 specific instance of discrete diffusion [Austin et al., 2021, Sahoo et al., 2024, Shi et al., 2024, Ou
 441 et al., 2024, Nie et al., 2024]. Notable developments include DiffuLLaMA [Gong et al., 2025a] and
 442 Dream [Ye et al., 2025], both adapted from pretrained autoregressive LLMs. LLaDA [Nie et al.,
 443 2025] represents a breakthrough as a masked diffusion LLM trained from scratch using full-attention,
 444 achieving performance comparable to similarly-sized autoregressive models. These approaches are
 445 predominantly based on masked modeling. Unlike these full-attention dLLMs, Block Diffusion
 446 [Arriola et al., 2025] introduced a hybrid approach that models sequences block-by-block while
 447 applying diffusion within each block, enabling flexible length generation and improved inference
 448 efficiency through kv-caching. Recent commercial models like Mercury [Inception Labs et al., 2025]
 449 and Gemini Diffusion [DeepMind, 2025] have demonstrated the practical viability of diffusion-based
 450 code generation, achieving performance comparable to leading autoregressive models while offering
 451 significantly faster inference. More recent works have introduced caching and parallel decoding
 452 algorithms [Wu et al., 2025, Liu et al., 2025a, Ma et al., 2025, Israel et al., 2025, Sahoo et al., 2025,
 453 Hu et al., 2025] that significantly improve inference efficiency for masked diffusion language models.
 454 In this work, we focus on full-attention masked dLLMs.

455 C.2 Reinforcement Learning for Diffusion Language Models

456 Applying reinforcement learning to diffusion language models presents unique challenges compared
 457 to autoregressive models. The primary obstacle is the intractability of likelihood functions in
 458 diffusion models, which necessitates approximating response likelihoods for policy optimization. This

requirement introduces computational overhead and potential bias, particularly when approximation errors occur in policy ratios used for importance sampling. d1 proposed diffu-GRPO [Zhao et al., 2025] which adopts an efficient approximation through mean-field approximation. MMaDA [Yang et al., 2025] and diffucoder’s coupled-GRPO [Gong et al., 2025b] further improve the masking strategy in log probabilities estimation to achieve better learning efficiency. LLaDA 1.5 [Zhu et al., 2025] tackles the variance issues in ELBO-based likelihood estimates through preference optimization. Recently, wd1 [Tang et al., 2025] addresses these challenges by reformulating policy optimization as a weighted likelihood objective that eliminates the need for policy ratios. SDPO [Han et al., 2025] decomposes the diffusion trajectory alignment problem into stepwise subproblems that align the posterior at each diffusion step. Our inpainting method can also be applicable to some of the above online RL methods.

Additionally, a closely related work in RL for AR LLMs is Prefix-RFT [Huang et al., 2025], which samples prefixes from demonstrations to guide online exploration, though this is limited to left-to-right sequential generation that does not leverage the bidirectional conditioning capabilities of diffusion LLMs.

D Detailed Experiments Results

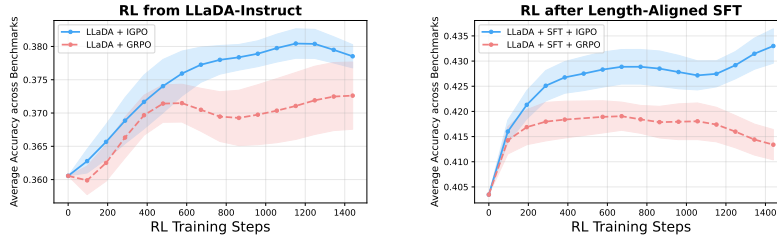


Figure 5: **RL training curves of IGPO versus normal GRPO sampling.** (a) Starting from LLaDA-8B-Instruct. (b) Starting from the *length-aligned* SFT checkpoint. IGPO exhibits superior and more stable training performance under both initialization checkpoints compared to standard GRPO sampling. Results are averaged over 3 random seeds across four mathematical reasoning benchmarks (GSM8K, MATH500, AMC and Minerva Math), with standard errors shown as shaded regions.

E Length-Aligned SFT: SFT trace revision length distribution comparison

As illustrated in Figure 6, the original OpenR1-Math-220K dataset exhibits substantial token length diversity, with reasoning traces extending beyond 10,000 tokens while LLaDA’s maximum context length is only 4096 tokens. Naively applying SFT on this dataset would result in many truncated sequences, and even for samples within the 4096-token limit, significant distribution mismatch persists across training phases—we use 256 tokens for RL sampling and 512 tokens for evaluation. Our rewriting using LLaMA-4-Maverick successfully constrains all traces to under 1500 tokens, creating alignment between SFT training, RL sampling, and evaluation phases. Additionally, while reflective behavior has been found helpful for LLaDA in prior work [Zhao et al., 2025], the excessive repeated reflective patterns in the original dataset are unsuitable for its constrained generation space. The rewriting process eliminates this redundancy while preserving essential reasoning structure.

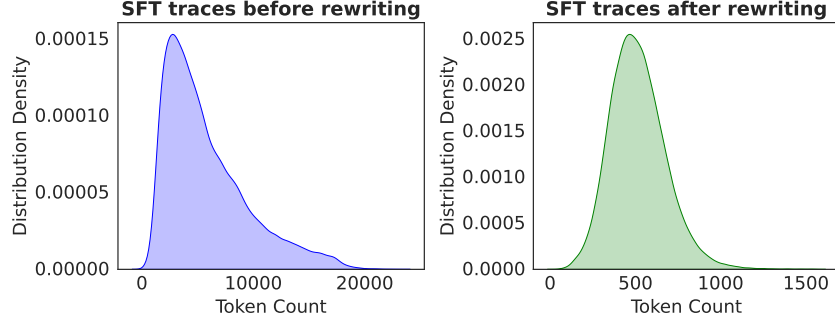


Figure 6: **Token Length Distribution of SFT Dataset Before and After Revision.** Comparison of token length distributions for the OpenR1-Math-220K dataset (94k math problems). After revision using LLaMA-4-Maverick, token lengths are constrained to below 1500 tokens, eliminating the extreme range of the original dataset where traces could exceed 20,000 tokens. This addresses the generation length mismatch across SFT training, RL sampling (256 tokens), and evaluation (512 tokens) phases.

486 F Temperature Selection for RL Training

487 Following the methodology established by Polaris An et al. [2025] for scaling reinforcement learning
 488 on advanced reasoning models, we conduct a systematic analysis to determine the optimal sampling
 489 temperature for our RL training process. We evaluate our model’s performance across different
 490 sampling temperatures by analyzing both Pass@5 and Average@5 scores on the MATH500 dataset.
 491 We also divide three temperature regions: low temperatures (≤ 0.8) yield high accuracy but reduced
 492 diversity in generated rollouts, restricting the model’s ability to explore diverse reasoning paths; high
 493 temperatures (≥ 1.6) preserve rollout diversity but significantly degrade accuracy due to increased
 494 noise in token generation; and the middle Controlled Exploration Zone (0.9-1.5) provides the optimal
 495 trade-off between maintaining reasonable accuracy and achieving sufficient diversity for effective RL
 496 training. Based on this analysis, we select temperature $T = 1.2$ to balance exploration with sample
 497 quality and provide sufficient diversity for RL training.

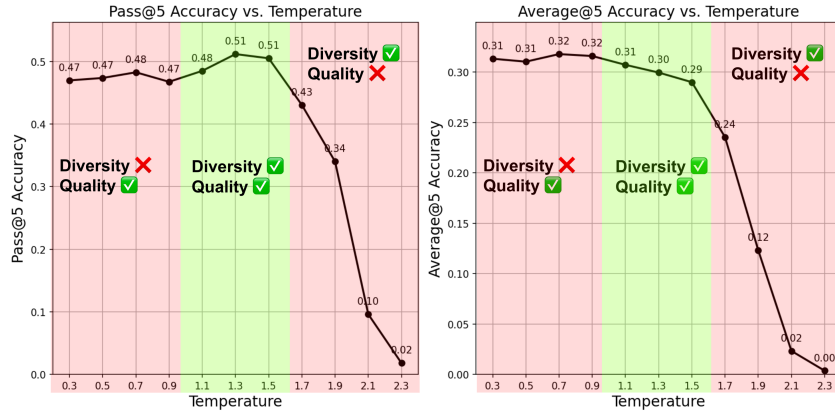


Figure 7: Performance analysis across different sampling temperatures on MATH500. The plot shows Pass@5 and Average@5 scores, revealing three distinct regions: low temperatures with high accuracy but low diversity, high temperatures with preserved diversity but degraded accuracy, and the middle region offering the desired trade-off. We select $T = 1.2$ for our RL training.

498 G Ablation: SFT on hint traces then apply GRPO vs IGPO

499 In our RL training setup, we assume access to ground-truth reasoning traces for every query in
 500 the training dataset. To investigate whether direct supervised fine-tuning on these traces provides
 501 comparable benefits to our elastic inpainting approach, we conduct an ablation study comparing
 502 two strategies: (1) applying SFT on the RL dataset’s reasoning traces followed by standard GRPO
 503 sampling, versus (2) directly applying IGPO with elastic hint injection only when all generated
 504 responses are incorrect.

505 Specifically, we first fine-tune the LLaDA-8B-Instruct model on the MetaMath dataset’s reasoning
 506 traces for 20 epochs, then apply standard GRPO sampling. We compare this against our IGPO
 507 approach, which selectively injects partial reasoning hints from the same MetaMath dataset only
 508 when zero-advantage scenarios occur (i.e., when all sampled responses yield incorrect rewards).

509 The results in Figure 8 demonstrate that IGPO consistently outperforms the SFT-first variant. Notably,
 510 after SFT on the MetaMath dataset for 20 epochs, the model’s initial performance drops significantly
 511 compared to the original LLaDA-8B-Instruct baseline. This degradation occurs because the MetaMath
 512 dataset contains very concise reasoning traces, many shorter than our 256-token generation length
 513 limit. Consequently, the model adopts overly concise reasoning patterns that prove insufficient for
 514 the challenging problems in our evaluation benchmarks (such as AMC and Minerva).

515 While subsequent RL training can recover performance to some extent—as evidenced by the rapid
 516 improvement in early training steps—it ultimately fails to match the effectiveness of IGPO. This
 517 comparison highlights two key advantages of our approach: (1) the effectiveness of applying inpaint-
 518 ing guidance selectively only when the model struggles with specific queries, rather than forcing a
 519 uniform reasoning style through SFT, and (2) the critical importance of reducing all-wrong group
 520 occurrences, which successfully recovers gradient signals from otherwise degenerate zero-advantage
 521 scenarios.

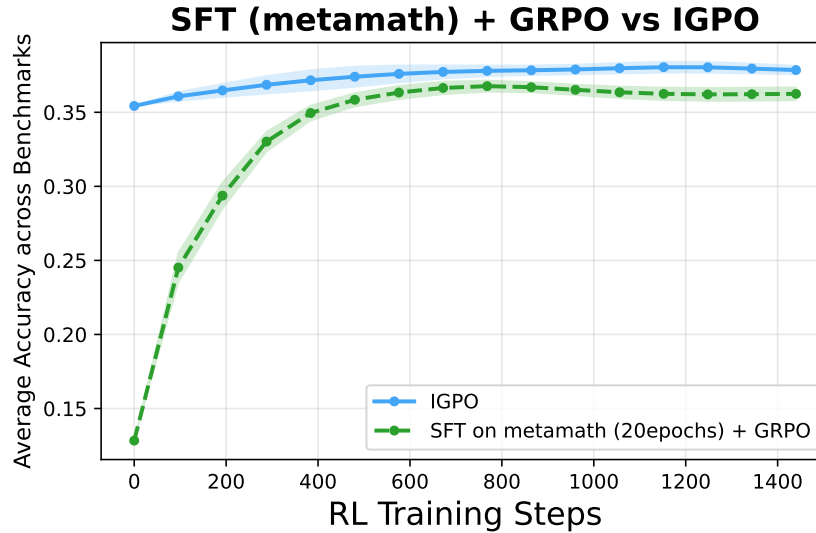


Figure 8: Comparison of SFT-first approach versus direct IGPO application. The SFT-first strategy involves fine-tuning on MetaMath reasoning traces for 20 epochs followed by standard GRPO, while IGPO applies inpainting-guided exploration elastically only during zero-advantage scenarios. IGPO demonstrates superior and more stable performance, avoiding the performance degradation caused by overly concise reasoning patterns learned during SFT on short traces. Results are averaged across four mathematical reasoning benchmarks with standard errors shown as shaded regions.

Table 2: Training Hyperparameters

Parameter	Value
SFT Training Parameters	
Per Device Train Batch Size	4
Hardware Configuration	8×8 H100 GPUs
Gradient Accumulation Steps	8
Learning Rate	5×10^{-6}
LR Schedule	Warmup-stable-decay
LR Warmup Steps	200
LR Min Value	1×10^{-6}
LR Decay Period	Final 10% of steps
Number of Epochs	100
RL Sampling Parameters	
RL Online Sampling Generation Length L	256
Diffusion Steps	128
Block Length	32
Sampling Temperature	1.2
Generations Per Group G	8
RL Training Parameters	
Per Device Train Batch Size	8
Hardware Configuration	8×8 H100 GPUs
Gradient Accumulation Steps	1
Effective Batch Size	512
KL Beta β	0.01
Policy Gradient Inner Iterations per Generation μ	4
Learning Rate	5×10^{-7}
LR Schedule	Linear decay to 0
LR Warmup Steps	50
LR Decay Period	10 epochs
Training Steps	1440
Clip Ratio Epsilon ε	0.2
IGPO Specific Parameters	
Chunk Size $ c_j \sim \mathcal{U}[s_{\min}, s_{\max}]$	$\mathcal{U}[5, 10]$
Inpainting Ratio $\eta_i \sim \mathcal{U}[\eta_{\text{low}}, \eta_{\text{high}}]$	$\mathcal{U}[0.2, 0.6]$
replacement fraction λ	0.5
Entropy-based Gradient Filtering for Inpainted Tokens τ	0.2

Blue text: Question Prompt **Green text:** Injected Hints **Black text:** Inpainted Generation

Problem: Circle C has radius 6 cm. How many square centimeters are in the area of the largest possible inscribed triangle having one side as a diameter of circle C ?

Inpainting Input At Denosing Step 0:

```
<|start_header_id|>user<|end_header_id|>
Respond in the following format:
<reasoning>
...
</reasoning>
<answer>
\ boxed{<Your answer>}
</answer>
Circle  $C$  has radius 6 cm. How many square centimeters are in the area
of the largest possible inscribed triangle having one side as a diameter of
circle  $C$ ?<|eot_id|><|start_header_id|>assistant<|end_header_id|>

<|mdm_mask|><|mdm_mask|>...<|mdm_mask|>
Since the diameter is twice the radius, the base of the triangle will have
length  $2 \times 6 = 12$  cm.
To <|mdm_mask|>...<|mdm_mask|> , we want the height to be as large
<|mdm_mask|>...<|mdm_mask|>
of the triangle is the distance from the midpoint of the base to the
circumference of the circle.
This is equal to the radius of the circle, <|mdm_mask|>...<|mdm_mask|>
<|mdm_mask|>...<|mdm_mask|> <|mdm_mask|>...<|mdm_mask|>
12 \ times <|mdm_mask|>...<|mdm_mask|>
```

Inpainted Generation After Denoising:

```
<reasoning>
We know to find the largest possible inscribed triangle in circle  $C$ , the base
of the triangle should be the diameter of the circle.
Since the diameter is twice the radius, the base of the triangle will have
length  $2 \times 6 = 12$  cm.
To find the largest possible area of the triangle, we want the height to be
as large as possible.
The largest possible height of the triangle is the distance from the midpoint of
the base to the
circumference of the circle.
This is equal to the radius of the circle, which is 6 cm.
Therefore, the area of the largest possible inscribed triangle is given by the
formula for the area of a triangle, which is  $\frac{1}{2} \times \text{base} \times \text{height} = \frac{1}{2} \times 12 \times 6 = 36$  square centimeters.
</reasoning>
<answer>
\ boxed{36}
</answer>
```

524 J Evaluation Hyperparameters

525 We evaluate all models with a fixed decoding setup per benchmark. Unless otherwise noted, the same
526 prompts and decoding hyperparameters are applied to both LLaDA-8B-Instruct and LLaDA-1.5.

Table 3: Evaluation hyperparameters by benchmark.

Benchmark	Temp.	Block Len.	Diffusion Steps	Gen. Len.	Metric
GSM8K	0.0	32	512	512	pass@1
MATH500	0.0	32	512	512	pass@1
AMC	0.5	128	512	512	avg@16
Minerva	0.0	128	512	512	pass@1

527 K Prompt for SFT traces revision

Prompt for SFT traces revision sent to LLaMA 4 Maverick

Please rewrite the original solution to make it more concise and easier to understand without changing the details. Please put the explanation in the solution between `<reasoning>` and `</reasoning>` and put the final numerical answer between `<answer>` and `</answer>` in boxed format. Please shorten or rewrite the rewritten solution to a random length between 100 and 1000 words while keeping sufficient details of the reasoning steps. Please do not return anything other than the rewritten solution.

Example:

```
<reasoning>
xxx
</reasoning>
```

```
<answer>
\boxed{14}
</answer>
```

Original solution: {generations}.

Your response:

SFT trace before rewritten by LLaMA 4 Maverick

Problem: Find all primes between 1 and 15.

<think>

Okay, so I need to find all the prime numbers between 1 and 15. Hmm, let's start by remembering what a prime number is. From what I recall, a prime number is a natural number greater than 1 that has no positive divisors other than 1 and itself. So, numbers like 2, 3, 5, etc., are primes because you can't divide them evenly by any other number besides 1 and the number itself. On the other hand, numbers like 4, 6, 8, and so on are not primes because they can be divided by other numbers. For example, 4 can be divided by 2, so it's not prime. Alright, so the task is to list all primes between 1 and 15. Let me list out all numbers from 1 to 15 first to check them one by one. The numbers are: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15.

Now, let's go through each number and see if it's prime. Starting with 1. [Wait, I remember that by definition, prime numbers are greater than 1.](#) So 1 is not considered a prime. So we can skip 1.

Next up is 2. Let's check if 2 is prime. The divisors of 2 are 1 and 2. There are no other divisors, so yes, 2 is a prime number.

Moving on to 3. Similarly, the divisors are 1 and 3. No other numbers divide into 3, so 3 is prime.

Next is 4. Let's check divisors. 4 divided by 2 is 2, which is an integer, so 4 is not prime. It can be divided by 2, so it's composite.

Then 5. The divisors of 5 are 1 and 5. No other numbers divide into 5, so 5 is prime.

6. Checking divisors. 6 divided by 2 is 3, so it's divisible by 2 and 3. Not prime.

7. Divisors are 1 and 7. No other divisors, so 7 is prime.

8. Divided by 2 gives 4, so 8 is not prime.

9. Divided by 3 gives 3, so 9 is not prime (since 3 times 3 is 9).

10. Divided by 2 is 5, so 10 is not prime.

11. Let's see, 11. Divided by 2? 11 divided by 2 is 5.5, not an integer. Divided by 3? 11 divided by 3 is approximately 3.666... Not an integer. Divided by 4? 2.75, nope. 5? 2.2, nope. Since 11 is a prime number because none of these divisions result in integers, so 11 is prime.

12. Divided by 2 is 6, so composite.

13. Checking divisors. 13 divided by 2 is 6.5, 13 divided by 3 is about 4.333..., 4? 3.25, 5? 2.6, 6? 2.166... None of these are integers. So 13 is prime.

14. Divided by 2 is 7, so composite.

15. Divided by 3 is 5, so composite.

Let me recap the numbers between 1 and 15 and their primality:

1: Not prime

2: Prime

3: Prime

4: Not

5: Prime

6: Not

7: Prime

8: Not

9: Not

10: Not

11: Prime

12: Not

13: Prime

14: Not

15: Not

So compiling the primes from this list: 2, 3, 5, 7, 11, 13.

[Wait a second, let me verify each one again just to be sure I didn't make a mistake.](#)

Starting with 2: Definitely prime. It's the only even prime.

3: Prime, yes.

5: Prime, yes.

7: Prime. Correct.

11: Let's check again. 11 divided by 2 is not whole. 3×3 is 9, 3×4 is 12. So no. 5×2 is 10, 5×3 is 15. So 11 is prime.

13: Similarly, 13 divided by 2 is 6.5, 3 is 4.333, 5 is 2.6. So, prime.

Yes, those all check out. So primes between 1 and 15 are 2, 3, 5, 7, 11, and 13.

But hold on, sometimes people might make a mistake with 9. Wait, 9 is 3 times 3, so that's composite, which I had down. 15 is 3 times 5, composite. So all the non-primes are accounted for.

Is there any number I might have missed? Let's see. Let's list all numbers again:

2, 3, 5, 7, 11, 13. Are there any others?

Well, 1 is excluded. 4,6,8,9,10,12,14,15 are composite. So no, those six numbers are all primes between 1 and 15.

Wait, but the question says "between 1 and 15." Does that include 15? Hmm, the wording is "between 1 and 15," which could be a bit ambiguous. But usually, when someone says "between 1 and 15," they mean including both endpoints. However, since 15 is composite, it's excluded either way. So primes are 2,3,5,7,11,13.

Yes, that seems correct. I don't think I made any mistakes here. Let me cross-verify with another method, like the Sieve of Eratosthenes. The Sieve is a method where you list all numbers and cross out multiples.

Starting with 2, cross out multiples: 4,6,8,10,12,14.

Then 3, cross out multiples: 6,9,12,15.

Then 5, cross out 10,15.

7, cross out 14.

11 and 13 are primes, and their multiples are beyond 15. So remaining numbers are 2,3,5,7,11,13. Which matches our previous list.

Therefore, the primes between 1 and 15 are 2,3,5,7,11,13.

****Final Answer****

The primes between 1 and 15 are 2, 3, 5, 7, 11, and 13.

</think>

To find all the prime numbers between 1 and 15, we start by recalling that a prime number is a natural number greater than 1 that has no positive divisors other than 1 and itself. We then check each number from 1 to 15:

- ****1****: Not a prime number (by definition).
- ****2****: Prime (divisors: 1, 2).
- ****3****: Prime (divisors: 1, 3).
- ****4****: Not prime (divisible by 2).
- ****5****: Prime (divisors: 1, 5).
- ****6****: Not prime (divisible by 2, 3).
- ****7****: Prime (divisors: 1, 7).
- ****8****: Not prime (divisible by 2).
- ****9****: Not prime (divisible by 3).
- ****10****: Not prime (divisible by 2, 5).
- ****11****: Prime (divisors: 1, 11).
- ****12****: Not prime (divisible by 2, 3, 4, 6).
- ****13****: Prime (divisors: 1, 13).
- ****14****: Not prime (divisible by 2, 7).
- ****15****: Not prime (divisible by 3, 5).

Using the Sieve of Eratosthenes method to cross out non-prime numbers confirms the primes listed above. Therefore, the prime numbers between 1 and 15 are:

2, 3, 5, 7, 11, 13

SFT trace after rewritten by LLaMA 4 Maverick

Problem: Find all primes between 1 and 15.

<reasoning>

To find all the prime numbers between 1 and 15, we recall that a prime number is a natural number greater than 1 that has no positive divisors other than 1 and itself.

We then check each number from 1 to 15 to determine if it is prime.

- **1***: Not a prime number (by definition).
- **2***: Prime (divisors: 1, 2).
- **3***: Prime (divisors: 1, 3).
- **4***: Not prime (divisible by 2).
- **5***: Prime (divisors: 1, 5).
- **6***: Not prime (divisible by 2, 3).
- **7***: Prime (divisors: 1, 7).
- **8***: Not prime (divisible by 2).
- **9***: Not prime (divisible by 3).
- **10***: Not prime (divisible by 2, 5).
- **11***: Prime (divisors: 1, 11).
- **12***: Not prime (divisible by 2, 3, 4, 6).
- **13***: Prime (divisors: 1, 13).
- **14***: Not prime (divisible by 2, 7).
- **15***: Not prime (divisible by 3, 5).

The Sieve of Eratosthenes method confirms the primes listed above.

</reasoning>

<answer>

2, 3, 5, 7, 11, 13

</answer>