BEYOND NEXT-TOKEN PREDICTION: EXPANDING LARGE LANGUAGE MODEL REASONING COVERAGE WITH BLOCK DIFFUSION OBJECTIVE

Anonymous authors
Paper under double-blind review

ABSTRACT

Autoregressive (AR) language models have long been the dominant paradigm for modern large language models (LLMs). Recently, Diffusion Language Models (dLMs) have emerged as an alternative paradigm, distinguished by their ability to generate sequences in arbitrary orders. However, recent studies suggest that arbitrary-order generation may, in fact, limit the reasoning capabilities of dLMs relative to conventional autoregressive decoding strategies. This observation raises an intriguing possibility: *if dLMs perform best under autoregressive decoding, can the diffusion training objective itself provide an alternative to the next-token prediction objective for training capable autoregressive models?* Through a fully controlled experimental framework, we systematically investigate how the two training objectives shape reasoning capabilities across pre-training, mid-training, and RL post-training, evaluated on both *extrapolative generalization* and *contextual generalization*. Our experiments reveal several key insights into the impact of the diffusion objective. **Firstly**, the diffusion objective can yield stronger reasoning potential and enables greater gains during reinforcement-learning post-training. **Secondly**, diffusion-trained models exhibit stronger contextual generalization and greater robustness to reward hacking. **Thirdly**, the two objectives exhibit distinct advantages during mid-training: next-token prediction provides stronger gains on in-domain tasks, whereas block diffusion is more effective for out-of-domain generalization. Together, these findings highlight the potential of mask diffusion as an alternative training objective for reasoning language models.

1 INTRODUCTION

Autoregressive (AR) language models have long been the dominant paradigm for modern large language models (LLMs). Recently, diffusion language models (dLMs) have emerged as a promising alternative (Nie et al., 2025; Ye et al., 2025b; Zhu et al., 2026; Khanna et al., 2025). Rather than generating tokens in a fixed left-to-right order, dLMs generate sequences through discrete denoising. This formulation offers two potential advantages over AR models: parallel decoding, which generates multiple tokens per forward pass, and the capability of arbitrary-order generation.

While the efficiency gain from parallel decoding has been well established (Wu et al., 2026b; Khanna et al., 2025; Song et al., 2025; Wu et al., 2026a), the benefits of arbitrary-order generation for reasoning remain questionable. Prior work has shown that highly parallel decoding often converges toward left-to-right generation (Nie et al., 2025; Israel et al., 2025; Wu et al., 2026b; Gong et al., 2026). More importantly, Ni et al. (2026) demonstrates that restricting dLMs to the standard AR order can expand the reasoning solution space compared to arbitrary-order generation. These findings not only challenge the view that arbitrary-order generation is responsible for the reasoning benefits of diffusion models, but also suggest that *models trained*

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

with a diffusion objective could also yield strong reasoning capabilities in AR order. Furthermore, prior work has highlighted the effectiveness of the mask diffusion, particularly block diffusion objectives, in extracting richer learning signals and being less prone to overfitting compared to the next-token prediction objective (Prabhudesai et al., 2025; Ni et al., 2025). This raises a fundamental question:

Can the block diffusion objective serve as a more effective alternative to next-token prediction for training capable autoregressive models?

Studying this question directly in LLM reasoning is challenging due to the lack of *controlled* training environments. Modern LMs are trained on massive pre-training datasets whose composition is largely unknown. Furthermore, differences in model architectures and training setups make it difficult to isolate the causal effects between the two training objectives.

To rigorously answer this question, we build upon the synthetic framework of Zhang et al. (2026a), which is designed around three principles: (i) *fully controllable reasoning tasks* with explicit atomic operations and DAG-defined dependency structures; (ii) *parseable reasoning processes* that enable process-level evaluation; and (iii) *systematic manipulation* of pre-/mid-/post-training distributions to attribute causal effects to each stage, along with two key generalization dimensions: *Extrapolative (Depth) generalization* assesses whether models can solve problems more complex than those encountered during pre-training by composing learned primitives in deeper structures, and *Contextual (Breadth) generalization* evaluates whether models can transfer reasoning skills across novel surface contexts that share equivalent underlying logic.

Using this framework, we conduct extensive experiments to systematically compare next-token prediction and mask diffusion objectives across different stages of training. Our experiments reveal several insights into how the choice of training objective shapes reasoning capabilities throughout pre-training, mid-training, and RL post-training as follows:

- **Pre-trained models with the mask diffusion objective yield stronger reasoning coverage than the next-token prediction objective.** Through Pass@k analysis, we show that under the same autoregressive decoding strategy, models trained with the mask diffusion objective yield broader reasoning coverage compared to next-token prediction-trained models, especially on challenging out-of-distribution tasks.

- **Reinforcement learning amplifies the benefits of block diffusion across both types of generalization.** We further investigate how RL fine-tuning affects models pretrained with different objectives. Our results show that block diffusion provides a stronger initialization for RL, leading to larger capability gains on challenging *extrapolative generalization* tasks. For *contextual generalization*, RL from a block-diffusion initialization can incentivize generalization even under extremely sparse exposure (e.g., 0.1%), whereas next-token initialization fails to transfer. Additionally, initializing with a diffusion objective provides stronger robustness to reward hacking during RL training under outcome-only rewards, suggesting that block diffusion not only improves capability gains from RL but also promotes more reliable generalization under imperfect supervision.

- **Mid-training with next-token prediction provides stronger gains for in-domain tasks, while block diffusion is superior for out-of-domain.** Under the same AR initialization, we further compare the two objectives during mid-training and find a clear trade-off: next-token prediction yields stronger improvements on in-domain tasks, whereas block diffusion consistently improves performance on challenging out-of-domain tasks. This suggests that the choice of mid-training objective should depend on the desired generalization regime, with next-token prediction favoring in-distribution tasks, while block diffusion provides superior performance on out-of-distribution tasks

2 Related Works

Diffusion language models. The role of diffusion models, particularly masked diffusion models, in discrete text generation has been the subject of extensive discussion. Early work has substantially narrowed the performance gap (Arriola et al., 2025; Lou et al., 2024; Sahoo et al., 2024; Dieleman, 2023), with large-scale models such as LLaDA Nie et al. (2025) and Dream Ye et al. (2025b) achieving performance competitive with AR models. Ni et al. (2025); Prabhudesai et al. (2025) further demonstrate that dLMs can outperform AR models in data-constrained settings. A key advantage of masked diffusion models is their non-AR formulation, which allows tokens to be generated in arbitrary order. This flexibility has been hypothesized to benefit complex reasoning by overcoming the myopic nature of AR generation (Kim et al., 2025; Ye et al., 2025a) and enabling greater creativity Nagarajan et al. (2025).

However, the extent to which arbitrary-order generation itself provides a fundamental advantage has recently been questioned. In particular, Ni et al. (2026) demonstrate that AR generation can represent a broader solution space than arbitrary-order generation. Similarly, Du et al. (2025) argue that requiring a model to learn across uniformly sampled permutations yields a sub-optimal approximation of the underlying data distribution, raising an important question regarding which aspects of dLMs provide advantages over AR models. In this work, we contribute to this discussion by providing empirical evidence that the diffusion objective is a key driver of stronger generalization compared to AR objective across pre-training, mid-training, and post-training.

Reinforcement learning for Reasoning. Reinforcement learning (RL) with verifiable rewards has become a standard paradigm for improving reasoning in language models (Zeng et al., 2025; Yu et al., 2025; Lambert et al., 2025). However, its effectiveness in expanding the reasoning boundary in the diffusion regime remains largely unexplored. While several studies (Gong et al., 2026; Shen et al., 2026; Hong et al., 2026) use Pass@k to characterize the reasoning capabilities of dLMs, they primarily focus on improving performance. Recently, Ni et al. (2026) showed that AR generation achieves broader coverage than arbitrary-order generation, and that autoregressive RL fine-tuning can further improve performance. However, their analysis focuses primarily on inference-time decoding and post-training, leaving open how generation order and RL interact across the full spectrum of reasoning capabilities, from pre-training to post-training.

Understanding LMs via Controlled Experiments. Recent studies have emphasized the importance of controlled experiments for understanding the capabilities of language models (Sun et al., 2026; Yuan et al., 2026; Liu et al., 2025). For example, Yuan et al. (2026) study the compositional generalization of LLM capabilities during RL fine-tuning, while Sun et al. (2026) introduce a fully out-of-distribution task to examine whether RL can genuinely acquire new reasoning strategies. More recently, Zhang et al. (2026a) synthesize GSM-style reasoning tasks to systematically track the emergence of LM reasoning capabilities from pre-training through post-training. Building upon this line of work, we conduct controlled experiments to compare diffusion and autoregressive models across different stages of training, providing insights into how their respective training objectives affect reasoning capabilities.

3 Preliminaries

3.1 Next-token Prediction Objective

Given a sequence x, the next-token prediction (NTP) objective maximizes the likelihood of left-to-right factorization:

$$\mathcal{L}_{\text{NTP}}(\theta) = -\mathbb{E}_{x\sim\mathcal{D}}\left[\sum_{k=1}^{L}\log p_\theta(x_k|x_{<k})\right]. \tag{1}$$

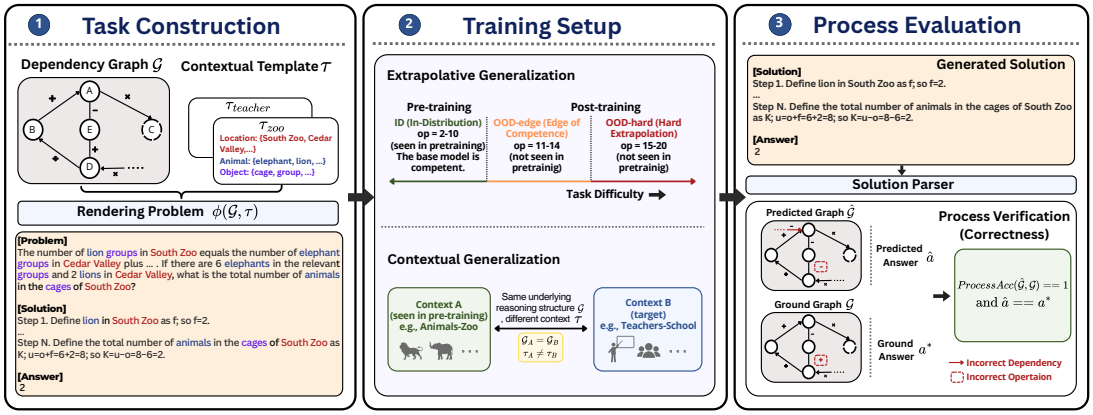

Figure 1: **Controlled synthetic reasoning framework for task construction, training, and process evaluation.** Each problem is specified by a dependency graph $\mathcal{G}$ and contextual template τ, the task setup for *extrapolative* and *contextual* generalization, and Process-verified evaluation to verify the correctness at each reasoning steps.

where L denotes the sequence length, x_k denotes the k-th token, and $x_{<k}$ denotes the subsequence preceding x_k.

3.2 Block Diffusion Objective

Block diffusion objective first partitions the sequence x into B contiguous, non-overlapping blocks $\{x^1, \ldots, x^B\}$, the training objective becomes the minimization of the expected conditional Negative Evidence Lower Bound over blocks:

$$\mathcal{L}_{\text{Diff}}(\theta, B) = -\mathbb{E}_{\substack{x\sim\mathcal{D},\\ b\sim\mathcal{U}[1,B]\\ t\sim\mathcal{U}[0,1]}} \left[\frac{1}{t} \sum_{k=1}^{L'} \mathbf{1}\left[x_{t,k}^b = [\text{MASK}] \log p_\theta(x_{0,k}^b | x_t^b, x_{<b})\right] \right] \tag{2}$$

where L' denotes the block size, x_t^b denotes the noised sequence for block b, and $x_{0,k}^b$ denotes the k-th original token in that block. The model is trained to reconstruct the original block x^b from its corrupted version x_t^b, conditioned on the preceding clean blocks $x_{<b}$. When the block size is $B = 1$, $\mathcal{L}_{\text{Diff}}(\theta, 1)$ reduces to the standard $\mathcal{L}_{\text{NTP}}$ objective.

3.3 Controlled Synthetic Reasoning Framework.

Setup. We follow the controllable synthetic reasoning data-generation setup of Zhang et al. (2026a), which builds upon the GSM-Infinite framework Zhou et al. (2025). Specifically, each reasoning problem is constructed from a *directed acyclic graph* (DAG) $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, which captures the underlying reasoning structure, where nodes $v \in \mathcal{V}$ correspond to variables and directed edges $e \in \mathcal{E}$ denote dependencies between them. The graph culminates in a designated answer node v^*, which yields the final answer a^*. The *reasoning complexity* of a task is controlled by the number of arithmetic operations:

$$\text{op}(\mathcal{G}) = |\mathcal{E}| \tag{3}$$

The dependency graph $\mathcal{G}$ is then rendered from a pre-defined *contextual template* τ (e.g., *animals–zoo, teachers–school*) to produce a complete mathematical problem. By varying both the underlying dependency graph $\mathcal{G}$ and the contextual template τ, we can systematically generate a diverse set of mathematical problems,

maintaining complete control over training distributions across the pre-, mid-, and post-training stages. To assess generalization, we evaluate reasoning capabilities along two complementary axes:

Extrapolative (Depth) Generalization. This dimension evaluates a model's ability to maintain correctness as *reasoning complexity* $\mathrm{op}(\mathcal{G})$ increases Zhang et al. (2026b). A model demonstrates strong extrapolative generalization if it can correctly solve problems with operation chains that exceed those encountered during training.

Contextual (Breadth) Generalization. This dimension evaluates whether a model can transfer its reasoning primitives to novel domains that differ in surface form but share the same underlying reasoning structure. A model exhibits strong contextual generalization when its performance remains stable across changes in *contextual templates*, given the underlying graph $\mathcal{G}$ remains unchanged.

Evaluation Protocol. For each instance with a ground-truth dependency graph $(\mathcal{G}, a^*)$, we parse the generated solution into a predicted dependency graph $\hat{\mathcal{G}}$ and predicted answer $\hat{a}$. The reasoning process is evaluated at the step level by comparing the predicted and ground-truth nodes, their dependencies, and their numeric values. All reported Pass@k metrics are computed under this strict evaluation criterion to ensure ground-truth correctness.

4 Does Mask Diffusion Enable Better Extrapolative Generalization?

In this section, we empirically evaluate the performance gap between models trained with next-token prediction and mask diffusion objectives on *Extrapolative Generalization*. For dLMs, unless otherwise specified, we use Autoregressive Order, which decodes tokens from left to right. By fixing the pre-training data, model architecture, and decoding strategy, we can determine whether the mask diffusion objective provides meaningful improvements over conventional next-token prediction. The experimental setup proceeds as follows:

Pre-training. We follow the setup of Zhang et al. (2026a), where we train a Qwen3 model Yang et al. (2025) with 100M parameters on 10B tokens of pre-training data. The pre-training dataset consists of `op=2-10` operations across templates. For evaluation, we define three problem categories based on operation counts: In-Distribution (ID) problems within the pre-training range (`op=2-10`); OOD-edge problems just beyond this range (`op=11-14`), and OOD-hard problems substantially beyond the pre-training distribution (`op=15-20`).

Post-training. We apply RL algorithms using a total of 200K samples drawn from two difficulty ranges: `op=11-14` (edge), and `op=(17-20)` (hard). For AR models, we employ the standard GRPO algorithm Shao et al. (2024), while for dLMs, we use JustGRPO Ni et al. (2026), a direct application of standard GRPO to dLMs.

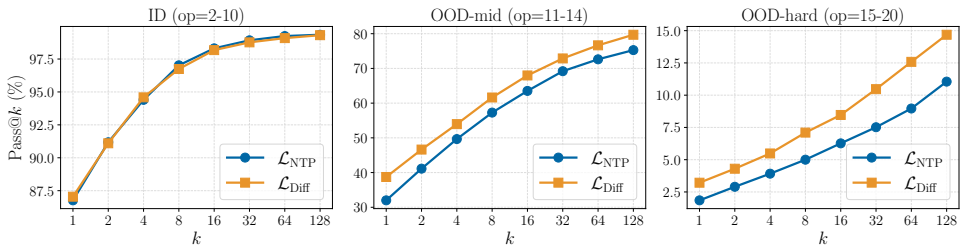

Figure 2: **Pass@k performance comparison between $\mathcal{L}_{\mathrm{NTP}}$ and $\mathcal{L}_{\mathrm{Diff}}$ objectives on three tasks:** ID (`op=(2-10)`), OOD-mid (`op=(11-14)`), OOD-hard (`op=15-20`). Models trained with $\mathcal{L}_{\mathrm{diff}}$ exhibit broader reasoning coverage on harder tasks than models trained with $\mathcal{L}_{\mathrm{NTP}}$.

Pass@k Analysis. As shown in Fig. 2, AR and dLMs exhibit nearly identical Pass@k performance across different k values on ID tasks (`op=2-10`). However, on OOD tasks (`op=11-14` and `op=15-20`), AR-order dLMs outperform the traditional AR model across different k values. Notably, this performance gap becomes more pronounced as the tasks shift toward the OOD-hard domain. These results suggest that the block diffusion objective can yield genuine capability gains beyond the next-token prediction objective, particularly on challenging OOD tasks.

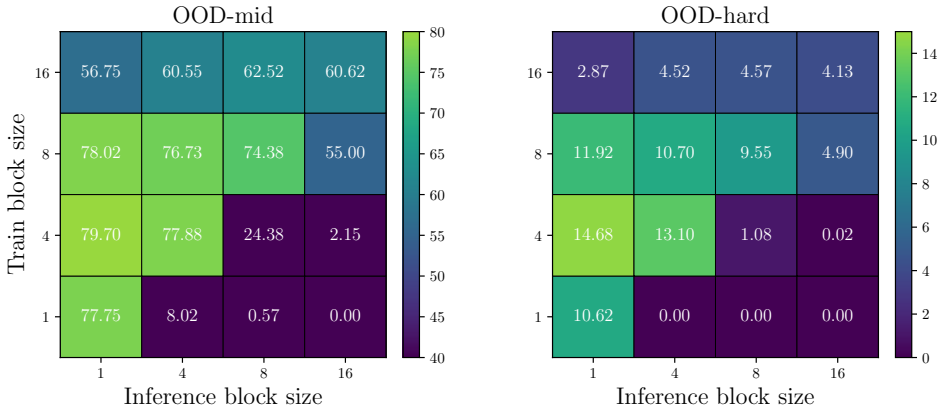

Figure 3: **Pass@128 performance across training and inference block sizes.** Models trained with smaller block sizes generally achieve stronger reasoning coverage under AR decoding, while models trained with larger block sizes yield suboptimal performance.

Robustness of AR decoding across training block sizes. In block diffusion objective, the block size B controls the model's contextual receptive field and parallelism. A smaller block size in *both* training and inference more closely approximates AR next-token prediction, while a larger block broadens the contextual receptive field and enables more flexible, arbitrary-order generation. We further analyze how varying block size during training affects the reasoning capabilities of dLMs with AR-order generation. As shown in Fig. 3, on both OOD-Mid and OOD-Hard, increasing the training block size from 1 to 4 or 8 improves reasoning coverage, with AR order (inference block size 1) achieving the highest performance, consistent with Ni et al. (2026). However, this trend deteriorates substantially when the training block size is increased to 16. Interestingly, models trained with a block size of 16 exhibit the opposite behavior during decoding, with larger inference block sizes yielding broader coverage than AR-order generation.

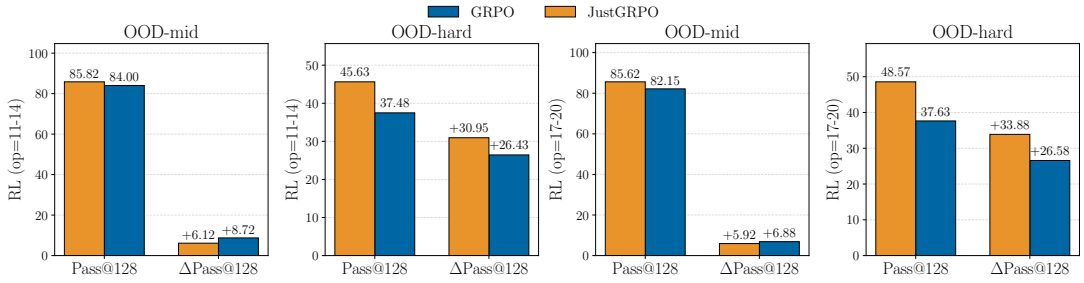

Figure 4: **Capability gains from RL fine-tuning on two difficulty ranges:** `op=11-14` **and** `op=17-20`. GRPO provides moderate gains in the OOD-mid task, while JusttGRPO shows larger improvement on OOD-hard task.

GRPO produces stronger capability gains on OOD-mid, whereas JustGRPO achieves substantially larger gains on OOD-hard. To quantify the capability gains from RL post-training, we report the relative change in Pass@k after RL fine-tuning compared to the base model:

$$\Delta\text{Pass@}k = \text{Pass@}k - \text{Base Pass@}k \tag{4}$$

As shown in Fig. 4, while JustGRPO still achieves the best Pass@128 performance on OOD-mid, GRPO provides larger improvements in Pass@128 on the OOD-mid tasks. However, on OOD-hard, JustGRPO exhibits substantially stronger generalization, achieving significant improvements over GRPO. This suggests that while GRPO can effectively improve performance on moderately challenging OOD settings, JustGRPO is better able to generalize reasoning capabilities to more challenging problems.

Main Takeaway 1

- **Next-token Prediction is not the only objective to train an Autoregressive model.** Block Diffusion objective yields broader reasoning coverage than next-token prediction, particularly on challenging OOD tasks in *Extrapolative Generalization*, with models trained with Block Diffusion objective consistently achieve best performance under AR-order decoding across different training block sizes.
- **RL produces stronger capability gains when initialized from block diffusion objective.** RL post-training further extends the reasoning capabilities gap between AR and block diffusion-trained models, especially on out-of-distribution tasks.

5 How Does Mask Diffusion Compare with Next-Token Prediction on Contextual Generalization?

Having shown that learning with diffusion yields broader reasoning coverage than the next-token prediction objective on *Extrapolative Generalization*. We next investigate *Contextual Generalization* to determine whether the mask diffusion objective provides an improvement in transferring underlying reasoning primitives to underrepresented contexts during pre-training.

Pre-training. We next study contextual generalization where the base model is pre-trained on 10B tokens consisting of `op=2-20` operations *context* A and long-tailed *context* B with atomic reasoning primitives (`op=2`), where we vary the ratio of atomic `op=2` examples to long-tailed *context* B exposure to assess whether the mask diffusion objective provides an advantage in transferring the underlying reasoning primitives to underrepresented contexts during pre-training.

Post-training. RL is applied on 200K samples, consisting of 50% context A and 50% context B, spanning `op=2-20`. In this setting, we consider *strict outcome reward*, which grants outcome rewards only when the entire reasoning process is verified as correct.

JustGRPO shows stronger contextual generalization even under minimal exposure. As shown in Fig. 5, GRPO incentivizes contextual generalization when the model has minimal exposure ($\geq 1\%$) to context B during pre-training. However, under very minimal exposure (0.1%), GRPO fails to transfer to context B, consistent with Zhang et al. (2026a). In contrast, JustGRPO still demonstrates a degree of generalization to *context* B even under a very low-exposure setting, suggesting stronger contextual generalization than GRPO.

Reward hacking in *outcome reward* supervision. We further investigate RL fine-tuning under *outcome reward*, where the reward depends solely on the final answer. At very low exposure rates (below 1%), we observe that both algorithms fail to generalize to the tail context due to *reward hacking*: both models tend to replicate the reasoning patterns learned from context A while ignoring the underlying reasoning dependencies in the context B graphs (Fig. 6). However, with minimal exposure (10%), JustGRPO achieves a non-trivial

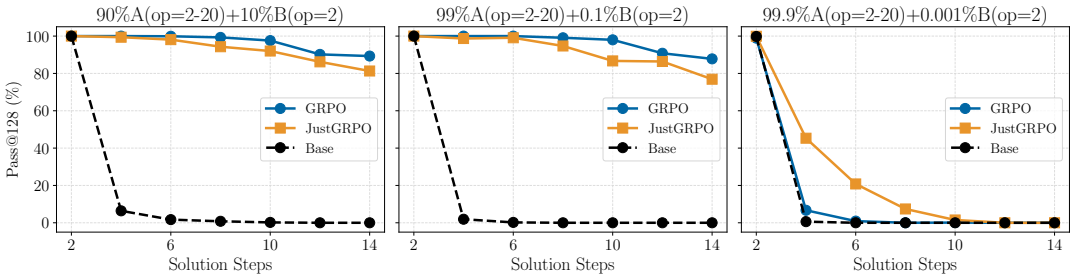

Figure 5: Pass@128 performance on *context* B for models post-trained with GRPO and JustGRPO with *strict outcome reward*, across different ratios of long-tailed *context* B.

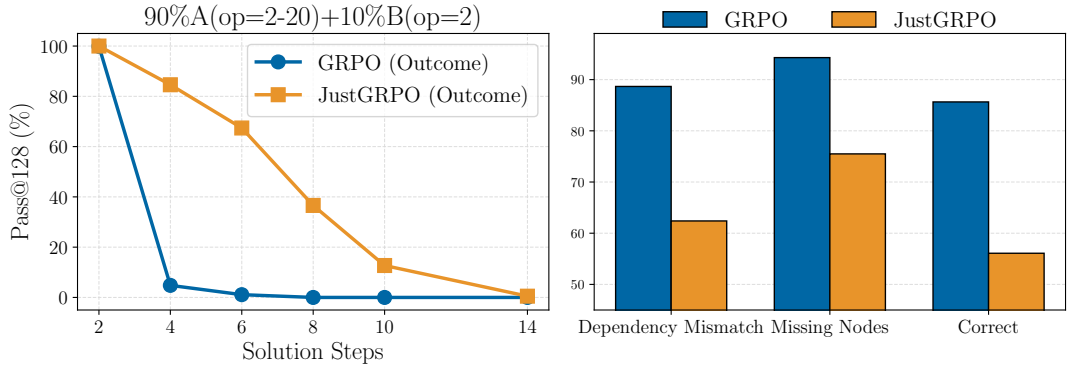

Figure 6: **Pass@128 comparison between GRPO and JustGRPO under *outcome reward*.** JustGRPO achieves stronger generalization beyond the atomic `op=2` examples seen during pre-training (left panel). In contrast, GRPO tends to exploit spurious shortcuts, yielding more correct solutions despite following invalid reasoning chains, as reflected by its increased structural error patterns (right panel).

level of generalization beyond the atomic `op=2` examples. This suggests that JustGRPO is more robust to reward hacking in contextual generalization.

Main Takeaway 2

- **JustGRPO incentivizes contextual generalization even when the base model is exposed to very minimal long-tail context.** With as little as 0.1% exposure to the new context during pre-training, JustGRPO can still exhibit cross-context generalization beyond the atomic examples observed during pre-training. In contrast, GRPO cannot induce transfer under the same limited exposure.
- **JustGRPO mitigates reward hacking and promotes more reliable generalization under outcome reward supervision.** Unlike GRPO, which can exploit spurious shortcuts to obtain correct final answers but invalid reasoning chains, JustGRPO better aligns outcome rewards with valid reasoning behavior, resulting in contextual generalization under low exposure settings ($\approx 10\%$).

6 How does Pre-training interact with Mid-training?

Mid-training is an intermediate phase between pre-training and post-training, designed to bridge data distributions and strengthen reasoning priors before downstream adaptation (Wang et al., 2025; Liu et al.,

2026). A common practice in training dLMs is to initialize from an AR checkpoint and mid-train it into a capable diffusion language model (Bie et al., 2025; Cheng et al., 2025).

This raises several key questions: *How does initialization from an AR checkpoint compare with initialization from a dLM? How does mid-training with the diffusion objective compare with mid-training with the next-token prediction objective?* In this section, we investigate the interplay between pre-training and mid-training, examining how pre-training shapes the generalization of reasoning capabilities during mid-training.

Task Setting. We perform mid-training on pre-trained checkpoints initialized from different AR/dLM configurations, while varying the training objective between next-token prediction and block diffusion. We use a default block size of $B = 4$ and train on the `op=11-14` range. Further details of the training setup are provided in Appendix B.

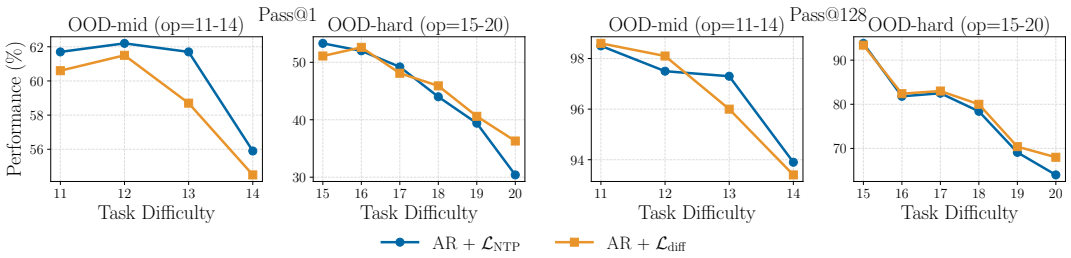

Figure 7: **Pass@1 and Pass@128 performance on extrapolative tasks with different training objectives under AR initialization.** AR initialization with next-token prediction yields the strongest performance on OOD-mid tasks, while AR initialization provides better starting points than dLM initialization and leads to strong improvements on OOD-hard tasks.

Results. As shown in Fig. 7, on **OOD-mid** tasks, AR initialization with the next-token prediction objective outperforms all other configurations, achieving the highest Pass@1 performance. In contrast, on **OOD-hard** tasks, AR initialization with the diffusion objective performs the best on the hardest instances, achieving the highest performance on both Pass@1 and Pass@128. These results suggest that next-token prediction is more effective for improving performance on tasks that are similar to mid-training data, whereas the diffusion objective provides stronger generalization to OOD tasks. Across both settings, AR initialization consistently provides the strongest starting point.

Main Takeaway 3

Mid-training objectives should be chosen based on the desired generalization regime. (i) when prioritizing in-distribution performance, next-token prediction provides stronger gains compared to the block diffusion objective; (ii) for out-of-distribution generalization, we should prioritize using the block diffusion objective instead.

7 Conclusion

In this paper, we systematically investigate the role of the block diffusion objective in shaping the reasoning capabilities of *Autoregressive* across pre-training, mid-training, and RL post-training compared to the next-token prediction objective. Through a fully controlled experimental framework, we show that the block diffusion objective can yield stronger reasoning capabilities and enable greater gains during RL post-training. Furthermore, we show that block-diffusion trained models not only yield stronger generalization but also more robustness to the reward-hacking issue. Together, these findings highlight the potential of mask diffusion as an alternative training objective for reasoning language models compared to next-token prediction objective.

AI USE STATEMENT

In this work, we used AI models solely for revising the writing of the paper. Note that revision here strictly means enhancing the clarity and readability of the text (e.g., fixing typos or constructing LaTeX tables), and not for any other purposes.

REFERENCES

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

Tiwei Bie, Maosong Cao, Kun Chen, Lun Du, Mingliang Gong, Zhuochen Gong, Yanmei Gu, Jiaqi Hu, Zenan Huang, Zhenzhong Lan, Chengxi Li, Chongxuan Li, Jianguo Li, Zehuan Li, Huabin Liu, Lin Liu, Guoshan Lu, Xiaocheng Lu, Yuxin Ma, Jianfeng Tan, Lanning Wei, Ji-Rong Wen, Yipeng Xing, Xiaolu Zhang, Junbo Zhao, Da Zheng, Jun Zhou, Junlin Zhou, Zhanchao Zhou, Liwang Zhu, and Yihong Zhuang. Llada2.0: Scaling up diffusion language models to 100b, 2025. URL https://arxiv.org/abs/2512.15745.

Shuang Cheng, Yihan Bian, Dawei Liu, Linfeng Zhang, Qian Yao, Zhongbo Tian, Wenhai Wang, Qipeng Guo, Kai Chen, Biqing Qi, and Bowen Zhou. Sdar: A synergistic diffusion-autoregression paradigm for scalable sequence generation, 2025. URL https://arxiv.org/abs/2510.06303.

Sander Dieleman. Diffusion language models, 2023. URL https://benanne.github.io/2023/01/09/diffusion-language.html.

Cunxiao Du, Xinyu Yang, Fengzhuo Zhang, Yunlong Hou, Sicheng Yu, Min Lin, and Chao Du. Understanding the limitations of diffusion llms through a probabilistic perspective. Notion blog, 2025. URL https://www.notion.so/Understanding-the-Limitations-of-Diffusion-LLMs-through-a-Probabilistic-Perspective-2ae0ba07baa88053b838d5bf0b0aad41.

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. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=58NA3unZj5.

Chunsan Hong, Seonho An, Min-Soo Kim, and Jong Chul Ye. Improving discrete diffusion unmasking policies beyond explicit reference policies. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=on6cb46OhD.

Daniel Mingyi Israel, Guy Van den Broeck, and Aditya Grover. Accelerating diffusion LLMs via adaptive parallel decoding. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=xwqTt26NJf.

Samar Khanna, Siddhant Kharbanda, Shufan Li, Harshit Varma, Eric Wang, Sawyer Birnbaum, Ziyang Luo, Yanis Miraoui, Akash Palrecha, Stefano Ermon, Aditya Grover, and Volodymyr Kuleshov. Mercury: Ultra-fast language models based on diffusion. *CoRR*, abs/2506.17298, June 2025. URL https://doi.org/10.48550/arXiv.2506.17298.

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

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James Validad Miranda, Alisa Liu, Nouha Dziri, Xinxi Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Christopher Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. Tulu 3: Pushing frontiers in open language model post-training. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=i1uGbfHHpH.

Emmy Liu, Graham Neubig, and Chenyan Xiong. Midtraining bridges pretraining and posttraining distributions. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=5PfEQzE9bf.

Mingjie Liu, Shizhe Diao, Ximing Lu, Jian Hu, Xin Dong, Yejin Choi, Jan Kautz, and Yi Dong. ProRL: Prolonged reinforcement learning expands reasoning boundaries in large language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=YPsJha5HXQ.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion language modeling by estimating the ratios of the data distribution, 2024. URL https://openreview.net/forum?id=71mqtQdKB9.

Vaishnavh Nagarajan, Chen Henry Wu, Charles Ding, and Aditi Raghunathan. Roll the dice & look before you leap: Going beyond the creative limits of next-token prediction. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=Hi0SyHMmkd.

Jinjie Ni, Qian Liu, Longxu Dou, Chao Du, Zili Wang, Hang Yan, Tianyu Pang, and Michael Qizhe Shieh. Diffusion language models are super data learners, 2025. URL https://arxiv.org/abs/2511.03276.

Zanlin Ni, Shenzhi Wang, Yang Yue, Tianyu Yu, Weilin Zhao, Yeguo Hua, Tianyi Chen, Jun Song, Cheng Yu, Bo Zheng, and Gao Huang. The flexibility trap: Rethinking the value of arbitrary order in diffusion language models. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=kpgURPRMGf.

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

Mihir Prabhudesai, Mengning Wu, Amir Zadeh, Katerina Fragkiadaki, and Deepak Pathak. Diffusion beats autoregressive in data-constrained settings. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=W5Ht05jF4c.

Subham Sekhar Sahoo, Marianne Arriola, Aaron Gokaslan, Edgar Mariano Marroquin, Alexander M Rush, Yair Schiff, Justin T Chiu, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. In *ICML 2024 Workshop on Structured Probabilistic Inference & Generative Modeling*, 2024. URL https://openreview.net/forum?id=JESNfpRIGj.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300, 2024. URL https://doi.org/10.48550/arXiv.2402.03300.

Yangyi Shen, Tianjian Feng, Jiaqi Han, Wen Wang, Tianlang Chen, Chunhua Shen, Jure Leskovec, and Stefano Ermon. Improving diffusion language model decoding through joint search in generation order and token space, 2026. URL https://arxiv.org/abs/2601.20339.

Yuxuan Song, Zheng Zhang, Cheng Luo, Pengyang Gao, Fan Xia, Hao Luo, Zheng Li, Yuehang Yang, Hongli Yu, Xingwei Qu, Yuwei Fu, Jing Su, Ge Zhang, Wenhao Huang, Mingxuan Wang, Lin Yan, Xiaoying Jia, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Yonghui Wu, and Hao Zhou. Seed diffusion: A large-scale diffusion language model with high-speed inference, 2025. URL https://arxiv.org/abs/2508.02193.

Yiyou Sun, Yuhan Cao, Pohao Huang, Haoyue Bai, Hannaneh Hajishirzi, Nouha Dziri, and Dawn Song. RL grokking recipe: How does RL unlock and transfer new algorithms in LLMs? In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=CJJ8VxOWbG.

Zengzhi Wang, Fan Zhou, Xuefeng Li, and Pengfei Liu. Octothinker: Mid-training incentivizes reinforcement learning scaling. In *2nd AI for Math Workshop @ ICML 2025*, 2025. URL https://openreview.net/forum?id=chCeUHjLzs.

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. In *The Fourteenth International Conference on Learning Representations*, 2026a. URL https://openreview.net/forum?id=1NZ3DHF9nT.

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. In *The Fourteenth International Conference on Learning Representations*, 2026b. URL https://openreview.net/forum?id=3Z3Is6hnOT.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Jiacheng Ye, Jiahui Gao, Shansan Gong, Lin Zheng, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Beyond autoregression: Discrete diffusion for complex reasoning and planning. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=NRYgUzSPZz.

Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Dream 7b: Diffusion large language models, 2025b. URL https://arxiv.org/abs/2508.15487.

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

Lifan Yuan, Weize Chen, Yuchen Zhang, Ganqu Cui, Hanbin Wang, Ziming You, Ning Ding, Zhiyuan Liu, Maosong Sun, and Hao Peng. From f(x) and g(x) to f(g(x)): LLMs learn new skills in RL by

composing old ones. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=jt7oCtYqHE.

Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun MA, and Junxian He. SimpleRL-zoo: Investigating and taming zero reinforcement learning for open base models in the wild. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=vSMCBUgrQj.

Charlie Zhang, Graham Neubig, and Xiang Yue. On the interplay of pre-training, mid-training, and RL on reasoning language models. In *Forty-third International Conference on Machine Learning*, 2026a. URL https://openreview.net/forum?id=TBaUfO9znF.

Kai Zhang, Xiangchao Chen, Bo Liu, Tianci Xue, Zeyi Liao, Zhihan Liu, Xiyao Wang, Yuting Ning, Zhaorun Chen, Xiaohan Fu, Jian Xie, Yuxuan Sun, Boyu Gou, Qi Qi, Zihang Meng, Jianwei Yang, Ning Zhang, Xian Li, Ashish Shah, Dat Huynh, Hengduo Li, Zi Yang, Xuefei Cao, Lawrence Keunho Jang, Shuyan Zhou, Jiacheng Zhu, Huan Sun, Jason E Weston, Yu Su, and Yifan Wu. Agent learning via early experience. In *Forty-third International Conference on Machine Learning*, 2026b. URL https://openreview.net/forum?id=N3dXUHY5dD.

Yang Zhou, Hongyi Liu, Zhuoming Chen, Yuandong Tian, and Beidi Chen. GSM-∞: How do your LLMs behave over infinitely increasing reasoning complexity and context length? In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=n52yyvEwPa.

Fengqi Zhu, Rongzhen Wang, Shen Nie, Xiaolu Zhang, Chunwei Wu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. LLaDA 1.5: Variance-reduced preference optimization for large language diffusion models. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11425–11460, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.524. URL https://aclanthology.org/2026.acl-long.524/.

A Data Generation Framework

In this section, the controllable data generation framework Ni et al. (2026) is defined that is used throughout the paper. We describe the graph-based representation of reasoning instances, the abstraction mechanism separating structure from numerical and linguistic instantiations, the contextual rendering function for generating natural-language problems, and the overall generation and deduplication pipeline.

A.1 Graph Construction

Each reasoning instance of the problem is represented by a directed acyclic graph (DAG) $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where each node $v_i \in \mathcal{V}$ represents a latent quantity, and a directed edge $(v_j \to v_i) \in E$ denotes a functional dependency. These dependencies are restricted to elementary arithmetic operations $f_i \in \{+, -, \times, \div\}$.

Given numerical assignments to all leaf nodes, we recursively evaluate the graph using

$$\mathrm{val}(v_i) = f_i\left(\{\mathrm{val}(v_j)\}_{j\in \mathrm{pa}(i)}\right),$$

with leaf values serving as the base cases. For a designated query node v^*, the ground-truth answer is defined as $a^* = \mathrm{val}(v^*)$. The query node v^* corresponds to the last numeric node in the topological order of the forward generator or to the distinguished unknown parameter in the equation-style reverse generator.

the structural complexity of a reasoning instance is defined by its number of arithmetic operations, which serves as the primary control variable for studying depth-wise generalization.

$$\mathrm{op}(\mathcal{G}) = |\mathcal{E}|$$

A.2 Abstract and Instance Parameters.

Following the abstraction framework described in Ni et al. (2026), we separate each reasoning instance into abstract and instance parameters, distinguishing the underlying reasoning structure from its numerical and linguistic realizations.

Abstract Parameters. Each graph $\mathcal{G}$ is defined by abstract parameters specifying its variables, dependency structure $\mathcal{E}$, and the arithmetic operation f_i associated with each node. These parameters define the symbolic reasoning structure independently of specific numerical values or linguistic realizations.

Instance Parameters. Given an abstract graph, instance parameters instantiate it with concrete numerical values to leaf nodes (e.g., "there are 12 adult lions and 7 elephant calves") and context-specific linguistic bindings (e.g., "adult lions in the city zoo"). Different numerical and linguistic instantiations can therefore share the same underlying reasoning structure.

Implicit Reasoning. The dependencies in $\mathcal{E}$ may be explicitly stated in the problem or left implicit. We partition the edge set as

$$\mathcal{E} = \mathcal{E}_{\text{explicit}} \cup \mathcal{E}_{\text{implicit}}, \qquad \mathcal{E}_{\text{explicit}} \cap \mathcal{E}_{\text{implicit}} = \emptyset,$$

where explicit edges correspond to relations stated in the text, while implicit edges represent dependencies that are part of the underlying reasoning graph but are not directly verbalized. This distinction allows us to study a model's ability to recover unspoken reasoning dependencies.

A.3 Contextual Rendering

We convert each symbolic graph into a natural-language instance through a rendering function

$$\Phi : \mathcal{G} \times \mathcal{T} \to \mathcal{X}, \qquad x = \Phi(\mathcal{G}, \tau), \tag{5}$$

where $\mathcal{G}$ denotes the underlying graph, $\tau \in \mathcal{T}$ specifies the contextual realization, and x is the resulting natural-language instance.

Contextual Templates. Each template defines a particular semantic domain in which the abstract graph is expressed. For example, the *animals–zoo* and *teachers–school* templates associate the same abstract variables with different entities and linguistic descriptions. In addition to determining the vocabulary used to describe graph variables, a template controls how graph dependencies are expressed in the problem statement. Consequently, the edges of $\mathcal{G}$ are divided into explicitly stated dependencies, $\mathcal{E}_{\text{explicit}}$, and dependencies that must be inferred, $\mathcal{E}_{\text{implicit}}$.

The rendering process preserves the underlying computational structure across different contexts. Specifically, for templates $\tau_a, \tau_b \in \mathcal{T}$, we have

$$\text{Struct}(\Phi(\mathcal{G}, \tau_a)) = \text{Struct}(\Phi(\mathcal{G}, \tau_b)) \,. \tag{6}$$

Thus, changing the contextual template modifies the semantic and linguistic realization of an instance without altering its underlying reasoning graph. This provides a controlled setting for evaluating whether a model can transfer its reasoning capability across different contextual domains.

Instance Representation. Each rendered graph is converted into an instance consisting of a question, its corresponding reasoning trace, and the final answer:

$$x = (x_{\text{q}}, x_{\text{s}}, x_{\text{a}}) \,. \tag{7}$$

Here, x_{q} specifies the natural-language problem and the target quantity or entity to be inferred. The reasoning component x_{s} provides a step-by-step reasoning following the dependency structure of $\mathcal{G}$, including the necessary intermediate computations. Finally, x_{a} denotes the final answer corresponding to the query.

This structure maintains an explicit correspondence between each natural-language instance and its underlying symbolic computation, allowing the same graph to be instantiated across multiple contexts while preserving the intended reasoning structure.

A.4 Task Setup

Reasoning generalization in language models can be characterized along two complementary dimensions: *depth*, which concerns the ability to handle increasingly complex reasoning, and *breadth*, which concerns the ability to transfer reasoning across different contextual domains. This controllable data generation framework enables these two dimensions to be isolated and systematically evaluated across the pre-, mid-, and post-training.

Model Notation. Let $f_{\theta_{\text{pre}}}$, $f_{\theta_{\text{mid}}}$, and $f_{\theta_{\text{post}}}$ denote the model obtained after pre-training, continued mid-training, and RL-based post-training, respectively. We use $\text{Correct}(f, \mathcal{G}, \tau)$ to denote whether model f correctly solves an instance generated from graph $\mathcal{G}$ under contextual template τ.

Extrapolative Generalization (Depth). We control reasoning depth through the number of operations required by a graph. For each training stage $\phi \in \{\text{pre}, \text{mid}, \text{post}\}$, let $\mathcal{O}_\phi$ denote the set of operation counts represented in its training distribution. The overall training range is therefore

$$\mathcal{O}_{\text{train}} = \mathcal{O}_{\text{pre}} \cup \mathcal{O}_{\text{mid}} \cup \mathcal{O}_{\text{post}}. \tag{8}$$

We evaluate models under two conditions. The *in-distribution (ID)* setting contains graphs whose operation counts satisfy

$$\text{op}(\mathcal{G}) \in \mathcal{O}_{\text{train}}, \tag{9}$$

whereas the *extrapolative out-of-distribution (OOD)* setting uses graphs with

$$\text{op}(\mathcal{G}) > \max \mathcal{O}_{\text{train}}. \tag{10}$$

A model demonstrates extrapolative generalization when it retains high process-verified accuracy on these longer reasoning chains while maintaining comparable performance on ID instances.

Contextual Generalization (Breadth). Contextual generalization evaluates breadth of whether the reasoning capability learned from one context transfers to another. Since the same graph can be rendered using different templates while preserving its structural representation,

$$\mathrm{Struct}(\Phi(\mathcal{G}, \tau_a)) = \mathrm{Struct}(\Phi(\mathcal{G}, \tau_b)), \tag{11}$$

we can vary the narrative context independently of the underlying reasoning structure.

Let $\mathcal{T}^{\phi}_{\mathrm{train}}$ denote the set of templates encountered during training at stage ϕ, and let $\mathcal{T}_{\mathrm{eval}}$ denote the broader set of evaluation templates, including contexts that are rarely or never observed during training. A model exhibits contextual generalization when its reasoning performance remains consistent across such contextual changes, i.e.,

$$\mathrm{Acc}\left(f_{\theta_\phi}, \mathcal{G}, \tau_a\right) \approx \mathrm{Acc}\left(f_{\theta_\phi}, \mathcal{G}, \tau_b\right), \qquad \tau_b \notin \mathcal{T}^{\phi}_{\mathrm{train}}. \tag{12}$$

Thus, contextual generalization measures whether a model acquires reusable reasoning patterns rather than relying on memorized task styles or linguistic cues. The resulting framework allows us to evaluate transfer across familiar, unseen, and long-tailed contextual domains while keeping the underlying reasoning structure fixed.

B Training setup

Model Architecture. Our experiments are conducted using Qwen3-decoder architecture Yang et al. (2025) models with 100M parameters, similar to Zhang et al. (2026a). The detailed architecture configurations are as in Table 1.

Table 1: Qwen3 model architecture configuration.

Component	Configuration
Model Type	Qwen3
Number of Layers	12
Hidden Size	768
Intermediate Size	3,072
Number of Attention Heads	12
Number of Key-Value Heads	2
Activation Function	SiLU
RMS Norm Epsilon	1×10^{-6}

Training hyperparameters. The complete training configurations and hyperparameters for all three stages are summarized in Table 2. For mid-training, we follow the setup of Zhang et al. (2026a), using a total training budget of 20B tokens.

B.1 Process-level Verification

We follow the process-verified evaluation protocol of prior work Ni et al. (2026). Given an input instance with ground-truth reasoning graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathrm{val})$ and final answer a^*, the model generates a free-form solution s. We deterministically convert s into a predicted dependency graph $\hat{\mathcal{G}} = (\hat{\mathcal{V}}, \hat{\mathcal{E}}, \widehat{\mathrm{val}})$ and extract its predicted final answer $\hat{a}$. Where $\hat{\mathcal{V}}$ corresponds to a named intermediate quantity in a solution, while an edge in $\hat{\mathcal{E}}$ represents a dependency on a quantity defined in an earlier step, $\widehat{\mathrm{val}}$ stores the inferred numeric value for each

Table 2: Training configurations and hyperparameters across different training stages.

Category	Parameter	Pre-training	Mid-training	Post-training
Training	Global batch size	512K tokens	512K tokens	1024 examples
	Max sequence length	2048	2048	2048
Optimization	Learning rate	2×10^{-4}	1×10^{-4}	1×10^{-6}
	Learning rate scheduler	Cosine decay	Cosine decay	Constant
	Minimum learning rate	3×10^{-5}	3×10^{-5}	–
	Warmup ratio	5%	15%	–
RL Fine-tuning	KL coefficient	–	–	10^{-3}
	Entropy bonus	–	–	0
	Mini-batch size	–	–	256
	Micro-batch size / GPU	–	–	16
Sampling	Temperature	–	–	1.0
	Top-p	–	–	1.0
Evaluation	Temperature	–	–	0.7
	Top-p	–	–	1.0
	Max new tokens	–	–	1,024

node, and $\hat{a}$ is the predicted final outcome. For every gold node $v \in \mathcal{V}$, we assign a binary step-level score:

$$s(v; \hat{\mathcal{G}}, \mathcal{G}) = \begin{cases} 1, & \text{if } v \in \hat{V},\ \mathrm{pa}_{\hat{\mathcal{G}}}(v) = \mathrm{pa}_{\mathcal{G}}(v),\ \widehat{\mathrm{val}}(v) = \mathrm{val}(v), \\ 0, & \text{otherwise,} \end{cases}$$

where $\mathrm{pa}_{\mathcal{G}}(v)$ denotes the set of parent dependencies of v in $\mathcal{G}$. Thus, a step is considered correct only when its corresponding intermediate quantity is present, has the correct dependencies, and has the correct value. The overall process accuracy is defined as the fraction of gold reasoning steps that are correctly recovered:

$$\mathrm{ProcessAcc}(\hat{\mathcal{G}}; \mathcal{G}) = \frac{1}{|\mathcal{V}|} \sum_{v \in \mathcal{V}} s(v; \hat{\mathcal{G}}, \mathcal{G}).$$

Then we define verified correctness to measure end-to-end performance by jointly considering the predicted answer $\hat{a}$ and the ground-truth answer a^*:

$$\mathrm{VerifiedCorrect}(\hat{a}, \hat{\mathcal{G}}; a^*, \mathcal{G}) = \begin{cases} 1, & \text{if } \mathrm{ProcessAcc}(\hat{\mathcal{G}}; \mathcal{G}) = 1 \text{ and } \hat{a} = a^*, \\ 0, & \text{otherwise.} \end{cases}$$

B.2 Pre-/Post-training Data Configuration

This section describes the data recipes used in Sec. 4, 5, and Sec. E. Table 3 summarizes the operation-count ranges, contextual templates, and training budgets used in each experimental setting.

C RL Training dynamics for Extrapolative Generalization.

We examine the RL training dynamics under different post-training data recipes for the *Extrapolative Generalization* experiments in Fig. 8.

Table 3: Data configuration for pre-/post-training experiments. $\mathrm{op}(\mathcal{G})$ ranges indicate the operation counts during each training phase. Contexts A, B, C correspond to distinct templates: A = `animals-zoo`, B = `teachers-school`, C = `movie-festival`.

	Pre-training			Post-training (RL)		
Section	**op($\mathcal{G}$)**	**Contexts**	**Training Budget**	**op($\mathcal{G}$)**	**Contexts**	**Training Budget**
Sec. 4	20% op=2-4 + 30% op=5-7 + 50% op=8-10	33% A + 33% B + 33% C	10B tokens	op=11-14 op=17-20	33% A + 33% B + 33% C	204.8K samples
Sec. 5	99.9% op=2-20 A + 0.1% op=2 B 99% op=2-20 A + 1% op=2 B 90% op=2-20 A + 10% op=2 B		10B tokens	op=2-20	50% A + 50% B	204.8K samples

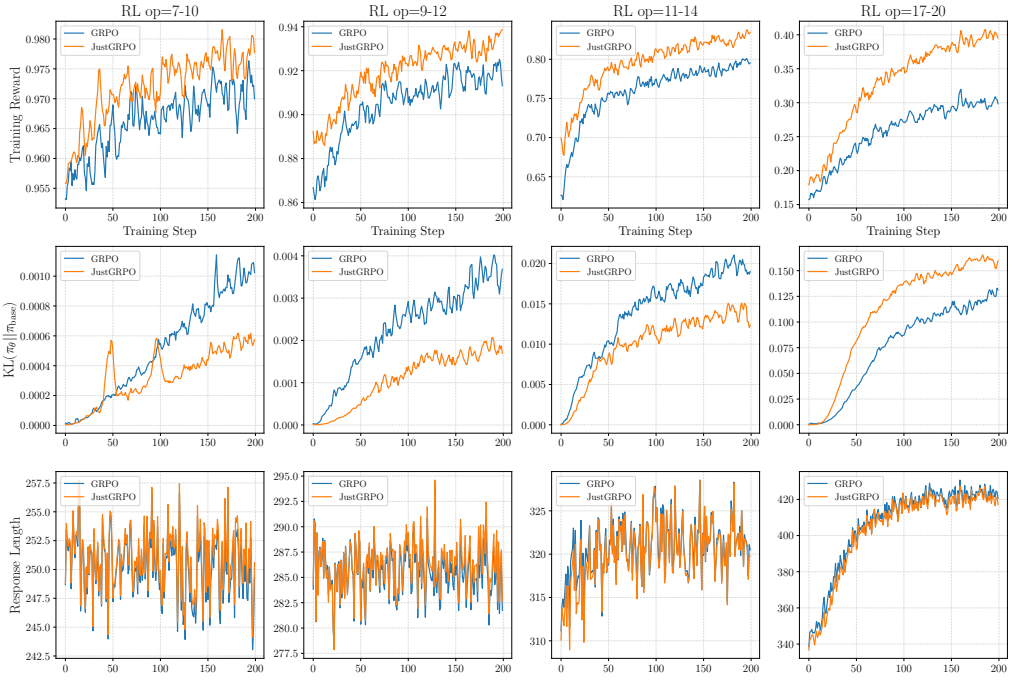

Figure 8: **Training dynamics across different post-training data recipes.** Block-diffusion pre-trained models achieve better training acorss post-training recipes while maintaining lower deviation from the base model.

Across all post-training recipes, models pretrained with the block diffusion objective consistently achieve higher rewards while maintaining a smaller KL divergence from the initial policy. This indicates that block diffusion enables more effective reward optimization while requiring less deviation from the pretrained policy.

D Contextual Generalization RL Training Dynamics.

We next examine the RL training dynamics under both *outcome* and *process-based* reward supervision in the *Contextual Generalization* experiments. These results allow us to distinguish improvements in training reward from genuine improvements in contextual generalization.

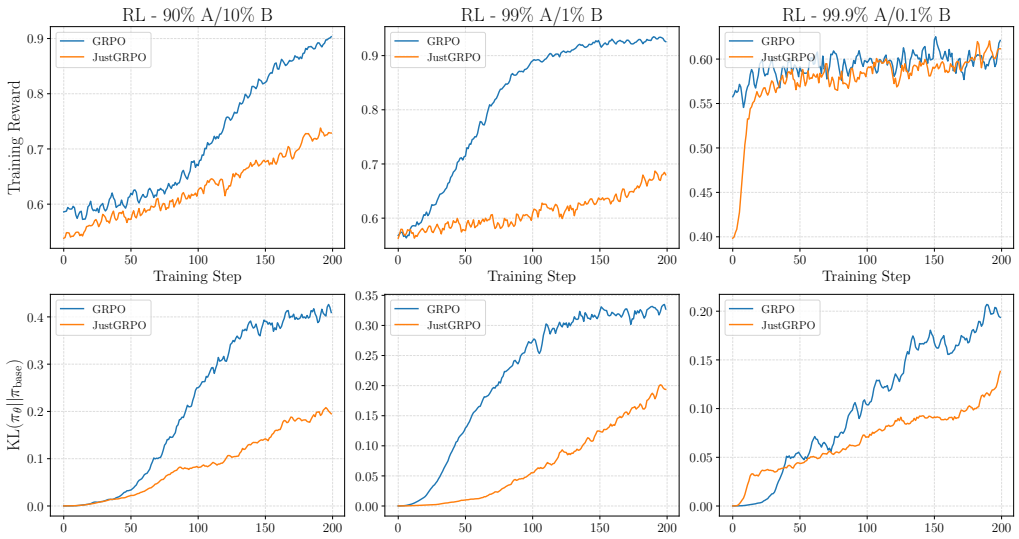

Figure 9: **Reward-KL dynamics under *outcome supervision*.** We observe Reward hacking phenomenon where NTP-trained models achieve near-perfect accuracy through training process but exhibits poor generalizattion in contrast to block-diffusion pre-trained models.

Under *outcome supervision* (Fig. 9), Next-token-prediction (NTP)-pretrained models rapidly increase their training reward to $\geq 90\%$ while substantially deviating from the base model. However, this increase in reward does not translate into improved contextual generalization. Instead, the models exploit a shortcut in the reward signal by copying the response surface associated with the dominant context, as illustrated in Fig. 9. This behavior suggests that the high training reward achieved by NTP-pretrained models is largely attributable to reward hacking rather than genuine contextual generalization.

Under *process-based supervision* (Fig. 10), the training dynamics differ from those observed under outcome supervision. NTP-pretrained models achieve higher reward and stronger accuracy at small-to-moderate levels of long-tailed context-B exposure. However, this advantage diminishes as the exposure becomes extremely limited: when context-B exposure is ($\leq 1\%$), NTP and block-diffusion-pretrained models exhibit comparable performance.

E Block Size Ablation in RL for Extrapolative Generalization

In this section, we ablate the block size of the diffusion pre-training objective and examine its impact on extrapolative generalization after RL fine-tuning. Specifically, we pre-train models with different block sizes and subsequently fine-tune them using JustGRPO Ni et al. (2026) on `op=11-14` examples. We compare these models against the next-token prediction objective and report the resulting Pass@k curves in Fig. 11.

The results reveal a non-monotonic effect of block size on extrapolative generalization. A moderate block size of $B = 8$ consistently yields stronger coverage than the next-token prediction objective, indicating that the benefits of block diffusion extend beyond the small block size of $B = 4$. However, increasing the block size further to $B = 16$ degrades performance, resulting in lower coverage than the next-token prediction baseline. These results suggest that while block diffusion can improve the coverage obtained through RL, excessively large block sizes may weaken this benefit and lead to suboptimal extrapolative generalization.

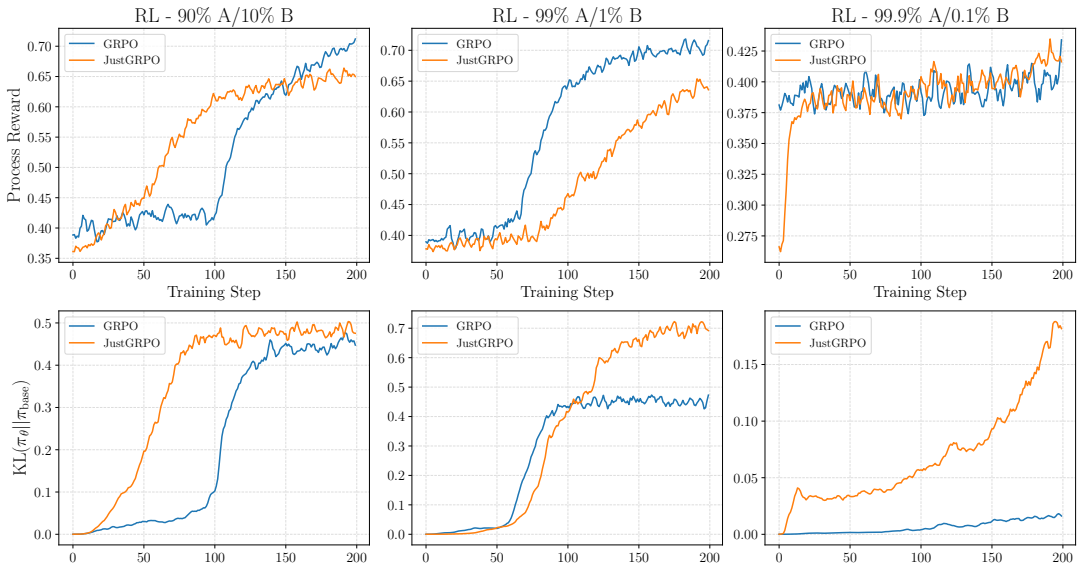

Figure 10: **Reward-KL dynamics under *process reward supervision*.** Under process-reward supervision, NTP pre-trained models achive higher reward while yield similar performance on extremely small exposure to conttext B.

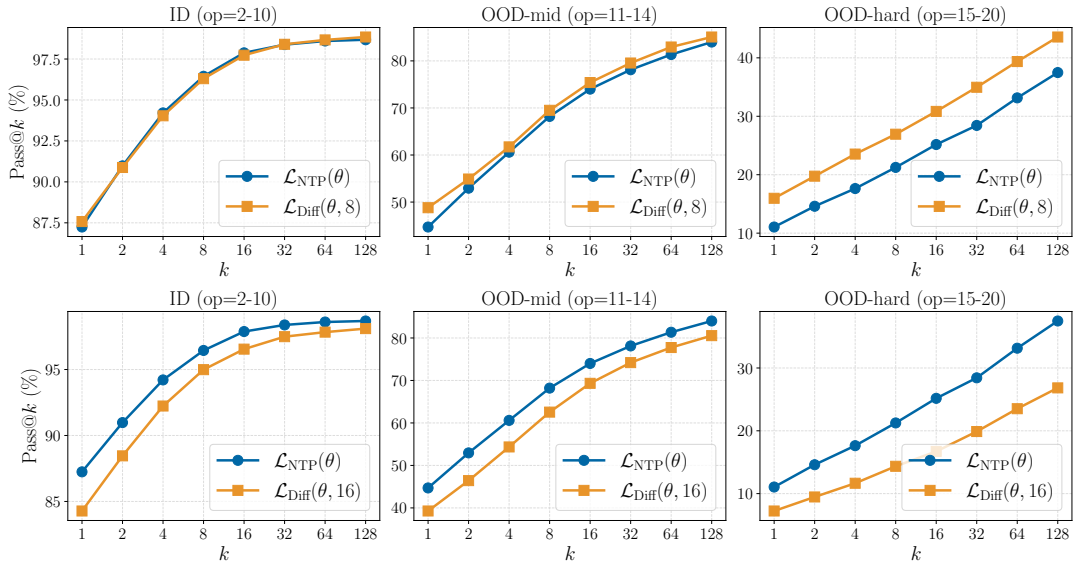

Figure 11: Pass@k curves of RL-trained models initialized from block-diffusion pre-training with block sizes $B = 8$ (upper) and $B = 16$ (lower).

F WHY DOES A LARGER TRAINING BLOCK YIELD SUB-OPTIMAL REASONING COVERAGE?

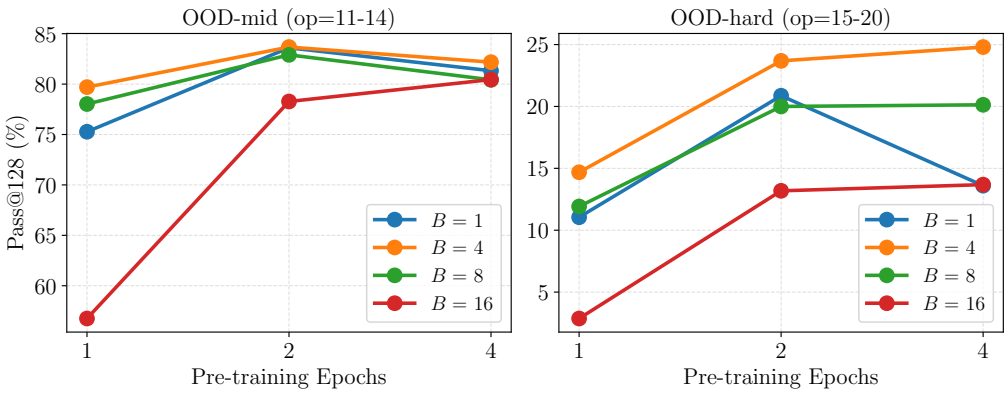

Figure 12: **Pass@k evolution across pre-training epochs with different training block sizes B.** Larger block sizes yield greater improvements with longer training, whereas smaller blocks show only modest gains and may even suffer from overfitting, resulting in narrower reasoning coverage.

The suboptimal reasoning coverage of models trained with larger block sizes could be primarily due to insufficient training. Specifically, larger block sizes increase the number of token orderings that the model must learn, resulting in a substantially larger space of possible generation trajectories (Prabhudesai et al., 2025; Ni et al., 2025). As a result, models trained with larger blocks may be under-trained relative to models trained with smaller blocks, leading to narrower reasoning coverage. To test this, we increase the number of pre-training epochs from 1 to 4 across different training blocks and measure the improved Pass@k performance.

As shown in Fig. 12, most training block sizes reach peak performance by epoch 2, with only marginal improvements or stable performance by epoch 4. While larger block sizes benefit more from extended training, smaller blocks (e.g., $B = 1$) are more susceptible to overfitting, leading to reduced reasoning coverage. Nevertheless, the additional training does not fully close the performance gap with models pretrained using smaller block sizes.