ProxyThinker: Test-Time Guidance through Small Visual Reasoners

Zilin Xiao[1], Jaywon Koo[1], Siru Ouyang[2], Jefferson Hernandez[1], Yu Meng[3], Vicente Ordonez[1]
[1]Rice University [2]University of Illinois Urbana-Champaign [3]University of Virginia
{zilin,vicenteor}@rice.edu

Abstract

Recent advancements in reinforcement learning with verifiable rewards have pushed the boundaries of the visual reasoning capabilities in large vision-language models (LVLMs). However, training LVLMs with reinforcement fine-tuning (RFT) is computationally expensive, posing a significant challenge to scaling model size. In this work, we propose PROXYTHINKER, an inference-time technique that enables large models to inherit the visual reasoning capabilities from small, slow-thinking visual reasoners without any training. By subtracting the output distributions of base models from those of RFT reasoners, PROXYTHINKER modifies the decoding dynamics and successfully elicits the slow-thinking reasoning demonstrated by the emerged sophisticated behaviors such as self-verification and self-correction. PROXYTHINKER consistently boosts performance on challenging visual benchmarks on mathematical and multi-disciplinary reasoning, enabling untuned base models to compete with the performance of their full-scale RFT counterparts. Code is available at `https://github.com/MrZilinXiao/ProxyThinker`.

1 Introduction

Recent advances in large language models have led to the development of systems capable of extended reasoning and deliberation, often referred to as *"slow-thinking"* models, such as OpenAI-o1 [10], DeepSeek-R1 [7], and QwQ [24]. Unlike *"fast-thinking"* models such as GPT-4o [9], *"slow-thinking"* models usually engage in multi-step self-reflection to produce an answer that resembles the thorough thinking process that humans make before producing a final answer for non-trivial problems. These models have achieved remarkable success in complex problem-solving benchmarks, particularly in mathematical and scientific reasoning domains [21, 32, 30]. Recent research has also extended such reflective reasoning to multimodal tasks [8, 3, 29, 34, 25], pushing large vision-language models (LVLMs) toward greater performance in scenarios that require structured and contextual understanding across modalities.

Many of the most effective *"slow-thinking"* models rely on reinforcement learning with verifiable rewards (RLVR) [5, 22, 28], a reinforcement fine-tuning (RFT) framework that encourages the model to generate intermediate reasoning steps that lead to a correct answer for automatically verifiable tasks. While effective, this approach is computationally intensive and resource-demanding. First, the process often requires maintaining multiple model copies when using algorithms such as Proximal Policy Optimization (PPO) [18] or Group Relative Policy Optimization (GRPO) [20], which significantly increases memory usage. Second, the training process typically alternates between rollout and optimization phases, resulting in significant complexity and extensive training time.

Due to these high training costs, prior work has rarely applied RFT to LVLMs with more than 7 billion parameters. Recent research findings [19, 31, 6, 15, 27] suggest that RFT does not teach new knowledge beyond the capabilities of the base model, but rather elicits and amplifies reasoning

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: MATH-AI.

behaviors that are already included in the sampling distributions of the base model. In this work, we introduce PROXYTHINKER, a simple yet effective inference-time method that allows for efficient transfer of visual reasoning capabilities without incurring any training costs. Motivated by the line of work that explores decoding-steering of language models [14, 12, 17, 11, 13], we propose using the difference between the last-token logits from a reasoning **Expert** model after RFT training and those from a non-RFT **Amateur** model to represent the reasoning abilities induced by RFT. Such a difference could steer a larger **Base** model toward the slow-thinking reasoning pattern.

We conduct experiments on mathematical and multi-disciplinary reasoning tasks using large base models with 32B and 72B parameters. Quantitative results show significant improvements on benchmarks such as MathVista [16], MathVerse [33], MathVision [26], and R1-OneVisionBench [29]. For example, on the MathVision test split, we improved the accuracy of Qwen2.5-VL-32B-Instruct [1] from 38.4% to 40.8% by integrating OpenVLThinker-7B [3] as a reasoning expert, despite the latter's poor accuracy of 25.3%. This even surpasses the 40.5% achieved by the full-scale RFT model VL-Rethinker-32B [25]. Ablation studies show that our method works robustly without any hyperparameter tuning to achieve substantial gains. We further conducted a comprehensive analysis to show emerging reasoning behaviors in PROXYTHINKER, hoping to shed light on future research work in decoding-time algorithms that enhance reasoning abilities.

2 Preliminaries

Vision-Language Model (VLM) decoding. A VLM defines a conditional probability distribution $p_{\boldsymbol{\theta}}$ over output sequences, parameterized by model weights $\boldsymbol{\theta}$, and conditioned on both a textual prompt $\mathbf{x} = [x_1, \ldots, x_n]$ and a set of input images $\mathcal{I} = \{I_1, \ldots, I_k\}$. The model autoregressively generates a response $\mathbf{y} = [y_1, \ldots, y_m]$ according to:

$$p_{\boldsymbol{\theta}}(\mathbf{y} \mid \mathbf{x}, \mathcal{I}) = \prod_{j=1}^{m} p_{\boldsymbol{\theta}}(y_j \mid \mathbf{x}, \mathcal{I}, y_{<j}).$$

Decoding-time algorithm refers to a technique that modifies a language model (LM) output distribution at inference time as a means to improve generation quality and control without training. DExperts [14] first proposes to steer the output of an LM toward desirable attributes, such as reducing toxicity and controlling sentiment, with a pair of *expert* and *anti-expert* LMs, encouraging safe continuation and penalizing toxic completions. Contrastive Decoding [12] improves open-ended text generation quality by contrasting the predictions of a large *expert* LM against those of a small *amateur* LM. The intuition behind this is that if both a big and small model are likely to produce an undesirable token (*e.g.*, a generic, repetitive word), that token score is suppressed, whereas tokens favored by the expert but not the amateur are boosted. DoLa [2] targets the pervasive issue of LLM hallucinations – generating text not supported by factual knowledge. It obtains the next-token distribution by contrasting the later layers of the model against its earlier layers, essentially subtracting or down-weighting the contributions of lower-layer representations. In contrast to these approaches, our motivation lies in transferring the reasoning abilities of a small visual reasoner, which is orthogonal to their goals and contributions.

3 PROXYTHINKER: Next-Token-Prediction with Test-time Guidance

There is increasing evidence that reinforcement fine-tuning (RFT) does not impart fundamentally new knowledge into a base model, but rather amplifies reasoning behaviors that the base model was already capable of in principle [31]. Or in other words, RFT shifts the probability mass of a model toward token sequences that exhibit structured, *"slow-thinking"* reasoning strategies, such as branching into sub-cases, backtracking after a contradiction [23], and self-checking intermediate answers [6]. These reasoning strategies are reflected in the high activation of relevant tokens at specific stages of the reasoning process.

In the upper part of Figure 1, we present an example from the MathVision dataset, where we provide three different vision-language models (VLMs) with the same incorrect reasoning process. Both large and small base models tend to directly provide an answer after reading the reasoning process, whereas a small RFT-trained expert exhibits reflective reasoning strategies. However, due to its limited model

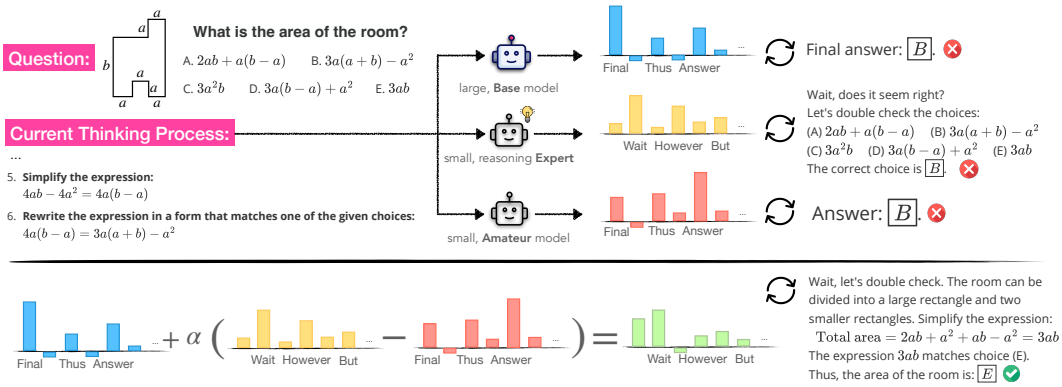

Figure 1: PROXYTHINKER with a case study from MathVision. When provided with the same incorrect thinking process, both the large and small base models tend to finalize the answer prematurely in the decoding process. The small reasoning expert shows signs of self-verification behaviors, *e.g.*, assigning high probabilities to "Wait", "However", "But". However, its limited capacity confines it to shallow reasoning, such as restating answer choices. Through logits manipulation, PROXYTHINKER transfers this reasoning behavior to the large base model, effectively triggering accurate self-correction and leading to the correct option.

capacity, this reflective behavior remains *shallow* and largely restricted to restating answer options. We therefore ask: Can the reasoning skills acquired during RFT be directly transferred to a larger model via logits delta? By combining the reasoning patterns of the small RFT expert with the enhanced capacity of the large base model, we anticipate that such transfer could deepen the model reasoning behaviors and improve performance on reasoning-intensive tasks. In the lower part of Figure 1, we observe that applying logits delta successfully elicits the effective reflection of the large base model and ultimately leads to a correct option.

Formally, consider a pretrained VLM Ψ, or **Base** model, which we wish to adapt toward improved *"slow-thinking"* reasoning behavior without updating its parameters. Given a set of input images $\mathcal{I} = \{I_1, I_2, \ldots, I_k\}$ and a text prompt $x_{<t}$ with t being the current decoding time step, Ψ produces a logit vector over the vocabulary, conditioned jointly on both modalities. The goal is to guide Ψ to behave as if it had undergone RFT while avoiding any parameter updates to Ψ. To achieve this, we introduce two auxiliary models: a small, base **Amateur** model ψ_0 and its RFT counterpart, reasoning **Expert** model ψ_1, which is much cheaper to tune than tuning the large model Ψ itself with the RFT method. The proposed PROXYTHINKER modifies the output distribution of Ψ at inference time using a logit shift computed from the difference between the output logits of ψ_1 and ψ_0.

At each decoding step t, we condition Ψ, ψ_1, and ψ_0 on the shared image set $\mathcal{I}$ and the current text prefix $x_{<t}$ to compute logits z_Ψ, z_{ψ_1}, and z_{ψ_0}, respectively. A hyperparameter $\alpha \in \mathbb{R}^+$ controls the influence of this difference signal. The adjusted distribution from PROXYTHINKER model $\hat{\Psi}$ is given by:

$$p_{\hat{\Psi}}\left(x_t \mid x_{<t}, \mathcal{I}\right) = \text{softmax}\left[z_\Psi\left(x_t \mid x_{<t}, \mathcal{I}\right) + \alpha \cdot \left(z_{\psi_1}\left(x_t \mid x_{<t}, \mathcal{I}\right) - z_{\psi_0}\left(x_t \mid x_{<t}, \mathcal{I}\right)\right)\right]. \quad (1)$$

A token x_t is then sampled from this adjusted distribution and appended to the input sequence $x_{<t}$, forming $x_{\leq t}$, used as the next-step input for all three models — Ψ, ψ_0, and ψ_1 — in the subsequent decoding iteration. This feedback loop continues autoregressively until the end of the generation.

4 Experiments

To investigate the generality and scalability of PROXYTHINKER, we examine whether it works on widely used mathematical and multi-disciplinary reasoning tasks using 32B and 72B models with different types of RFT reasoning experts. The prompt template for each reasoning expert is attached in Appendix A.1. We report these results in Table 1. To explore the upper bound of our method, we also use two larger models, VL-Rethinker-32B and VL-Rethinker-72B, which are directly trained via RFT, as ceiling performance references.

Table 1: Performance (Accuracy %) on mathematical and multi-disciplinary reasoning benchmarks. α is set to 0.5 for PROXYTHINKER methods. R1-Bench stands for R1-Onevision-Bench. Overall best PROXYTHINKER method is marked with light blue. Ceiling performance of full-scale RFT expert is highlighted with gray. 💡 indicates an RFT reasoning expert, and 🤖 denotes a PROXYTHINKER variant.

Model	RFT Expert	MathVista	MathVerse	MathVision	MMMU	R1-Bench
Qwen2.5-VL-7B	–	68.2	46.3	25.1	55.6†	32.1
💡 OpenVLThinker-7B	–	70.2	47.9	25.3	56.2*	32.9*
💡 ThinkLite-VL-7B	–	75.1	50.7	32.9	54.6	39.0*
💡 VL-Rethinker-7B	–	74.9	54.2	32.3	56.7	47.2*
Qwen2.5-VL-32B	–	74.7	53.8*	38.4	**67.5**†	49.4*
🤖 Qwen2.5-VL-32B	OpenVLThinker-7B	77.4 (+2.7)	53.8 (0.0)	**40.8** (+2.4)	67.1 (-0.4)	**53.0** (+3.6)
🤖 Qwen2.5-VL-32B	ThinkLite-VL-7B	77.6 (+2.9)	**56.0** (+2.2)	38.8 (+0.4)	67.1 (-0.4)	49.7 (+0.3)
🤖 Qwen2.5-VL-32B	VL-Rethinker-7B	**78.1** (+3.4)	55.1 (+1.3)	39.2 (+0.8)	67.1 (-0.4)	52.5 (+3.1)
💡 VL-Rethinker-32B	–	78.8	56.9	40.5	65.6	50.8*
Qwen2.5-VL-72B	–	74.8	55.1*	38.1	68.0†	52.0
🤖 Qwen2.5-VL-72B	OpenVLThinker-7B	77.8 (+3.0)	56.4 (+1.3)	36.2 (-1.9)	**69.5** (+1.5)	50.4 (-1.6)
🤖 Qwen2.5-VL-72B	ThinkLite-VL-7B	**78.7** (+3.9)	57.2 (+2.1)	**40.4** (+2.3)	69.0 (+1.0)	50.2 (-1.8)
🤖 Qwen2.5-VL-72B	VL-Rethinker-7B	78.1 (+3.3)	**58.6** (+3.5)	39.5 (+1.4)	68.5 (+0.5)	**54.4** (+2.4)
💡 VL-Rethinker-72B	–	80.3	61.7	43.9	68.8	57.9*

† indicates reproduced results by us due to challenges in reproducing the original evaluation setup.
* indicates reproduced results by us because the original authors did not conduct such an evaluation.

PROXYTHINKER provides consistent improvements on nearly all benchmarks. With the exception of the MMMU validation set, we observe consistent performance improvements across all benchmark tasks and model-expert combinations. For example, using OpenVLThinker-7B as an expert improves Qwen2.5-VL-32B-Instruct's MathVision test accuracy from 38.4% to 40.8%, surpassing even the fully RFT-trained VL-Rethinker-32B (40.5%). This improvement is unlikely due to knowledge transfer, as OpenVLThinker-7B achieves only 25.3% on MathVision. Rather, it suggests that the reasoning patterns of the small expert have been effectively extracted and applied to enhance the large base model's reasoning abilities that otherwise would require full-scale RFT to activate.

Quality of RFT expert generally determines the degree of improvement. Consistent with our intuition, a stronger RFT expert tends to provide more structured reasoning paths, enhancing the base model's reasoning abilities more effectively. VL-Rethinker-7B, the most competitive of the experts, achieves the best overall results with both Qwen2.5-VL-32B and 72B. Nonetheless, there are a few exceptions across benchmarks.

5 Conclusion

We present PROXYTHINKER, a simple yet effective decoding-time algorithm for transferring visual reasoning capabilities from small visual reasoning models. PROXYTHINKER leverages the token-level logits difference between an RFT expert and an amateur model to effectively steer a large model's generation toward *"slow-thinking"*, multi-step reasoning behaviors. Through extensive experiments on vision-centric and multimodal reasoning tasks, we demonstrate that PROXYTHINKER can consistently enhance performance across model sizes, including substantial improvements on spatial, mathematical, and multi-disciplinary reasoning benchmarks. We believe PROXYTHINKER provides a promising direction for efficient reasoning transfer in large vision-language models and offers insights into the understanding of how RFT influences model behavior.

References

[1] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

[2] Yung-Sung Chuang, Yujia Xie, Hongyin Luo, Yoon Kim, James R. Glass, and Pengcheng He. Dola: Decoding by contrasting layers improves factuality in large language models. In *The Twelfth International Conference on Learning Representations*, 2024.

[3] Yihe Deng, Hritik Bansal, Fan Yin, Nanyun Peng, Wei Wang, and Kai-Wei Chang. Openvlthinker: An early exploration to complex vision-language reasoning via iterative self-improvement. *arXiv preprint arXiv:2503.17352*, 2025.

[4] Haodong Duan, Junming Yang, Yuxuan Qiao, Xinyu Fang, Lin Chen, Yuan Liu, Xiaoyi Dong, Yuhang Zang, Pan Zhang, Jiaqi Wang, et al. Vlmevalkit: An open-source toolkit for evaluating large multi-modality models. In *Proceedings of the 32nd ACM international conference on multimedia*, pages 11198–11201, 2024.

[5] Hugging Face. Open r1: A fully open reproduction of deepseek-r1, January 2025.

[6] Kanishk Gandhi, Ayush Chakravarthy, Anikait Singh, Nathan Lile, and Noah D Goodman. Cognitive behaviors that enable self-improving reasoners, or, four habits of highly effective stars. *arXiv preprint arXiv:2503.01307*, 2025.

[7] 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.

[8] Wenxuan Huang, Bohan Jia, Zijie Zhai, Shaosheng Cao, Zheyu Ye, Fei Zhao, Zhe Xu, Yao Hu, and Shaohui Lin. Vision-r1: Incentivizing reasoning capability in multimodal large language models. *arXiv preprint arXiv:2503.06749*, 2025.

[9] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

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

[11] Sicong Leng, Hang Zhang, Guanzheng Chen, Xin Li, Shijian Lu, Chunyan Miao, and Lidong Bing. Mitigating object hallucinations in large vision-language models through visual contrastive decoding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13872–13882, 2024.

[12] Xiang Lisa Li, Ari Holtzman, Daniel Fried, Percy Liang, Jason Eisner, Tatsunori Hashimoto, Luke Zettlemoyer, and Mike Lewis. Contrastive decoding: Open-ended text generation as optimization. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12286–12312, Toronto, Canada, July 2023. Association for Computational Linguistics.

[13] Alisa Liu, Xiaochuang Han, Yizhong Wang, Yulia Tsvetkov, Yejin Choi, and Noah A. Smith. Tuning language models by proxy. In *First Conference on Language Modeling*, 2024.

[14] Alisa Liu, Maarten Sap, Ximing Lu, Swabha Swayamdipta, Chandra Bhagavatula, Noah A. Smith, and Yejin Choi. DExperts: Decoding-time controlled text generation with experts and anti-experts. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6691–6706, Online, August 2021. Association for Computational Linguistics.

[15] Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025.

[16] Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. In *International Conference on Learning Representations (ICLR)*, 2024.

[17] Sean O'Brien and Mike Lewis. Contrastive decoding improves reasoning in large language models. *arXiv preprint arXiv:2309.09117*, 2023.

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

[19] Darsh J Shah, Peter Rushton, Somanshu Singla, Mohit Parmar, Kurt Smith, Yash Vanjani, Ashish Vaswani, Adarsh Chaluvaraju, Andrew Hojel, Andrew Ma, et al. Rethinking reflection in pre-training. *arXiv preprint arXiv:2504.04022*, 2025.

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

[21] 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, 2024.

[22] Yi Su, Dian Yu, Linfeng Song, Juntao Li, Haitao Mi, Zhaopeng Tu, Min Zhang, and Dong Yu. Expanding rl with verifiable rewards across diverse domains. *arXiv preprint arXiv:2503.23829*, 2025.

[23] Gokul Swamy, Sanjiban Choudhury, Wen Sun, Zhiwei Steven Wu, and J Andrew Bagnell. All roads lead to likelihood: The value of reinforcement learning in fine-tuning. *arXiv preprint arXiv:2503.01067*, 2025.

[24] Qwen Team. Qwq-32b: Embracing the power of reinforcement learning, March 2025.

[25] Haozhe Wang, Chao Qu, Zuming Huang, Wei Chu, Fangzhen Lin, and Wenhu Chen. Vl-rethinker: Incentivizing self-reflection of vision-language models with reinforcement learning. *arXiv preprint arXiv:2504.08837*, 2025.

[26] Ke Wang, Junting Pan, Weikang Shi, Zimu Lu, Houxing Ren, Aojun Zhou, Mingjie Zhan, and Hongsheng Li. Measuring multimodal mathematical reasoning with math-vision dataset. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024.

[27] Yiping Wang, Qing Yang, Zhiyuan Zeng, Liliang Ren, Lucas Liu, Baolin Peng, Hao Cheng, Xuehai He, Kuan Wang, Jianfeng Gao, et al. Reinforcement learning for reasoning in large language models with one training example. *arXiv preprint arXiv:2504.20571*, 2025.

[28] Tong Wei, Yijun Yang, Junliang Xing, Yuanchun Shi, Zongqing Lu, and Deheng Ye. Gtr: Guided thought reinforcement prevents thought collapse in rl-based vlm agent training. *arXiv preprint arXiv:2503.08525*, 2025.

[29] Yi Yang, Xiaoxuan He, Hongkun Pan, Xiyan Jiang, Yan Deng, Xingtao Yang, Haoyu Lu, Dacheng Yin, Fengyun Rao, Minfeng Zhu, et al. R1-onevision: Advancing generalized multimodal reasoning through cross-modal formalization. *arXiv preprint arXiv:2503.10615*, 2025.

[30] Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

[31] Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *arXiv preprint arXiv:2504.13837*, 2025.

[32] 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. *arXiv preprint arXiv:2503.18892*, 2025.

[33] Renrui Zhang, Dongzhi Jiang, Yichi Zhang, Haokun Lin, Ziyu Guo, Pengshuo Qiu, Aojun Zhou, Pan Lu, Kai-Wei Chang, Yu Qiao, et al. Mathverse: Does your multi-modal llm truly see the diagrams in visual math problems? In *European Conference on Computer Vision*, pages 169–186. Springer, 2024.

[34] Hengguang Zhou, Xirui Li, Ruochen Wang, Minhao Cheng, Tianyi Zhou, and Cho-Jui Hsieh. R1-zero's" aha moment" in visual reasoning on a 2b non-sft model. *arXiv preprint arXiv:2503.05132*, 2025.

A Technical Appendices and Supplementary Material

General Conversation Template

<lim_start|>**system**
You are a helpful assistant.<lim_endl>
<lim_start|>**user**<|vision_start|><|image_pad|><|vision_end|>
{query_question}
{instruction_prompt} <lim_endl>
<lim_start|>assistant

Instruction Prompt for ThinkLite-VL

You FIRST think about the reasoning process as an internal monologue and then provide the final answer. The reasoning process MUST BE enclosed within <think> </think> tags. The final answer MUST BE put in **\boxed{}**.

Instruction Prompt for OpenVLThinker

Your final answer MUST BE put between **<answer> </answer>**.

Instruction Prompt for VL-Rethinker

Please reason step by step, and put your final answer within **\boxed{}**.

Figure 2: Prompt templates of different RFT experts in §4.

A.1 Prompt Templates

We provide the prompt templates of different reasoning experts in Figure 2. For MathVerse evaluation, we employ the prompt template in Figure 3 to first extract the answer and then score the answer using `gpt-4o-mini` as a judge, following VLMEvalKit [4].

A.2 Failure Case Analysis

Although PROXYTHINKER achieves consistent improvements across multiple reasoning-intensive datasets, we observe that it struggles to deliver statistically significant gains on certain knowledge-intensive benchmarks, such as MMMU – a limitation also present in full-scale RFT experts. To illustrate this, we present a test case from the MMMU Val set in Figure 4, comparing the reasoning processes of the large base model (Qwen2.5-VL-32B-Instruct), the small reasoning expert (VL-Rethinker-7B), and PROXYTHINKER. The results show that the small reasoning expert fails to accurately validate the knowledge content of answer choices, likely due to its limited model capacity. This type of knowledge verification is particularly challenging to learn via reinforcement learning with verifiable rewards. As a result, ProxyThinker inherits this limitation as well.

MathVerse Answer Extraction Prompt

I am providing you a response from a model to a math problem, termed 'Model Response'. You should extract the answer from the response as 'Extracted Answer'. Directly output the extracted answer with no explanation.\n\n

MathVerse Answer Score Prompt

Below are two answers to a math question. Question is [Question], [Standard Answer] is the standard answer to the question, and [Model_answer] is the answer extracted from a model's output to this question. Determine whether these two answers are consistent.
Please note that only when the [Model_answer] completely matches the [Standard Answer] means they are consistent. For non-multiple-choice questions, if the meaning is expressed in the same way, it is also considered consistent, for example, 0.5m and 50cm.
If they are consistent, Judement is 1; if they are different, Judement is 0.

Figure 3: MathVerse extraction and scoring prompt for `gpt-4o-mini` as a judge.

General Conversation Template

<lim_startl>**system**
You are a helpful assistant.<lim_endl>
<lim_startl>**user**<lvision_startl><limage_padl><lvision_endl>
{query_question}
{instruction_prompt} <lim_endl>
<lim_startl>assistant

Instruction Prompt for ThinkLite-VL

You FIRST think about the reasoning process as an internal monologue and then provide the final answer. The reasoning process MUST BE enclosed within <think> </think> tags. The final answer MUST BE put in **\boxed{}**.

Instruction Prompt for OpenVLThinker

Your final answer MUST BE put between **<answer> </answer>**.

Instruction Prompt for VL-Rethinker

Please reason step by step, and put your final answer within **\boxed{}**.

Figure 4: A knowledge-intensive test case from the MMMU Val set with reasoning process from Qwen2.5-VL-32B-Instruct, VL-Rethinker-7B and PROXYTHINKER-32B.