Mirage or Method? How Model–Task Alignment Induces Divergent RL Conclusions

Haoze Wu[1*] **Cheng Wang**[2*] **Wenshuo Zhao**[1] **Junxian He**[3]
[1]Zhejiang University [2]National University of Singapore [3]HKUST
waithz@zuaa.zju.edu.cn wangcheng@u.nus.edu junxianh@cse.ust.hk

Abstract

Recent advances in applying reinforcement learning (RL) to large language models (LLMs) have led to substantial progress. In particular, a series of remarkable yet often counterintuitive phenomena have been reported in LLMs, exhibiting patterns not typically observed in traditional RL settings (e.g., spurious rewards, one-shot RL). However, the precise conditions under which these observations hold remain unclear. In this work, we identify a key factor that differentiates RL observations: whether the pretrained model already exhibits strong *Model-Task Alignment*, as measured by pass@k on the target task. Through systematic experiments across diverse models and tasks, we find that while standard RL remains robust, many counterintuitive results emerge only under strong model-task alignment.

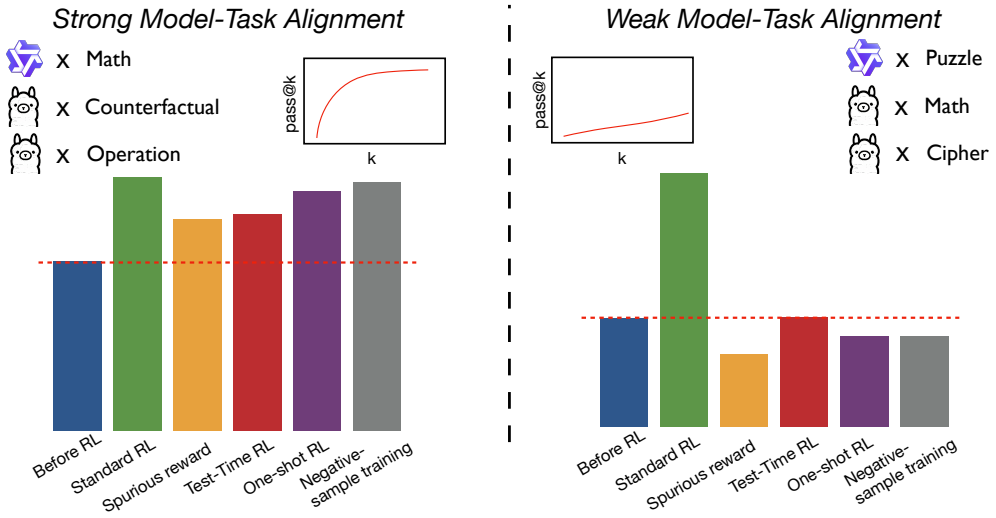

Figure 1: Model-task alignment, which is measured by pass@k accuracy on the evaluated task, drives distinct outcomes from the same series of RL approaches.

1 Introduction

While RL yields significant performance improvements in LLM reasoning [7, 5, 20, 19]—mirroring the success of RL in traditional domains such as games [16, 17]—we also observe several remarkable yet often counterintuitive empirical phenomena. These effects appear to be unique to LLMs and would be considered unexpected in traditional RL settings. For instance, single training examples

[*]Equal Contribution. Work done during visit to HKUST.

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

can match or rival full-dataset training performance [21], ground-truth reward may be surprisingly dispensable [15], and training with negative samples alone can match sophisticated reward-based methods [1]. While these findings have sparked considerable interest, the conditions under which they hold remain underexplored—particularly concerning, given their potential implications for RL practice, as existing conclusions rely heavily on limited settings dominated by Qwen models [14] trained on mathematical tasks.

To this end, we conduct a systematic empirical study of several prominent RL claims, rigorously validated across diverse model architectures and task domains—including both Qwen and non-Qwen models on mathematical and non-mathematical tasks. Our controlled experiments reveal that *Model-Task Alignment*, defined as the degree to which a model's inherent capabilities match task requirements and measured by pass@k accuracy, is a key predictor of when counterintuitive RL phenomena arise. Our findings challenge the notion that spurious rewards work solely due to data leakage. While concurrent work [22] attributes their efficacy to test-set contamination in Qwen models, we show that spurious rewards remain effective even in clean settings—provided strong model-task alignment is already present. This supports our broader hypothesis: alignment, not reward fidelity, often governs success in LLM-based RL, especially as we scale to more challenging, low-alignment regimes.

2 Hypothesis: Model-Task Alignment Dependency

Most counterintuitive RL findings arise from Qwen on math tasks [14, 23], casting doubt on generalization—e.g., spurious rewards fail with Llama [13] on the same tasks [15]. Rather than viewing "Qwen+math" as an outlier, we propose Model-Task Alignment Dependency: the efficacy of these phenomena depends on how well a model's capabilities match task demands, quantified via pass@k accuracy, which we use to classify settings as aligned or misaligned.

Strategic Model and Task Selection. To operationalize our hypothesis, we quantify alignment via pass@k across all model-task pairs (Figure 2; full results in Appendix A), identifying strong alignment (e.g., Qwen2.5 on math; both models on KOR-Bench's Operation/Counterfactual subsets) and weak alignment (e.g., Llama3.1 on math; both models on most other logical tasks). This enables us to test whether counterintuitive RL phenomena stem from alignment-specific conditions or reflect fundamental RL properties. We describe these models and tasks in Appendix B.

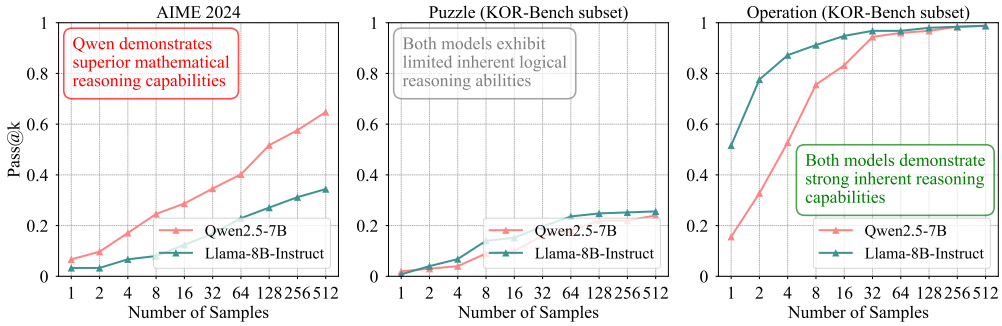

Figure 2: Pass@k for different tasks. Different LLMs have significantly different abilities on different tasks, which will affect how the RL techniques perform across model-task combinations.

The Contamination Hypothesis. Concurrent work [22] attributes the efficacy of spurious rewards to pretraining data contamination, confirming leakage in Qwen models on several math benchmarks. While we acknowledge contamination as a concern, we argue that inherent model-task alignment—distinct from data leakage—is the more fundamental factor: models can exhibit strong task performance even without test-set contamination. To verify this, we extend contamination analysis beyond Qwen-math settings using the partial-prompt method [22] (details in Appendix C), measuring exact match and ROUGE-L (where 1.0 indicates perfect reconstruction). As shown in Table 1 and Appendix C, tasks like Operation and Counterfactual from KOR-Bench show no contamination yet yield high pass@k scores, indicating strong inherent alignment.

Model	Portion	AMC 23		MATH500		Puzzle		Operation	
		ROUGE	EM	ROUGE	EM	ROUGE	EM	ROUGE	EM
Qwen2.5-7B	0.4	63.78	23.91	50.36	8.20	19.56	0.00	21.37	0.00
	0.6	64.42	33.73	60.98	21.20	19.62	0.00	24.25	0.00
	0.8	73.23	49.39	66.42	40.20	19.24	0.00	20.18	0.00
Llama-3.1-8B	0.4	27.18	0.00	23.09	0.60	18.27	0.00	21.83	0.00
	0.6	30.64	0.00	40.56	3.80	17.31	0.00	18.34	0.00
	0.8	44.54	4.81	48.33	17.8	15.85	0.00	16.75	0.00

Table 1: Contamination Analysis across model-task combinations. Portion refers to the truncation ratio of the prompt used to test whether models can complete the remaining content. **Red** indicates potential contamination with strong model-task alignment; **Gray** indicates no contamination with weak model-task alignment; **Green** indicates no contamination with strong model-task alignment. We maintain this color scheme throughout the paper to indicate the categories of experimental settings.

3 RQ1 – Reward Signal: How Critical Is It?

	Math Tasks			Logic Tasks							
							KOR Benchmark				
	AIME24	MATH500	AMC	SynLogic	BBH	BBEH	OP	CF	Puzzle	Logic	Cipher
Qwen2.5-7B Family											
Base	3.3	40.8	31.0	1.5	45.2	1.2	27.2	17.2	0.8	8.0	4.8
RLVR (External Reward)											
Correct	14.2$_{+10.9}$	71.0$_{+30.2}$	62.4$_{+31.4}$	42.6$_{+41.1}$	62.7$_{+17.5}$	6.8$_{+5.6}$	82.4$_{+55.2}$	79.6$_{+62.4}$	16.8$_{+10.0}$	46.4$_{+38.4}$	20.4$_{+15.6}$
Random	10.0$_{+6.7}$	57.5$_{+16.7}$	45.7$_{+14.7}$	10.2$_{+8.7}$	32.7$_{-12.5}$	0.0$_{-1.2}$	53.6$_{+26.4}$	30.8$_{+13.6}$	1.2$_{+0.4}$	6.8$_{-1.2}$	3.6$_{-1.2}$
Incorrect	6.7$_{+3.4}$	57.0$_{+16.2}$	43.1$_{+12.1}$	0.0$_{-1.5}$	30.3$_{-14.9}$	0.0$_{-1.2}$	60.8$_{+33.6}$	12.8$_{-4.4}$	0.4$_{-0.4}$	6.4$_{-1.6}$	3.2$_{-1.6}$
Format	6.7$_{+3.4}$	55.3$_{+14.5}$	48.9$_{+17.9}$	1.5$_{0.0}$	44.4$_{-0.8}$	2.4$_{+1.2}$	37.2$_{+10.0}$	21.6$_{+4.4}$	0.8$_{0.0}$	6.8$_{-1.2}$	4.4$_{-0.4}$
Self-Rewarded Reinforcement Learning											
Vote	13.3$_{+10.0}$	69.4$_{+28.6}$	58.2$_{+27.2}$	2.8$_{+1.3}$	33.6$_{-11.6}$	0.0$_{-1.2}$	56.4$_{+29.2}$	16.3$_{-0.9}$	0.8$_{0.0}$	6.8$_{-1.2}$	3.2$_{-1.6}$
EM	11.6$_{+8.3}$	70.8$_{+30.0}$	57.8$_{+26.8}$	1.5$_{0.0}$	37.5$_{-7.7}$	0.0$_{-1.2}$	67.2$_{+40.0}$	27.2$_{+10.0}$	0.8$_{0.0}$	6.8$_{-37.6}$	3.2$_{-13.6}$
Llama3.1-8B-Instruct Family											
Base	3.3	32.5	20.2	0.8	38.6	4.1	60.4	86.4	2.0	28.8	8.4
RLVR (External Reward)											
Correct	6.7$_{+3.4}$	38.6$_{+6.1}$	25.1$_{+4.9}$	21.0$_{+20.2}$	49.1$_{+10.5}$	4.3$_{+0.2}$	76.0$_{+15.6}$	88.8$_{+2.4}$	15.6$_{+13.6}$	34.4$_{+7.6}$	11.6$_{+3.2}$
Random	3.3$_{0.0}$	26.8$_{-5.7}$	21.3$_{+1.1}$	0.0$_{-0.8}$	32.1$_{-6.5}$	4.1$_{0.0}$	69.2$_{+8.8}$	87.2$_{+0.8}$	0.8$_{-1.2}$	23.6$_{-5.2}$	4.4$_{-4.0}$
Incorrect	2.1$_{-1.2}$	26.4$_{-6.1}$	18.7$_{-1.5}$	0.8$_{0.0}$	30.2$_{-8.4}$	3.8$_{-0.3}$	70.0$_{+9.6}$	83.2$_{-3.2}$	0.8$_{-1.2}$	19.2$_{-9.6}$	4.4$_{-4.0}$
Format	3.1$_{-0.2}$	31.5$_{-1.0}$	18.7$_{-1.5}$	0.8$_{0.0}$	36.4$_{-2.2}$	4.1$_{0.0}$	68.8$_{+8.4}$	85.6$_{-0.8}$	2.0$_{0.0}$	28.0$_{-0.8}$	6.4$_{-2.0}$
Self-Rewarded Reinforcement Learning											
Vote	4.6$_{+1.3}$	37.7$_{+5.2}$	23.0$_{+2.8}$	1.5$_{+0.7}$	35.9$_{-2.7}$	4.3$_{+0.2}$	67.2$_{+6.8}$	83.2$_{-3.2}$	2.0$_{0.0}$	28.0$_{-0.8}$	8.8$_{+0.4}$
EM	5.1$_{+1.8}$	38.3$_{+5.8}$	25.0$_{+4.8}$	0.8$_{0.0}$	34.8$_{-3.8}$	4.1$_{0.0}$	73.6$_{+13.2}$	87.2$_{+0.8}$	2.0$_{0.0}$	23.6$_{-5.2}$	7.6$_{-0.8}$

Table 2: Comprehensive evaluation of different reward signals in RL. "Vote" denotes Majority Voting, "EM" means entropy minimization on self-generated samples only; OP: Operation ; CF: Counterfactual.

This section examines how reward signal quality affects RL performance in LLMs. Prior work shows that more accurate rewards do not always improve outcomes [4], and that strong models are surprisingly robust to noisy rewards, while weaker ones are not [11, 15]. We extend this analysis across diverse reward signals and model-task combinations (implementation details in Appendix D.1). We present results in Table 2. From the results, we identify three critical findings regarding the impact of reward signal quality on model performance (Appendix E.1 provides additional discussion):

Ground Truth Rewards Are Optimal. Across all models and tasks, ground truth rewards consistently yield the strongest gains—e.g., Qwen2.5-7B improves from 3.3 to 14.2 on AIME24 and from 40.8 to 71.0 on MATH500—establishing them as the gold standard for RL in reasoning.

Alignment Governs Noisy-Reward Robustness. Robustness to spurious rewards depends on model-task alignment. In strong-alignment settings (Red, Green), models tolerate random or inaccurate rewards—Qwen2.5 maintains math performance, and both models improve on Operation/Counterfac-

tual tasks. In weak-alignment settings (Gray), spurious rewards fail (e.g., Llama3.1 on math, both models on hard logical tasks), confirming alignment—not contamination—as the key factor.

Self-Rewarded Methods Are Limited. Self-rewarded approaches (e.g., majority voting, entropy minimization) consistently lag behind external rewards. Though majority voting reaches 69.4 on MATH500 with Qwen2.5, it falls short of ground truth rewards and generalizes poorly to logical reasoning across models. Test-Time Reinforcement Learning (TTRL) [26] is essentially no different from Self-Rewarded Reinforcement Learning when majority voting is employed. Thus, we are also curious whether TTRL remains effective for different models and in domains beyond mathematics. The results can be seen in Appendix E.2.

4 RQ2 – Is One-shot Enough for RL to Work?

Dataset	Math Tasks			Logic Tasks							
							KOR Benchmark				
	AIME24	MATH500	AMC	SynLogic	BBH	BBEH	OP	CF	Puzzle	Logic	Cipher
Qwen2.5-7B											
∅	3.3	40.8	31.0	1.5	45.2	1.2	27.2	17.2	0.8	8.0	4.8
full set	14.2$_{+10.9}$	71.0$_{+30.2}$	62.4$_{+31.4}$	42.6$_{+41.1}$	62.7$_{+17.5}$	6.8$_{+5.6}$	82.4$_{+55.2}$	79.6$_{+62.4}$	16.8$_{+10.0}$	46.4$_{+38.4}$	20.4$_{+15.6}$
random-1	10.7$_{+7.4}$	58.7$_{+17.9}$	53.1$_{+22.1}$	0.8$_{-0.7}$	40.2$_{-5.0}$	0.0$_{-1.2}$	60.4$_{+33.2}$	36.8$_{+19.6}$	0.8$_{0.0}$	6.4$_{-1.6}$	4.4$_{-0.4}$
random-2	12.5$_{+9.2}$	63.0$_{+22.2}$	55.7$_{+22.7}$	2.4$_{+0.9}$	43.1$_{-2.1}$	1.2$_{0.0}$	67.2$_{+40.0}$	56.8$_{+39.6}$	2.0$_{+1.2}$	3.2$_{-4.8}$	4.8$_{0.0}$
selected-1	12.3$_{+9.0}$	65.2$_{+24.4}$	55.2$_{+24.2}$	0.8$_{-0.7}$	39.9$_{-5.3}$	0.0$_{-1.2}$	69.2$_{+42.0}$	38.4$_{+21.2}$	0.8$_{0.0}$	8.0$_{0.0}$	6.4$_{+1.6}$
Llama3.1-8B-Instruct											
∅	3.3	32.5	20.2	0.8	38.6	4.1	60.4	86.4	2.0	28.8	8.4
full set	6.7$_{+3.4}$	38.6$_{+6.1}$	25.1$_{+4.9}$	21.0$_{+20.2}$	49.1$_{+10.5}$	4.3$_{+0.2}$	76.0$_{+15.6}$	88.8$_{+2.4}$	15.6$_{+13.6}$	34.4$_{+7.6}$	11.6$_{+3.2}$
random-1	3.8$_{+0.5}$	30.5$_{-2.0}$	21.1$_{+0.9}$	0.8$_{0.0}$	35.1$_{-3.5}$	3.8$_{-0.3}$	73.6$_{+13.2}$	85.6$_{-0.8}$	1.2$_{-0.8}$	28.0$_{-0.8}$	8.8$_{+0.4}$
random-2	2.7$_{-0.6}$	33.1$_{+0.6}$	21.1$_{+0.9}$	0.8$_{0.0}$	36.7$_{-1.9}$	4.1$_{0.0}$	70.0$_{+9.6}$	86.4$_{0.0}$	2.8$_{+0.8}$	27.2$_{-1.6}$	8.4$_{0.0}$
selected-1	3.7$_{+0.4}$	30.3$_{-2.2}$	22.3$_{+2.1}$	0.8$_{0.0}$	34.4$_{-4.2}$	3.8$_{-0.3}$	69.2$_{+8.8}$	88.8$_{+2.4}$	2.0$_{0.0}$	19.2$_{-9.6}$	6.8$_{-1.6}$

Table 3: One-shot RL Results. OP: Operation; CF: Counterfactual. We only observe the effectiveness of one-shot reinforcement learning in settings with strong model-task alignment (red and green).

Existing work [21] showed that training on a single carefully selected question can match full-dataset performance, challenging conventional RL data requirements. They select samples using reward-variance-based criteria, denoted as $m_{selected}$ (math) and $l_{selected}$ (logic). For comparison, we also use one or two randomly chosen samples: (m_{random}, l_{random}) and $(m'_{random}, l'_{random})$. Specific examples are in Appendix H; all other settings follow Appendix D, with 300 training steps. We present results in Table 3. Based on the experimental results, we identify two critical findings regarding the effectiveness of one-shot RL (Appendix F provides additional discussion):

One-shot RL succeeds only under strong model-task alignment. In aligned settings (Red, Green), models generalize well from a single example: Qwen2.5-7B nearly matches full-dataset performance on MATH500 (65.2 vs. 71.0), and both models improve significantly on Operation/Counterfactual tasks (e.g., Llama on Operation: 69.2 vs. baseline 60.4). In weak-alignment settings (Gray), gains are minimal, indicating one-shot RL is effective only when strong foundational capabilities exist.

Sample selection strategy has limited impact. The reward-variance-based selection offers little consistent advantage over random sampling: on MATH500, Qwen2.5 achieves 65.2 (selected) vs. 58.7–63.0 (random); for Llama3.1, differences are negligible across tasks—challenging the presumed superiority of sophisticated selection.

5 RQ3 — Does RL Work with Only Negative Samples?

Recent work [25] has demonstrated that training exclusively on negative samples can be surprisingly effective for model reasoning. However, these findings are primarily observed in scenarios with strong model-task alignment. We investigate whether negative-only training generalizes to weak model-task alignment scenarios, where models lack strong foundational capabilities.

Table 4 summarizes the performance of NSR and PSR relative to the full-signal DAPO baseline across our three experimental categories. It reveals distinct patterns based on model-task alignment (more discussion can be seen in Appendix G):

	Math Tasks			Logic Tasks							
							KOR Benchmark				
	AIME24	MATH500	AMC	SynLogic	BBH	BBEH	OP	CF	Puzzle	Logic	Cipher
Qwen2.5-7B	3.3	40.8	31.0	1.5	45.2	1.2	27.2	17.2	0.8	8.0	4.8
DAPO	14.2$_{+10.9}$	71.0$_{+30.2}$	62.4$_{+31.4}$	42.6$_{+41.1}$	62.7$_{+17.5}$	6.8$_{+5.6}$	82.4$_{+55.2}$	79.6$_{+62.4}$	16.8$_{+10.0}$	46.4$_{+38.4}$	20.4$_{+15.6}$
NSR	13.9$_{+10.6}$	68.7$_{+27.9}$	63.5$_{+32.5}$	1.5$_{0.0}$	41.2$_{-4.0}$	1.6$_{+0.4}$	60.4$_{+33.2}$	36.8$_{+19.6}$	2.0$_{+1.2}$	6.8$_{-1.2}$	4.8$_{0.0}$
PSR	14.0$_{+10.7}$	70.3$_{+29.5}$	63.1$_{+32.1}$	24.8$_{+23.3}$	57.1$_{+11.9}$	4.3$_{+3.1}$	73.6$_{+46.4}$	38.4$_{+21.2}$	9.2$_{+8.4}$	31.2$_{+23.2}$	11.2$_{+6.4}$
Llama3.1-8B	3.3	32.5	20.2	0.8	38.6	4.1	60.4	86.4	2.0	28.8	8.4
DAPO	6.7$_{+3.4}$	38.6$_{+6.1}$	25.1$_{+4.9}$	21.0$_{+20.2}$	49.1$_{+10.5}$	4.3$_{+0.2}$	76.0$_{+15.6}$	88.8$_{+2.4}$	15.6$_{+13.6}$	34.4$_{+7.6}$	11.6$_{+3.2}$
NSR	7.9$_{+4.6}$	36.9$_{+4.4}$	24.7$_{+4.5}$	0.0$_{-0.8}$	34.2$_{-4.4}$	4.3$_{+0.2}$	67.2$_{+6.8}$	86.4$_{0.0}$	2.0$_{0.0}$	28.0$_{-0.8}$	5.2$_{-3.2}$
PSR	7.9$_{+4.6}$	35.7$_{+4.2}$	23.6$_{+3.4}$	13.0$_{+11.5}$	43.3$_{+4.7}$	4.1$_{0.0}$	69.2$_{+8.8}$	89.6$_{+3.2}$	12.0$_{+11.2}$	34.4$_{+7.6}$	10.8$_{+2.4}$

Table 4: Results of NSR and PSR under different settings. When Model-Task alignment is strong, both NSR and PSR yield pronounced performance gains for all models (Red and Green). Conversely, under weak alignment, NSR-trained models exhibit no noticeable improvement (Gray).

Strong Model-Task Alignment Enables Effective Negative-Sample Learning. In strong alignment settings (Red and Green), both negative-sample-only (NSR) and positive-sample-only (PSR) training recover nearly all the gains of full-signal DAPO. For example, Qwen2.5-7B on MATH500 achieves 68.7 (NSR) and 70.3 (PSR) versus DAPO's 71.0—demonstrating that when models already possess strong domain capabilities, either signal alone can effectively drive learning.

Weak Alignment Reveals the Advantage of Positive-Only Signals. In weak alignment settings (Gray), PSR consistently outperforms NSR across logical reasoning tasks. On SynLogic, PSR yields substantial improvements (Qwen2.5-7B: 1.5 $\rightarrow$ 24.8; Llama3.1-8B: 0.8 $\rightarrow$ 13.0), whereas NSR provides minimal gains. This indicates that while both approaches work under strong alignment, PSR is significantly more robust when models lack foundational expertise.

6 Conclusion

This work reveals that *Model-Task Alignment* strength, measured by pass@k accuracy, serves as the fundamental determinant of when counterintuitive RL phenomena emerge in language model reasoning. We demonstrate that remarkable behaviors—including robustness to spurious rewards, one-shot training effectiveness, and negative-only signal sufficiency—manifest primarily when models already possess strong foundational capabilities in the target domain, functioning more as capability elicitation mechanisms rather than genuine learning drivers for unfamiliar tasks.

References

[1] Shivam Agarwal, Zimin Zhang, Lifan Yuan, Jiawei Han, and Hao Peng. The unreasonable effectiveness of entropy minimization in llm reasoning, 2025.

[2] AIME. Art of Problem Solving — artofproblemsolving.com. `https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions`, 2024. [Accessed 26-08-2025].

[3] AMC. Art of Problem Solving — artofproblemsolving.com. `https://artofproblemsolving.com/wiki/index.php/AMC_12_Problems_and_Solutions`, 2023. [Accessed 26-08-2025].

[4] Yanjun Chen, Dawei Zhu, Yirong Sun, Xinghao Chen, Wei Zhang, and Xiaoyu Shen. The accuracy paradox in rlhf: When better reward models don't yield better language models. *arXiv preprint arXiv:2410.06554*, 2024.

[5] 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.

[6] 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, 2021.

[7] 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.

[8] Mehran Kazemi, Bahare Fatemi, Hritik Bansal, John Palowitch, Chrysovalantis Anastasiou, Sanket Vaibhav Mehta, Lalit K. Jain, Virginia Aglietti, Disha Jindal, Peter Chen, Nishanth Dikkala, Gladys Tyen, Xin Liu, Uri Shalit, Silvia Chiappa, Kate Olszewska, Yi Tay, Vinh Q. Tran, Quoc V. Le, and Orhan Firat. Big-bench extra hard, 2025.

[9] Junteng Liu, Yuanxiang Fan, Zhuo Jiang, Han Ding, Yongyi Hu, Chi Zhang, Yiqi Shi, Shitong Weng, Aili Chen, Shiqi Chen, Yunan Huang, Mozhi Zhang, Pengyu Zhao, Junjie Yan, and Junxian He. Synlogic: Synthesizing verifiable reasoning data at scale for learning logical reasoning and beyond, 2025.

[10] Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Tianjun Zhang, Erran Li, Raluca Ada Popa, and Ion Stoica. Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl. Link, 2025. Notion Blog.

[11] Ang Lv, Ruobing Xie, Xingwu Sun, Zhanhui Kang, and Rui Yan. The climb carves wisdom deeper than the summit: On the noisy rewards in learning to reason. *arXiv preprint arXiv:2505.22653*, 2025.

[12] Kaijing Ma, Xinrun Du, Yunran Wang, Haoran Zhang, Zhoufutu Wen, Xingwei Qu, Jian Yang, Jiaheng Liu, Minghao Liu, Xiang Yue, Wenhao Huang, and Ge Zhang. Kor-bench: Benchmarking language models on knowledge-orthogonal reasoning tasks, 2024.

[13] Meta. The llama 3 herd of models, 2024.

[14] Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025.

[15] Rulin Shao, Shuyue Stella Li, Rui Xin, Scott Geng, Yiping Wang, Sewoong Oh, Simon Shaolei Du, Nathan Lambert, Sewon Min, Ranjay Krishna, et al. Spurious rewards: Rethinking training signals in rlvr. *arXiv preprint arXiv:2506.10947*, 2025.

[16] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. Mastering chess and shogi by self-play with a general reinforcement learning algorithm, 2017.

[17] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy P. Lillicrap, Fan Hui, L. Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of go without human knowledge. *Nature*, 550:354–359, 2017.

[18] Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, , and Jason Wei. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*, 2022.

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

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

[21] Yiping Wang, Qing Yang, Zhiyuan Zeng, Liliang Ren, Liyuan 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.

[22] Mingqi Wu, Zhihao Zhang, Qiaole Dong, Zhiheng Xi, Jun Zhao, Senjie Jin, Xiaoran Fan, Yuhao Zhou, Yanwei Fu, Qin Liu, et al. Reasoning or memorization? unreliable results of reinforcement learning due to data contamination. *arXiv preprint arXiv:2507.10532*, 2025.

[23] 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.

[24] An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement, 2024.

[25] Xinyu Zhu, Mengzhou Xia, Zhepei Wei, Wei-Lin Chen, Danqi Chen, and Yu Meng. The surprising effectiveness of negative reinforcement in llm reasoning, 2025.

[26] Yuxin Zuo, Kaiyan Zhang, Li Sheng, Shang Qu, Ganqu Cui, Xuekai Zhu, Haozhan Li, Yuchen Zhang, Xinwei Long, Ermo Hua, et al. Ttrl: Test-time reinforcement learning. *arXiv preprint arXiv:2504.16084*, 2025.

A More Pass@k Results

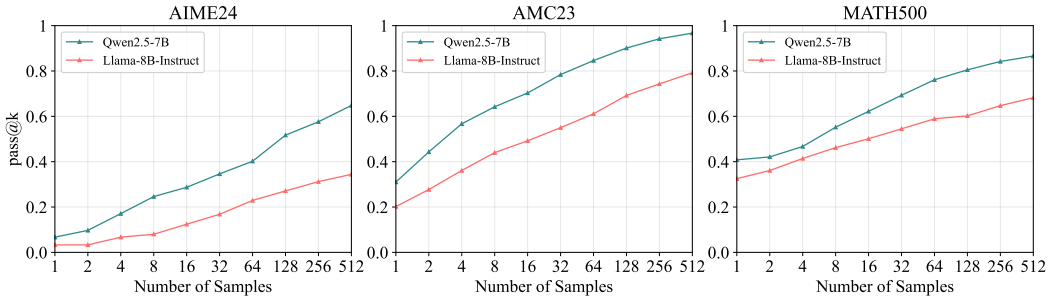

Figure 3: Pass@k for math tasks. Qwen demonstrates strong capabilities across all three mathematical evaluation datasets.

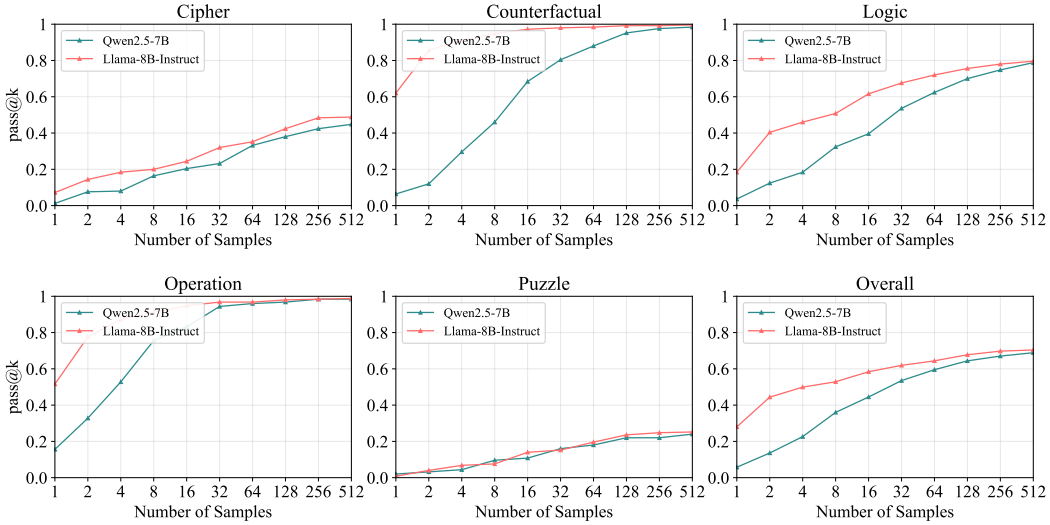

Figure 4: Pass@k for KOR-Bench. Both models demonstrate strong inherent reasoning capabilities in Operation and Counterfactual subtasks, but exhibit limited inherent logical reasoning abilities in Cipher, Puzzle and Logic.

B Model and Evaluated Tasks

Building on our *Model-Task Alignment Dependency* hypothesis, we strategically design model-task combinations that test the boundaries of current claims in RL for language model reasoning. Our experimental design is motivated by the critical need to distinguish between findings that represent universal RL properties versus those that emerge from specific model-task capability alignments. We evaluate two representative language models from different families: Qwen2.5-7B-Base [14] and Llama-3.1-8B-Instruct [13], enabling systematic comparison across model architectures with varying baseline capabilities while controlling for architectural differences at comparable parameter scales.

Our evaluation encompasses mathematical and logical reasoning domains. For mathematical reasoning, we employ AIME24 [2], MATH500 [6] and AMC23 [3]. For logical reasoning, we utilize SynLogic [9] (synthetic puzzles with 35 task types, we use the validation split), BBH [18] (multi-step reasoning tasks), BBEH [8] (extended-difficulty version), and KOR-Bench [12] (knowledge-orthogonal reasoning across five categories).

C Contamination Evaluation

C.1 Implementation Details

Our contamination analysis follows a systematic prompt truncation methodology to evaluate potential data leakage across model-task combinations. Original prompts are truncated at varying ratios (0.4, 0.6, and 0.8) while preserving word boundaries, and models are asked to complete the remaining content using greedy decoding for deterministic outputs. We measure contamination using ROUGE-L scores between model completions and the actual remaining prompt content, where a perfect score of 1.0 indicates complete reconstruction and potential contamination. The evaluation pipeline employs distributed processing to handle complex mathematical expressions and prevent evaluation timeouts, with results aggregated across multiple rollouts to ensure statistical reliability.

C.2 More Contamination Results

Task Type	Benchmark	Model	Portion=0.4		Portion=0.6		Portion=0.8	
			ROUGE	EM	ROUGE	EM	ROUGE	EM
Math Tasks	AMC 23	Qwen2.5-7B	63.78	23.91	64.42	33.73	73.23	49.39
		Llama-3.1-8B	27.18	0.00	30.64	0.00	44.54	4.81
	MATH500	Qwen2.5-7B	50.36	8.20	60.98	21.20	66.42	40.20
		Llama-3.1-8B	23.09	0.60	40.56	3.80	48.33	17.8
	AIME24	Qwen2.5-7B	44.64	10.00	48.69	13.33	60.08	30.00
		Llama-3.1-8B	26.08	0.00	30.80	0.00	50.50	13.33
Logic Tasks	Puzzle	Qwen2.5-7B	19.56	0.00	19.62	0.00	19.24	0.00
		Llama-3.1-8B	18.27	0.00	17.31	0.00	15.85	0.00
	Operation	Qwen2.5-7B	21.37	0.00	24.25	0.00	20.18	0.00
		Llama-3.1-8B	21.83	0.00	18.34	0.00	16.75	0.00
	Counterfactual	Qwen2.5-7B	18.88	0.00	19.96	0.00	18.66	0.00
		Llama-3.1-8B	19.02	0.00	19.39	0.00	18.94	0.00
	Logic	Qwen2.5-7B	22.08	0.00	27.28	0.00	28.23	0.00
		Llama-3.1-8B	21.38	0.00	28.37	0.00	28.42	0.00
	Cipher	Qwen2.5-7B	34.61	0.00	41.03	0.00	44.77	0.00
		Llama-3.1-8B	29.59	0.00	36.95	0.00	42.93	0.00

Table 5: Extended Contamination Analysis across model-task combinations. **Red** indicates potential contamination with strong baseline performance; **Gray** indicates no contamination with weak baseline performance; **Green** indicates no contamination with strong baseline performance.

D Experimental Setup

Training Datasets and Evaluation. Except for the experiments on Test-Time RL (Appendix E.2), we use DeepScaleR [10] as the training set for mathematical tasks and the training split of SynLogic-Easy [9] for logical tasks. Evaluation datasets are as described in Appendix B. Following SynLogic [9], all evaluations are conducted in a zero-shot setting, with avg@8 metrics computed for AIME 2024 and SynLogic to mitigate variance.

Training Configuration. Our experiments default to using the DAPO algorithm unless otherwise specified. We set $\epsilon_{low} = 0.2, \epsilon_{high} = 0.28$, max promt length=2048, max generation length=8192. We use dynamic sampling, and set max_num_gen_batches $= 2$. We found that for logical task training, each sampled batch often contains very few samples with non-zero reward variance. We made two improvements: (1) When neither of the two generated sampling batches contains any samples

with non-zero reward variance (which usually happens in the early stages of SynLogic training when the model cannot get any questions right), we use the second generated batch as the training batch. (2) When the number of available samples from the two generations is less than the training batch size, we duplicate the samples to match the training batch size. We don't use length penalty. During most training experiments, we set $lr = 1e^{-6}$, batch size $= 128$, mini batch size $= 64$, temperature $= 1.0$.

D.1 Implementation Details of RQ1

Following the setting described in Appendix D, we train with different rewards for 300 steps on mathematical and logical reasoning tasks, respectively. The format reward is different from that of [15], we use the same template as SynLogic. In addition to this, the definition of the reward functions is consistent.

D.2 Implementation Details of RQ3

In our implementation, Negative Sample Reinforcement (NSR) masks out all trajectories with reward 1 (correct answers) when computing the policy gradient, leaving only negative-rewarded samples to drive updates. Conversely, Positive Sample Reinforcement (PSR) ignores trajectories with reward 0 and optimizes only on positively rewarded samples. All other hyperparameters remain identical to the DAPO baseline described in Appendix D.

E More RQ1 Experimental Result

E.1 Discussion about RQ1

How Different Reward Signals Affect the Behavior of LLMs. In mathematical tasks, employing ground truth rewards decreases the frequency of code usage in model responses[15]. Their study also revealed that, in contrast to Qwen2.5-Math [24], the accuracy improvement of the Qwen2.5 Base model was primarily attributed to a shift from code-based reasoning to language-based reasoning. As shown in Table 6, we identify analogous trends in mathematical tasks. Specifically, for logic puzzles, the application of ground truth rewards similarly reduces the incidence of code in responses. However, other types of rewards, particularly format and random rewards, do not demonstrate a significant impact on diminishing code usage frequency. We speculate that, throughout the RL training process, ground truth rewards can steer the model away from its old reasoning pattern (i.e., producing reasoning responses with code) and toward a more natural, language-based reasoning pattern.

Reward Type	MATH500		SynLogic	
	Before RL	After RL	Before RL	After RL
Correct	89.1	**12.4**	57.3	**21.7**
Random		94.2		48.2
Format		96.7		50.7
Incorrect		28.1		28.3

Table 6: Code Usage Count of Qwen2.5-7B before and after RL training with different rewards.

As shown in Table 2, spurious rewards are effective only on the Operation and Counterfactual for the Llama model; consequently, we also report the frequency of code-based reasoning before and after training on these two tasks. As shown in Table 7, we observe that, both before and after RL training, Llama almost never invokes code during the reasoning process. We attribute the sporadic use of code (0.8) to the fact that some SynLogic tasks explicitly require outputs to be presented as code blocks. This indicates that Llama and Qwen exhibit distinct reasoning patterns even though they both benefit from noisy reward signals in these settings.

E.2 Test-Time RL

Test-Time Reinforcement Learning (TTRL) [26] addresses a fundamental challenge in LLM development: how to improve model performance on unlabeled test data without access to ground-truth labels for reward signals. It prompts the model to generate multiple responses to each test question and use the most frequent answer as the label for reward signals. Although the model is trained on

Reward Type	Operation		Counterfactual	
	Before RL	After RL	Before RL	After RL
Correct	0.0	0.8	0.0	0.0
Random		0.0		0.0
Format		0.0		0.0
Incorrect		0.0		0.0

Table 7: Code Usage Count of Llama-3.1-8B-Instruct before and after RL training on two tasks.

the unlabeled test set, this approach is essentially no different from Self-Rewarded Reinforcement Learning when majority voting is employed. Thus, we are also curious whether TTRL remains effective for different models and in domains beyond mathematics.

Table 8 shows the results of the Qwen and Llama models on different tasks. Due to the limited scale of the test dataset, we trained for 30 steps on all test datasets. It could be observed that in settings where the model–task alignment is strong, TTRL yields substantial improvements, as exemplified by Qwen on math tasks and Operation subset. For tasks in which the model lacks initial prior knowledge, TTRL fails to deliver improvements or yields only marginal gains. As discussed by [26], majority voting is the foundation of TTRL. We also recorded the variation of Maj@16 during the training process; the results are shown in Table 9. We can observe that, in settings where TTRL yields substantial improvements, Maj@16 consistently rises throughout training. Especially for Qwen on Operation subset, it achieves an absolute gain of 16.4 points. This further underscores that TTRL's efficacy hinges on strong model–task alignment, rather than on contamination.

Model	MATH500	SynLogic	OP	Model	MATH500	SynLogic	OP
Qwen2.5-7B	40.8	1.5	27.2	Llama-3.1-8B-Instruct	32.5	0.8	60.4
+TTRL	62.1$_{+21.3}$	1.8$_{+0.3}$	55.6$_{+28.4}$	+TTRL	41.2$_{+8.7}$	0.8$_{0.0}$	83.6$_{+23.2}$

Table 8: Test-Time Reinforcement Learning (TTRL) performance changes. TTRL produces significant gains only when model-task alignment is strong (red and green cells).

	Step 0	Step 5	Step 10	Step 15	Step 20	Step 25	Step 30
Qwen+Math500	54.2	60.6	64.3	68.2	67.1	69.3	70.5+16.3
Qwen+SynLogic	2.2	3.0	3.7	4.4	4.4	4.4	5.2+3.0
Qwen+OP	46.0	53.6	55.6	57.2	58.8	60.0	60.0+16.4
Llama+Math500	46.3	48.6	51.3	53.2	53.9	55.0	54.7+8.4
Llama+SynLogic	1.5	1.5	2.2	1.5	2.2	2.2	2.2+0.7
Llama+OP	73.6	78.0	79.6	84.0	83.6	86.8	88.4+14.8

Table 9: The variation of Maj@16 as training progresses. In tasks where TTRL brings significant improvements (red and green), Maj@16 continues to improve with training.

F Discussion of RQ2

Training on a single sample for mathematical tasks can quickly improve the accuracy of that sample and also lead to improvements on the test set [21]. We attempt to verify this conclusion on logical tasks. Considering that the initial rollout accuracy of the model on $l_{selected}$ is 0, we additionally sample two examples whose initial rollout accuracies on Qwen2.5-7B are $5/16$ and $1/16$ (on Llama-3.1-8B-Instruct are $3/16$ and $1/16$), denoted as l_{simple} and l_{mid}. During training, we track three metrics: the rollout accuracy of these examples acc_{1-shot}, the accuracy of the subtask to which this example belongs (in-distribution) acc_{id}, and the accuracy of other subtasks in SynLogic (out-of-distribution) acc_{ood}. The results are shown in Figure 5.

One-shot RL possesses the ability to generalize within the distribution. When the problem is relatively simple (with an initial rollout accuracy that is not zero), the model's rollout accuracy on that sample quickly increases. Although the initial rollout accuracy of l_{mid} on Qwen is only one-fifth that of l_{simple} (on Llama is one-third), it still attains a high rollout accuracy within a few dozen steps.

Since GRPO and DAPO compute advantages via intra-group normalization, the model is unable to derive any informative feedback from samples whose initial rollout accuracy is zero. Moreover, we observe that the test accuracy for the same subtask also continues to improve, demonstrating effective within-distribution generalization.

One-shot RL struggles to generalize to other types of logic puzzles. We find that while models can improve on tasks similar to their training example, they fail to transfer learning to different puzzle types. This suggests that one-shot learning primarily exploits existing model capabilities rather than developing new reasoning skills.

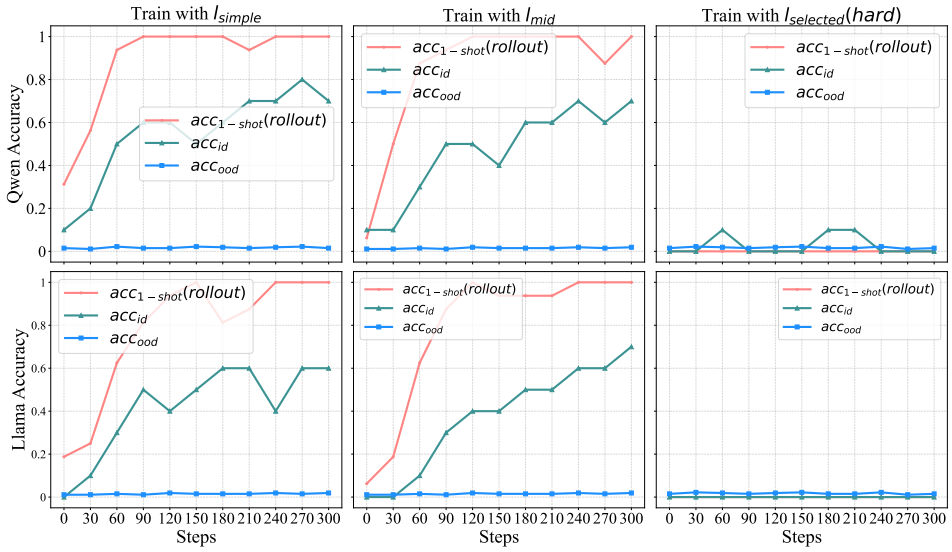

Figure 5: The changes in two models' accuracy during the training. If the initial rollout accuracy is non-zero, both models rapidly fit the employed samples (l_{simple}, l_{mid}) and exhibit generalization within the same subtask; however, we observe no generalization to puzzles of other types.

F.1 More Discussion about Difficult Example in One-shot RL

During training with $l_{selected}$, apart from the rollout accuracy (reward) remaining consistently at 0, metrics such as entropy and response length also exhibit almost no changes. As shown in Figure 6, after 300 training steps, the model still maintains a large reinforcement learning exploration space.

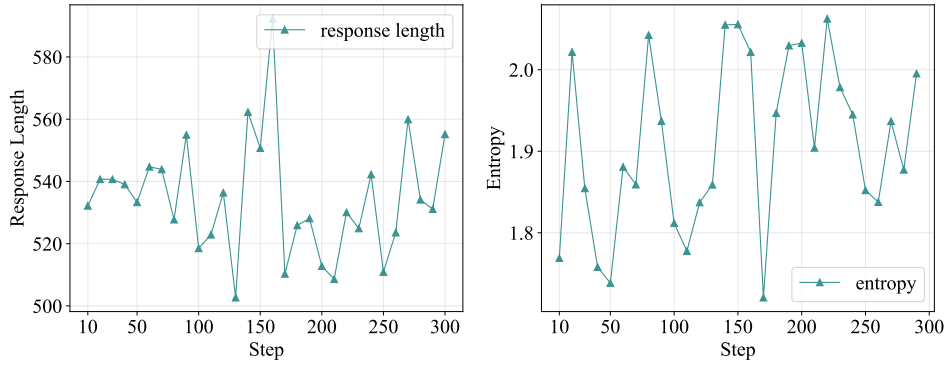

Figure 6: Training Dynamics of Qwen2.5-7B when trained with $l_{selected}$. Entropy and response length exhibit almost no changes.

G Discussion of RQ3

The relationship between positive and negative samples in reinforcement learning is fundamentally connected to the exploration-exploitation trade-off, with entropy serving as a key mediator. To

elucidate these dynamics in our experimental context, we examine how different sample types affect the exploration-exploitation balance through their impact on training entropy.

Negative Signals Help Maintain Exploration. Figure 7 plots token-level entropy throughout training. Consistent with [25], NSR slows entropy collapse, especially on mathematical tasks—suggesting that penalising only erroneous trajectories can preserve output diversity. However, the flatter entropy curve on logical tasks corresponds to poorer final accuracy.

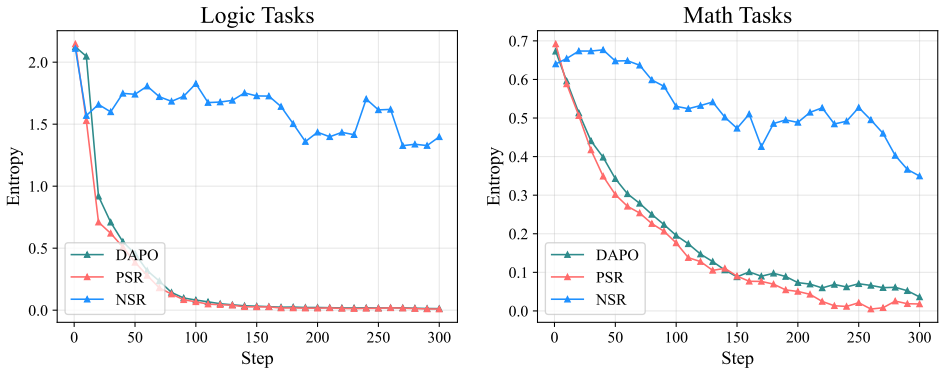

Figure 7: Entropy Dynamics of Qwen2.5-7B during Training. NSR can maintain the exploration space of RL, but a larger exploration space is not always favorable, as in logical tasks.

H Few-shot RL Example Details

Details of example $m_{selected}$

How many positive divisors do 9240 and 13860 have in common?

Details of example m_{random}

The angles of quadrilateral $PQRS$ satisfy $\angle P = 3\angle Q = 4\angle R = 6\angle S$. What is the degree measure of $\angle P$?

Details of example m'_{random}

Given a finite sequence $S = (a_1, a_2, \ldots, a_n)$ of n real numbers, let $A(S)$ be the sequence $\left(\frac{a_1+a_2}{2}, \frac{a_2+a_3}{2}, \ldots, \frac{a_{n-1}+a_n}{2}\right)$ of $n-1$ real numbers. Define $A^1(S) = A(S)$ and, for each integer m, $2 \leq m \leq n-1$, define $A^m(S) = A(A^{m-1}(S))$. Suppose $x > 0$, and let $S = (1, x, x^2, \ldots, x^{100})$. If $A^{100}(S) = \left(\frac{1}{2^{50}}\right)$, then what is x? **AND** If $x, 2x+2, 3x+3, \ldots$ are in geometric progression, the fourth term is:

Details of example $l_{selected}$

Here's a mathematical expression: ?-?+(6%5)*2-?+?/?/?/4/2 = 2. The digits on the left side of the equation have been replaced with question marks. Each question mark corresponds to a digit between 0 and 9. You need to try replacing the question marks with the correct digits to restore the expression.Please put the complete expression with the filled - in digits between [[and]] at the end of your response, with no other content, like this: [[2 + 4 * 3 - 4 = 10]]

Details of example l_{random}

Solve this cryptarithm: RRYUU + UYR + U = RYUUU (where RRYUU is a 5-digit number, UYR is a 3-digit number, U is a 1-digit number, and RYUUU is a 5-digit number). Each letter represents a unique digit. Find the digit substitution that makes the equation true.

Details of example l'_{random}

```
In this Number Wall puzzle, add walls (marked as 'A') to divide the
grid into islands.  Each island must contain exactly one number, and
its size must equal that number.
Grid:
+--+--+--+
| X | 3 | X |
+--+--+--+
| X | X | X |
+--+--+--+
| X | X | X |
+--+--+--+
Rules:
- Each island must contain exactly one number.
- The total number of cells in an island (including the number cell)
must equal the value of that number.
- All cells within an island must be connected horizontally or
vertically.
- Walls (marked as 'A') cannot form 2×2 or larger continuous
rectangles.
- All islands must be separated by walls.
```

AND

```
In the cryptarithm:  MMII + MIXIMM = MMXIIX, each letter stands for a
different digit (MMII is 4 digits, MIXIMM is 6 digits, and MMXIIX
is 6 digits).  Determine what each letter represents to make the
equation true.
```

Details of example l_{simple}

```
In this word sorting challenge, you need to rearrange words in
increasing based on a modified alphabet where l,z and a are the first
letters.  Words to sort:  yachted,coelomic,harateen.  Write your
final answer inside:  \boxed,like this:  \boxedword1,word2,word3.
```

Details of example l_{mid}

```
You are an expert proficient in Dyck language, where you must
complete all types of unclosed brackets (e.g., [], , <>) in
language sequences.  You need to analyze the steps of bracket
pairing according to Dyck language rules.  Given an initial Dyck
language sequence and steps for deriving the closed bracket sequence
(presented in a thinking process format), your task is to identify
locations with incorrect reasoning in the Dyck language, and there
may be multiple errors.  This could be forgetting to close a bracket,
using the wrong closing bracket, or incorrectly copying a subsequence
of closing brackets in the next step.  Task:  Check the sequence to
ensure brackets are properly closed.  Input:  [[(){}]]{}
Thought 1:  We should process the input one by one and track the
stack configuration.
Thought 2:  Stack:  Empty
Thought 3:  [ ; Stack:  Empty
Thought 4:  [ ; Stack:  [[
Thought 5:  ( ; Stack:  [[(
Thought 6:  )  ; Stack:  [[
Thought 7:  { ; Stack:  [[{
Thought 8:  } ; Stack:  [[
Thought 9:  ]  ; Stack:  [
Thought 10:  ]  ; Stack:  Empty
Thought 11:  { ; Stack:  {
Thought 12:  } ; Stack:  Empty
Thought 13:  Now, we have reached the end.  The final stack is empty.
Question:  Are there any reasoning errors in this sequence?
```