RLPR: Extrapolating RLVR to general domains without verifiers

Anonymous ACL submission

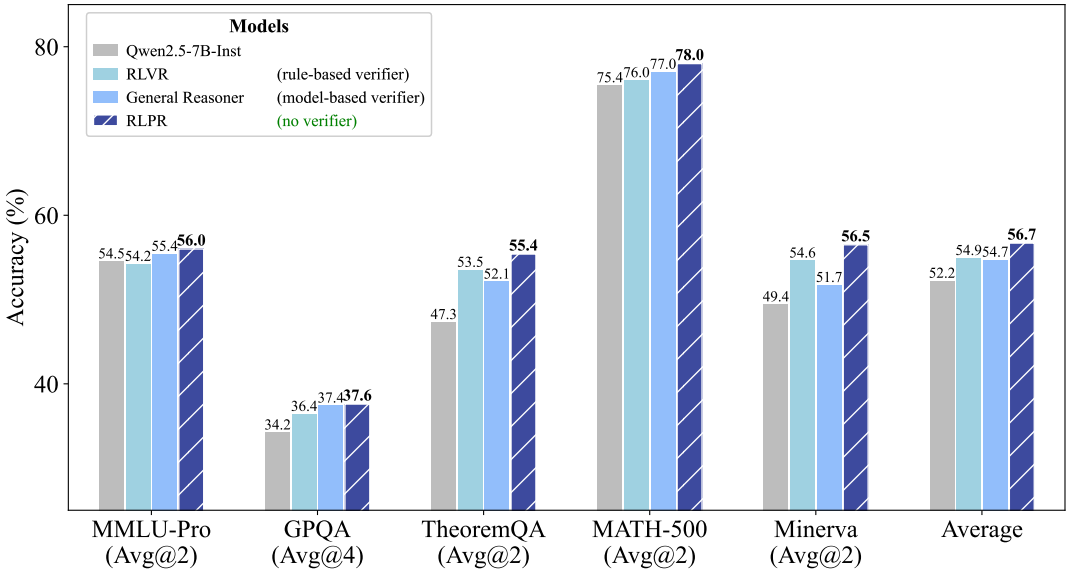


Figure 1: Overall performance on general-domain and mathematical reasoning benchmarks. By simply replacing the rule-based verifier reward of RLVR with the proposed LLM's intrinsic probability reward, **RLPR** achieves consistent improvements in both mathematical and general domains, even outperforming strong RL methods driven by model-based verifier reward. Average: average accuracy of five benchmarks. Verifier requirements of different methods are listed in parentheses.

Abstract

Reinforcement Learning with Verifiable Rewards (RLVR) demonstrates promising potential in advancing the reasoning capabilities of LLMs. However, its success remains largely confined to mathematical and code domains. This primary limitation stems from the heavy reliance on domain-specific verifiers, which results in prohibitive complexity and limited scalability. To address the challenge, our key observation is that LLM's intrinsic probability of generating a correct free-form answer directly indicates its own evaluation of the reasoning reward (i.e., how well the reasoning process leads to the correct answer). Building on this insight, we propose **RLPR**, a simple verifier-free framework that extrapolates RLVR to broader general domains. **RLPR** uses the LLM's own token probability scores for reference answers as the reward signal and maximizes the expected reward during training. We find that addressing the high variance of this noisy probability reward is crucial to make it work, and propose prob-to-reward and stabilizing methods to ensure a precise and stable reward from LLM intrinsic probabilities. Comprehensive experiments in four general-domain benchmarks and three mathematical benchmarks show that **RLPR** consistently improves reasoning capabilities in both areas for Gemma, Llama, and Qwen based models. Notably, **RLPR** outperforms concurrent VeriFree by 7.6 points on TheoremQA and 7.5 points on Minerva, and even surpasses strong verifier-model-dependent approaches General-Reasoner by 1.6 average points across seven benchmarks.

1 Introduction

Reinforcement Learning with Verifiable Rewards (RLVR) has emerged as a promising paradigm to advance the reasoning capabilities of Large Language Models (LLMs) (Jaech et al., 2024; DeepSeek-AI et al., 2025; Hu et al., 2025b). This paradigm not only shows the power of scaling test-time computation to address complex problems, but also sheds valuable light on paths to AGI with incentivized exploration and evolution.

However, in contrast to the pretraining of LLMs that can learn foundational capabilities from general domain data, most RLVR methods are confined to mathematics (Hu et al., 2025b; Liu et al., 2025b; Zeng et al., 2025; Yu et al., 2025) and code gen-

eration (Luo et al., 2025a; He et al., 2025; Cui et al., 2025a). The primary reason is that existing RLVR methods heavily rely on domain-specific verifiers to obtain reward, as shown in Figure 2. The most widely adopted verifiers are handcrafted rules (Hu et al., 2025b; Liu et al., 2025b; Zeng et al., 2025). Extending these rule-based reward systems to new models and domains typically requires prohibitive heuristic engineering. Moreover, for general-domain reasoning with free-form answers, it is even impossible to devise rule-based verifiers due to the high diversity and complexity of natural language. Recent works attempt to address this problem by training specialized LLMs as verifier models (Ma et al., 2025). However, training LLMs for general reward evaluation requires non-trivial and extensive data annotation, which often leads to unsatisfactory reward quality in practice. Involving separate verifier models also complicates the RL training framework and introduces additional computation cost. As a result, this scalability problem prevents existing RLVR methods from utilizing rich general-domain data and limits the potential of broader reasoning capabilities.

To address the problem, we propose the **RLPR** framework (**R**einforcement **L**earning with Reference **P**robability **R**eward) that extrapolates general-domain RLVR without external verifiers. *The basic insight is that LLM's intrinsic probability of generating a correct answer directly indicates its own evaluation of the reasoning reward* (i.e., how well the reasoning process leads to the correct answer). It also reflects the policy by measuring how likely the LLM is to take the correct action. Therefore, we can directly leverage this probability signal as a reward to incentivize reasoning for the correct answer in general domains. Since this probability score is a natural built-in of LLM's foundational capabilities, it offers good coverage and potential for reward evaluation even without any specialized fine-tuning. It can also better deal with the complexity and diversity of free-form natural language answers, giving reasonable reward even to partially correct answers.

Specifically, **RLPR** introduces two key innovations: (1) At the reward modeling level, we propose a simple and scalable alternative to the explicit reward from external verifiers with an intrinsic Probability-based Reward (PR), calculated by the average decoding probabilities of the reference answer tokens. Compared with naive sequence likelihood as reward (Zhou et al., 2025), the proposed PR shows better robustness and higher reward quality on quantitative examinations (see Figure 4). Moreover, we propose a simple debiasing method to eliminate the reward bias from text by optimizing the reward advantage over the same prompt without reasoning. (2) At the training level, we propose an adaptive curriculum learning mechanism to stabilize training. We adaptively remove prompts yielding low reward standard deviation (indicating prompts that are too easy or too complex), using a dynamic threshold based on the exponential moving average of past rewards' standard deviation. We find that this approach can well keep up with the reward distribution shifts, and improves both the training stability and final performance.

Comprehensive experiments on seven benchmarks show that, without any external verifiers, **RLPR** substantially enhances reasoning capabilities in both mathematical and general domains. Leveraging Qwen2.5-7B (Team, 2024) as base model, **RLPR** achieves 56.0 on MMLU-Pro and 55.4 on TheoremQA, even surpassing the strong General Reasoner-7B (Ma et al., 2025) that utilizes a specially trained 1.5B verifier model. Furthermore, compared with VeriFree (Zhou et al., 2025), a concurrent verifier-free approach, **RLPR** achieves significant improvement of 7.6 on TheoremQA and 7.5 on Minerva. We also evaluate **RLPR** on models from Llama3.1 (Grattafiori et al., 2024) and Gemma2 (Team et al., 2024), achieving improvements of 6.4 and 6.1 average points across seven benchmarks respectively.

The contribution of this work can be summarized as fourfold: (1) We present **RLPR**, a simple and scalable framework that extends RLVR to general domains without using external verifiers. (2) We propose a novel probability reward that eliminates the need for external verifiers and achieves better reward quality than naive likelihood as a reward. (3) We introduce a novel standard deviation filtering strategy that effectively stabilizes training by removing samples with low reward standard deviation. (4) We conduct comprehensive experiments to demonstrate the effectiveness of our framework on various models from Qwen, Llama and Gemma. All the codes, data, and model weights will be released to facilitate future research.

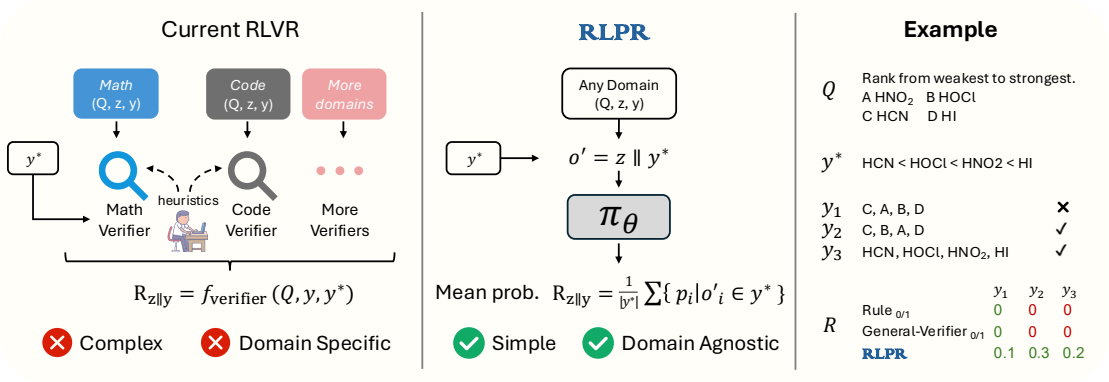


Figure 2: Existing RLVR methods rely on specialized verifiers for each domain, suffering from high complexity and limited scalability. We propose the **RLPR** framework, which replaces the complex verifier-based reward with a simple probability-based reward generated by the policy model π_θ. Q: input question, z: generated reasoning content before final answer, y: generated final answer, y^*: reference answer. As shown in the example on the right side, rules and verifier models wrongly label both y_2 and y_3 as incorrect due to their limited capability of handling natural language complexity.

2 RLPR

In this section, we first introduce the fundamentals of RLVR and describe the procedure to calculate the probability reward for **RLPR**. Then we introduce the debiasing method and the standard deviation filtering approach.

2.1 Reinforcement Learning from Verifiable Rewards

Reinforcement learning from verifiable reward (RLVR) is a general post-training paradigm in which a rule-based verifier assigns a scalar reward score to each generated response. Specifically, given a prompt x, the policy π_θ produces reasoning content z and the final answer y. Then the expected verifier score is optimized:

$$\mathcal{J}(\theta) = \mathbb{E}_{z,y\sim\pi_\theta(\cdot|x)}\left[f_{\text{verifier}}(y, y^*)\right], \qquad (1)$$

where f_{verifier} is a task-specific, rule-based verifier checking whether the generated answer y passes the test defined by ground truth y^*. Common instantiations include symbolic verifiers (Hynek and Greg, 2025) for mathematical problems or sandboxed execution (Bytedance-Seed-Foundation-Code-Team et al., 2025) for code generation. However, building rule-based verifiers is a laborious, systematic effort that involves designing handcrafted rules and edge case handling. This restricts the application of RLVR to new domains.

2.2 Probability Reward

Motivated by the observation that the LLM's intrinsic probability of generating a correct answer directly indicates its internal evaluation of the reasoning quality, we use per-token decoding probabilities of the reference answer as the reward signal. Unlike existing methods that rely on domain-specific verifiers (Cui et al., 2025a; Luo et al., 2025a), which require substantial manual heuristics and engineering effort for the construction of verifiers, our reward computation process involves only the model itself. An overview of the process is shown in Figure 2.

We denote each response to question Q with $o = (o_0, \cdots, o_N)$, where o_i is a token in the response. To obtain probabilities, we extract the generated answer y from the full response and denote the remaining content as reasoning z. We then construct a modified sequence $o' = (o'_0, \cdots, o'_{N'})$ by replacing the generated answer with the reference from the training data. This sequence is fed to the policy model to get probabilities $(p_0, \cdots, p_{N'})$. The probability reward is computed as:

$$r = f_{\text{seq}}(\{p_i | o'_i \in y^*\}), \qquad (2)$$

where f_{seq} aggregates per-token probabilities into a single reward scalar for the response o. While using $f_{\text{seq}} = \sqrt[n]{\prod}$ (the normalized product of probabilities, i.e., sequence likelihood) reflects the overall likelihood of the reference answer, we observe that it introduces high variance and is overly sensitive to minor variations, such as synonyms. For instance, the token probability sequences (0.01, 0.7, 0.9) and (0.05, 0.7, 0.9) yield vastly different scores under the product, despite only a small difference on the first token. To address this issue, we instead adopt $f_{\text{seq}} = \frac{1}{|y^*|}\sum$ (mean probabilities), which yields a more robust reward signal and demonstrates superior correlation with answer

quality in our analyses (see Fig 4). We observe that probability reward values are highly consistent with the quality of generated answer y, where high rewards are gained when the predicted answer is semantically similar to the reference answer and low rewards are assigned otherwise. Note that the length-normalization step is redundant for GRPO (Shao et al., 2024) but could be crucial for algorithms like REINFORCE++ (Hu et al., 2025a) which do not include group-normalization.

2.3 Reward Debiasing

Although the probability-based rewards correlate strongly with response quality, they are also influenced by various latent factors. We denote the contributors to the probability reward r as U_r, which decomposes into two latent factors:$U_r = U_z + U_{\text{others}}$, where U_z represents the effects of the reasoning content, and U_{others} captures the characteristics of other related factors, including the question and reference answer. Using r directly as a reward introduces bias associated with the unobserved factor U_{other}, potentially degrading the reward quality. To mitigate this, we introduce a base score r' by computing the probability score of directly decoding the reference answer y^*, without intermediate reasoning z, using Eq 2. This gives $U_z = U_r - U_{r'}$, and the debiased probability reward is then calculated as with $\hat{r} = \text{clip}(0, 1, r - r')$, where the clipping operation ensures that the reward remains within a favorable numeric range $[0, 1]$. This formulation effectively removes the potential bias from U_Q and U_{y^*} and models PR as the improvement in probability given the generated reasoning z. We find this debiasing step stabilizes training and enhances reward robustness. The final gradient estimator of our objective function is:

$$\begin{aligned}\nabla \mathcal{J}_{\text{RLPR}}(\theta) &= \nabla \mathbb{E}_{o \sim \pi_\theta(\cdot|x)} \left[\hat{r}\right] \\ &= \sum_o \hat{r}\ \pi_\theta(o|x) \nabla \log \pi_\theta(o|x) \\ &= \mathbb{E}_{o \sim \pi_\theta(\cdot|x)} \left[\hat{r} \nabla \log \pi_\theta(o|x)\right], \quad (3)\end{aligned}$$

where we optimize the expected reward on the whole response $o = z||y$.

2.4 Standard deviation filtering

Existing RLVR methods employ accuracy filtering (Cui et al., 2025a) to stabilize training by excluding too difficult and too easy prompts. Typically, this involves filtering entirely correct or incorrect prompts. However, the continuous nature of PR makes it challenging to directly apply accuracy filtering since it is hard to set a universal threshold for response correctness.

Through the analysis of accuracy filtering, we observe that filtering prompts with low standard deviation in reward values can effectively achieve a similar effect. Specifically, prompts that consistently yield all high or all low scores exhibit low standard deviation due to the boundedness of PR (i.e., all reward values lie within $[0, 1]$). Meanwhile, the overall standard deviation distribution continuously shifts during training, and a fixed threshold may cause either too strict or loose filtering at different training stages. To address this, we adopt an exponential moving average to dynamically update the filtering threshold β using the average standard deviation of each training step. By filtering the prompts whose reward standard deviation is less than β, we introduce an adaptive curriculum learning mechanism to improve both the training stability and final performance.

3 Experiments

In this section, we empirically investigate the effectiveness of **RLPR** in enhancing LLM reasoning capabilities. In addition to evaluating model performance, we also analyze reward quality of our proposed PR, the efficacy of different components, and the potential of applying **RLPR** to verifiable domains such as mathematics.

3.1 Experimental Setup

Models. We conduct experiments on Gemma2 (Team et al., 2024), Llama3.1 (Grattafiori et al., 2024) and Qwen2.5 (Team, 2024) models. Unless otherwise specified, experiments are conducted on Qwen2.5-7B-Base.

Training Data. We adopt the collection of prompts released by (Ma et al., 2025), which includes high-quality reasoning questions across multiple domains. To focus on the effectiveness of **RLPR** in general domains, we only use non-mathematics prompts from the data. We ask GPT-4.1 (OpenAI, 2025) to filter out prompts that are too easy and finally get 77k prompts for training.

Evaluation. We evaluate the reasoning capabilities with multiple general reasoning and mathematical benchmarks. For math reasoning, we include MATH-500 (Cobbe et al., 2021), Minerva (Lewkowycz et al., 2022), and AIME24. For general domains, we adopt four benchmarks: (1)

MMLU-Pro (Wang et al., 2024) is a widely used multitask language understanding benchmark. We randomly sample 1000 prompts from the benchmark to strike a balance between efficiency and variance. (2) GPQA (Rein et al., 2023) includes graduate-level questions from multiple disciplines. We use the highest-quality GPQA-diamond subset. (3) TheoremQA (Chen et al., 2023) assesses a model's ability to apply theorems to solve complex science problems. We remove the 53 multimodal instructions. (4) WebInstruct. We hold out a validation split from (Ma et al., 2025) as a more accessible benchmark for medium-sized models. Unlike the aforementioned benchmarks, this benchmark is designed to be less challenging while still assessing multidisciplinary reasoning. We randomly held-out 1k prompts from the training set and remove potential data contamination by applying 10-gram deduplication, resulting in 638 distinct questions.

Baselines. We compare against the following established and contemporaneous methods: (1) Base models. We include the Qwen2.5 (Team, 2024) model family for comparison, reporting results for both Qwen2.5-7B. We also compare with Gemma2-2B-it and Llama3.1-8B-Inst. (2) TTRL (Zuo et al., 2025) eliminates the reliance on labeled reference answers and instead uses majority voting to assign pseudo-labels to sampled responses. We report the result of the model trained on MATH-500 (Zuo et al., 2025) prompts. (3) SimpleRL-Zoo (Zeng et al., 2025) trains the model using rule-based rewards. (4) General Reasoner (Ma et al., 2025) conducts RLVR in multiple domains by introducing an additional verifier model, which is distilled from Gemini 2.0 (Google DeepMind, 2024) to verify general-domain responses. (5) VeriFree (Zhou et al., 2025) is a concurrent work that uses the likelihood of reference answers (for those shorter than 7-tokens) as the reward signal and incorporates an auxiliary fine-tuning loss. As results were only released for the Qwen3 (Team, 2025a) model series, we reproduce their method on Qwen2.5-7B using the official repository. For fair comparison, we evaluate both their provided prompt and our training prompt, finding that the original prompt yields better results. Therefore, we adopt this configuration for this baseline.

Implementation Details. We adopt the verl (Sheng et al., 2024) framework for efficient training. In each rollout step, we sample eight responses per prompt for a batch of 768 prompts using a temperature of 1, and subsequently perform 4 policy updates on the collected responses. The clip range is (0.8, 1.27) to prevent entropy collapse (Yu et al., 2025; Cui et al., 2025b), and filtering uses $\beta = 0.5$. For evaluation, we set temperature to 1 and report Avg@k over multiple runs to reduce the evaluation variance. The max generation length for training and evaluation is 3072, with minimal truncation observed. Baseline evaluations follow original papers or default to our setup if the original paper uses greedy decoding. For reliable answer extraction, we adopt the "<think></think><answer></answer>" template of R1 (Liu et al., 2025b) during training and use the striped content inside answer tags as the generated answer. For Gemma/Llama, we change the training and evaluation temperature to 0.6 and remove the <think> part in templates to avoid generation degradation. We observe that rule-based scoring scripts introduce errors in benchmarks containing question formats beyond multiple-choice. To address this, we deploy a Qwen2.5-7B-Inst model server for evaluation, and additionally leverage GPT-4.1 for more complex benchmarks, such as TheoremQA.

3.2 Main Results

The main experimental results are reported in Table 1, from which we observe that: (1) **RLPR** significantly improves general-domain reasoning performance. Without any external verifier, our method improves the average performance on four general-domain reasoning benchmarks by 24.9% on Qwen2.5-7B. (2) **RLPR** exceeds the RLVR baseline on Qwen, Llama and Gemma. Specifically, we achieve larger general reasoning performance improvement over RLVR for 1.4, 3.9 and 1.4 average points on Gemma, Llama and Qwen respectively. (3) **RLPR** exhibits even better performance compared with methods that require trained verifier models, surpassing General Reasoner, which uses a trained 1.5B-parameter verifier model to judge each sampled response, by 1.6 on average across all seven reasoning benchmarks. (4) **RLPR** achieves a significant performance advantage compared with concurrent verifier-free methods, with improvement of 7.6 points on TheoremQA and 7.5 points on Minerva over VeriFree (Zhou et al., 2025).

3.3 Probability-based Reward Analysis

We first illustrate a token-level probability example in Figure 3, where response sequence $o2$ receives a substantially lower score on the "HO" token, precisely reflecting the error made by response se-

Model	Base	Verifier	MMLU-Pro Avg@2	GPQA Avg@4	TheoremQA Avg@2	WebInst. Avg@2	MATH-500 Avg@2	Minerva Avg@2	AIME 24 Avg@16	General -	All -
Gemma Models											
Gemma2-2B-it	Base	–	27.9	19.3	16.4	33.5	26.6	15.9	0.0	24.3	19.9
RLVR	Inst	Rule	31.6	25.8	20.1	**52.3**	**30.7**	16.5	**0.2**	32.4	25.3
RLPR	Inst	✗	**33.5**	**28.5**	**21.2**	52.0	30.4	**17.1**	**0.2**	**33.8**	**26.0**
Llama Models											
Llama3.1-8B-Inst	Base	–	46.4	31.6	31.3	54.7	50.1	32.7	4.2	40.5	35.6
RLVR	Inst	Rule	49.3	36.0	32.0	60.2	51.9	35.2	4.6	44.4	38.5
RLPR	Inst	✗	**53.6**	**36.5**	**35.5**	**68.5**	**54.1**	**39.0**	**8.8**	**48.5**	**42.3**
Qwen Models											
Qwen2.5-7B	–	–	45.3	32.4	41.4	60.4	63.0	37.6	6.5	44.9	40.9
TTRL	Base	Rule	51.1	34.1	48.8	68.0	**82.1**	52.8	15.8	50.5	50.4
SimpleRL-Zoo	Base	Rule	54.1	36.2	49.5	70.7	76.3	49.2	14.8	52.6	50.1
RLVR	Base	Rule	55.1	36.2	52.2	75.3	76.5	54.9	17.7	54.7	52.6
General Reasoner	Base	Model	55.4	37.4	52.1	74.5	77.0	51.7	16.0	54.8	52.0
VeriFree	Base	✗	53.8	36.7	47.6	72.5	73.5	49.0	12.5	52.6	49.4
RLPR	Base	✗	**56.0**	37.6	**55.4**	**75.5**	78.0	**56.5**	16.3	**56.1**	**53.6**

Table 1: Overall performance of Gemma, Llama, Qwen series models on seven reasoning benchmarks. WebInst.: held-out evaluation set from WebInstruct. General: Average of MMLU-Pro, GPQA, TheoremQA and WebInst.

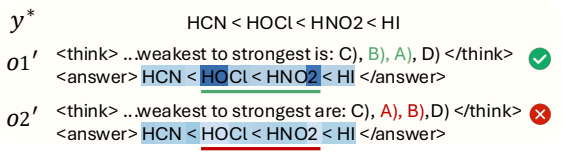


Figure 3: Token-level probability visualization, where deeper colors represent higher values. The underlined part highlights that probabilities precisely reflect that response sequence $o2$ incorrectly place option B after A, resulting lower scores at the corresponding positions in the reference answer. The question is shown in Figure 2.

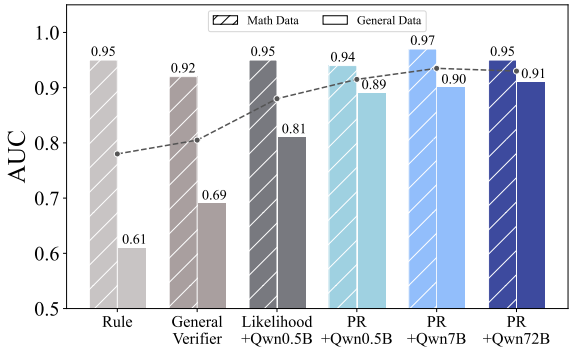


Figure 4: Reward quality comparison. We report the AUC on both math and general data, and highlight the average score with the dashed line. Qwn: Qwen2.5.

quence $o2$ (i.e., placing option A before option B). For quantitative analysis of the Probability-based Reward (PR) quality, we sample eight responses for each prompt from the WebInstruct (Ma et al., 2025) and DeepScale (Luo et al., 2025b) datasets. To ensure a fair evaluation, we use the publicly released model from (Hu et al., 2025b). Human annotators then evaluate the correctness of each response. To maintain robustness and control labeling costs, we randomly keep 50 prompts from each dataset that contain both correct and incorrect responses.

PR discriminates correct responses better than the rule-based verifier on general data. To evaluate the ability of different reward to distinguish between correct and incorrect responses (i.e., assign higher rewards to correct responses), we rank responses for each prompt according to the respective rewards and compute the ROC-AUC (Bradley, 1997) metric using human annotations as ground truth. Higher AUC values indicate stronger discrimination capability. As shown in Figure 4, while the rule-based verifier achieves reasonable performance on mathematical prompts, it struggles on general-domain prompts, achieving an AUC of only 0.61. The primary flaw of the rule-based verifier in general domains is that it overlooks correct responses due to its limited capability of processing natural language complexity. We show an example in Figure 2 to illustrate the phenomenon. In contrast, PR consistently delivers high-quality rewards across both mathematical and general domains.

PR outperforms verifier models across both mathematical and general domains. While the General-Verifier achieves improvement over rule-based reward on general data (0.61→0.69), its performance declines on mathematical prompts (0.95→0.92) as shown in Figure 4. We attribute this limitation to the finetuning-based paradigm, which requires extensive task-specific data and struggles to generalize across domains. In con-

Data	Verifier	TheoremQA Avg@2	Minerva Avg@2
DAPO	Rule	50.3	50.6
WebInstruct	Rule	52.2	54.9
	✗	**55.4**	**56.5**

Table 2: Effect of different RLVR training data and reward mechanisms.

Method	TheoremQA	Minerva
RLPR	**55.4**	**56.5**
w/o debiasing	52.7-2.7	54.1-2.4
w/o std-filtering	52.5-2.9	55.1-1.4
w/o token prob.	33.5-21.9	34.2-22.3

Table 3: Ablation experimental results. Token prob.: token probability average. Avg@2 results are reported.

trast, our proposed PR achieves improvements of at least 2% on mathematical data and 20% on general-domain data compared with the verifier model. Upon analyzing the General-Verifier's judgments, we find that its main errors stem from limited comprehension of complex responses and challenges in output parsing. By leveraging the intrinsic capabilities of LLMs, PR directly produces high-quality reward scores in a single forward pass, also eliminating the need for any text post-processing.

PR is effective with even small-scale models. We compare the quality of PR using models of varying sizes. As shown in Figure 4, even the smallest Qwen2.5-0.5B outperforms the specifically trained General-Verifier on both mathematical and general data. While increasing the model size further improves the performance on general-domain data, gains on mathematical data are marginal due to the already high absolute scores.

PR is robust over entropy and length distribution. We also analyze the robustness of PR by analyzing the correlation between PR values and factors, including length and decoding entropy of generated responses. For each prompt, we calculate the Spearman correlation coefficient and p-value. We observe that only 8% prompts get a p-value smaller than 0.05, and the average coefficient is -0.060 for length and 0.059 for entropy. These results indicate that the probability reward values show negligible correlation with both entropy and length. This indicates that our proposed reward serves as a robust reward mechanism.

PR is essential for utilizing general-domain data. We compare the performance of models trained exclusively on mathematical prompts (Yu et al., 2025) versus those trained on general-domain prompts, as shown in Table 2. The results demonstrate that general-domain data enhances the performance on both benchmarks (+1.9 on TheoremQA, +4.3 on Minerva). However, general-domain data also includes additional challenges for rule-based verifiers. Consequently, directly adopting existing rule-based verifiers gives obvious diminished performance.

3.4 Ablation Study

To investigate the contribution of different design choices in **RLPR**, we perform an ablation study.

Effect of per-token probability as reward. We compare our per-token probability-based reward with naive sequence likelihood as the reward signal. In the calculation of likelihood, low-probability tokens can dramatically affect the final reward. For instance, probabilities of $1e^{-4}$ versus $1e^{-5}$ can lead to a tenfold difference in reward, despite their small absolute difference. This issue becomes more pronounced for longer answers, which are more likely to contain at least one low-probability token. (Zhou et al., 2025) addresses this instability by filtering out prompts whose reference answers exceed seven tokens. However, this also significantly limits the data diversity. In contrast, using the mean per-token probability is much more robust and yields better performance, as shown in Table 3. We compare the quality of the likelihood reward and our proposed PR in Figure 4, where PR consistently achieves better results on both domains.

Effect of reward debiasing and standard deviation filtering. We compare our final debiased reward $\hat{r}$ with directly using the reward in Eq 2. Results in Table 3 shows that the performance on both benchmarks is worse with original reward, demonstrating the effectiveness of the debiasing operation. To quantify the effectiveness of the standard deviation filtering approach, we also train a model without any filtering mechanism. The results in Table 3 show that the filtering strategy is important for the final performance of models by removing prompts that do not get diverse responses.

3.5 Robustness Analysis

Compared with rule-based rewards, the distribution of our proposed probability-based reward (PR) may be influenced by variations in training prompt templates. To evaluate the robustness of **RLPR** with different templates, we consider three prompt settings: p_1 from VeriFree (Zhou et al., 2025), p_2 used in DeepSeek-R1 (DeepSeek-AI et al., 2025) and

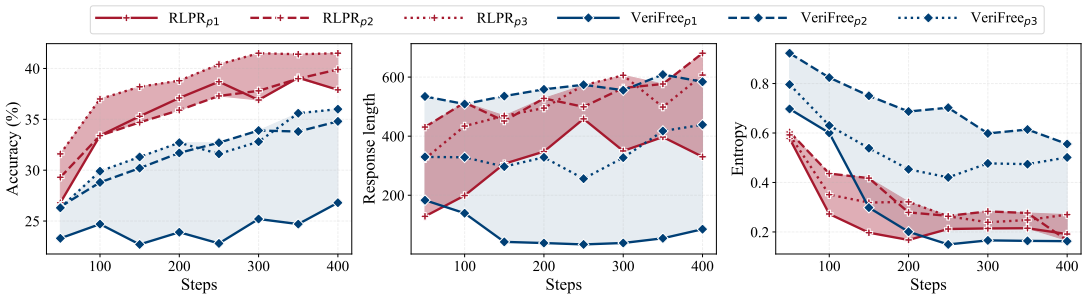


Figure 5: Robustness across different training prompt templates. **RLPR** yields consistently higher performance compared with VeriFree. Left: average performance on seven benchmarks. Middle: response length. Right: response entropy during training.

p_3 which moves the format requirement to user prompt. To reduce training costs, we switch the base model to Qwen2.5-3B, decrease the batch size to 128, and apply a single update per training step. For fair comparison, we adopt the origin dataset from VeriFree for this experiment. Figure 5 presents the comparison of performances, response length, and entropy across different training steps. We observe that **RLPR** maintains consistent performance regardless of prompt choice, while VeriFree exhibits high sensitivity, with a notable performance drop of by 8.0 at step-400 when using p_1. Furthermore, the response length of **RLPR** under all prompts converges to a similar level, and the entropy remains within a reasonable range with no signs of entropy collapse (Cui et al., 2025b).

4 Related Works

We introduce the most related background works in this section and refer readers to the Appendix for a more detail review of related works.

Reinforcement Learning with Verifiable Rewards. Reinforcement learning from binary verifiable rewards (Cui et al., 2025a; Yu et al., 2025; Luo et al., 2025c; Team, 2025b; DeepSeek-AI et al., 2025) recently demonstrates strong reasoning capabilities on math and code tasks, and has emerged as a common practice. However, this paradigm is restricted to domains where robust verifiers are available. In this work, we propose to extend RLVR practices to domains without robust verifiers.

Reasoning in General Domains. One line of work is generative reward models (Mahan et al., 2024), where a generative model judges the rollouts, which was further extended to the verifiers based on a generative model (Ma et al., 2025; Liu et al., 2025a). In this work, we demonstrate that reinforcement learning for general-domain reasoning can rely on the decoding probability of the reference answer as a reward signal. Concurrent to our work, (Zhou et al., 2025) utilizes policy likelihood for reference answer as rewards, while limited to short answers less than 7 tokens and requires an auxiliary fine-tuning-based objective. Instead, we observe the robustness of per-token probability as a reward signal and extend RLVR to general domains without length constraints.

Self-Reward Optimization. The common idea behind the practice of self-reward is raising the probability of consistent answers (Zuo et al., 2025). Recent literature (Agarwal et al., 2025; Li et al., 2025) shows that entropy minimization is a sugar for reasoning tasks, though restricted to certain model families. However, such practice might be problematic for restricting exploration (Cui et al., 2025b; Hochlehnert et al., 2025). In contrast to self-rewarding methods that remove diversity to exploit existing reasoning ability, our approach builds the reward based on the reference answer, yielding reasoning performance with healthy token entropy from the clip-high trick (Yu et al., 2025).

5 Conclusion

In this work, we present **RLPR** , a novel framework that extends this paradigm to broader general domains. Comprehensive experimental results on Gemma, Llama and Qwen show that our method achieves significant improvement on both general and mathematical reasoning tasks without using external verifiers. We propose a novel probability reward (PR) and reward debiasing strategy to enhance its quality further. By replacing rule-based reward with PR, we eliminate the need for external verifiers and achieve better performance than using naive likelihood as a reward or using verifier models. Moreover, we propose a simple standard deviation filtering strategy that stabilizes training by removing samples with low reward standard deviation. In the future, we will extend **RLPR** to multimodal domain and larger models.

6 Limitations

Though RLPR extrapolate RLVR to generate domains, it still requires reference answers to provide supervision. Further investigation is required for scenarios where reference answers are hard to acquire, such as scientific research problems.

References

Shivam Agarwal, Zimin Zhang, Lifan Yuan, Jiawei Han, and Hao Peng. 2025. The unreasonable effectiveness of entropy minimization in llm reasoning. *arXiv preprint arXiv:2505.15134*.

Andrew P. Bradley. 1997. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7):1145–1159.

Bytedance-Seed-Foundation-Code-Team, Yao Cheng, Jianfeng Chen, Jie Chen, Li Chen, Liyu Chen, Wentao Chen, Zhengyu Chen, Shijie Geng, Aoyan Li, Bo Li, Bowen Li, Linyi Li, Boyi Liu, Jiaheng Liu, Kaibo Liu, Qi Liu, Shukai Liu, Siyao Liu, and 36 others. 2025. Fullstack bench: Evaluating llms as full stack coders. *Preprint*, arXiv:2412.00535.

Wenhu Chen, Ming Yin, Max Ku, Pan Lu, Yixin Wan, Xueguang Ma, Jianyu Xu, Xinyi Wang, and Tony Xia. 2023. Theoremqa: A theorem-driven question answering dataset. *Preprint*, arXiv:2305.12524.

Xiusi Chen, Gaotang Li, Ziqi Wang, Bowen Jin, Cheng Qian, Yu Wang, Hongru Wang, Yu Zhang, Denghui Zhang, Tong Zhang, and 1 others. 2025. Rm-r1: Reward modeling as reasoning. *arXiv preprint arXiv:2505.02387*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, Qixin Xu, Weize Chen, and 1 others. 2025a. Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*.

Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, and 1 others. 2025b. The entropy mechanism of reinforcement learning for reasoning language models. *arXiv preprint arXiv:2505.22617*.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, and 181 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Google DeepMind. 2024. Gemini 2.0: Our latest, most capable ai model yet. First Gemini 2.0 Flash announced December 11, 2024; multimodal support for text, image, audio, native tool use.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Jujie He, Jiacai Liu, Chris Yuhao Liu, Rui Yan, Chaojie Wang, Peng Cheng, Xiaoyu Zhang, Fuxiang Zhang, Jiacheng Xu, Wei Shen, Siyuan Li, Liang Zeng, Tianwen Wei, Cheng Cheng, Bo An, Yang Liu, and Yahui Zhou. 2025. Skywork open reasoner series. Notion Blog.

Andreas Hochlehnert, Hardik Bhatnagar, Vishaal Udandarao, Samuel Albanie, Ameya Prabhu, and Matthias Bethge. 2025. A sober look at progress in language model reasoning: Pitfalls and paths to reproducibility. *arXiv preprint arXiv:2504.07086*.

Jian Hu, Jason Klein Liu, and Wei Shen. 2025a. Reinforce++: An efficient rlhf algorithm with robustness to both prompt and reward models. *Preprint*, arXiv:2501.03262.

Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. 2025b. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *Preprint*, arXiv:2503.24290.

Kydlíček Hynek and Gandenberger Greg. 2025. Math-verify.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, Alex Iftimie, Alex Karpenko, Alex Tachard Passos, Alexander Neitz, Alexander Prokofiev, Alexander Wei, Allison Tam, Ally Bennett, Ananya Kumar, and 242 others. 2024. Openai o1 system card. *Preprint*, arXiv:2412.16720.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. 2022. Solving quantitative reasoning problems with language models. *Preprint*, arXiv:2206.14858.

Pengyi Li, Matvey Skripkin, Alexander Zubrey, Andrey Kuznetsov, and Ivan Oseledets. 2025. Confidence is all you need: Few-shot rl fine-tuning of language models. *Preprint*, arXiv:2506.06395.

Qianchu Liu, Sheng Zhang, Guanghui Qin, Timothy Ossowski, Yu Gu, Ying Jin, Sid Kiblawi, Sam Preston, Mu Wei, Paul Vozila, and 1 others. 2025a. X-reasoner: Towards generalizable reasoning across modalities and domains. *arXiv preprint arXiv:2505.03981*.

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

Michael Luo, Sijun Tan, Roy Huang, Ameen Patel, Alpay Ariyak, Qingyang Wu, Xiaoxiang Shi, Rachel Xin, Colin Cai, Maurice Weber, Ce Zhang, Li Erran Li, Raluca Ada Popa, and Ion Stoica. 2025a. Deepcoder: A fully open-source 14b coder at o3-mini level. Notion Blog.

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. 2025b. Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl. Notion Blog.

Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Tianjun Zhang, Li Erran Li, and 1 others. 2025c. Deepscaler: Surpassing o1-preview with a 1.5 b model by scaling rl. *Notion Blog*.

Xueguang Ma, Qian Liu, Dongfu Jiang, Ge Zhang, Zejun Ma, and Wenhu Chen. 2025. General-reasoner: Advancing llm reasoning across all domains. *Preprint*, arXiv:2505.14652.

Dakota Mahan, Duy Van Phung, Rafael Rafailov, Chase Blagden, Nathan Lile, Louis Castricato, Jan-Philipp Fränken, Chelsea Finn, and Alon Albalak. 2024. Generative reward models. *arXiv preprint arXiv:2410.12832*.

OpenAI. 2025. Introducing gpt-4.1 in the api. Accessed: 2025-06-03.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2023. Gpqa: A graduate-level google-proof q&a benchmark. *Preprint*, arXiv:2311.12022.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *Preprint*, arXiv:2402.03300.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2024. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, Pouya Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, and 179 others. 2024. Gemma 2: Improving open language models at a practical size. *Preprint*, arXiv:2408.00118.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Qwen Team. 2025a. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Qwen Team. 2025b. Qwq-32b: Embracing the power of reinforcement learning.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. 2024. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Preprint*, arXiv:2406.01574.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, and 1 others. 2025. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*.

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

Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. 2025. Simplerl-zoo: Investigating and taming zero reinforcement learning for open base models in the wild. *Preprint*, arXiv:2503.18892.

Xuandong Zhao, Zhewei Kang, Aosong Feng, Sergey Levine, and Dawn Song. 2025. Learning to reason without external rewards. *arXiv preprint arXiv:2505.19590*.

Xiangxin Zhou, Zichen Liu, Anya Sims, Haonan Wang, Tianyu Pang, Chongxuan Li, Liang Wang, Min Lin, and Chao Du. 2025. Reinforcing general reasoning without verifiers. *arXiv preprint arXiv:2505.21493*.

Yuxin Zuo, Kaiyan Zhang, Li Sheng, Shang Qu, Ganqu Cui, Xuekai Zhu, Haozhan Li, Yuchen Zhang, Xinwei Long, Ermo Hua, Biqing Qi, Youbang Sun, Zhiyuan Ma, Lifan Yuan, Ning Ding, and Bowen Zhou. 2025. Ttrl: Test-time reinforcement learning. *Preprint*, arXiv:2504.16084.

A Appendix

A.1 Experimental Details

Our experiments are conducted on Qwen2.5-7B (Team, 2024) if not additionally specified. Following most RLVR practices, we forgo the supervised fine-tuning process and directly post-train on the base model, and use GRPO algorithm by default. We change the prompt template during training and validation time in our main experiments to control the response structure to have extractable thoughts and answers. The prompt template is shown in Table 4.]

A.1.1 Parameter Settings

Each experiment is trained on 32 NVIDIA A100 GPUS. We use a 1e-3 entropy penalty coefficient and no KL penalty. The learning rate for the policy model is 5e-7.

A.1.2 Training Logs

We monitor key training metrics of our methods in Fig. 6. During training, the response length (Fig. 6a) steadily increases, allowing more profound reasoning behaviors and no sign of degeneration. In Fig. 6b, the policy model quickly learns to follow the response structure. Moreover, as shown in Fig. 6c, our training entropy exhibits neither collapses as a result of the clip-high trick, nor abrupt increases. This ensures the balance between exploration and exploitation.

A.2 Pass@k Evaluation

To further test the impact on the potential reasoning boundary of our method, we provide pass@k results on various tasks in Fig. 7 following (Yue et al., 2025). Compared to standard RLVR and General Reasoner, **RLPR** shows comparable or better pass@k accuracy, indicating that our method is not trading reasoning potential for pass@1 improvements.

A.3 Training Data

We adopt WebInstruct (Ma et al., 2025) as our training dataset, excluding math-related prompts to focus on general-domain reasoning. To ensure the quality and difficulty of training samples, we apply a multi-stage filtering strategy: First, we remove history-related questions and those targeting elementary or middle school levels to avoid commonsense or overly simple content. Finally, leveraging GPT-4.1-mini's reasoning scores (1–4, see Table 5), we retain only highly challenging samples (score ≥ 3). This process reduces the dataset from 231,833 to 77,687 samples, yielding a focused and high-quality corpus for complex non-mathematical reasoning.

A.4 Implementation Details

This section provides additional implementation details to supplement Section 3.1. The policy model generates 8 responses per question, using a learning rate of 1e-6. We remove the KL divergence term by setting the KL coefficient to 0. Detailed configurations are presented in Table 6, where the number of policy updates per step and the value of β are empirically determined to be optimal for their respective scenarios.

RLVR baselines are trained under the same setting with corresponding **RLPR** results, except using rule-based verifiers and accuracy filtering. For RLVR training on Llama and Gemma, we find accuracy filtering can remove over 90% training prompts and thus significantly increase the training cost and find small batch size causes entropy blow up. So we do not apply accuracy filtering for these two experiments and conduct only one update for each batch to stabilize training.

A.5 Detailed Related Works

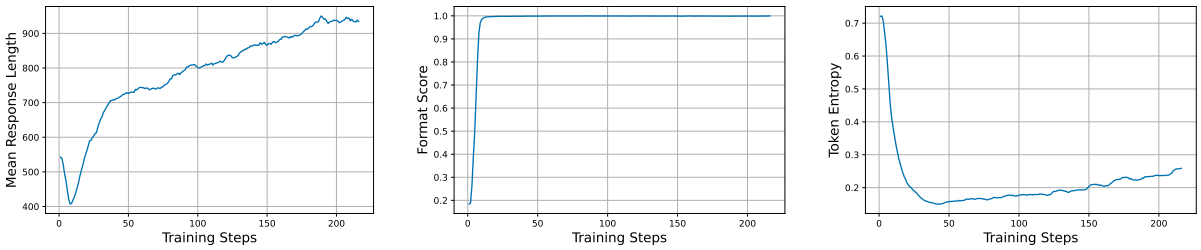
Reinforcement Learning with Verifiable Rewards. Reinforcement learning from binary verifiable rewards (Cui et al., 2025a; Yu et al., 2025; Luo et al., 2025c; Team, 2025b; DeepSeek-AI et al., 2025) recently demonstrates strong reasoning capabilities on math and code tasks, and has emerged as a common practice. These practices utilize verifiers such as Math-Verify (Hynek and Greg, 2025), SandboxFusion (Bytedance-Seed-Foundation-Code-Team et al., 2025), and custom implemented ones (Cui et al., 2025a), which effectively judge the correctness of model rollouts and forgo the need for preference annotations. However, this paradigm is restricted to domains where robust verifiers are available. Moreover, existing implementations of verifiers show inconsistencies (He et al., 2025) since the complexity for rule-based verifiers to handle edge cases is nontrivial. In this work, we propose to extend RLVR practices to domains without robust verifiers.

Reasoning in General Domains. Previous research explores reasoning in general domains, a vital part of which is how to obtain reliable reward signals. One line of work is generative re-

RLPR training prompt

```
<|im_start|>system
A conversation between User and Assistant. The user asks a question, and the Assistant solves it.
The assistant first thinks about the reasoning process in the mind and then provides the user
with the answer. The reasoning process and answer are enclosed within <think> </think> and
<answer> </answer> tags, respectively, i.e., <think> reasoning process here </think> <answer>
answer here </answer>.
<|im_end|>
<|im_start|>user
{{question}}<|im_end|>
<|im_start|>assistant
```

Table 4: We adopt the training prompt of R1 (DeepSeek-AI et al., 2025) for **RLPR**.



(a) Mean response length (b) Format reward (c) Token entropy

Figure 6: Training dynamics of **RLPR** on Qwen2.5-7B

ward models (Mahan et al., 2024), where a generative model judges the rollouts. This concept has been extended to the implementation of verifiers based on a generative model (Ma et al., 2025; Liu et al., 2025a) and enhancements of the judge model itself as a reasoner (Chen et al., 2025). In this work, we demonstrate that reinforcement learning for general-domain reasoning can rely on the decoding probability of the reference answer as a reward signal. Concurrent to our work, (Zhou et al., 2025) utilizes policy likelihood for reference answer as rewards, while limited to short answers less than 7 tokens and requires an auxiliary fine-tuning-based objective. Instead, we observe the robustness of per-token probability as a reward signal and extend RLVR to general domains without length constraints.

Self-Reward Optimization. Unsupervised reinforcement learning on language models using the policy model itself as a reward has recently emerged as an embarrassingly effective approach (Zuo et al., 2025; Zhao et al., 2025). The common idea behind the practice of self-reward is raising the probability of consistent answers (Zuo et al., 2025), intuitively from the observation that concentrating on the majority brings free improvements (Wang et al., 2022). Recent literature (Agarwal et al., 2025; Li et al., 2025) shows that entropy minimization, which naively degrades generation diversity, is a sugar for reasoning tasks, though restricted to certain model families. However, such practice might be problematic for restricting exploration (Cui et al., 2025b; Hochlehnert et al., 2025; Yu et al., 2025). In contrast to self-rewarding methods that remove diversity to exploit existing reasoning ability, our approach builds the reward based on the reference answer, yielding reasoning performance with healthy token entropy from the clip-high trick (Yu et al., 2025).

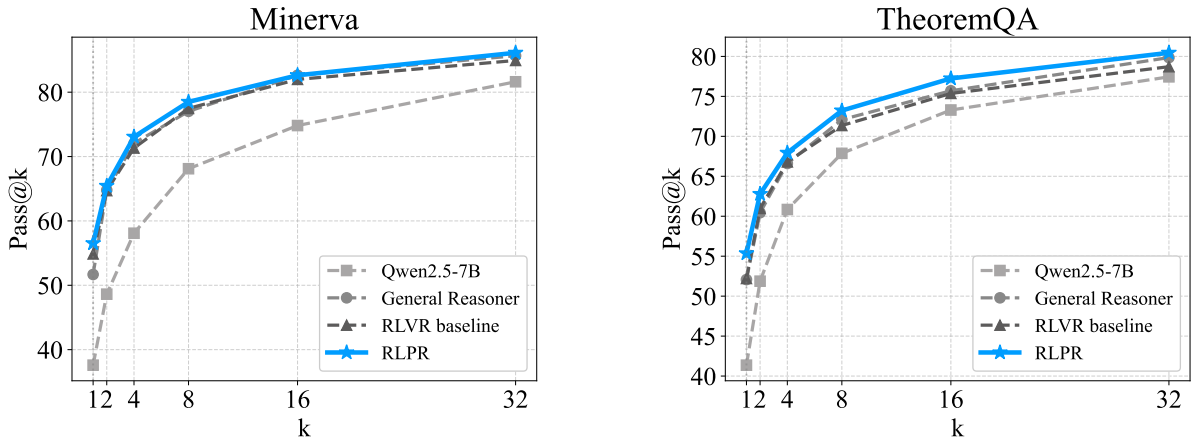


Figure 7: Pass@k curves for **RLPR** and baselines.

Prompt for GPT-4.1 to assess reasoning complexity

Description

You are asked to evaluate the reasoning level requirement of problems. Problems are scored from 1 to 4, with higher scores indicating greater reasoning demands. You should make your decision based on the following detail instructions.

1 Point: No reasoning requirements.

Problems requiring direct recall of specific facts and commonsense knowledge.
Examples:
- What is Fermat's Last Theorem. (Requires only recalling facts)
- What are the five quantitative forecasting models? (Requires only recalling facts)
- What is the capital of China? (Requires only recalling commonsense knowledge)
- When was Mark Twain born? (Requires only recalling commonsense knowledge)

2 Points: No reasoning skill requirements.

Problems that do not require reasoning skills. Either because (1) it is too simply and reasoning skills don't help, or (2) it is too hard to clearly rank different answers since the question is too open-ended and reasoning skills also do not help.

Examples:
- Solve x + 1 = 10, what is the value of x. (Too simple)
- What would you do if you have four legs. (Too open-ended to determine response quality)

3 Points: Moderate level reasoning skills and knowledge are enough.

Problems requiring moderate level reasoning skills and knowledge. Such as problems that are mostly likely to be solved by any random undergraduate student regardless of their majors.

Examples:
- Solve a quadratic equation: x^2 - 5x + 6 = 0.
- Find all solutions to $[\sqrt{x} + 2\sqrt{x^2 + 7x} + \sqrt{x + 7} = 35 - 2x]$. Enter all the solutions, separated by commas.
- Summarize the main causes of World War I. (Requires recalling and organizing established historical factors).
- Describe the importance of empathy in storytelling to someone unfamiliar with the concept, using no more than 4 sentences, and ensure all text is in lowercase. Include a quote from a famous author at the end.

4 Points: Long-time analysis and deep understanding of relevant knowledge are required.

Problems requiring long-time to analyze and solve, and depend on deep understanding of relevant knowledge. Such as designing a complex system, developing a comprehensive strategy or providing detail and easy-to-understand solution for realworld problems.

Examples:
- Design a scalable and secure REST API for a large e-commerce platform, considering microservices architecture, data consistency, fault tolerance, and evolving business needs.
- Develop a comprehensive urban planning strategy for sustainable development in a rapidly growing city, integrating environmental, social, economic, and infrastructural considerations.
- Conduct a thorough root cause analysis for a major systemic failure (e.g., a financial crisis or a large-scale environmental disaster) and propose multi-level preventative and corrective policy measures.
- The polynomial $P(x) = (1 + x + x^2 + \ldots + x^{17})^2 - x^{17}$ has 34 complex zeros of the form $z_k = r_k \left[\cos(2\pi\alpha_k) + i \sin(2\pi\alpha_k)\right]$, $k = 1, 2, 3, \ldots, 34$, with $0 < \alpha_1 \leq \alpha_2 \leq \alpha_3 \leq \cdots \leq \alpha_{34} < 1$ and $r_k > 0$. Find $\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 + \alpha_5$.

Please score the following question: Q: `{question}`
You should first explain your reasoning briefly, then give the final score in following format:
Reasoning score: [1-4]

Table 5: Prompt for GPT-4.1 to assess reasoning complexity.

Experiment name	Table / Figure	Batch Size	Update per Step	Clip Threshold	β	Temperature
Main experiment	Figures 1, 6, 7	768	4	(0.8, 1.27)	0.5	1.0
	Qwen in Table 1	768	4	(0.8, 1.27)	0.5	1.0
	Llama in Table 1	256	4	(0.8, 1.27)	0.9	0.6
	Gemma in Table 1	256	4	(0.8, 1.27)	1.0	0.6
RLPR vs. RLVR	Table 2	768	4	(0.8, 1.27)	0.5	1.0
Ablation study	Table 3	768	4	(0.8, 1.27)	0.5	1.0
Robustness analysis	Figure 5	128	1	(0.8, 1.27)	0.5	1.0

Table 6: Implementation setup for each experiment. Default settings align with Sections 3.1 and A.4.