PRIME: A Process-Outcome Alignment Benchmark for Verifiable Reasoning in Mathematics and Engineering

Xiangfeng Wang[*1,2], **Hangyu Guo**[*2], **Yanlin Lai**[*2,3], **Mitt Huang**[*2], **Liang Zhao**[2], **Chengyuan Yao**[2], **Yinmin Zhang**[2], **Qi Han**[2], **Xiaoxiao Ren**[2], **Chun Yuan**[†3], **Tong Xu**[†1]
Zheng Ge[2], **Xiangyu Zhang**[2], **Daxin Jiang**[2]
[1]University of Science and Technology of China [2]Stepfun [3]Tsinghua University

Github: https://github.com/wonderful9462/PRIME

Huggingface: https://huggingface.co/collections/wonderful9462/prime

Abstract

While model-based verifiers are essential for scaling Reinforcement Learning with Verifiable Rewards (RLVR), current outcome-centric verification paradigms primarily focus on the consistency between the final result and the ground truth, often neglecting potential errors in the derivation process. This leads to assigning positive rewards to correct answers produced from incorrect derivations. To bridge this gap, we introduce **PRIME**, a benchmark for evaluating verifiers on **PR**ocess-outcome alignment verification **I**n **M**athematics and **E**ngineering. Curated from a comprehensive collection of college-level STEM problems, PRIME comprises 2,530 high-difficulty samples through a consistency-based filtering pipeline. Through extensive evaluation, we find that current verifiers frequently fail to detect derivation flaws. Furthermore, we propose a process-aware RLVR training paradigm utilizing verifiers selected via PRIME. This approach substantially outperforms the outcome-only verification baseline, achieving absolute performance gains of **8.29%**, **9.12%**, and **7.31%** on AIME24, AIME25, and Beyond-AIME, respectively, for the Qwen3-14B-Base model. Finally, we demonstrate a strong linear correlation ($R^2 > 0.92$) between verifier accuracy on PRIME and RLVR training effectiveness, validating PRIME as a reliable predictor for verifier selection. We release the benchmark and code at https://anonymous.4open.science/r/7E7D/.

1 Introduction

Reinforcement learning with verifiable reward (RLVR) has proven highly effective for Large Reasoning Models (LRMs) in STEM domains (Shao et al., 2024; Guo et al., 2025). Given that rule-based verification frameworks, exemplified by Math-Verify[1], often struggle with flexible output formats (Wang et al., 2024a), the field has transitioned toward model-based verifiers to provide more robust reward signals (Liu et al., 2025a; Team et al., 2025; Liu et al., 2025b). However, it has become increasingly evident that the efficacy of RLVR is strictly bounded by the verifier's own capability, as variability in verification precision directly dictates the quality of the training outcome (Mahan et al., 2024; Zhang et al., 2025). In response to this dependency, recent research has concentrated on rigorously benchmarking and optimizing verifier performance. Benchmarks such as VerifierBench (Liu et al., 2025b) and VAR (Chen et al., 2025) have been established to evaluate judgment reliability, guiding the development of specialized methods (Zheng et al., 2025; Feng et al., 2025) aimed at empowering precise verification across

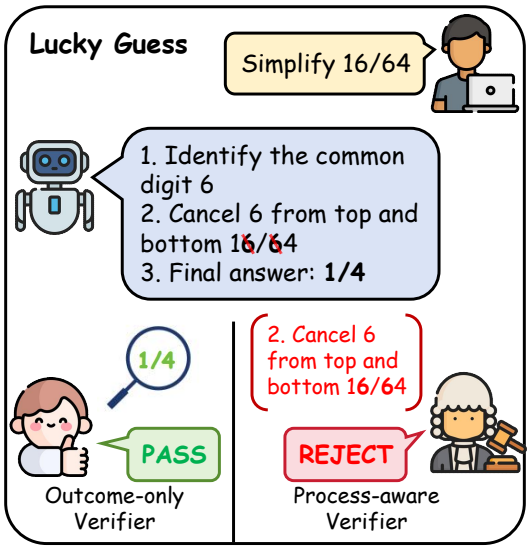


Figure 1: **An illustration of a "lucky guess"** where the model arrives at the correct answer via an incorrect derivation. While an outcome-only verifier erroneously accepts this false positive due to the correct final answer, a process-aware verifier detects the logical inconsistency and correctly penalizes the invalid derivation.

[*]Equal contribution.
[†]Corresponding author.

[1]https://github.com/huggingface/Math-Verify

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 14973–14985

complex STEM tasks.

However, current verification benchmarks primarily focus on outcome correctness while neglecting the derivation process. As illustrated in Figure 1, this outcome-only approach erroneously validates "lucky guesses," where the model arrives at the correct answer through incorrect derivation. To quantify this, we collect 2,500 STEM prompts and generate multiple responses using diverse open-source and closed-source models. Through rigorous human annotation, we identify that 16.98% of the responses exhibit such "lucky guesses," where the outcome is correct, but the derivation is flawed. Given this high frequency, verifiers guided by existing outcome-level verification benchmarks provide inaccurate rewards that reinforce flawed derivation, potentially limiting the performance of RLVR training.

To bridge this gap, we introduce **PRIME**, a benchmark for **PR**ocess–outcome alignment verification **I**n **M**athematics and **E**ngineering domains. Unlike evaluations limited solely to the outcome level, PRIME assesses verifiers' capability to detect both outcome accuracy and process-outcome logical consistency. To ensure the benchmark's reliability, we first collect a large pool of college-level STEM questions and apply model-based filtering to retain only verifiable problems with reliable reference answers. Second, we generate multiple solution trajectories for each retained problem using a diverse set of open source and commercial LRMs to cover a broad range of reasoning behaviors. Third, we perform difficulty-oriented selection to prioritize cases that are hard to verify. Finally, to guarantee the reliability of the dataset, we employ 18 domain experts to rigorously annotate each sample for both process-outcome consistency and outcome-ground truth alignment. The resulting benchmark comprises approximately 2,530 annotated instances, spanning 16 major STEM categories with fine-grained coverage across 480 subdisciplines.

In summary, our main contributions are threefold:

- We introduce **PRIME**, a high-quality benchmark specifically designed to evaluate process-outcome alignment in Mathematics and Engineering. Comprising 2,530 expert-annotated samples, it effectively detects spurious correctness and fills the gap in existing benchmarks that neglect derivation correctness.
- We propose a process-aware RLVR training paradigm where employing a process-aware verifier grants rewards only when the LRM's final answer matches the ground truth and the derivation maintains logical consistency with its outcome. Compared to the outcome-only verifier, our approach achieves substantial performance gains, with absolute improvements of **8.29%**, **9.12%**, and **7.31%** on AIME24, AIME25, and Beyond-AIME, respectively, on Qwen3-14B-Base model.
- We demonstrate a strong linear correlation (with $R^2 > 0.92$) between a verifier's accuracy on PRIME and the resulting performance improvement in RLVR training. This validates PRIME as a reliable predictor for assessing and selecting effective verifiers for LRMs training.

2 Related Work

2.1 Large Reasoning Models

Recent advancements in Large Reasoning Models (LRMs) have been significantly propelled by Reinforcement Learning with Verifiable Rewards (RLVR), which incentivizes structured, multi-step problem-solving capabilities. Currently, mainstream RLVR paradigms primarily focus on outcome-centric verification, where the reward signal is determined solely by the consistency between the final result and the ground truth. Within this framework, verification methods are generally categorized into rule-based and model-based approaches. Rule-based approaches predominantly rely on deterministic systems, such as compilers for code or symbolic solvers for mathematics, to generate rigorous reward signals. Pioneering models like DeepSeek-R1 (Guo et al., 2025) and Qwen2.5-Math (Yang et al., 2024) exemplify this direction, utilizing strict answer matching to drive large-scale reinforcement learning. To address the limitations of rigid string matching in flexible scenarios, model-based approaches employ learned verifiers to assess the semantic equivalence between the model's final answer and the ground truth (Seed et al., 2025; Zhang et al., 2024). However, regardless of whether they are rule-based or model-based, these outcome-centric verifiers share a critical vulnerability: they are unable to detect inconsistencies in the intermediate reasoning process. This limitation prevents the effective suppression of the "lucky

guess" phenomenon, where models receive positive reinforcement for correct answers derived from flawed logic.

2.2 Verifier and Verifier Evaluation

Recent advancements in model-based verification can be broadly categorized into outcome-centric and process-centric approaches. Outcome-centric methods, such as CompassVerifier (Liu et al., 2025b) and xVerify (Chen et al., 2025), focus on assessing the semantic equivalence between generated answers and ground truth, with extensions like SCI-Verifier (Zheng et al., 2025) further handling symbolic domains. However, by prioritizing the final result, these approaches often overlook the underlying derivation, failing to detect spurious correctness arising from "lucky guesses." In contrast, process-centric approaches, known as Process Reward Models (PRMs) (Uesato et al., 2022; Lightman et al., 2023), provide step-aware supervision to mitigate reasoning fallacies, with techniques like Math-Shepherd (Wang et al., 2024a) automating verification via Monte Carlo rollouts. Despite the evolution of these verification paradigms, the evaluation of verifiers remains underdeveloped. Current benchmarks primarily test answer extraction capabilities rather than the ability to detect logical inconsistencies between the derivation and the final result. PRIME addresses this limitation by introducing a specialized benchmark for Process-Outcome Alignment, which rigorously evaluates whether a verifier grants rewards only when the final answer matches the ground truth and the derivation maintains logical consistency with its outcome.

3 Benchmark Construction

To systematically evaluate the capability of verifiers to judge the process-outcome consistency of LRMs, we introduce PRIME. The benchmark is constructed through a multi-stage pipeline that integrates automated processing with human-in-the-loop verification. As illustrated in Figure 2, the construction encompasses five key stages, and we detail each one in the subsequent sections.

3.1 STEM Data Collection and Filtering

To comprehensively evaluate the verification capability of large reasoning models across complex scientific domains, we construct an initial pool $\mathcal{Q}_{\text{raw}}$ containing over 7M problems collected primarily from undergraduate and graduate-level textbooks, as well as university final exams. These problems span Mathematics, Physics, Chemistry, and Engineering, requiring deep domain knowledge and complex multi-step reasoning. Raw data from educational resources often contains open-ended inquiries or errors unsuitable for objective verification. We further apply a two-stage model filtering pipeline to sanitize $\mathcal{Q}_{\text{raw}}$:

1. **Verifiability Check:** We prompt GPT-OSS-120B to directly assess the problem statement for verifiability. This step filters out ambiguous or open-ended questions lacking a unique ground truth.

2. **Correctness Validation:** To maximally eliminate errors in textbook answers, we employ top-performing models, including Gemini-3-Pro, GPT-5, and Claude-Sonnet-4.5, to cross-validate the ground truth a_{gt}. We filter out instances where none of the models yield a result matching a_{gt}, retaining only those validated by at least one model.

After filtering, we organize valid problems into a hierarchical structure encompassing 16 fine-grained sub-domains, such as Topology, Organic Polymer, and Systems Robotics. To prevent data imbalance caused by the uneven distribution of source materials, we apply a downsampling strategy to over-represented categories. We denote this final filtered dataset as $\mathcal{Q}_{\text{clean}} = \{(q_i, a_i)\}$, where q_i represents the question statement and a_i denotes the standard ground truth answer (typically a final value or option without reasoning steps). The subject composition is presented in Figure 3.

3.2 Diverse Response Generation and Difficulty-Aware Selection

To assess verifier robustness against diverse reasoning patterns, we construct a response set containing varying qualities and error types. For each valid problem $(q, a) \in \mathcal{Q}_{\text{clean}}$, we employ M distinct Large Reasoning Models (LRMs) to serve as candidate solvers. Each model $\mathcal{M}_m$ generates a single reasoning trajectory r_m, yielding a set of M triplets per question:

$$\mathcal{T} = \{(q_i, a_i, r_i^m) \,|m \in \{M\}, (q_i, a_i) \in Q_{\text{clean}}\} \tag{1}$$

where r_i^m represents the response generated by the m-th model.

To ensure the benchmark focuses on samples that effectively discriminate verifier capabilities,

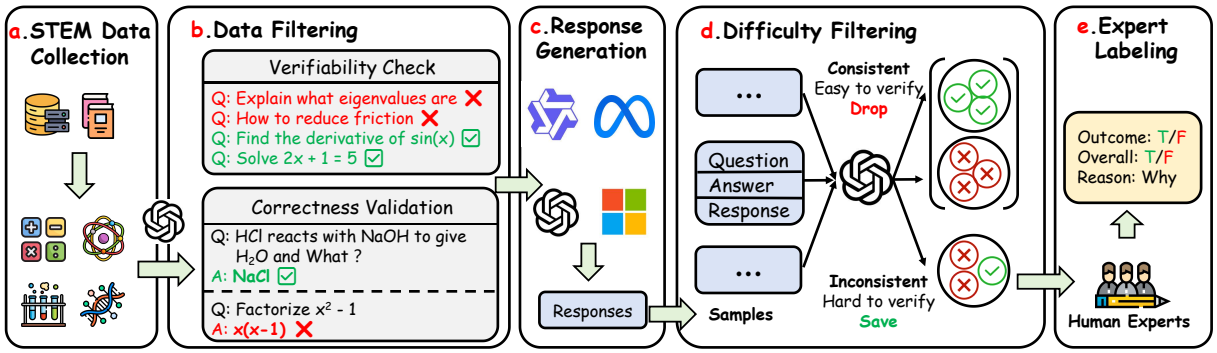


Figure 2: Overview of the PRIME construction pipeline. The process comprises five stages: (a) Extensive STEM data collection with diversity control; (b) Two-stage automated filtering for verifiability and correctness; (c) Heterogeneous response generation using diverse LRMs; (d) Difficulty-aware filtering to select discriminative samples; (e) Fine-grained expert labeling focusing on process-outcome alignment.

Figure 3: **Subject distribution of PRIME.** The inner ring represents the four major STEM disciplines, while the outer ring details the 16 fine-grained sub-domains.

we implement a *Verification Difficulty Filtering* mechanism. We select responses that are intrinsically hard to verify. We utilize GPT-OSS-120B as a proxy verifier to assess each triplet $(q, a, r) \in \mathcal{T}$. The verifier is prompted to check whether the response r is correct regarding both the reasoning process and the final result.

To capture the uncertainty in verification, we perform $k = 8$ independent verification trials for each triplet. Let $v_j \in \{0, 1\}$ denote the verdict of the j-th trial, where 1 indicates the response is judged as correct and 0 otherwise. We define the *Verification Consensus Score* $C(q, r)$ as the average of positive judgments: $C(q, r) = \sum_{j=1}^{k} v_j / k$. Based on this score, we categorize the samples:

- **Easy-to-Verify (Drop):** $C(q, r) = 1$ or $C(q, r) = 0$. These are samples where the proxy verifier is unanimous across all trials (either consistently correct or consistently incorrect). These cases represent "obvious" decisions that offer minimal signal for distinguishing strong verifiers.

- **Hard-to-Verify (Save):** $0 < C(q, r) < 1$. These samples lie on the decision boundary of the verifier, where the proxy model exhibits inconsistency in its judgment. Such instances typically contain subtle errors or deceptive steps that are challenging to verify reliably.

We retain only the response triplets from the "Hard-to-Verify" category to form the final evaluation set $\mathcal{T}_{\text{final}}$. This strategy ensures that PRIME specifically targets the difficult regime of process supervision.

3.3 Fine-grained Expert Labeling and Evaluation Metrics

To ensure the accuracy and reliability of our PRIME, we employ domain experts to perform fine-grained annotation on the selected samples in $\mathcal{T}_{\text{final}}$. Instead of relying on traditional outcome-only checks, we introduce a process-aware labeling. Specifically, for each trajectory (q, r, a), experts provide a structured label tuple $L = (y_{\text{outcome}}, y_{\text{overall}})$:

- $y_{\text{outcome}} \in \{0, 1\}$: **Outcome Label.** Validity of the extracted final answer a against the ground truth a_{gt}.

- $y_{\text{overall}} \in \{0, 1\}$: **Overall Label (Process-Outcome Consistency).** This is the primary metric of PRIME. It is marked as valid (1) *if and only if* the reasoning process s is logically

sound *and* correctly derives a. This strictly penalizes "lucky guesses", where the answer is correct by chance but the derivation is flawed.

Consequently, the evaluation score on PRIME is calculated based on the prediction of the overall Label (y_{overall}), rather than the outcome-only label. This ensures that verifiers are assessed on their ability to validate the entire derivation process and final answer. A summary of the annotation protocol is provided in Appendix A.1. The detailed prompts used for the verifier in our evaluation are provided in Appendix Figure 9.

4 Analysis of Performance on PRIME

In this section, we conduct a comprehensive evaluation on PRIME to assess the capability of current models in ensuring process-outcome alignment. We benchmark a diverse array of models, and present an in-depth analysis of performance trends, the impact of reasoning capabilities, and the trade-off between accuracy and efficiency.

4.1 Experimental Settings.

Models. We categorize the evaluated models into three distinct groups: (1) **Open-source Models**, which encompass standard instruction-tuned models such as Qwen3-Instruct (Yang et al., 2025), DeepSeek-V3.2 (Liu et al., 2025a), Phi-4 (Abdin et al., 2024), and GLM-4.6 (Z.ai Team, 2025), alongside reasoning-enhanced variants like Qwen3-thinking, DeepSeek-V3.2-thinking, and Kimi-K2-thinking (Team et al., 2025). This category also includes the GPT-OSS series (20B and 120B) (Agarwal et al., 2025). (2) **Closed-source Models**, comprising proprietary systems divided into standard chat versions (e.g., GPT-5-chat (OpenAI, 2025), Claude-Sonnet/Opus (Anthropic, 2025a,b)) and reasoning-oriented models tailored for complex tasks, such as Gemini-3.0/2.5-Pro-thinking (Comanici et al., 2025; Google DeepMind, 2025), GPT-5-thinking (OpenAI, 2025), Grok-4-thinking (xAI Team, 2025), and Seed-1.6-thinking (ByteDance Seed Team, 2025). (3) **Specialized Verifiers**, including models explicitly engineered for verification and reward modeling tasks, such as CompassVerifier-32B, xVerify-9B-C, SCI-Verifier-4B, and Tencent-Qwen2.5-7B-RLVR (Su et al., 2025). We also include MathVerify (Kydlíček and Hugging Face Team, 2025), a rule-based verifier where we employ GPT-OSS-120B to extract the final answer from the model's response for direct comparison with the ground truth.

Evaluation Paradigm. We adopt distinct evaluation strategies based on the model type. For specialized verifiers, we strictly adhere to their official implementations to extract verification predictions, directly comparing them against the corresponding ground truth labels. For general-purpose models, we employ a structured prompting approach (Figure 9) that asks the model to independently check two aspects: whether the derivation logic is sound (`<process>`) and whether the final answer is correct (`<outcome>`). To determine the model's overall judgment, we simply consider a solution valid only when the model approves *both* the process and the outcome. Since our benchmark provides specific ground truth labels for both the outcome and the overall correctness, we calculate performance on two corresponding metrics: (1) **Outcome Accuracy**, measured by comparing the model's `<outcome>` tag directly with the Outcome ground truth y_{outcome}; and (2) **Overall Accuracy**, measured by comparing the model's combined judgment against the Overall ground truth y_{overall}.

4.2 Performance Analysis.

General Performance Trends. As shown in Table 1, performances on PRIME are strongly related to both model scale and model family. Closed-source models generally achieve the best results, with Gemini-2.5-Pro-thinking reaching an overall accuracy of 88.56%. Among open-source models, accuracy increases steadily with parameter count, rising from Qwen3-4B (75.23%) to GPT-OSS-120B, which attains the best open-source overall accuracy of 81.64%. These results suggest that large open-source models are becoming increasingly competitive on demanding verification tasks.

Reasoning is Important for Verification. Table 1 also shows that models with explicit reasoning consistently outperform their instruction-tuned counterparts. For example, Qwen3-4B-thinking exceeds Qwen3-4B-Instruct by more than 8%. This gap indicates that verification on PRIME requires more than surface-level semantic matching; it often requires following the derivation to catch logical errors. Overall, strong reasoning ability appears necessary for a high-quality verifier.

Model	Math		Physics		Chemistry		Biology		Average	
	Outcome	Overall	Outcome	Overall	Outcome	Overall	Outcome	Overall	Outcome	Overall
Rule-based Verifier										
Math-Verify	65.56	63.67	63.17	65.85	69.69	65.47	73.80	72.04	67.71	65.93
Open-source Models										
Qwen3-4B-Instruct-2507	81.22	64.78	78.35	68.97	80.05	70.20	77.33	74.56	79.74	73.13
Phi-4	81.44	60.00	81.70	70.09	80.05	63.81	76.57	69.52	80.29	64.46
DeepSeek-V3.2	87.56	72.44	87.50	79.02	88.24	77.62	87.91	84.63	87.81	77.13
Qwen3-4B	86.33	69.56	87.72	79.46	86.57	75.83	84.89	82.12	86.43	75.23
Qwen3-4B-thinking-2507	89.22	76.22	89.51	82.81	91.30	83.76	90.68	86.65	90.15	81.36
Qwen3-8B	87.78	72.89	88.17	80.13	87.47	77.37	87.91	84.13	87.77	77.32
Qwen3-30B-A3B	90.44	79.67	87.28	83.26	88.24	83.38	90.18	87.66	89.16	82.71
GPT-OSS-20B	89.67	75.56	88.62	83.26	88.24	79.80	89.67	85.14	89.04	79.74
GPT-OSS-120B	91.22	78.33	88.39	82.37	90.03	79.80	88.92	85.14	89.99	81.64
DeepSeek-V3.2-thinking	90.67	**83.89**	89.73	**87.72**	**93.35**	88.49	**91.18**	**90.43**	**91.41**	**87.02**
GLM-4.6	90.22	81.00	89.51	85.27	89.00	84.91	87.66	84.38	89.32	83.50
Kimi-K2-thinking	**91.78**	82.56	**90.40**	87.05	92.46	**89.00**	89.67	87.66	**91.41**	86.15
Closed-source Models										
Claude-Sonnet-4	89.44	74.78	88.39	80.58	88.62	77.75	84.63	81.61	88.25	77.80
Claude-Sonnet-4.5	90.22	74.00	86.83	79.69	88.36	77.49	83.88	79.60	88.05	76.97
Claude-Opus-4	89.33	75.33	88.39	81.47	89.26	80.43	88.41	84.89	89.00	79.50
Claude-Opus-4.5	90.89	75.67	90.18	81.47	89.13	79.67	86.65	81.86	89.55	78.91
GPT-5-chat	88.11	71.11	86.61	77.23	82.74	68.67	83.12	77.33	85.40	72.42
GPT-5.2-chat	90.78	81.89	92.19	88.17	91.30	86.45	91.44	89.42	91.29	85.60
Claude-Sonnet-4-thinking	90.78	81.00	89.73	85.94	90.28	85.29	86.65	85.39	89.79	83.89
Claude-Sonnet-4.5-thinking	**92.44**	81.00	91.07	**89.96**	**92.20**	88.11	90.43	88.66	**91.81**	85.99
Claude-Opus-4-thinking	91.59	80.89	91.07	87.28	91.43	86.57	88.41	86.65	90.94	79.50
Claude-Opus-4.5-thinking	88.89	79.33	89.73	86.61	87.85	81.84	85.64	85.14	88.21	82.31
Gemini-3.0-pro-thinking	90.56	81.67	88.39	85.49	89.26	86.19	86.15	85.39	89.08	84.33
Gemini-2.5-Pro-thinking	92.11	86.11	91.52	88.17	91.30	**90.15**	**91.94**	**91.44**	91.52	**88.56**
GPT-5-thinking	91.11	**86.67**	90.85	**89.96**	89.64	88.62	88.16	88.16	90.15	88.09
GPT-5.2-thinking	92.11	86.00	**92.41**	**89.96**	91.30	88.49	89.92	89.17	91.57	87.97
Grok-4-thinking	91.33	85.78	89.51	87.72	91.82	89.00	90.18	89.42	90.98	87.69
Seed-1.6-thinking	90.00	74.78	87.72	79.91	89.13	79.80	86.65	82.87	88.80	78.51
Verifier Model										
General-Verifier	76.78	57.78	78.57	68.53	77.49	63.04	75.57	68.77	77.13	63.04
Tencent-Qwen2.5-7B-RLVR	86.22	66.33	82.14	72.77	84.65	71.23	85.14	78.34	84.84	70.87
xVerify-9B-C	87.11	67.44	81.92	73.44	83.89	72.55	86.39	80.10	84.92	71.98
CompassVerifier-32B	**88.33**	67.33	**88.62**	77.46	**89.13**	**73.66**	88.66	81.36	**88.68**	73.29
SCI-Verifier-4B	87.56	**68.78**	87.05	**77.68**	87.98	73.53	**91.18**	**84.38**	88.17	**74.28**

Table 1: Performance of different models on PRIME. The best and second-best results are highlighted in **bold** and underlined, respectively.

4.3 Further Analysis

Efficiency vs. Performance Trade-off. To guide the selection of a verifier for resource-intensive tasks like RLVR, we analyzed the trade-off between computational cost and verification performance. In Figure 4, we plot the verification accuracy on the PRIME benchmark against a proxy for inference cost (calculated as the product of activated parameters and the number of generated tokens). The visualization reveals a general trend where higher accuracy correlates with an increased proxy cost, indicating that greater reasoning effort often leads to better performance. This analysis provides a practical framework for selecting an appropriate verifier by balancing the required accuracy against the available computational budget for scalable training.

5 Analysis of Downstream RLVR

We bridge our PRIME analysis to the practical task of RLVR. To validate the benchmark's utility, we first demonstrate the critical impact of process consistency via an ablation study. We then investigate the correlation between verifier scores and resulting policy performance, establishing PRIME as a strong predictor of downstream RLVR outcomes.

5.1 Experimental Settings.

Reinforcement Learning Details. The reward signal is designed to strictly enforce both outcome

Verifier Model	AIME24	AIME25	Beyond-AIME	GPQA-D	SuperGPQA	MATH-500	MMLU-Pro	AVG
Qwen3-8B-Base								
GPT-OSS-120B-Outcome	21.04	18.65	7.94	37.18	26.91	83.20	64.90	37.12
GPT-OSS-120B	**29.64**	**22.40**	**10.94**	**45.45**	**31.00**	**86.40**	**66.12**	**41.70**
Qwen3-14B-Base								
GPT-OSS-120B-Outcome	28.85	21.61	11.27	45.17	**41.26**	85.00	68.86	43.15
GPT-OSS-120B	**37.14**	**30.73**	**18.58**	**50.66**	31.60	**91.20**	**69.61**	**47.07**

Table 2: Performance comparison between models trained with Outcome-only verification and our proposed Process-aware verification. The best results are highlighted in **bold**.

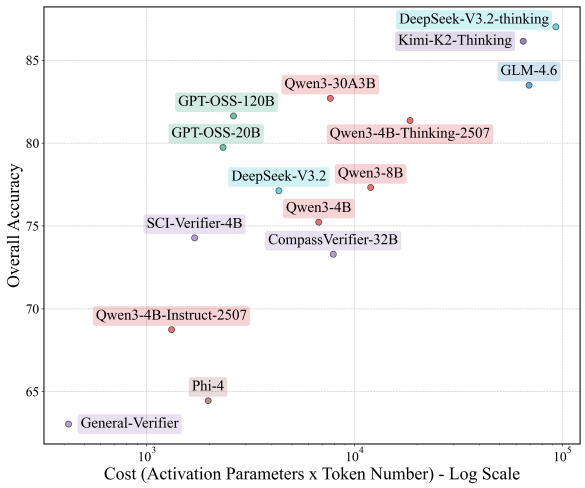


Figure 4: Efficiency-Performance Trade-off Analysis. The scatter plot visualizes the relationship between the average token consumption on PRIME and the overall accuracy for various models. The red dashed line delineates the boundary between open-source models (below) and commercial closed-source models (above).

accuracy and the logical consistency of the derivation. Formally, for a given question x, model response y, and ground truth answer a^*, the reward $R(x, y, a^*)$ is defined as:

$$R(x, y, a^*) = \mathbb{I}(\mathcal{O}) \cdot \mathbb{I}(\mathcal{C}) \tag{2}$$

where $\mathbb{I}(\cdot)$ is the indicator function. Here, $\mathcal{O}$ represents the outcome correctness, verifying whether the final answer extracted from y matches the ground truth a^*. $\mathcal{C}$ denotes the process consistency, a judgment provided by the verifier on whether the model's reasoning trajectory is logically sound and consistent with the final result. This dual-verification mechanism ensures that the model is penalized for "lucky guesses" where the answer is correct but the reasoning is flawed. We perform RLVR directly on two base models Qwen3-8B-Base and Qwen3-14B-Base. The training utilizes the `WebInstruct-verified` dataset. We configure the training with a global batch size of 64 and a rollout number of 16. All experiments are conducted on a cluster of 40 H800 GPUs, and the models are trained for a total of 300 steps with PPO.

Benchmarks We evaluate LRMs trained via RLVR using verifiers selected by our benchmark across diverse domains. Our suite includes AIME 2024, AIME 2025, and Beyond-AIME (ByteDance-Seed, 2025) for advanced math; GPQA-Diamond (GPQA-D) (Rein et al., 2024) and SuperGPQA (Du et al., 2025) for scientific reasoning; plus MATH-500 and MMLU-Pro (Wang et al., 2024b). We report average@16 accuracy for the AIME suite, average@4 for GPQA-D and MATH-500, and pass@1 for others.

5.2 Impact of Process Consistency.

To verify the necessity of the process consistency judgment in our reward formulation, we conduct ablation experiments comparing our proposed process-aware RLVR against a standard outcome-only baseline. We use GPT-OSS-120B as the verifier. In the baseline setting, the reward is determined solely by the final answer correctness, ignoring the logical validity of the intermediate reasoning trajectory. The results are presented in Table 2. We can see that incorporating process-aware supervision yields consistent and significant improvements across both model scales. For the Qwen3-8B-Base model, the inclusion of process verification boosts the average accuracy from 37.12% to 41.70%. Similarly, for the Qwen3-14B-Base model, the performance rises from 43.15% to 47.07%. The gains are particularly pronounced in high-difficulty reasoning benchmarks; for instance, on AIME 2024, the 8B model improves drastically from 21.04% to 29.64%, and the 14B model improves from 28.85% to 37.14%. This indicates that relying solely on outcome verification makes the policy model prone to reward hacking via "lucky guesses," whereas strictly enforcing process consistency ensures the model learns robust

Verifier Model	AIME24	AIME25	Beyond-AIME	GPQA-D	SuperGPQA	MATH-500	MMLU-Pro	AVG
Qwen3-8B Base								
-	4.53	2.86	2.23	20.61	17.66	40.20	31.79	17.13
Math-Verify	15.47	14.43	7.28	40.15	26.24	78.40	63.28	35.04
CompassVerifier-32B	13.39	10.99	7.59	38.64	30.46	78.80	59.33	34.17
Qwen3-4B	18.65	15.78	6.92	34.38	27.01	82.80	55.24	34.40
Qwen3-8B	24.74	18.91	8.58	37.72	27.84	83.60	58.73	37.16
GPT-OSS-20B	27.19	20.00	**11.72**	**45.93**	**32.17**	**86.60**	65.20	41.26
GPT-OSS-120B	**29.64**	**22.40**	10.94	45.45	31.00	86.40	**66.12**	**41.70**
Qwen3-14B Base								
-	7.60	4.53	2.89	32.01	23.52	57.60	45.35	24.79
Math-Verify	20.68	18.54	7.91	47.00	28.19	84.40	66.41	39.07
CompassVerifier-32B	21.93	13.02	8.98	46.24	**40.50**	85.00	69.08	40.68
Qwen3-4B	28.07	21.67	10.94	46.53	35.40	85.60	67.21	42.20
Qwen3-8B	22.40	18.75	10.92	49.21	39.03	86.40	68.27	42.14
GPT-OSS-20B	30.36	21.51	12.92	**51.58**	40.44	87.60	69.32	44.82
GPT-OSS-120B	**37.14**	**30.73**	**18.58**	50.66	31.60	**91.20**	**69.61**	**47.07**

Table 3: Downstream performance comparison. Evaluation of Qwen3-8B Base and Qwen3-14B Base trained via RLVR using different verifiers. The best results are highlighted in **bold** and the second best are underlined.

and generalized reasoning patterns, leading to superior performance on complex tasks.

5.3 Correlation and Performance.

Verifier Selection for Downstream Validation. To assess whether PRIME performance reliably predicts utility in practical RLVR pipelines, we select a representative subset of verifiers from Section 4. Instead of relying solely on top-scoring models, we cover a capability spectrum (70%–85% Overall Accuracy) by selecting five models: CompassVerifier-32B, Qwen3-4B, Qwen3-8B, GPT-OSS-20B, and GPT-OSS-120B. This diversity allows us to directly correlate benchmark scores with the downstream performance of policy models trained on their reward signals.

Correlation between Performance in PRIME and Downstream Performance. The comparative results are presented in Table 3 and further visualized in Figure 5. From Table 3, we can see that the performance of the trained policy models improves significantly as the capability of the verifier increases. Notably, using our best-performing verifier, GPT-OSS-120B, boosts the average accuracy of the Qwen3-14B Base model from 24.79% to 47.07%, surpassing the gains achieved by smaller verifiers like CompassVerifier-32B (40.68%) or Qwen3-4B (42.20%). Combining this with the trend analysis in Figure 5, we observe a strong positive linear correlation between the verifier's Overall Accuracy on PRIME and the policy model's final performance, with coefficient of determination (R^2) values reaching 0.937 and 0.920 for the 8B and 14B models, respectively. This indicates that PRIME serves as a highly reliable predictor of a reward model's utility. A verifier that demonstrates high process consistency on our benchmark is statistically more likely to guide the policy model toward correct reasoning paths, thereby effectively preventing reward hacking and enhancing training stability. Furthermore, the training dynamics, visualized in Appendix Figure 8, are provided to validate the stability of our RLVR process. The reward

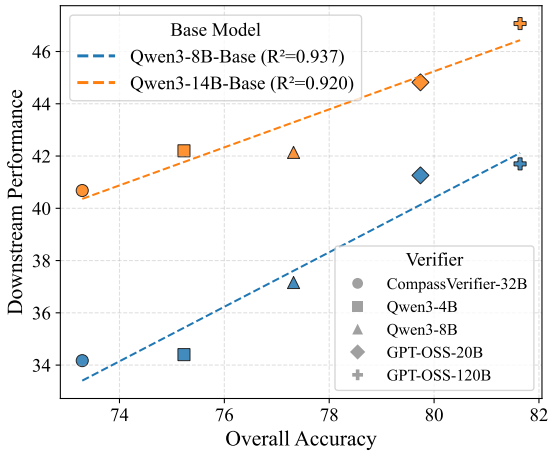


Figure 5: Correlation between verifier performance and downstream policy improvement with verifier. The x-axis represents the Overall Accuracy on our PRIME benchmark, and the y-axis represents the average score on the benchmarks in Table 3

curves exhibit a steady increase before reaching a plateau, indicating that each policy model was trained to convergence.

6 Conclusion

We introduce PRIME to evaluate the process verification capabilities of Large Reasoning Models. Our benchmarking reveals that strong reasoning is a prerequisite for verification, and process-aware verifiers significantly outperform outcome-only baselines. We further validate the benchmark's utility by demonstrating a strong correlation between PRIME scores and downstream RLVR policy improvement, confirming that high-quality process supervision leads to substantial gains in complex reasoning tasks.

Limitation

A primary limitation of this study is that we did not train a specialized process-aware verifier model from scratch. Instead, we leveraged existing high-performance open-source model selected by our benchmark to serve as the verifier in RLVR. Our empirical findings indicate that these general-purpose strong reasoners already possess sufficient capability to distinguish between valid reasoning and spurious "lucky guesses," thereby providing highly effective supervision signals without the need for costly specialized training. Future work will focus on scaling this verification paradigm to even larger LRMs and extending the evaluation to agentic scenarios, where multi-step planning and tool use require even more process consistency checks.

Ethical Considerations

The development of PRIME adheres to strict ethical standards for AI research. Our initial corpus of 7 million problems is sourced from university-level textbooks and public examinations. This data is intended solely for academic purposes to evaluate reasoning capabilities, and the final 2,530 benchmark samples are the result of an original, multi-stage processing and annotation pipeline. Professional domain experts in STEM performed all fine-grained labeling and were compensated fairly for their professional expertise. To protect privacy, no personally identifiable information of these experts is included in the released dataset. A primary goal of this work is to mitigate reward hacking and "lucky guesses" in RLVR by enforcing strict process-outcome alignment. By penalizing pseudo successes, we promote the development of more trustworthy and logically sound reasoning models for scientific applications. While we release the benchmark and trained models to facilitate open research, they are not intended for autonomous decision-making in high-stakes environments without human oversight. We do not foresee significant dual-use risks, as the content is centered on fundamental mathematics and engineering disciplines.

Acknowledgments

This work was supported in part by the grants from National Science and Technology Major Project (No. 2023ZD0121104), and the Anhui Natural Science Foundation (No. 2508085ZD006).

References

Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J Hewett, Mojan Javaheripi, Piero Kauffmann, and 1 others. 2024. Phi-4 technical report. *arXiv preprint arXiv:2412.08905*.

Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, and 1 others. 2025. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*.

Anthropic. 2025a. Claude Opus 4 and Claude Sonnet 4 System Card. System card, Anthropic. Accessed: 2026-01-05.

Anthropic. 2025b. Claude Sonnet 4.5 System Card. System card, Anthropic. Accessed: 2026-01-05.

ByteDance-Seed. 2025. Beyondaime: Advancing math reasoning evaluation beyond high school olympiads. https://huggingface.co/datasets/ByteDance-Seed/BeyondAIME.

ByteDance Seed Team. 2025. Seed 1.6: Multimodal models with adaptive deep thinking. https://seed.bytedance.com/en/seed1_6. Technical Report. Accessed: 2026-01-05.

Ding Chen, Qingchen Yu, Pengyuan Wang, Wentao Zhang, Bo Tang, Feiyu Xiong, Xinchi Li, Minchuan Yang, and Zhiyu Li. 2025. xverify: Efficient answer verifier for reasoning model evaluations. *CoRR*, abs/2504.10481.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, and 1 others. 2025. Gemini 2.5: Pushing the frontier with

advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.

Xinrun Du, Yifan Yao, Kaijing Ma, Bingli Wang, Tianyu Zheng, King Zhu, Minghao Liu, Yiming Liang, Xiaolong Jin, Zhenlin Wei, and 1 others. 2025. Supergpqa: Scaling llm evaluation across 285 graduate disciplines. *arXiv preprint arXiv:2502.14739*.

Ruixiang Feng, Zhenwei An, Yuntao Wen, Ran Le, Yiming Jia, Chen Yang, Zongchao Chen, Lisi Chen, Shen Gao, Shuo Shang, and 1 others. 2025. Cosineverifier: Tool-augmented answer verification for computation-oriented scientific questions. *arXiv preprint arXiv:2512.01224*.

Google DeepMind. 2025. Gemini: The Most Capable and General Model We've Ever Built. Accessed: 2026-01-05.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Hynek Kydlíček and Hugging Face Team. 2025. Math-Verify: A robust mathematical expression evaluation system. Accessed: 2026-01-05.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *arXiv preprint arXiv:2305.20050*.

Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, and 1 others. 2025a. Deepseek-v3. 2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*.

Shudong Liu, Hongwei Liu, Junnan Liu, Linchen Xiao, Songyang Gao, Chengqi Lyu, Yuzhe Gu, Wenwei Zhang, Derek F. Wong, Songyang Zhang, and Kai Chen. 2025b. Compassverifier: A unified and robust verifier for llms evaluation and outcome reward. *CoRR*, abs/2508.03686.

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. GPT-5 System Card. System card, OpenAI. Accessed: 2026-01-05.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. 2024. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

ByteDance Seed, Jiaze Chen, Tiantian Fan, Xin Liu, Lingjun Liu, Zhiqi Lin, Mingxuan Wang, Chengyi Wang, Xiangpeng Wei, Wenyuan Xu, and 1 others. 2025. Seed1. 5-thinking: Advancing superb reasoning models with reinforcement learning. *arXiv preprint arXiv:2504.13914*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Yi Su, Dian Yu, Linfeng Song, Juntao Li, Haitao Mi, Zhaopeng Tu, Min Zhang, and Dong Yu. 2025. Crossing the reward bridge: Expanding RL with verifiable rewards across diverse domains. *CoRR*, abs/2503.23829.

Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, and 1 others. 2025. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. 2022. Solving math word problems with process-and outcome-based feedback. *arXiv preprint arXiv:2211.14275*.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2024a. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9426–9439.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, and 1 others. 2024b. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37:95266–95290.

xAI Team. 2025. Grok 4 Model Card. Model card, xAI. Accessed: 2026-01-05.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, and 1 others. 2024. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*.

Z.ai Team. 2025. GLM-4.6: Advanced Agentic, Reasoning and Coding Capabilities. Accessed: 2026-01-05.

Kaiyan Zhang, Yuxin Zuo, Bingxiang He, Youbang Sun, Runze Liu, Che Jiang, Yuchen Fan, Kai Tian, Guoli Jia, Pengfei Li, and 1 others. 2025. A survey of reinforcement learning for large reasoning models. *arXiv preprint arXiv:2509.08827*.

Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. 2024. Generative verifiers: Reward modeling as next-token prediction, 2024. *URL https://arxiv.org/abs/2408.15240*, 1.

Shenghe Zheng, Chenyu Huang, Fangchen Yu, Junchi Yao, Jingqi Ye, Tao Chen, Yun Luo, Ning Ding, Lei Bai, Ganqu Cui, and Peng Ye. 2025. Sci-verifier: Scientific verifier with thinking. *CoRR*, abs/2509.24285.

A Appendix

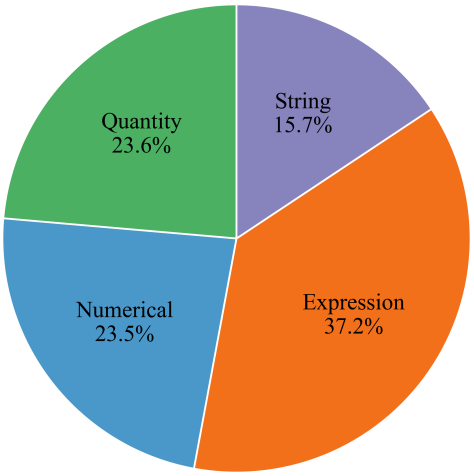


Figure 6: **Answer type distribution of PRIME.**

Model	Accuracy
CompassVerifier	4.66
xVerify-9B-C	11.89
Tencent-Qwen-7B-RLVR	8.86
Sci-Verifier	9.09
Qwen3-4B	22.14
Qwen3-8B	29.37
GPT-OSS-20B	36.83
GPT-OSS-120B	45.92

Table 4: Performance comparison on the "Lucky Guess" subset in our PRIME. This subset contains samples with correct final outcomes but flawed derivations, measuring the verifiers' sensitivity to derivation false positives.

A.1 Annotation Details

We recruited 18 annotators with STEM backgrounds across four disciplines: Mathematics (5), Engineering (5), Chemistry (5), and Biology (3). Before formal annotation, each annotator completed a 20-item discipline-specific screening set, and only those exceeding an 80% threshold were admitted.

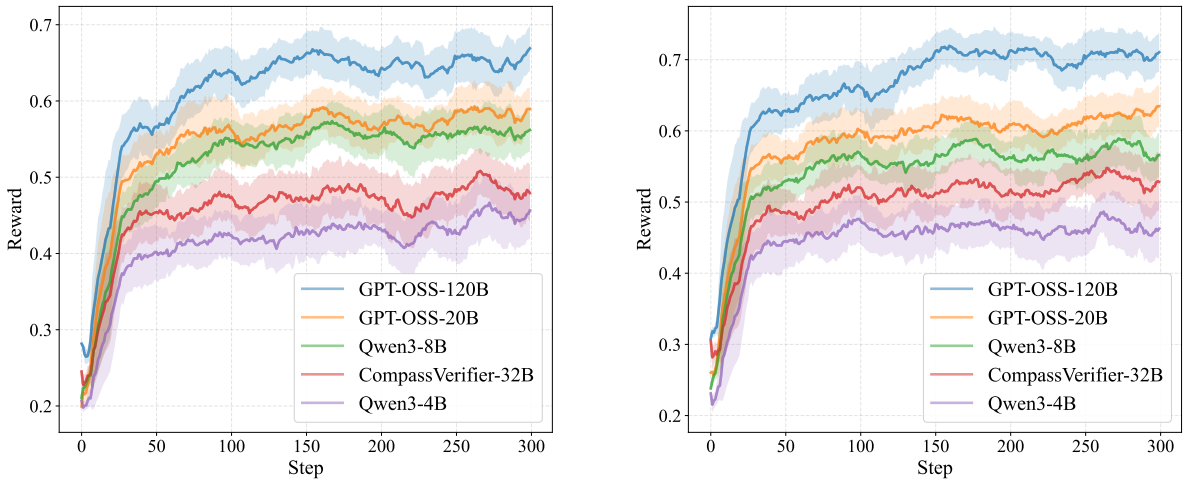
During annotation, instances were processed in batches of 300, with 50 randomly sampled for independent audit by a senior PhD-level inspector. If more than 10% of the audited labels were problematic, the entire batch was rejected and reassigned for re-annotation until it passed audit.

A.2 Case Study

Figure 7 illustrates a "lucky guess" scenario where the model incorrectly uses the circumference formula ($C = 2\pi r$) to calculate the area of a circle ($r = 2$) yet coincidentally arrives at the correct value 4π. This case highlights the limitations of baseline verifiers. The rule-based verifier rejects the response due to rigid formatting constraints as it fails to match "4pi" with "4π", resulting in a false negative regarding the outcome. In contrast, the outcome-only verifier accepts the response based solely on the correct final answer and ignores the flawed derivation. This generates false positive signals that encourage reward hacking in RLVR. Finally, a process-aware verifier (such as GPT-OSS-120B) successfully detects the logical fallacy of applying the circumference formula for area calculation and correctly penalizes the response. It further indicates that process-aware verification is essential to filter out false positive responses, ensuring that rewards are assigned solely for reasoning that is both accurate in outcome and valid in derivation.

Evaluation Example	Rule-based Verifier	Outcome-only Verifier	Process-aware Verifier
Question: Calculate the **area** of a circle with a radius of r=2. **Answer**: 4π **Response**: The formula for the area of a circle is C=2πr. So the final answer is 2π x 2 = 4pi. **Outcome**: True **Overall**: False	**Output**: False (Because it cannot identify 4pi as 4π.) **Outcome Score: 0** **Overall Score: 1**	**Output**: The final answer "4pi" is semantically equivalent to the reference answer "4π". <verdict>**True**</verdict> **Outcome Score: 1** **Overall Score: 0**	**Output**: The model incorrectly applies the circumference formula (2πr) to calculate the **area**. The correct answer is **coincidental**. <result>True</result> <overall>False</overall> **Outcome Score: 1** **Overall Score: 1**

Figure 7: Qualitative comparison of verification paradigms.



(a) Qwen3-8B-Base

(b) Qwen3-14B-Base

Figure 8: RLVR training reward curves using different verifiers. Higher-performing verifiers identified by PRIME consistently provide more stable and higher reward signals during the training process.

The Evaluation Prompt for Process-Outcome Consistency Verification

```
# Role
You are an extremely meticulous and conscientious exam grader. Your task is to assess whether a
student's submitted answer is correct.
# Input Variables
- **[Question]**: The problem that needs to be solved.
- **[Reference Answer]**: Typically only the final result, as a reference.
- **[Student Answer]**: Includes the full solution steps and the final result.
# Primary Principles
I will provide you with a question and the corresponding reference answer. You need to determine:
1. Whether the student's solution **process** is correct.
2. Whether the student's final **result** is correct.
3. Whether the student's answer is **perfect**.
# Core Logic ## Process Correctness
You must carefully check whether there are problems in each step of the student's solution. Your
judgment should follow the principle of "presumption of correctness": the process is considered
correct unless one of the following fundamental errors occurs.
**Fundamental Errors**:
1. Logical errors, calculation errors, or factual errors in reasoning.
2. Inconsistencies within reasoning, such as inconsistency between statements.


**Note**: If the student did not provide the solution process, it is considered **incorrect by
default**.
## Result Correctness
Similarly, your judgment of the final result also follows the principle of "presumption of
correctness": the answer is considered correct unless one of the following fundamental errors
occurs.
**Fundamental Errors**:
1. The student fails to provide a clear final answer.
2. The student's final answer does not satisfy the problem requirements.
3.  The student's final result does not match the reference answer (unless you can prove the
student's result is also correct).
**Note**:  In numerical calculations, discrepancies due to reasonable rounding are **not
considered** fundamental errors but should be carefully distinguished from computational errors.
## Answer Perfection
An answer is considered perfect only when both the process and result are correct, and none of
the following imperfections occur. Any answer with a fundamental error is necessarily imperfect.
Imperfection reference standards:
1. Insufficient precision: The solution does not provide the exact form (e.g., √2) but only an
approximate value (e.g., 1.414).
2. Not simplified: Fractions, radicals, etc., are not expressed in simplest form.
3. Redundant content: Contains information clearly irrelevant to the solution.


4. Lack of process: Missing necessary solution steps.
5. Other flaws you consider imperfect but not fundamental errors.
**Note**: If the problem's requirements conflict with the standards of perfection, the problem
requirements take precedence.
# Special Cases
- If the student's answer contains a large amount of repetitive content, both the process and
the result should be considered incorrect.
- If the student's answer is incomplete (e.g., cut off), both the process and the result should
be considered incorrect.
# Output Format
You need to output a result in XML format with the following fields:
<process> [Your judgment on the correctness of the solution process, only True or False] </process>
<outcome> [Your judgment on the correctness of the final result, only True or False] </outcome>
<perfect> [Your judgment on whether the answer is perfect, only True or False] </perfect> <reason>
[Your reasoning for the judgment in a few brief sentences] </reason>
{few_shots}
# Task
Now, please strictly follow the above principles, logic, and format to evaluate the following
input and return the result in XML format.
<question> question </question>
<student_answer> student_answer </student_answer>


<reference_answer> reference_answer </reference_answer>
```

Figure 9: Detailed evaluation prompt used for assessing verifier performance on PRIME. This prompt enforces a multi-dimensional check on process correctness, result accuracy, and overall perfection.