PhysPRM: A Generative Process Reward Model with Fine-grained Diagnosis for Physics Problem Solving

Yuxuan Dong[1,2], **Xinyu Zhang**[1,2*], **Lingling Zhang**[1,2], **Han Lai**[1,2], **Pengyu Li**[1,2], **Bifan Wei**[1,2], **Yaqiang Wu**[4], **Jun Liu**[1,3]

[1]School of Computer Science and Technology, Xi'an Jiaotong University
[2]Ministry of Education Key Laboratory of Intelligent Networks and Network Security, China
[3]Shaanxi Province Key Laboratory of Big Data Knowledge Engineering, China
[4]Lenovo Research
yuxuandong@stu.xjtu.edu.cn, zhang1393869716@stu.xjtu.edu.cn

Abstract

Despite the remarkable progress of Large Language Models (LLMs) in abstract reasoning tasks, they continue to struggle with physics problem solving due to difficulties in decoding implicit constraints and maintaining physical consistency. To address these challenges, Process Reward Models (PRMs) have emerged as a promising approach to verify intermediate reasoning steps. Existing PRMs attempt to mitigate reasoning errors but typically rely on scalar scoring, which lacks the explanatory power necessary to diagnose complex physical misconceptions. In this work, we introduce **PhysPRM**, a Generative PRM that treats evaluation as a generative task to produce fine-grained diagnoses comprising critiques, final judgments, and specific error types. To facilitate this, we develop an automated data synthesis pipeline to construct PhysPRM30K, a comprehensive training dataset, and PhysProcessBench, a rigorously human-verified benchmark. By employing a two-stage training paradigm that integrates Supervised Fine-Tuning with Group Relative Policy Optimization, PhysPRM significantly enhances the physics reasoning capabilities of various LLMs. Extensive experiments demonstrate that PhysPRM boosts performance by up to 8.9 and 10.3 points in Best-of-N and critique refinement strategies, respectively. [1]

1 Introduction

With the remarkable success of Large Language Models (LLMs) in natural language processing (NLP), they have also achieved significant advancements across reasoning-intensive domains, such as mathematics (Yan et al., 2025b; Lightman et al., 2024) and logic (Xu et al., 2025a). Despite their strong performance in these abstract tasks, a large gap remains in physics-based reasoning. Physics problem solving inherently demands the tight integration of domain knowledge with real-world constraints. Specifically, a robust physics solver is required to: (1) decode implicit physical constraints embedded in natural language descriptions, e.g., interpreting "smooth circular track" in the question as a frictionless constraint that implies energy conservation, as illustrated in Figure 1 (a); (2) maintain physical consistency because the laws of physics remain invariant across different scenarios and reasoning trajectories. However, LLMs often exhibit unstable performance in these core capabilities, particularly when generating long reasoning chains where physics errors stemming from these fundamental deficits frequently occur.

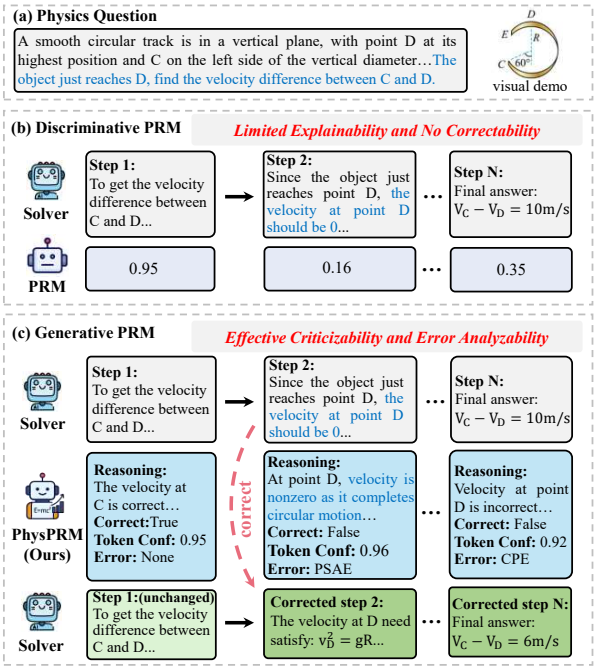


Figure 1: Comparison between discriminative PRMs and our proposed generative PRM (PhysPRM) for assisting LLMs in physics problem solving.

The unstable performance of LLMs in physics problem solving often leads to low success rates in single-pass attempts. However, given that cor-

*Corresponding author
[1]Code is available at https://github.com/fakedyx/PhysPRM.

Findings of the Association for Computational Linguistics: ACL 2026, pages 9410–9427

rect solutions often exist within multiple sampled trajectories, Process Reward Models (PRMs) have emerged as a promising approach. These models evaluate each reasoning step to select the best solution from a set of candidates generated by LLMs. Currently, most methods in other reasoning domains are discriminative PRMs (Wang et al., 2025b; Rizvi et al., 2025; Sun et al., 2025; Pala et al., 2025) that typically produce scalar scores or simple labels, as shown in Figure 1 (b). Consequently, these approaches suffer from a lack of explanatory power (Zhao et al., 2025). In the physics domain, where errors arise from diverse sources, including the misapplication of physical mechanisms, incorrect state or process analysis, and calculation errors, a simple scalar score or "incorrect" label is insufficient for effective diagnosis.

In this work, we introduce **PhysPRM**, a generative PRM designed for physics problem solving. Unlike discriminative PRMs, PhysPRM treats process supervision as a generative task rather than mere scalar scoring. It integrates Chain-of-Thought (CoT) (Wei et al., 2022) to produce fine-grained diagnoses, comprising detailed critiques, final judgments, and specific error types, while also leveraging token-level probabilities to quantify confidence in its judgment, as shown in Figure 1 (c). Based on this diagnosis, PhysPRM not only evaluates each reasoning step to select the optimal solution, but also provides actionable guidance for reasoning refinement, enabling the LLM to correct the first erroneous step encountered in a reasoning chain.

To facilitate model development, we design an automated data synthesis pipeline that leverages LLMs to generate high-quality data. Specifically, this pipeline proceeds through four sequential stages: *Diverse Solution Generation*, *Automated Process Annotation*, *Data Filtering*, and *Curriculum-based Organization*. Using this pipeline, we construct PhysPRM30K, a comprehensive training dataset containing approximately 30,000 samples, and PhysProcessBench, a human-verified benchmark designed to evaluate the ability of models to detect and classify erroneous steps.

Building upon the dataset and benchmark, we train PhysPRM through Supervised Fine-Tuning (SFT) for initialization and Group Relative Policy Optimization (GRPO) for refinement. Experimental results demonstrate that PhysPRM significantly enhances LLMs' physics reasoning capabilities across different model families and scales. Specifically, PhysPRM improves the overall performance of Llama3.1-8B, Qwen2.5-7B-Instruct, and Qwen2.5-32B-Instruct by 5.9, 8.9, and 7.0 points, respectively, across seven physics benchmarks under the Best-of-8 setting. In the critique refinement strategy, PhysPRM surpasses Gemini-2.5-Flash with an average 10.3-point gain over three turns. Our main contributions are as follows:

(1) We propose an automated data synthesis pipeline to construct PhysPRM30K, a comprehensive training dataset, and PhysProcessBench, a rigorously human-verified benchmark.

(2) We develop **PhysPRM**, a generative PRM designed for physics problem solving that offers fine-grained diagnoses and serves as a plug-and-play module to enhance LLMs through both Best-of-N and critique refinement strategies.

(3) Extensive experiments across seven physics benchmarks demonstrate that PhysPRM boosts LLMs' performance by up to 8.9 and 10.3 points in Best-of-N and critique refinement, respectively.

2 Related Work

Process Reward Models (PRMs). PRMs have emerged as a key technique for complex reasoning, providing process supervision through step-by-step evaluation (Lightman et al., 2024; Zhang et al., 2025c). Existing approaches are broadly categorized into two streams: discriminative and generative. Discriminative PRMs function as classifiers, encoding the context to predict a scalar correctness score for each step (Wang et al., 2024; Luo et al., 2024; Pala et al., 2025; Chen et al., 2025a). Although straightforward, they often lack interpretability. In contrast, generative PRMs leverage LLMs' capabilities not only to evaluate reasoning steps but also to explain rationales and guide corrections. Emerging works (Zhang et al., 2025a; Khalifa et al., 2025; Zhao et al., 2025) incorporate generative analysis into the evaluation process.

Physics Problem Solving. To improve physics problem solving, research has evolved from basic CoT prompting (Kojima et al., 2022) to tool-augmented frameworks that utilize external APIs for formula retrieval and calculations (Pang et al., 2025; Ma et al., 2024; Zhang et al., 2026a; Zhu et al., 2025; Zhang et al., 2026b). More recently, specialized training and reinforcement learning have further enhanced the reasoning capabilities of LLMs (Dan et al., 2025; Chen et al., 2025b). Despite these advancements, most existing methods still struggle with error cascading in complex

reasoning chains due to the absence of step-level supervision. PhysPRM addresses this gap by providing fine-grained diagnoses to guide LLMs through the physics reasoning process.

3 Methodology

We introduce PhysPRM, a generative PRM designed to evaluate step-by-step reasoning in physics problems. Given a context C and a question Q, an LLM π generates a solution $S = \{s_1, \ldots, s_n\}$ via $S = \pi(C \parallel Q)$. Unlike discriminative models that assign scalar scores, PhysPRM generates a sequence of fine-grained diagnoses for each step:

$$f_{\text{PhysPRM}} : (Q, S) \mapsto (d_1, d_2, \ldots, d_n) \quad (1)$$

where f_{PhysPRM} denotes the inference of PhysPRM and d_i represents the diagnosis for step s_i. Specifically, each diagnosis d_i is defined as a tuple (t_i, j_i, e_i) that encapsulates the full assessment of the step. Here, t_i is the detailed critique providing the rationale for judgment j_i and error type e_i.

3.1 How to Synthesize Data for PhysPRM?

To address the data requirements of PhysPRM, we propose our pipeline for high-quality data synthesis. As illustrated in Figure 2 (a), the pipeline consists of four key stages: **Diverse Solution Generation**, **Automated Process Annotation**, **Data Filtering**, and **Curriculum-based Organization**.

Diverse Solution Generation. We aggregate problems from various physics datasets. To induce diversity in reasoning paths, we employ an ensemble of distinct LLMs to generate multiple solutions per problem. We then implement a strict retention strategy: all incorrect solutions are preserved to maximize error exposure, while correct solutions are down-sampled to balance the dataset. The selected solutions are subsequently split into individual steps using the "\n\n" separator. To reduce data construction costs, we merge adjacent steps if the total step count exceeds 10.

Automated Process Annotation. We utilize the advanced LLM Gemini-2.5-Flash (Comanici et al., 2025) as an automated judge to conduct fine-grained evaluations. Specifically, each reasoning step s_i is verified sequentially against the ground truth (S^*, a^*) and the history of previously generated diagnoses $d_{1:i-1}$. This process continues until either the first error is detected or the entire sequence is confirmed as correct. For each step under evaluation, the judge first generates a textual critique t_{j_i} and a step judgment j_i:

$$f_{\text{Judge}} : (Q, S^*, a^*, s_{1:i}, d_{1:i-1}) \mapsto (t_{j_i}, j_i) \quad (2)$$

Subsequently, if a step is judged as correct, the error type is set to *None*. Conversely, if the step is deemed incorrect, the judge performs an *Error Type Analysis* to generate a classification rationale t_{e_i} and determine the corresponding error type e_i:

$$f_{\text{Judge}} : (Q, S^*, a^*, s_{1:i}, t_{j_i}, j_i) \mapsto (t_{e_i}, e_i) \quad (3)$$

The identified error e_i is classified into one of four categories[2]: *Physical Mechanism Application Error(**PMAE**)*, *Physical State Analysis Error(**PSAE**)*, *Physical Process Understanding Error(**PPUE**)*, or *Calculation Process Error(**CPE**)*. This taxonomy is inspired by prior research Zhang et al. (2025b) and refined through our statistical analysis of common physics reasoning failures. The final diagnosis is constructed as $d_i = (t_i, j_i, e_i)$, where t_i concatenates the critique t_{j_i} and the rationale t_{e_i}. Ultimately, the sequence of diagnoses, encompassing steps from the start up to the first error or the entire sequence if correct, is structured into a multi-turn dialogue format.

Data Filtering. To mitigate false negatives in error annotation, we employ Monte Carlo (MC) estimation (Wang et al., 2024). For a step s_i initially labeled incorrect, we generate $K = 4$ rollouts. The MC estimation mc_i is then calculated as:

$$mc_i = \frac{1}{K} \sum_{k=1}^{K} \mathbb{I}(a^k = a^*) \quad (4)$$

where $\mathbb{I}(\cdot)$ is the indicator function. The step is confirmed as incorrect only if none of the rollouts reach the correct answer a^* (i.e., $mc_i = 0$). We retain only those samples where the initial annotation and MC estimation are consistent.

Curriculum-based Organization. Inspired by cognitive learning patterns, we employ a hierarchical curriculum (Yan et al., 2025a) to enhance model generalization. To stratify difficulty, samples are organized across three quantitative dimensions:

1. Reasoning Depth: Reflects the contextual depth required to identify an error. We calculate this simply using the step index: $\mathcal{D}_{\text{dep}} = i/10$, where i denotes the index of the erroneous step.

[2]Full definitions and examples are detailed in Appendix A.

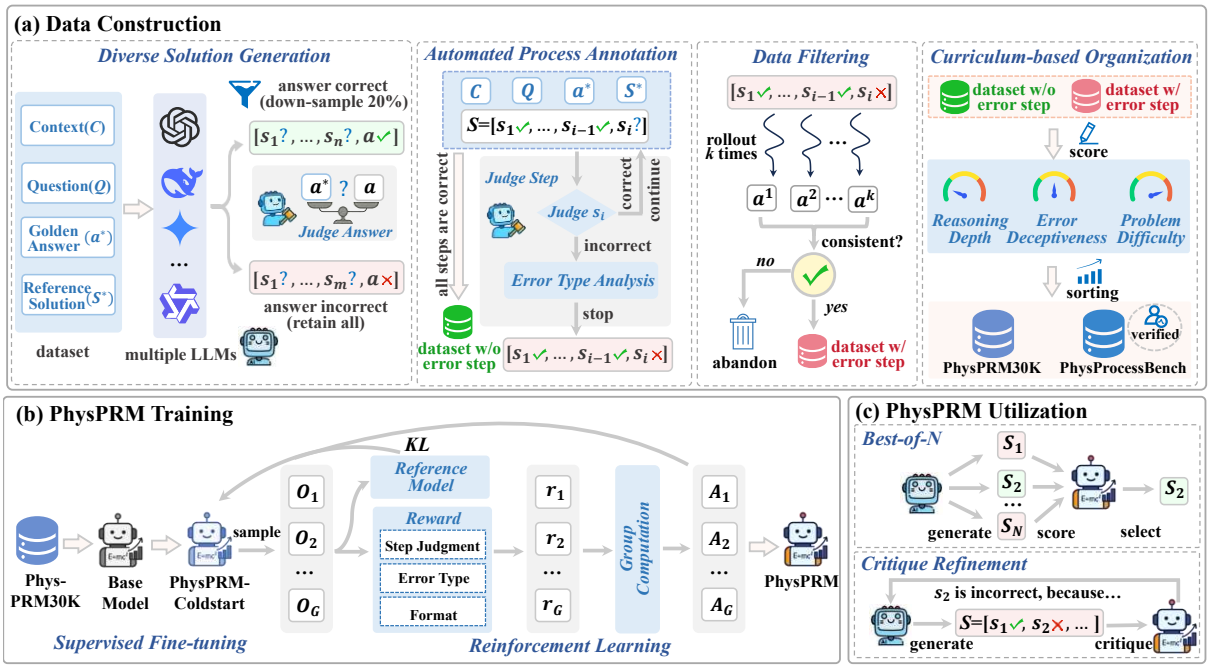


Figure 2: The overall framework of PhysPRM comprises three parts: (a) Data Construction, (b) PhysPRM Training, and (c) PhysPRM Utilization. Specifically, Data Construction involves four stages: *Diverse Solution Generation*, *Automated Process Annotation*, *Data Filtering*, and *Curriculum-based Organization*. PhysPRM Training proceeds in two stages: SFT for initialization and RL via GRPO for refinement. Finally, PhysPRM Utilization demonstrates the application of the model in both Best-of-N and Critique Refinement strategies.

2. Error Deceptiveness: Quantifies the deceptiveness of an error in inducing consistent incorrect answers. Obvious errors lead to divergent outcomes, while deceptive errors induce consistent but incorrect results. Using the MC rollouts from the *Data Filtering* phase, we calculate this metric as:

$$\mathcal{D}_{\text{dec}} = \frac{1}{K}\sum_{k=1}^{K}\mathbb{I}(a^k = a_{\text{most}}) \tag{5}$$

where a_{most} is the most frequent incorrect answer.

3. Problem Difficulty: Measures the difficulty inherent to the problem. This metric is derived from the pass rate (R_{pass}) in the *Diverse Solution Generation* phase: $\mathcal{D}_{\text{diff}} = 1 - R_{\text{pass}}$.

We normalize these metrics to $[0, 1]$ using Min-Max scaling and compute an equal-weighted score:

$$\mathcal{D} = \hat{\mathcal{D}}_{\text{dep}} + \hat{\mathcal{D}}_{\text{dec}} + \hat{\mathcal{D}}_{\text{diff}} \tag{6}$$

where $\hat{\mathcal{D}}$ denotes normalized values. Incorrect samples are sorted in ascending order of D, guiding the model from detecting simple, obvious to hard, deceptive errors, while correct samples (sorted by steps) are distributed uniformly to ensure stability.

Based on our pipeline, we construct the PhysPRM30K training dataset and the PhysProcessBench evaluation benchmark. PhysPRM30K aggregates 12K problems from eight sources, with diverse solutions generated by an ensemble of five LLMs. PhysProcessBench, derived from five test sets, serves as a gold-standard benchmark where every reasoning step has been rigorously verified by human experts. The task involves a step-level correctness assessment conditioned on the problem and the history. Comprehensive details on data sources, model configurations, and statistics are provided in Appendix B.1, B.3 and B.4.

3.2 How to Train PhysPRM?

The training of PhysPRM proceeds in two stages: Supervised Fine-Tuning (SFT) for initialization and Reinforcement Learning (RL) via Group Relative Policy Optimization (GRPO) (DeepSeek-AI, 2025) for refinement, as shown in Figure 2 (b).

Stage 1: Supervised Fine-Tuning. We first employ SFT on a uniformly sampled 60% subset of PhysPRM30K to provide the model with a high-quality initialization, serving as the *cold start* stage. This process equips the model with structured reasoning capabilities from the outset, laying a solid foundation for the subsequent RL stage.

Stage 2: Reinforcement Learning. To further refine the accuracy of step judgment and error type, while ensuring format adherence of the

diagnosis, we adopt GRPO on the remaining portion of PhysPRM30K. By estimating the baseline directly from the average reward of multiple outputs sampled per query, this mechanism optimizes PhysPRM based on relative performance within each group. To incentivize the generation of high-quality diagnoses, we formulate a composite reward function r_i consisting of three components:

1. Step Judgment Reward: Measures the accuracy of the predicted judgment ($\text{pred}_{\text{judge}}$) against the ground truth judgment (gt_{judge}) for a step.

$$r_{\text{judge}} = \mathbb{I}(\text{pred}_{\text{judge}} = \text{gt}_{\text{judge}}) \tag{7}$$

2. Error Type Reward: Evaluates the consistency between the predicted error type ($\text{pred}_{\text{type}}$) and the ground truth error type (gt_{type}) for a step.

$$r_{\text{type}} = \mathbb{I}(\text{pred}_{\text{type}} = \text{gt}_{\text{type}}) \tag{8}$$

3. Reasoning Format Reward: Verifies structural adherence to the expected reasoning format.

$$r_{\text{form}} = \mathbb{I}_{\{\text{<reasoning>},\text{<correct>},\text{<error\_type>}\} \subseteq \text{output}} \tag{9}$$

The final reward r_i is defined as:

$$r_i = r_{\text{judge}} + r_{\text{type}} + r_{\text{form}} \tag{10}$$

3.3 How to Utilize PhysPRM?

When applying Test-time Scaling (TTS) (Snell et al., 2024) for LLMs, PhysPRM serves as both a verifier and a critic to enhance the performance of LLMs in solving physics problems. Specifically, it supports the Best-of-N (BoN) strategy when acting as a verifier and critique refinement strategy when acting as a critic, as shown in Figure 2 (c).

BoN. In the BoN strategy, an LLM generates N candidate solutions. PhysPRM evaluates each step by extracting the probability of the correctness token from the corresponding diagnosis. Among the N candidates, if solutions exist where all reasoning steps are judged correct, we calculate the average step score for these solutions and select the one with the highest value. Conversely, if all N solutions contain incorrect steps, we calculate the average score of all steps for each candidate and select the solution with the highest overall score.

Critique Refinement. In the critique refinement strategy, an LLM generates a single initial solution. PhysPRM evaluates each step to identify the first erroneous step. By providing error analysis within the diagnosis as feedback, the LLM is prompted to refine and correct the solution. Finally, we verify the correctness of the newly generated solution.

4 Experiments

4.1 Setup

Benchmarks. We evaluate the training performance of PhysPRM on PhysProcessBench. Furthermore, we assess the capability of PhysPRM in BoN and critique refinement across nine diverse datasets, including PhysReason (Zhang et al., 2025b), PHYSICS (Pang et al., 2025), CMPhysBench (Wang et al., 2025a), PhysicsEval (Siddique et al., 2025), UGPhysics (Xu et al., 2025b), FormulaReasoning (Li et al., 2024), agieval-gaokao-physics (Zhong et al., 2023), MVPBench (Dong et al., 2025), and Phyx (Shen et al., 2025). These benchmarks cover difficulty levels ranging from junior high to university curricula. Notably, MVPBench and Phyx are multimodal datasets. For PhysReason, we substitute images with the provided captions to evaluate it as a text-only benchmark.

Settings. We employ PhysPRM as the verifier for BoN evaluation and set N to 8 by default. The policy model is required to generate N distinct solutions with a temperature of 0.7. For comparison, we use the average accuracy of N sets of answers generated by policy models as baselines.

Training Details. To train PhysPRM, we implement a sequential strategy on two NVIDIA A800 GPUs using LoRA. We first perform SFT on Qwen2.5-7B-Instruct (Team, 2024) for 2 epochs with a learning rate of 5e-6 as a cold start. The LoRA configuration includes a rank of 8 and alpha of 32. Following this, we refine the model through GRPO for 3 epochs. In this stage, we set the KL divergence coefficient beta to 0.04 and generate 8 completions per prompt with a temperature of 0.7.

4.2 Performance and Capability of PhysPRM

Results on PhysProcessBench. As shown in Table 2, most existing LLMs struggle to verify the correctness of individual steps. Many open-source models achieve F1 scores around 50.0, comparable to random guessing. This is because these models have a significant positive bias and tend to label most steps as correct. For example, in PhysReason subset of PhysProcessBench, Qwen2.5-7B-Instruct achieves an F1 score of 71.6 for positive steps but only 14.2 for negative ones. The large gap indicates that the model rarely identifies errors. In contrast, PhysPRM performs much better with an average F1 score of 89.0, outperforming proprietary models like Gemini-2.0-Flash and o3-mini.

Since no specialized physics PRMs existed prior

Model	PhysReason	PHYSICS	CMPhys-Bench	Physics-Eval	UGPhysics	Formula-Reasoning	agieval-gaokao-physics	Avg.
InternLM2.5-7B	14.4	15.0	6.0	21.6	12.5	26.4	37.5	19.1
+ PhysPRM	17.9	19.7	10.0	26.2	14.1	31.6	44.5	23.4
Improvement	**+3.5**	**+4.7**	**+4.0**	**+4.6**	**+1.6**	**+5.2**	**+7.0**	**+4.4**
Llama3.1-8B	15.6	17.5	17.5	30.5	15.3	46.8	30.5	24.8
+ PhysPRM	20.4	23.7	25.0	36.6	18.6	53.9	37.0	30.7
Improvement	**+4.8**	**+6.2**	**+7.5**	**+6.1**	**+3.3**	**+7.1**	**+6.5**	**+5.9**
Qwen2.5-7B-instruct	34.5	20.6	13.3	45.3	20.4	69.2	59.2	37.5
+ PhysPRM	45.2	28.9	23.0	54.2	22.8	77.9	73.0	46.0
Improvement	**+10.7**	**+8.3**	**+9.7**	**+8.9**	**+2.4**	**+8.7**	**+13.8**	**+8.9**
Qwen3-8B	39.9	26.8	25.3	57.1	31.6	74.7	71.3	46.7
+ PhysPRM	48.4	36.2	36.0	63.5	35.7	84.3	81.0	55.0
Improvement	**+8.5**	**+9.4**	**+10.7**	**+6.4**	**+4.1**	**+9.6**	**+9.7**	**+8.3**
Qwen2.5-32B-instruct	40.4	32.4	38.0	60.3	37.9	78.8	77.1	52.1
+ PhysPRM	47.0	38.5	46.0	67.5	43.7	86.9	84.0	59.1
Improvement	**+6.6**	**+6.1**	**+8.0**	**+7.2**	**+5.8**	**+8.1**	**+6.9**	**+7.0**
Qwen2.5-72B-instruct	43.5	35.6	42.3	62.8	41.3	80.4	79.3	55.0
+ PhysPRM	47.8	40.1	48.0	67.6	46.8	84.8	84.5	59.9
Improvement	**+4.3**	**+4.5**	**+5.7**	**+4.8**	**+5.5**	**+4.4**	**+5.2**	**+4.9**
Deepseek-V3.1	43.9	42.8	56.5	63.1	45.3	79.9	58.0	55.6
+ PhysPRM	48.2	45.7	62.0	68.5	50.2	84.6	69.5	61.2
Improvement	**+4.3**	**+2.9**	**+5.5**	**+5.4**	**+4.9**	**+4.7**	**+11.5**	**+5.6**
Gemini-2.0-Flash	53.9	51.2	59.0	68.9	51.8	86.4	75.6	63.8
+ PhysPRM	59.1	56.6	66.0	72.6	58.9	89.1	81.0	68.9
Improvement	**+5.2**	**+5.4**	**+7.0**	**+3.7**	**+7.1**	**+2.7**	**+5.4**	**+5.2**

Table 1: Results with BoN evaluation. Percentage accuracy (%) of multiple LLMs across seven physics datasets. By using PhysPRM as a verifier, existing LLMs achieve significant improvements in reasoning performance under the Best-of-8 setting. Only positive improvements are **bold**. All values are rounded to one decimal place.

to this work, we adapted a high-performing and established mathematical PRM, Qwen2.5-Math-PRM-7B (Zhang et al., 2025d), to serve as a strong baseline for our physics tasks. In direct verification, PhysPRM achieves an average F1 score of 89.0, which is 12.2 points higher than the 76.8 achieved by Qwen2.5-Math-PRM-7B, demonstrating that PhysPRM significantly outperforms the capable mathematical PRM across the board.

Results with BoN evaluation. As shown in Table 1, PhysPRM significantly boosts reasoning performance across various model scales and families. Specifically, for models with fewer than 10 billion parameters, including InternLM2.5-7B, Llama3.1-8B, Qwen2.5-7B-Instruct, and Qwen3-8B, the average performance improves by 4.4, 5.9, 8.9, and 8.3 points, respectively. Larger open-source models like Qwen2.5-32B-Instruct and Qwen2.5-72B-Instruct achieve gains of 7.0 and 4.9 points, while proprietary models like DeepSeek-V3.1 and Gemini-2.0-Flash improve by 5.6 and 5.2 points. These results further validate the capability of PhysPRM for TTS across different model sizes.

Results of critique refinement. As shown in Table 3, PhysPRM demonstrates superior critique capabilities compared to both the generator itself and Gemini-2.5-Flash, significantly boosting the performance of the generators. Notably, these gains scale with the number of refinement turns. Specifically, guided by PhysPRM, the generators achieve average improvements of 6.1, 8.8, and 10.3 points across the three refinement turns, respectively.

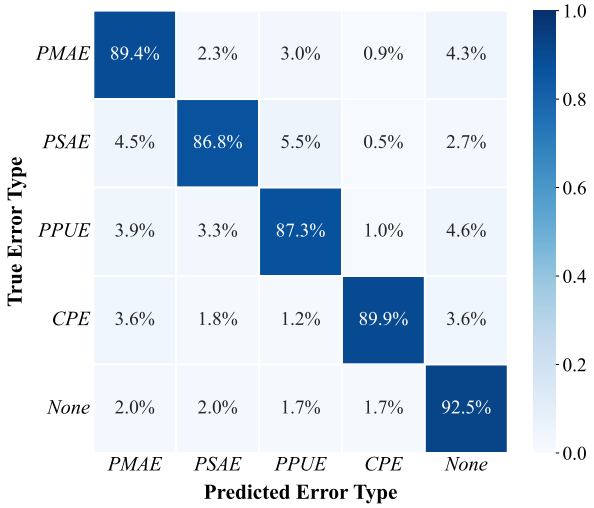


Figure 3: Confusion matrix of predicted error types versus ground truth on PhysProcessBench.

4.3 Analysis on PhysPRM Performance

Effects of diagnosis components. We analyze the impact of components by training PhysPRM under four settings ranging from judgment labels only to

Model	PhysicalEval	PHYBench	PHYSICS	PhysReason	UGPhysics	Overall
Random Guessing	50.0	50.0	50.0	50.0	50.0	50.0
Proprietary LLMs						
Deepseek-v3-1	59.6	53.2	55.4	58.9	52.6	55.9
Gemini-2.0-Flash	63.8	57.8	60.8	62.6	58.6	60.7
o3-mini	67.2	64.2	63.3	67.2	62.8	64.9
Open-source LLMs						
Qwen2.5-7B-Instruct	46.4	46.8	40.4	42.9	41.6	43.6
Qwen2.5-32B-Instruct	52.8	50.7	48.6	50.4	49.5	50.4
Qwen2.5-72B-Instruct	57.2	55.8	54.2	56.5	53.3	55.4
PRMs						
Qwen2.5-Math-PRM-7B	75.8	72.1	76.7	79.2	75.2	76.8
PhysPRM	91.1	89.7	89.0	90.4	84.8	89.0

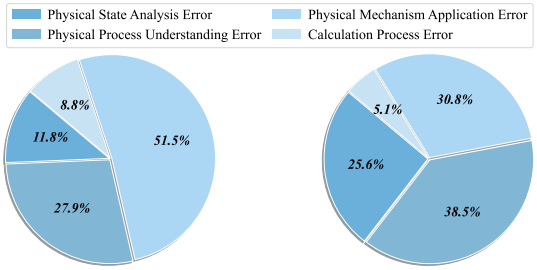
Table 2: Results on PhysProcessBench. We report the F1 scores of the correct and incorrect steps. The overall score is the average of the scores from five different data sources. The results of PhysPRM are shaded.

Critic Model	*Qwen2.5-7B-Instruct as Generator*				*Gemini-2.0-Flash as Generator*				Avg.
	PhysReason	PHYSICS	CMPhysBench	Avg.	PhysReason	PHYSICS	CMPhysBench	Avg.	
Zero-shot	34.7	20.4	13.0	22.7	53.8	49.0	56.0	52.9	37.8
Turn 1									
Generator	35.5	22.7	17.0	25.1	53.8	51.0	57.0	53.9	39.5
Gemini-2.5-Flash	39.1	24.3	19.0	27.5	54.4	53.3	58.0	55.2	41.4
PhysPRM	43.9	25.7	20.0	29.9	57.7	55.9	60.0	57.9	43.9
Turn 2									
Generator	36.1	24.0	19.0	26.4	54.0	52.6	58.0	54.9	40.7
Gemini-2.5-Flash	42.5	25.4	22.0	30.0	56.3	56.3	60.0	57.5	43.8
PhysPRM	46.8	27.3	24.0	32.7	60.3	58.2	63.0	60.5	46.6
Turn 3									
Generator	36.5	23.4	19.0	26.3	54.2	52.0	57.0	54.4	40.4
Gemini-2.5-Flash	43.7	26.6	25.0	31.8	57.9	57.6	60.0	58.5	45.2
PhysPRM	48.2	29.9	27.0	35.0	61.5	58.9	63.0	61.1	48.1

Table 3: Results of critique refinement. Accuracy (%) of the two generator models across each refinement turn using different critics: the generator itself, Gemini-2.5-Flash, and PhysPRM. The results of PhysPRM are shaded.

Training		Physics-Eval	Phys-Reason	PHYSICS
CoT	Error type			
✗	✗	62.1	69.1	64.8
✗	✓	61.3	68.9	60.4
✓	✗	85.4	83.6	83.1
✓	✓	**91.1**	**90.4**	**89.0**

Table 4: Ablation results on PhysProcessBench of PhysPRM with different diagnosis components. The best results are shown in **bold**.



(a) Qwen2.5-7B-Instruct (b) Gemini-2.0-Flash

Figure 4: Distribution of successfully refined error categories across three successive turns on the PhysReason dataset, with PhysPRM acting as the critic for generators (a) Qwen2.5-7B-Instruct and (b) Gemini-2.0-Flash.

the full diagnosis. As shown in Table 4, our evaluation on PhysProcessBench highlights that CoT is the primary contributor to performance improvement. Furthermore, although generating error types enhances process verification, it requires the presence of CoT. Incorporating error types in isolation actually reduces model performance.

To better understand the role of error types, we evaluate the classification performance of PhysPRM on PhysProcessBench, as shown in Figure 3. The accuracy for all categories exceeds 86%. Notably, the model demonstrates high recall for correct steps and high precision for incorrect steps. This indicates that while the model might miss some subtle errors, the errors it identifies are highly

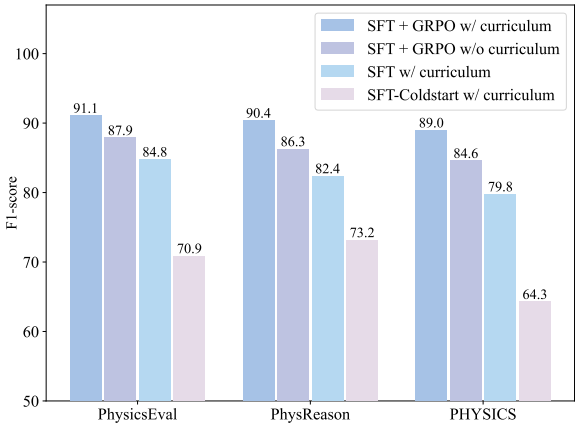


Figure 5: Results on PhysProcessBench of PhysPRM with different training strategies.

Model	Phyx	MVPBench	Avg.
Large Language Models			
Qwen2.5-7B	35.2	20.9	28.1
+PhysPRM	39.4	28.0	33.7
Improvement	**+4.2**	**+7.1**	**+5.6**
Qwen2.5-32B	67.2	31.5	49.4
+PhysPRM	70.0	35.4	52.7
Improvement	**+2.8**	**+3.9**	**+3.3**
Qwen2.5-72B	66.4	33.1	49.8
+PhysPRM	68.8	37.3	53.1
Improvement	**+2.4**	**+4.2**	**+3.3**
Multimodal Large Language Models			
Qwen2.5-7B-VL	30.0	16.7	23.4
+PhysPRM	35.4	22.2	28.8
Improvement	**+5.4**	**+5.5**	**+5.4**
Qwen2.5-32B-VL	43.4	33.4	38.4
+PhysPRM	46.8	37.3	42.1
Improvement	**+3.4**	**+3.9**	**+3.7**
Qwen2.5-72B-VL	46.8	37.0	41.9
+PhysPRM	49.4	43.1	46.3
Improvement	**+2.6**	**+6.1**	**+4.4**

Table 5: Results on visual physics benchmarks. Under Best-of-8 evaluation, PhysPRM enhances visual reasoning performance across both LLMs and MLLMs.

reliable. This characteristic aligns well with our core design objective for the PRM.

Figure 4 shows the distribution of error types resolved during three-turn refinement on PhysReason. For Qwen2.5-7B-Instruct, PhysPRM primarily addresses *Physical Mechanism Application Error* (fundamental errors). Conversely, Gemini-2.0-Flash shows balanced improvements, notably in *Physical Process Understanding* and *State Analysis Errors*, reflecting corrections in complex reasoning.

Effects of training strategies. We analyze the impact of various training strategies, including different combinations of SFT, GRPO, and curriculum. As shown in Figure 5, combining SFT with GRPO yields the best results, validating the effectiveness

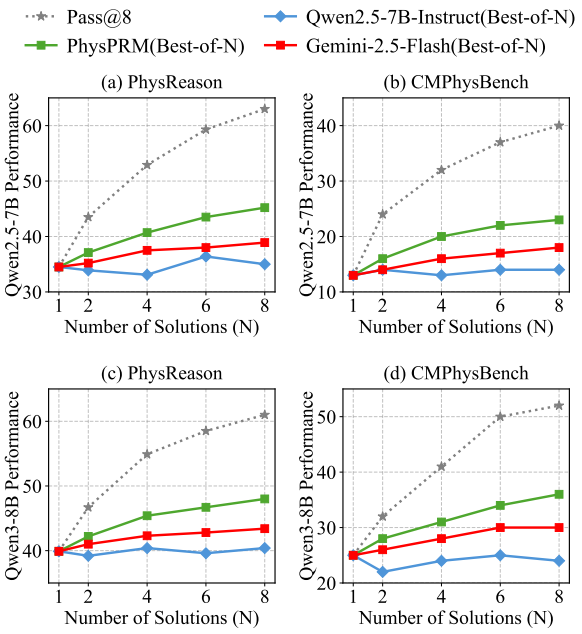


Figure 6: BoN results on the PhysReason and CMPhysBench benchmarks across different policy models and verifiers. PhysPRM consistently enhances reasoning performance as N increases, outperforming the improvements achieved by the Gemini-2.5-Flash verifier.

of this training paradigm. Moreover, applying curriculum learning to organize the data boosts performance, resulting in an average gain of 3.9 points on the three subsets in PhysProcessBench.

4.4 Analysis on PhysPRM Capability

Effects of BoN. Following the TTS technique, we vary the number of solutions N in the BoN strategy to evaluate PhysPRM against Qwen2.5-7B-Instruct and Gemini-2.5-Flash as verifiers. As shown in Figure 6, increasing N consistently boosts the reasoning performance of both Qwen2.5-7B and Qwen3-8B policy models. Specifically, on PhysReason, PhysPRM outperforms Gemini-2.5-Flash verifier by 1.6, 3.2, 4.7, and 5.5 points across the Best-of-2, 4, 6, and 8 settings, respectively. These results indicate that the performance gap between PhysPRM and other verifiers widens as N increases.

Results on visual physics problems. As shown in Table 5, we evaluate the Qwen2.5 series on Phyx and MVPBench under the Best-of-8 setting, finding that PhysPRM consistently improves visual reasoning performance. Specifically, for the Qwen2.5-Instruct series of 7B, 32B, and 72B, the average performance improves by 5.6, 3.3, and 3.3 points, respectively. Similarly, multimodal large language models including the Qwen2.5-VL series of 7B, 32B, and 72B achieve average gains of 5.4, 3.7,

and 4.4 points. These results validate the effectiveness of PhysPRM in visual scenarios.

5 Conclusion

We introduce PhysPRM, a generative PRM that transforms simple process scoring into fine-grained diagnoses containing detailed critiques, step judgments, and specific error types. To facilitate this, we design an automated synthesis pipeline to construct the PhysPRM30K dataset and PhysProcess-Bench benchmark. By employing a two-stage training paradigm that integrates SFT with GRPO, we demonstrate that PhysPRM significantly enhances the capabilities of various LLMs across seven datasets in both Best-of-N and critique refinement. Ultimately, our findings highlight the effectiveness of fine-grained supervision in addressing complex physical constraints, offering a robust path toward more reliable physics problem solving.

6 Acknowledgments

This work was supported by Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China (JYB2025XDXM116), National Natural Science Foundation of China (No. 62137002, 62293550, 62293553, 62293554, 62437002, 62477036, 62477037, 62192781), the Shaanxi Provincial Social Science Foundation Project (No. 2024P041), the Youth Innovation Team of Shaanxi Universities "Multi-modal Data Mining and Fusion", and Xi'an Jiaotong University City College Research Project (No. 2024Y01).

7 Limitations

Despite the strong performance of PhysPRM across various benchmarks, we identify two primary limitations regarding inference efficiency and optimization strategies. First, the reliance on generative feedback introduces additional computational overhead during inference. Unlike simple scoring models, PhysPRM generates a fine-grained diagnosis for each step, which requires more resources. Second, our current framework exclusively employs the Group Relative Policy Optimization algorithm for the reinforcement learning stage. We have not yet fully explored how other advanced algorithms might further benefit our approach. In future work, we aim to investigate methods to dynamically prune the reasoning process to improve speed and explore alternative optimization strategies to enhance robustness.

8 Ethical Statement

In developing PhysPRM, we have carefully considered the ethical implications of our research, particularly regarding data compliance and responsible AI usage. Our experiments use publicly available physics datasets and models (including the Qwen series and Gemini), and we strictly follow their licensing agreements and usage policies. To support future research, we will publicly release the complete datasets, models, and scripts under appropriate licenses (MIT and CC BY-NC-SA). Additionally, we used LLMs to help check grammarand improve the code structure of this submission.

References

Hongzhan Chen, Tao Yang, Shiping Gao, Ruijun Chen, Xiaojun Quan, Hongtao Tian, and Ting Yao. 2025a. Discriminative policy optimization for token-level reward models. *arXiv preprint arXiv:2505.23363*.

Jiacheng Chen, Qianjia Cheng, Fangchen Yu, Haiyuan Wan, Yuchen Zhang, Shenghe Zheng, Junchi Yao, Qingyang Zhang, Haonan He, Yun Luo, and 1 others. 2025b. P1: Mastering physics olympiads with reinforcement learning. *arXiv preprint arXiv:2511.13612*.

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*.

Nifu Dan, Yujun Cai, and Yiwei Wang. 2025. Symbolic or numerical? understanding physics problem solving in reasoning llms. *arXiv preprint arXiv:2507.01334*.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Zhuobai Dong, Junchao Yi, Ziyuan Zheng, Haochen Han, Xiangxi Zheng, Alex Jinpeng Wang, Fangming Liu, and Linjie Li. 2025. Seeing is not reasoning: Mvpbench for graph-based evaluation of multi-path visual physical cot. *arXiv preprint arXiv:2505.24182*.

Muhammad Khalifa, Rishabh Agarwal, Lajanugen Logeswaran, Jaekyeom Kim, Hao Peng, Moontae Lee, Honglak Lee, and Lu Wang. 2025. Process reward models that think. *arXiv preprint arXiv:2504.16828*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Xiao Li, Bolin Zhu, Kaiwen Shi, Sichen Liu, Yin Zhu, Yiwei Liu, and Gong Cheng. 2024. Formulareasoning: A dataset for formula-based numerical reasoning. *arXiv preprint arXiv:2402.12692*.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let's verify step by step. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Meiqi Guo, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, and 1 others. 2024. Improve mathematical reasoning in language models by automated process supervision. *arXiv preprint arXiv:2406.06592*.

Yubo Ma, Zhibin Gou, Junheng Hao, Ruochen Xu, Shuohang Wang, Liangming Pan, Yujiu Yang, Yixin Cao, Aixin Sun, Hany Awadalla, and 1 others. 2024. Sciagent: Tool-augmented language models for scientific reasoning. *arXiv preprint arXiv:2402.11451*.

Tej Deep Pala, Panshul Sharma, Amir Zadeh, Chuan Li, and Soujanya Poria. 2025. Error typing for smarter rewards: Improving process reward models with error-aware hierarchical supervision. *arXiv preprint arXiv:2505.19706*.

Xinyu Pang, Ruixin Hong, Zhanke Zhou, Fangrui Lv, Xinwei Yang, Zhilong Liang, Bo Han, and Changshui Zhang. 2025. Physics reasoner: Knowledge-augmented reasoning for solving physics problems with large language models. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 11274–11289.

Md Imbesat Hassan Rizvi, Xiaodan Zhu, and Iryna Gurevych. 2025. Spare: Single-pass annotation with reference-guided evaluation for automatic process supervision and reward modelling. *arXiv preprint arXiv:2506.15498*.

Hui Shen, Taiqiang Wu, Qi Han, Yunta Hsieh, Jizhou Wang, Yuyue Zhang, Yuxin Cheng, Zijian Hao, Yuansheng Ni, Xin Wang, and 1 others. 2025. Phyx: Does your model have the" wits" for physical reasoning? *arXiv preprint arXiv:2505.15929*.

Oshayer Siddique, JM Alam, Md Jobayer Rahman Rafy, Syed Rifat Raiyan, Hasan Mahmud, and Md Kamrul Hasan. 2025. Physicseval: Inference-time techniques to improve the reasoning proficiency of large language models on physics problems. *arXiv preprint arXiv:2508.00079*.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling LLM test-time compute optimally can be more effective than scaling model parameters. *CoRR*, abs/2408.03314.

Lin Sun, Chuang Liu, Xiaofeng Ma, Tao Yang, Weijia Lu, and Ning Wu. 2025. Freeprm: Training process reward models without ground truth process labels. *arXiv preprint arXiv:2506.03570*.

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

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2024. 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), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 9426–9439. Association for Computational Linguistics.

Weida Wang, Dongchen Huang, Jiatong Li, Tengchao Yang, Ziyang Zheng, Di Zhang, Dong Han, Benteng Chen, Binzhao Luo, Zhiyu Liu, and 1 others. 2025a. Cmphysbench: A benchmark for evaluating large language models in condensed matter physics. *arXiv preprint arXiv:2508.18124*.

Weiyun Wang, Zhangwei Gao, Lianjie Chen, Zhe Chen, Jinguo Zhu, Xiangyu Zhao, Yangzhou Liu, Yue Cao, Shenglong Ye, Xizhou Zhu, and 1 others. 2025b. Visualprm: An effective process reward model for multimodal reasoning. *arXiv preprint arXiv:2503.10291*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Fangzhi Xu, Qika Lin, Jiawei Han, Tianzhe Zhao, Jun Liu, and Erik Cambria. 2025a. Are large language models really good logical reasoners? a comprehensive evaluation and beyond. *IEEE Transactions on Knowledge and Data Engineering*.

Xin Xu, Qiyun Xu, Tong Xiao, Tianhao Chen, Yuchen Yan, Jiaxin Zhang, Shizhe Diao, Can Yang, and Yang Wang. 2025b. Ugphysics: A comprehensive benchmark for undergraduate physics reasoning with large language models. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*. OpenReview.net.

Bi-Cheng Yan, Hsin-Wei Wang, Fu-An Chao, Tien-Hong Lo, Yung-Chang Hsu, and Berlin Chen. 2025a. Hippo: Exploring a novel hierarchical pronunciation assessment approach for spoken languages. *arXiv preprint arXiv:2512.04964*.

Yibo Yan, Jiamin Su, Jianxiang He, Fangteng Fu, Xu Zheng, Yuanhuiyi Lyu, Kun Wang, Shen Wang, Qingsong Wen, and Xuming Hu. 2025b. A survey

of mathematical reasoning in the era of multimodal large language model: Benchmark, method & challenges. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 11798–11827, Vienna, Austria. Association for Computational Linguistics.

Jian Zhang, Zhangqi Wang, Haiping Zhu, Kangda Cheng, Kai He, Bo Li, Qika Lin, Jun Liu, and Erik Cambria. 2026a. Mars: Multi-agent adaptive reasoning with socratic guidance for automated prompt optimization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(19):16307–16315.

Jian Zhang, Zhiyuan Wang, Zhangqi Wang, Fangzhi Xu, Qika Lin, Lingling Zhang, Rui Mao, Erik Cambria, and Jun Liu. 2026b. Maps: Multi-agent personality shaping for collaborative reasoning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(19):16316–16324.

Jianghangfan Zhang, Yibo Yan, Kening Zheng, Xin Zou, Song Dai, and Xuming Hu. 2025a. Gm-prm: A generative multimodal process reward model for multimodal mathematical reasoning. *arXiv preprint arXiv:2508.04088*.

Xinyu Zhang, Yuxuan Dong, Yanrui Wu, Jiaxing Huang, Chengyou Jia, Basura Fernando, Mike Zheng Shou, Lingling Zhang, and Jun Liu. 2025b. PhysReason: A comprehensive benchmark towards physics-based reasoning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16593–16615, Vienna, Austria. Association for Computational Linguistics.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025c. The lessons of developing process reward models in mathematical reasoning. *arXiv preprint arXiv:2501.07301*.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025d. The lessons of developing process reward models in mathematical reasoning. *arXiv preprint arXiv:2501.07301*.

Jian Zhao, Runze Liu, Kaiyan Zhang, Zhimu Zhou, Junqi Gao, Dong Li, Jiafei Lyu, Zhouyi Qian, Biqing Qi, Xiu Li, and 1 others. 2025. Genprm: Scaling test-time compute of process reward models via generative reasoning. *arXiv preprint arXiv:2504.00891*.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. 2023. Agieval: A human-centric benchmark for evaluating foundation models. *Preprint*, arXiv:2304.06364.

Erle Zhu, Yadi Liu, Zhe Zhang, Xujun Li, Jin Zhou, Xinjie Yu, Minlie Huang, and Hongning Wang. 2025. MAPS: advancing multi-modal reasoning in expert-level physical science. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

A Error Type Details

We categorize reasoning errors into four distinct types, with detailed definitions and examples provided below.

Physical Mechanism Application Error occurs when physical laws, principles, or formulas are applied incorrectly. This represents a misunderstanding of *why* a principle applies. The solver might recall the law but use it in conditions where it is invalid.

- Applying Newton's Laws in a non-inertial reference frame without considering fictitious forces.
- Using conservation of energy in systems where friction does work.
- Using formulas outside their valid range (e.g., using small-angle approximations for large angles).
- Misremembering the specific conditions required for a law to hold true.

Physical State Analysis Error relates to the initial setup and analysis of the system. This error type reflects a misunderstanding of *what* exists in the scenario. It involves incorrect assessments of system boundaries, forces, or objects.

- Neglecting significant forces, such as friction in a non-ideal system.
- Incorrectly defining the system boundary, leading to errors in conservation laws.
- Failing to distinguish between isolated and open systems.
- Missing or adding non-existent forces acting on an object.

Physical Process Understanding Error stems from a misunderstanding of *how* the phenomenon evolves over time. Unlike state analysis, this error concerns the dynamic behavior and cause-and-effect relationships.

- Misunderstanding projectile motion (e.g., thinking horizontal and vertical motions affect each other).
- Confusing different forms of energy transformation.
- Mispredicting the direction of motion based on forces.
- Holding misconceptions, such as believing a continuous force is needed to maintain constant velocity.

Calculation Process Error is purely mathematical mistakes that happen during the execution phase. These errors are independent of the physical reasoning logic.

- Making algebraic errors when rearranging equations.
- Incorrect unit conversions (e.g., mixing meters and centimeters).
- Simple arithmetic mistakes in addition or multiplication.
- Copying numbers incorrectly or calculator usage errors.

B Dataset Details

B.1 Sources of Training Data

As shown in Table 6, the initial training data consists of 12,508 samples collected from eight different datasets. These problems cover various difficulty levels, including middle-school, high-school, and university physics. Specifically, for the PhysReason dataset, we decompose multi-part questions into separate samples, ensuring that each entry contains only a single question.

Benchmark	Split Strategy	# Samples
high-school-physics	Full	400
pocket-physics	Full	1,000
FormulaReasoning	Train	1,000
PhysReason	Non-test Samples	2,608
UGPhysics	Non-test Samples	1,600
PHYSICS	Non-test Samples	1,400
PhysicsEval	Non-test Samples	2,500
Phyx	Non-test Samples	2,000

Table 6: Detailed statistics and split strategies for the initial training data sources.

B.2 PhysPRM30K

As summarized in Table 7, PhysPRM30K comprises 30,679 samples. The dataset is constructed by aggregating physics problems from eight distinct sources, as detailed in Table 6. To generate process annotations, we leverage our automated data synthesis pipeline incorporating seven models: Qwen3-8B, Qwen2.5-32B-Instruct, Qwen2.5-72B-Instruct, GPT-4o, DeepSeek-V3-1, Gemini-2.0-Flash, and o3-mini.

Statistics	# Samples
Total Samples	30,679
- PhysReason	10,800
- PhysicsEval	6,407
- Phyx	4,402
- UGPhysics	3,241
- PHYSICS	2,332
- FormulaReasoning	1,584
- pocket-physics	1,349
- high-school-physics	564
Error Type Distribution	30,679
- Physical Mechanism Application	6,304
- Physical State Analysis	4,674
- Physical Process Understanding	4,620
- Calculation Process	4,149
- None	10,932
Step Count Distribution	30,679
- Short (1-4 steps)	6,640
- Medium (5-7 steps)	10,544
- Long (8-10 steps)	13,495

Table 7: Statistics of PhysPRM30K.

B.3 Benchmark

Table 8 provides more details regarding the BoN and critique refinement test benchmarks.

Benchmark	Split Strategy	# Samples
agieval-gaokao-physics	Full	200
FormulaReasoning	Test	421
PhysReason	Test-mini	504
CMPhysBench	With Answers	100
PHYBench	With Answers	100
PHYSICS	Random Selection	304
PhysicsEval	Random Selection	587
UGPhysics	Random Selection	263
MVPBench	Physics Subset	311
Phyx	Random Selection	500

Table 8: Detailed statistics of the evaluation benchmarks used for BoN and critique refinement testing. "Random Selection" indicates that a subset of samples is randomly chosen from the original dataset to serve as the test set.

B.4 PhysProcessBench

As summarized in Table 9, PhysProcessBench consists of 1,962 samples. To construct this benchmark, we collect problems from five datasets listed in Table 8: PhysicsEval, PHYBench, PHYSICS, PhysReason, and UGPhysics. We use our automated data synthesis pipeline along with three models, including Qwen3-32B, Qwen2.5-72B-Instruct, and DeepSeek-V3-1, to generate process annotation samples. The final samples are then manually checked and filtered to ensure high quality.

Statistics	# Samples
Total Samples	1962
- PhysicsEval	472
- PHYBench	154
- PHYSICS	265
- PhysReason	760
- UGPhysics	311
Error Type Distribution	1962
- Physical Mechanism Application	529
- Physical State Analysis	342
- Physical Process Understanding	306
- Calculation Process	169
- None	616
Source Solutions	1962
- Qwen3-32B	744
- Qwen2.5-72B-Instruct	610
- Deepseek-v3.1	608

Table 9: Statistics of PhysProcessBench.

C Experimental Details

C.1 Prompt in Data Synthesis

We describe the prompts used in our automated data synthesis pipeline to generate reasoning processes and error type analyses. The prompt used to guide Gemini-2.5-Flash in evaluating step correctness and its reasoning process is shown in Figure 7. Additionally, the prompt for generating detailed error type analysis is presented in Figure 8.

C.2 Prompt in Experiments

We present the prompts used in the critique refinement experiment to guide Gemini-2.5-Flash and Qwen2.5-7B-Instruct in generating critiques, as shown in Figure 9.

D Efficiency Analysis

In terms of computational efficiency, PhysPRM exhibits a trade-off compared to discriminative PRM models. Specifically, compared to the Qwen2.5-Math-PRM-7B discriminative model, PhysPRM incurs higher inference latency, requiring approximately 15.6 seconds per step versus 0.5 seconds for the discriminative model. Additionally, PhysPRM generates an average of 238 tokens per step to produce detailed diagnostic feedback. However, this increase in computational cost is accompanied by significant improvements in diagnostic capability.

E Cases

To provide a comprehensive understanding of PhysPRM, we present two representative cases from the critique refinement process in Figures 10 and 11. The outputs generated by PhysPRM comprise both reflections on errors. By systematically organizing accumulated correction experiences according to error types, we can efficiently retrieve and apply the most relevant successful modifications when encountering similar errors in subsequent tasks.

Prompt for Step Judgement

[System]:
You are a physics teacher. Your task is to reason whether a single step in a solution is correct.

[User]:
The materials you will need are provided below.

Context and Question:
{context}

The steps of a correct solution:
{gt_steps}

Previous correct steps:
{llm_steps}

Current step to evaluate:
{cur_step}

[Your Task]:
For the current step only:

1. Understand the overall correct solution path and the logic of the correct approach. Keep in mind that this may not be the only valid solution.

2. Break down what the current step is trying to accomplish. Identify the physical principles and mathematical reasoning involved. If needed, refer to previous steps to gain a better understanding of the current step. If the current step doesn't involve any calculation or reasoning but simply restates the problem, judge it as "correct" and provide simple reasoning indicating that the step is a restatement of the problem.

3. Decide whether this step is correct or incorrect based on your reasoning process, not merely by matching it to the reference solution. If incorrect, provide the correct step in reasoning process.

4. Provide a reasoning process, then conclude with a verdict ("correct" or "incorrect") in JSON format.

[Response Format]:
Output the following JSON:
{
"reasoning": "Your reasoning process here, based on physical and mathematical reasoning. Escape all quotes and special characters as needed.",
"correct_verdict": "correct | incorrect"
}

Figure 7: Prompt for step-level correctness evaluation. This prompts used to guide the model to evaluate the correctness of each reasoning step and provide a detailed critique.

Prompt for Error Analysis

[System]:
*You are a physics teacher. Your task is to determine the **type of error** in a current solution step.*

[User]:
The materials you will need are provided below.

Context and Question:
{context}

The steps of a correct solution:
{gt_steps}

Previous correct steps:
{llm_steps}

Current step to evaluate:
{cur_step}

Correctness verdict:
The step is {correct_verdict}.

Explanation:
{correct_explanation}

[Your Task]:
For the current step only:
1. Understand the "Error Type Categories" provided below.
2. Reason which error type applies to the current step.
3. Identify the error type for the current step from the following categories in JSON format.

Error Type Categories

1. Physical State Analysis Errors: Errors related to the incorrect assessment of the physics system's boundaries, the forces acting on it, or its constituent components. Identifying incomplete, excessive, or incorrect components such as forces and energy within the system. This involves a misunderstanding of 'what' is happening in the system. Examples include neglecting significant friction, incorrectly defining system boundaries leading to errors in conservation laws, misjudging whether a system is isolated, failing to account for all relevant forces on an object, or misidentifying interacting components.

2. Physical Process Understanding Errors: Errors stemming from a misunderstanding of how a physics phenomenon develops, how states change, or the causal relationships between events. Identifying incorrect target states or erroneous motion processes. This involves a misunderstanding of 'how' things are happening. Examples include misunderstanding the motion of a projectile, confusing energy transformation mechanisms, mispredicting motion direction based on forces, or having misconceptions about the nature of a physics process (e.g., believing a continuous force is needed for constant velocity).

3. Physical Theorem Application Errors: Errors arising from the incorrect application of physics theorems, principles, laws, concepts, formulas or using them in situations where they are not valid. This includes both misremembering the law itself and misapplying a correctly remembered law. This involves a misunderstanding of 'why' a principle applies. Examples include applying Newton's Laws of Motion to a non-inertial reference frame without considering fictitious forces, using the conservation of energy in systems with significant non-conservative forces like friction, applying formulas outside their valid range (e.g., using small-angle approximations for large angles), and misunderstanding the conditions under which a law applies.

4. Calculation Process Errors: Errors occurring during the mathematical manipulation of equations, the derivation of formulas, or the substitution of numerical values. These are purely mathematical mistakes. Examples include making algebraic errors when rearranging equations, performing incorrect unit conversions (e.g., mixing meters with centimeters), making arithmetic mistakes (e.g., addition or multiplication errors), incorrectly substituting values into formulas, and errors when using a calculator.

[Response Format]:
Output the following JSON:

```
{
    "error_analysis": "Reason the useful cause of the error briefly, pointing out what the step misunderstood or did incorrectly.",
    "error_type": "State the specific type of error (choose from the provided categories)"
}
```

Figure 8: Prompt for error type classification. This prompt is used to guide the model in identifying and categorizing specific physical reasoning errors based on the provided categories.

Prompt for Critique Generation

[System]:
You are a physics teacher. Your task is to review and critique a current solution step by step. Pay attention that you should not solve the problem or give the final answer. All of your task is to critique.

[User]:
The materials you will need are provided below.

Question:
{question}

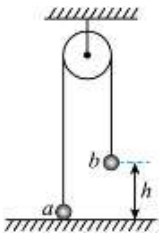
Solution:
<step_1> {solution_step_1} </step_1>
<step_2> {solution_step_2} </step_2>
...
<step_n> {solution_step_n} </step_n>

[Your Task]:
For each step of the solution:
1. Review the reasoning and calculation.
2. Identify any errors, omissions, or conceptual misunderstandings.
3. Provide a critique, focusing on the physical reasoning, mathematical modeling, or conceptual correctness.
4. At the end of each step's critique, output your judgment in the following format: "Yes" if the step is correct, or "No" if it contains errors.

[Response Format]:
Output the following JSON:
```
{
    "step_index": "step_idx",
    "critique": "Your critique here.",
    "verdict": "Yes|No"
}
```

Figure 9: Prompt for critique generation. This prompt is used to guide the model in reviewing and critiquing a solution step by step based on physical reasoning.

An inextensible, soft, and light rope passes over a smooth fixed pulley, with small spheres a and b attached to each end of the rope. Sphere a has a mass of 1 kg and is at rest on the ground; sphere b has a mass of 3 kg and is held by a hand at a height of h = 2.5m, at which point the rope is just taut. Starting from rest, sphere b is released. Neglecting air resistance and the size of the pulley, and taking the gravitational acceleration g as 10m/s^2, determine: What is the maximum height that the ball a can reach?



visual demo

Original Solution

Step 1: Sphere a: Mass = 1 kg, initially at rest. Sphere b: Mass = 3 kg, initial height = 2.5 m. Gravity force: Fa = 10 N down, Fb = 30 N down. ✓

Step 2: Using Newton's second law: For sphere a: T - 10 = a. For sphere b: 30 - T = 3a. Solving gives system acceleration a = 5 m/s^2. ✓

Step 3: When sphere b reaches the ground, all of its initial potential energy will have been converted into kinetic energy and then into the potential energy of sphere a... ✗

PhysPRM Fine-grained Diagnosis:
<reasoning>Step incorrectly assumes all of sphere b's lost potential energy converts to sphere a's potential energy, ignoring kinetic energy of both spheres at the moment b hits ground...**</reasoning>**
<correct>
The step is \boxed{false}.
</correct>
<error_type>
Physical Process Understanding Error
</error_type>

correct

Corrected Solution

Step 3: Balls a and b start from rest with ac celeration a = 5 m/s^2 over s = h = 2.5 m. Kinematics: v^2 = v_0^2 + 2as, v_0 = 0. v ^2 = 0 + 2 ×5 × 2.5 = 25, v = 5 m/s. At thi s instant, ball a is at height 2.5 m with upw ard velocity 5 m/s. ✓

Step 4: After ball b lands, rope slackens; ball a becomes an upward projectile with initial velocity v = 5 m/s. Additional height h' from energy conservation or kinematics: 0^2 = 5^2 + 2 ×(-10) × h', 0 = 25 - 20h', h' = 25 / 20 = 1.25 m. ✓

Step 5: Maximum total height H of ball a is sum: H = h + h' = 2.5 m + 1.25 m = 3.75 m. Final answer: 3.75m. ✓

Figure 10: The first case of PhysPRM.

A small cup with a mass of $0.5\\mathrm{kg}$ contains $1.5\\mathrm{kg}$ of water. The cup is tied with a rope and performs a \"water meteor\" performance in a vertical plane, with a rotation radius of $1\\mathrm{m}$. The speed of the cup at the highest point is $4\\mathrm{m/s}$, and ${\\bf g}$ is taken as $10\\mathrm{m}/\\mathrm{s}^{2}$. What is the pressure of the water on the bottom of the small cup at the highest point?

visual demo

Original Solution

Step 1: Determine the water's weight. Given: water mass = 1.5 kg, gravity = 10 m/s². Water weight = 1.5 kg × 10 m/s² = 15 N. ✓

Step 2: Determine the total force exerted on the bottom of the cup by the water, which is stated as the sum of the weight of the water and the calculated centripetal force.. F_{total} = W_{water} + F_{centripetal}... ✗

PhysPRM Fine-grained Diagnosis:

<reasoning>Step incorrectly attempts to find the normal force (F_N) by adding the weight of the water (W_{water}) to the total centripetal force for the system. The correct approach requires analyzing the water alone: the net force on the water provides its centripetal acceleration, meaning one must apply F_N + W_{water}=F_{centripetal}...
</reasoning>
<correct>The step is \boxed{false}.
</correct>
<error_type>
Physical State Analysis Error
</error_type>

correct

Corrected Solution

Step 2: Determine the total force exerted o n the bottom of the cup by the water, F_N + W_{water}=F_{centripetal}$. ✓

Step 3: The centripetal force required for the water's circular motion at the top point is: F{centripetal} = m_w \frac{v^2}{R} = 1.5×\frac{4^2}{1} = 1.5×16 = 24 N. This centripetal force is directed downward, toward the center of the circle. ✓

Step 4: F_N = F_{centripetal}$ - W_{wat er}=24 – 15 = 9 N. ✓

Step 5: According to Newton's third law, t he pressure exerted by the water on the bott om of the cup is $9 \\mathrm{N}$, directe d vertically upward. ✓

Figure 11: The second case of PhysPRM.