Retrieval-Augmented Process Reward Model for Generalizable Mathematical Reasoning

Jiachen Zhu[1], **Congmin Zheng**[1], **Jianghao Lin**[1], **Kounianhua Du**[1]
Ying Wen[1*], **Yong Yu**[1], **Jun Wang**[2], **Weinan Zhang**[1*]
[1]Shanghai Jiao Tong University, [2]University College London
{gebro13,desp.zcm,chiangel,kounianhuadu,ying.wen,wnzhang}@sjtu.edu.cn,
yyu@apex.sjtu.edu.cn
jun.wang@cs.ucl.ac.uk

Abstract

While large language models (LLMs) have significantly advanced mathematical reasoning, Process Reward Models (PRMs) have been developed to evaluate the logical validity of reasoning steps. However, PRMs still struggle with out-of-distribution (OOD) challenges. This paper identifies key OOD issues, including step OOD—caused by differences in reasoning patterns across model types and sizes—and question OOD, which arises from dataset shifts between training data and real-world problems. To address these issues, we introduce Retrieval-Augmented Process Reward Model (Retrieval-PRM), a novel framework designed to tackle these OOD issues. By utilizing a two-stage retrieval-enhanced mechanism, RetrievalPRM retrieves semantically similar questions and steps as a warmup, enhancing PRM's ability to evaluate target steps and improving generalization and reasoning consistency across different models and problem types. Our extensive experiments demonstrate that RetrievalPRM outperforms existing baselines across multiple real-world datasets. Our open-source contributions include a retrieval-enhanced dataset, a tuning framework for PRM training, and the RetrievalPRM model, establishing a new standard for PRM performance.

1 Introduction

While large language models (LLMs) have advanced mathematical reasoning (OpenAI, 2023; Dubey et al., 2024; Zhu et al., 2024; Shao et al., 2024; Yang et al., 2024b), they remain prone to critical flaws: explicit errors (e.g., miscalculations, logical inconsistencies) and implicit risks where correct answers mask flawed intermediate steps. Even when final results are accurate, LLMs often generate plausible-but-incorrect reasoning chains, eroding trust in their problem-solving processes (Lightman et al., 2023). To address this, Process Reward Models (PRMs) (Lightman et al., 2023; Wang et al., 2024b) have been developed to rigorously evaluate the logical validity of intermediate steps (Cobbe et al., 2021), mirroring human pedagogical practices that prioritize reasoning quality over answer correctness.

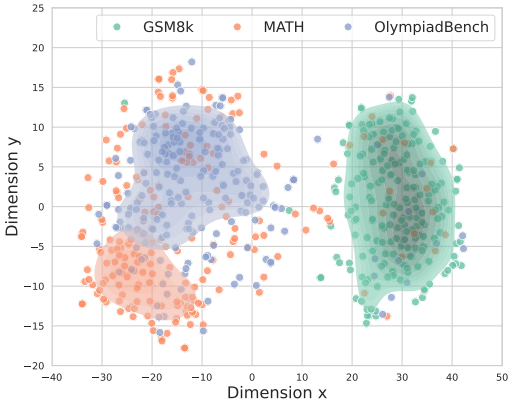

Figure 1: The distribution differences across three datasets: GSM8K, MATH and Olympiad. We use sentence-bert to encode these questions and perform t-sne visualization.

Existing works (Wang et al., 2024a; o1 Team, 2024; Zheng et al., 2024) frame PRM as a binary classification problem. They train PRM on open-source base LLMs such as Qwen (Yang et al., 2024b) or Llama (Dubey et al., 2024) using human-annotated dataset (Lightman et al., 2023) or automated process supervision method (Wang et al., 2024b; Luo et al., 2024; Qin et al., 2024). Although these approaches show great performance and empirical success, they still face kinds of out-of-distribution challenges. We believe the out-of-distribution (OOD) problem can be viewed from the following perspectives:

Firstly, **Step OOD** may occur because of different processes generated by different models. Due to the high cost of manual annotation, there are very few accurately labeled PRM expert datasets, such as PRM800K and ProcessBench, with processes generated by GPT (OpenAI, 2023) and

*Corresponding author

Findings of the Association for Computational Linguistics: ACL 2025, pages 8453–8468
July 27 - August 1, 2025

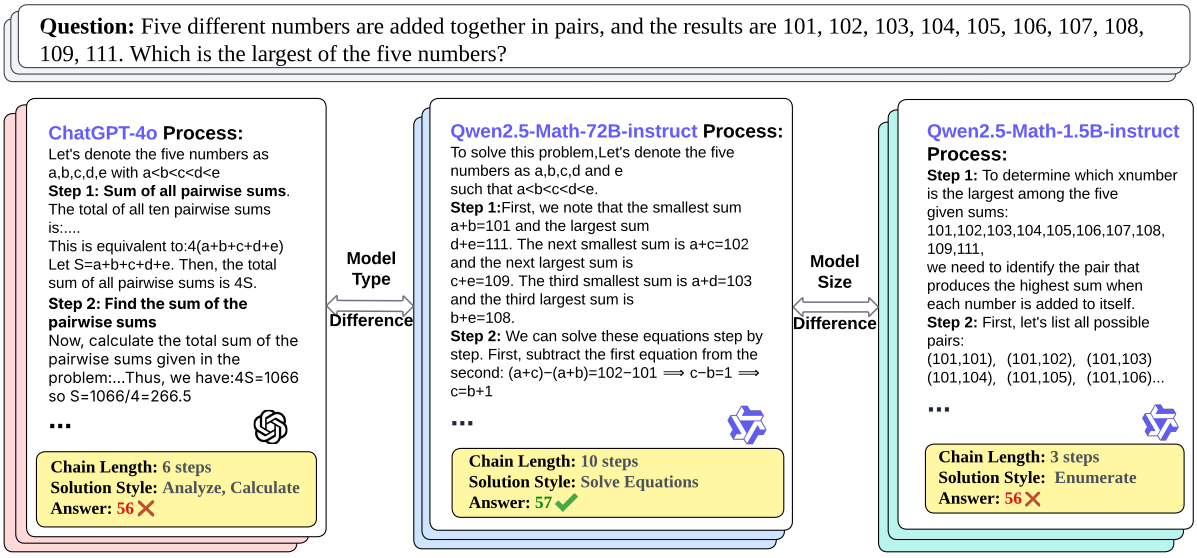

Figure 2: Processes and problem-solving ideas for the same question vary from different models with the perspectives of model types and model sizes. GPT tends to analyze and calculate, while Qwen-72B tends to solve equations. Qwen-1.5B is small and relatively weak. It can only enumerate, and its thinking chain is short, so its answers are also very wrong.

Qwen (Yang et al., 2024b), respectively. However, different model types (e.g., GPT, Qwen, Llama(Dubey et al., 2024)) approach problem-solving differently. As is shown in Figure 2, when facing the same question, GPT-4o tends to analyze and calculate, while Qwen-72B tends to solve questions directly. They have different solution styles. Therefore, using process data generated by one model to train a PRM and then applying it to guide another model leads to an OOD issue. Moreover, models of different sizes also exhibit different reasoning processes. Larger models, like exceptional students, tend to have clearer and more accurate reasoning steps, while smaller models tend to have very short reasoning chains, as shown in Figure 2.

Secondly, **Question OOD** emerges because of dataset shift. Current PRM datasets contain only a limited number of problems. For example, Math Shepherd and PRM800K cover problems from the GSM8K and MATH datasets, with GSM8K being at the elementary to middle school level and MATH at the high school to university level. However, real-world problems are far more diverse, such as those in the Olympic math competition dataset (He et al., 2024), leading to OOD issues in other datasets. As shown in the Figure 1, we used Sentence-BERT (Reimers, 2019) to encode all the problems from the three datasets and visualized the distribution with t-SNE. It is evident that the distributions differ, and since both Olympic and MATH problems are typically from high school-level exams, they are semantically closer to each other than to GSM8K.

To address this issue, we propose a new framework, Retrieval Augmented Process Reward Model (**RetrievalPRM**), which leverages a Two-stage Retrieval-enhanced Mechanism to help PRMs solve the OOD problem. we retrieve relevant questions and steps in these two stages to address the issues of question OOD and step OOD, respectively. Specifically, when predicting a step for a given question, we select semantically similar questions based on their embeddings, placing them at the beginning of the entire prompt. Additionally, we select more fine-grained, similar steps and use them as references when predicting the correctness of the step. These retrieved questions and steps serve as a kind of warm-up for PRM, acting as example problems for reference. They not only help stimulate PRM's potential by warming up but also allow the system to handle more difficult problems by identifying similarities, thus alleviating OOD issues.

Our main contributions are summarized as follows:

- To the best of our knowledge, we are the first to highlight the key OOD problems in Process Reward Models (PRMs), particularly the question OOD and step OOD, which arise due to differences in reasoning patterns across model types (e.g., GPT, Qwen), model sizes (1.5B, 72B) and varying problem difficulties in real-world datasets.

- We introduce the Retrieval-Augmented Process Reward Model (**RetrievalPRM**) framework, which utilizes a Two-stage Retrieval-enhanced

Mechanism to address OOD issues by incorporating both Question-level Retrieval and Step-level Retrieval, thereby enhancing PRM's ability to generalize across diverse problem-solving scenarios.

- We build a Retrieval-enhanced dataset for training PRM using RetrievalPRM framework. We have made our code publicly available.[1] Our dataset[2] and model[3] are open-sourced.

- Extensive experiments on the ProcessBench (Zheng et al., 2024) on four public real-world datasets demonstrate that RetrievalPRM outperforms strong baselines and that the Out-of-distribution issue has been alleviated due to our retrieval approach.

2 Preliminary

In this section, we formulate the whole problem and introduce PRM as a binary classification model.

2.1 Problem Formulation

We denote the Math dataset as $\mathcal{D} = \{(q_i, \mathbf{s}_i, \mathbf{y}_i)\}_{i=1}^{N}$, where N is the number of data instances. The input q_i is the i^{th} Math question. $\mathbf{s}_i = \{s_i^1, s_i^2, \ldots, s_i^{n_i}\}$ are the solution steps, where n_i is the step number of solution $\mathbf{s}_i$. $\mathbf{y}_i = \{y_i^1, y_i^2, \ldots, y_i^{n_i}\}$ and the label y_i^j indicates the correctness from the 1^{st} step to the j^{th} step.

$$y_i^j = \begin{cases} 1, & (s_i^1, \ldots, s_i^j) \text{ is correct for } q_i; \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

2.2 ORM vs. PRM

Outcome-supervised Reward Models are introduced (ORM) by (Cobbe et al., 2021), where verifiers are trained for judging the final correctness of generated solutions. ORM only predicts the final label $\hat{y}_i^{n_i}$, which can be formulated as

$$\forall i, \hat{y}_i^{n_i} = \text{ORM}(q_i, s_i^1, \ldots, s_i^{n_i}). \tag{2}$$

Building on this, the concept of process reward models (PRM) is introduced as a more granular and transparent approach. Not only does PRM evaluate the final solutions but it also assesses intermediate processes, where $\hat{y}_i^j$ represents the predicted label for the j^{th} step by PRM.

$$\forall i, j, \hat{y}_i^j = \text{PRM}(q_i, s_i^1, \ldots, s_i^j). \tag{3}$$

2.3 Large Language Model for PRM scoring

When directly adopting LLMs as the PRM for scoring, we need to convert the data $(q_i, \mathbf{s}_i, \mathbf{y}_i)$ with a hard prompt template. The whole template example is illustrated in Appendix B.2.

The textual input consists of the question q_i and steps $\mathbf{s}_i$, followed by a binary question about the correctness of these steps.

To obtain the floating-point correctness estimation $\hat{y}_i^j \in [0, 1]$ instead of discrete word tokens '+' or '-', we apply bidimensional softmax over the corresponding logits of the binary key answer tokens (ie., + & -) from LLMs to accomplish the correctness estimation during evaluation:

$$\hat{y}_i^j = \frac{\exp(l_{i,+})}{\exp(l_{i,+}) + \exp(l_{i,-})} \in (0, 1). \tag{4}$$

where $l_{i,+}$ and $l_{i,-}$ are the logits of token + and - in the i^{th} instance, respectively.

It is important to note that the estimated PRM scoring $\hat{y}_i^j$ is used solely for evaluation on the testing set. If training is involved, we maintain the standard instruction tuning and causal language modeling paradigm for LLMs. In this way, we don't need to replace the language model head with binary classification head which is the last layer of LLM.

3 Methodology

In this section, we introduce our proposed ***RetrievalPRM*** framework in detail.

3.1 Overview of RetrievalPRM

The RetrievalPRM is developed to address the problem of out-of-distribution (OOD) scenarios in mathematical problem-solving, specifically focusing on both question OOD and step OOD. According to Figure 3, traditional PRM models are constrained by predefined solution steps and are unable to handle unseen questions or steps effectively, especially when the problem context shifts or the solution process deviates from previously seen examples. RetrievalPRM overcomes this challenge by incorporating a Two-stage Retrieval-enhanced Mechanism that dynamically fetches relevant questions and steps from a large pool of questions and their

[1] https://github.com/Gebro13/PRM_research

[2] https://huggingface.co/datasets/gebro13/RetrievalPRM_Dataset

[3] https://huggingface.co/gebro13/RetrievalPRM

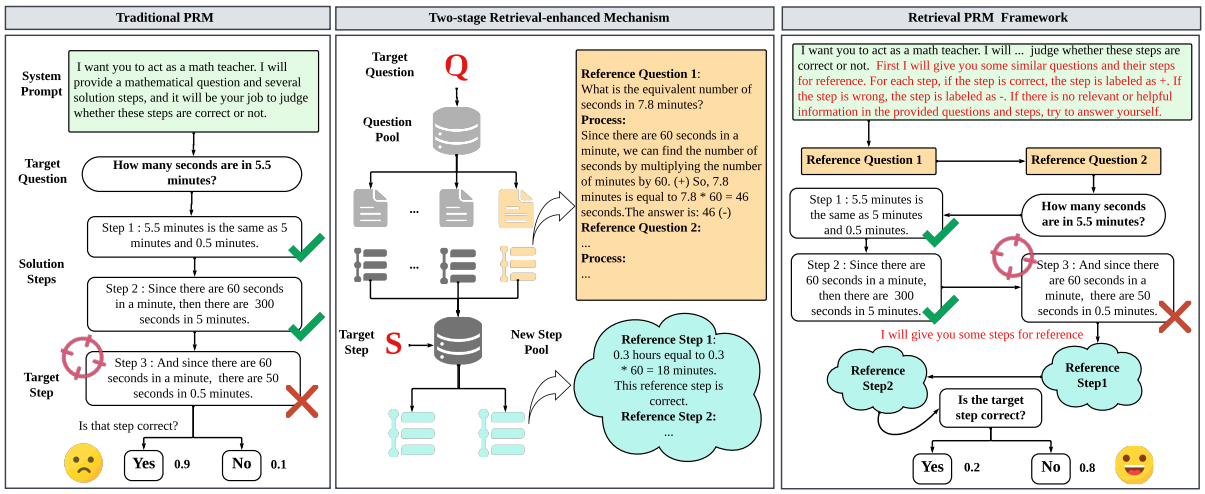

Figure 3: The model structure of our proposed RetrievalPRM framework and its difference with traditional PRM. We design a Two-stage Retrieval Module to retrieve reference questions and steps in each stage.

solutions. These retrieved questions and steps serve as a kind of warm-up for PRM, acting as example problems for reference. They not only help stimulate PRM's potential by warming up but also allow the system to handle more difficult problems by identifying similarities.

3.2 Two-stage Retrieval-enhanced Mechanism

The core of RetrievalPRM is the Two-stage Retrieval-enhanced Mechanism, which consists of two key phases: Question-level Retrieval and Step-level Retrieval.

3.2.1 Question-level Retrieval

The first stage of retrieval tackles the question OOD issue. As is shown in Figure 3, the retrieval pool is the question database $\mathbb{D}_q = \{q_i\}_{i=1}^N$. During retrieval process, we treat:

- Query: the target question q_t.
- Key: all q_i in the retrieval pool.
- Value: all the $(q_i, \mathbf{s}_i)$ pair in the retrieval pool.

We calculate their similarities $< q_i, q_t >$ to match the most similar n questions. Specifically, all questions will first pass through a Sentence-BERT model to encode questions and obtain their semantic representations.

$$\{e_{q_i}\}_{i=1}^N = \text{SentenceBERT}(\{q_i\}_{i=1}^N) \quad (5)$$

where $e_{q_i} \in \mathbb{R}^D$ is the embedding vector of the question q_i.

And then all the embeddings undergo Principle Component Analysis (PCA) (Kurita, 2021) for dimensionality reduction to extract the most important dimensions.

$$\{e'_{q_i}\}_{i=1}^N = \text{PCA}(\{e_{q_i}\}_{i=1}^N) \quad (6)$$

where $e'_{q_i} \in \mathbb{R}^d$ is the embedding after dimension reduction.

Finally, we compute the cosine similarity between the target question and the entire question pool, selecting the top-k most similar questions and inputting them into the text.

$$\langle q_i, q_t \rangle = \frac{e'_{q_t} \cdot e'_{q_i}}{|e'_{q_t}| \cdot |e'_{q_i}|}. \quad (7)$$

Now we sort the vector $\{\langle q_i, q_t \rangle\}_{i=1}^N$ of similarity and choose top-k $(q_i, \mathbf{s}_i)$ pairs as reference questions q_r and put them in RetrievalPRM's input together with the target question. Furthermore, we store all the solutions $\{\mathbf{s}_i\}_{i=1}^m$ of top-m $(m > k)$ questions in a new database to conduct a further step-level retrieval.

3.2.2 Step-level Retrieval

We place step-level retrieval in the second stage of the two-stage retrieval process, rather than as a separate module, for two key reasons:

Firstly, for a solution to be meaningful, both the question and the steps must be similar. For example, two different types of questions might both use the letter "p" to represent an unknown variable, but in some problems, "p" represents a prime number, while in others, it represents probability. This results in steps that may appear similar but have entirely different meanings, rendering the retrieved steps potentially unhelpful.

Secondly, since there are many possible solutions to a question, this leads to a large number of steps. If the majority of these steps are irrelevant, the time spent calculating similarities becomes inefficient. By placing step-level retrieval in the second stage, we can save both time and computational resources.

Therefore, after retrieving the top-*m* most similar questions, we inject all their solutions into a new steps database $\mathbb{D}_s$. Then, we use the target step as the query to retrieve reference steps from this new database. The similarity for retrieval is still calculated using Sentence-BERT, PCA, and cosine similarity, as mentioned in 3.2.1.

3.3 Retrieval-based System Prompt

In RetrievalPRM, The system prompt serves as the instruction set for the model, framing the problem and directing it to evaluate each step of the solution. Besides the traditional system prompt for PRM, the Retrieval-based System Prompt (RetSP) is extended with additional instructions, as shown in the red sentence in Figure 3, which encourages the model to leverage knowledge from reference questions. For example, we inform PRM that step labels "+" and "-" represent correct and incorrect steps, respectively. At the same time, to avoid noise, we specify that if the reference question or step contains no relevant or helpful information, it should not be considered. These retrieval-based system prompts give PRM a more flexible thinking process, enabling it to actively decide whether to use retrieval-based knowledge.

We define reference questions of q_i as $\mathbf{q}_i^r$ and reference steps as $\mathbf{s}_i^r$. The whole input $\mathbf{x}_i^j$ of predicting the j_{th} step of q_i in RetrievalPRM can be formulated as:

$$\begin{aligned} \mathbf{x}_i^j &= (RetSP, \mathbf{q}_i^r, q_i, s_i^1, \ldots, s_i^{j-1}, \mathbf{s}_i^r, s_i^j, y_i^j), \\ \hat{y}_i^j &= \text{PRM}(\mathbf{x}_i^j) \end{aligned} \tag{8}$$

where s_i^j is the j_{th} step of solution $\mathbf{s}_i$.

According to the input template above, it is worth noting that when predicting step n, we assume that steps 1 through n-1 are correct (Luo et al., 2024; Zheng et al., 2024). At this point, the most important task for PRM is to predict step n, so PRM can only access the reference steps for step n and cannot see the reference steps for steps $1 \sim n-1$.

4 Experiments

In this section, we present the experimental settings and results. Our implementation code of RetrievalPRM is publicly available.

4.1 Experiment Setup

4.1.1 Datasets

Datasets are categorized into two kinds: Math reasoning datasets, and prm training datasets.

Math Reasoning Datasets

We conduct experiments on four public and widely used datasets in mathematical reasoning tasks: *GSM8K* (Cobbe et al., 2021) which contains math problems from elementary to middle school, *MATH* (Hendrycks et al., 2021) which contains math problems from basic to university level, *OlympiadBench* (He et al., 2024) which involves questions from the Mathematical Olympiad, *Omni-MATH* (Gao et al., 2024b) which covers multi-domain high-difficulty problems. Further details are provided in Appendix C.

Except for GSM8K, which focuses on grade school math problems, the other three datasets feature problems of competition or Olympiad-level difficulty.

PRM training datasets

We conduct experiments on two publicly available datasets for PRM:

PRM800K (Lightman et al., 2023): Based on the MATH dataset, it contains 800,000 manually annotated step-level correctness labels for training the Process Reward Model. It relies on expensive manual annotations.

Math-Shepherd (Wang et al., 2024b): It generates 400,000 machine-annotated step-level labels (covering MATH and GSM8K datasets) by automatically building process supervision data, without manual annotation.

4.1.2 Evaluation Metrics

We evaluate our model in a public PRM benchmark ProcessBench (Zheng et al., 2024). The aim is to judge whether PRM can find the first wrong step. It divides data into two parts: samples with incorrect and correct final answers and then conducts harmonic mean on the accuracy of these two parts to get the final F1-score. Moreover, we think since the sample number of each part isn't balanced, We add an additional metric: weighted arithmetic mean of these two parts, which is shown in Table 1 as ArithACC.

4.1.3 Baselines

Following (Zheng et al., 2024), we divide all baselines into two parts:

(1) *Open-source PRM*, including Skywork (o1 Team, 2024), Qwen2.5-PRM (Zheng

Table 1: The performance of different models on ProcessBench. The best result is given in bold, and the second-best value is underlined. See Table 6 in Appendix D for breakdown of evaluation results.

	Model	GSM8k		MATH		OlympiadBench		OmniMATH		Avg.F1
		ArithACC	F1	ArithACC	F1	ArithACC	F1	ArithACC	F1	
Open-source PRM	RetrievalPRM-7B(Ours)	**76.0**	**74.6**	**70.6**	**71.1**	**59.1**	**60.2**	**55.2**	**57.33**	**65.8**
	Qwen2.5-Math-7B-PRM800K	73.5	68.2	65.1	62.6	53.2	50.7	43.4	44.3	56.5
	Skywork-PRM-7B	71.6	70.8	54.5	53.6	25.6	22.9	23.7	21.0	42.1
	Skywork-PRM-1.5B	59.9	59.0	49.1	48.0	20.5	19.3	19.7	19.2	36.4
	Math-Shepherd-PRM-7B	58.3	47.9	45.1	29.5	39.7	24.8	34.8	23.8	31.5
	RLHFlow-PRM-Mistral-8B	62.3	50.4	42.1	33.4	22.3	13.8	19.1	15.8	28.4
	RLHFlow-PRM-Deepseek-8B	56.9	38.8	45.1	33.8	26.5	16.9	23.2	16.9	26.6
Language Models as Critic	QwQ-32B-Preview	**87.9**	**88.0**	**78.5**	**78.7**	59.2	**57.8**	**61.1**	**61.3**	**71.5**
	GPT-4o	80.2	79.2	63.4	63.6	50.1	51.4	50.1	53.5	61.9
	Qwen2.5-72B-Instruct	77.9	76.2	65.4	61.8	**59.8**	54.6	55.1	52.2	61.2
	Llama-3.3-70B-Instruct	83.7	82.9	63.7	59.4	54.3	46.7	51.0	43.0	58.0
	Qwen2.5-Coder-32B-Instruct	72.0	68.9	64.5	60.1	57.0	48.9	52.5	46.3	56.1
	Llama-3.1-70B-Instruct	75.3	74.9	52.6	48.2	50.0	46.7	43.2	41.0	52.7
	Qwen2.5-14B-Instruct	72.3	69.3	59.2	53.3	50.2	45.0	43.5	41.3	52.2
	Qwen2-72B-Instruct	67.8	67.6	52.3	49.2	43.3	42.1	39.3	40.2	49.8
	Qwen2.5-32B-Instruct	70.6	65.6	61.9	53.1	53.5	40.0	47.7	38.3	49.3
	Qwen2.5-Math-72B-Instruct	70.3	65.8	59.6	52.1	56.1	32.5	55.1	31.7	45.5
	Qwen2.5-Coder-14B-Instruct	61.9	50.1	54.2	39.9	51.4	34.0	55.6	27.3	37.8
	Qwen2.5-7B-Instruct	37.8	36.5	36.9	36.6	29.9	29.7	27.3	27.4	32.6
	Meta-Llama-3-70B-Instruct	62.4	52.2	48.3	22.8	46.2	21.2	44.8	20.0	29.1
	Qwen2.5-Math-7B-Instruct	54.4	26.8	50.3	25.7	43.1	14.2	41.6	12.7	19.9
	Qwen2-7B-Instruct	25.1	8.4	20.4	19.0	16.1	14.7	13.8	12.1	13.6
	Meta-Llama-3-8B-Instruct	27.1	13.1	17.3	13.8	14.2	4.8	19.7	12.6	11.1
	Qwen2.5-Coder-7B-Instruct	49.1	14.3	46.3	6.5	47.2	4.1	48.9	1.8	6.7
	Llama-3.1-8B-Instruct	27.3	10.9	20.5	5.1	16.0	2.8	15.0	1.6	5.1

Table 2: The performance of different variants of RetrievalPRM on ProcessBench. We remove different components of RetrievalPRM to evaluate the contribution of each part to the model. The best result is given in bold, and the second-best value is underlined. See Table 7 in Appendix D for breakdown of evaluation results.

Retrieval Components		GSM8k		MATH		OlympiadBench		OmniMATH		Avg.F1
Question-level	Step-level	ArithACC	F1	ArithACC	F1	ArithACC	F1	ArithACC	F1	
✓	✓	76.0	74.6	70.6	71.1	**59.1**	**60.2**	**55.2**	**57.3**	**65.8**
✓	×	**77.8**	**74.9**	**70.7**	**71.2**	58.4	59.8	50.5	54.4	65.0
×	✓	73.8	67.5	69.5	69.2	58.2	58.9	52.2	56.3	63.0
×	×	71.0	65.6	67.3	67.5	54.3	55.8	47.2	50.9	59.9

et al., 2024), Math-Shepherd (Wang et al., 2024b) and RLHFlow (Xiong et al., 2024). These models are binary classification PRMs.

(2) *Language Models as Critic*, including Llama (Dubey et al., 2024), Qwen2 (Yang et al., 2024b), Qwen2.5 (Team, 2024), Qwen2.5-MATH (Yang et al., 2024a), Qwen2.5-Coder (Hui et al., 2024), GPT-4o (OpenAI et al., 2024). These models are promoted to judge the steps with the help of majority voting.

Further details of these baselines are provided in Appendix A due to article length limitations.

4.1.4 Implementation Details

Details like base models, hyperparameters, prompts, and training sizes are provided in Appendix B due to the article length limitations.

4.2 Overall Performance

We evaluate RetrievalPRM against existing baselines on ProcessBench, and the results are presented in Table 1. The findings are as follows:

- RetrievalPRM-7B surpasses all open-source PRM baselines, achieving the highest performance. Notably, the most significant improvement is observed on OmniMATH, the most challenging dataset, with performance gains increasing as dataset difficulty rises. This phenomenon may stem from the fact that most baseline PRMs are trained on human- or machine-annotated datasets such as PRM800K or Math-Shepherd, which primarily focus on GSM8K or MATH and exhibit OOD issues when applied to more complex datasets. In contrast, our RetrievalPRM effectively mitigates the OOD problem through its

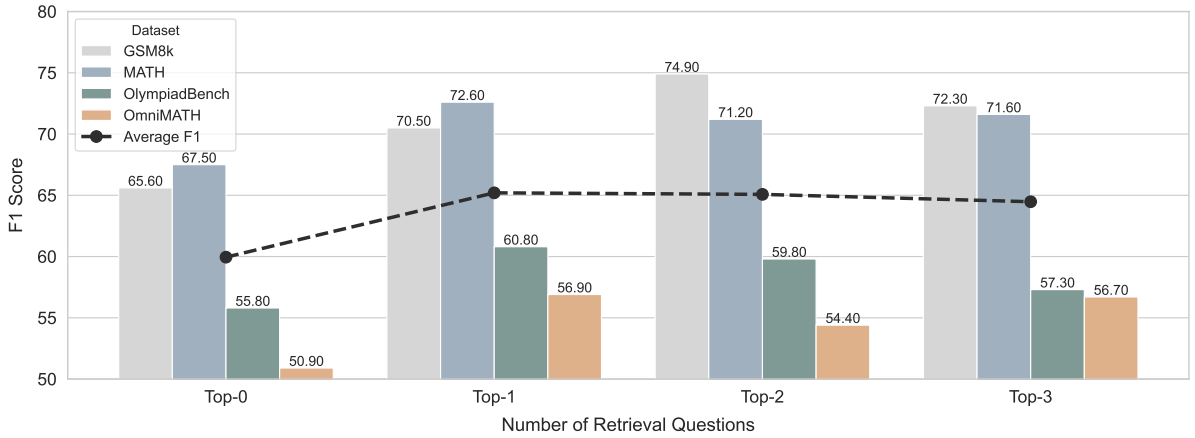

Figure 4: We show the F1 scores of Retrieval-PRM on four datasets and their average, as the number of retrieval questions varies. Specifically, Top-0 means no retrieval questions.

retrieval-based approach, demonstrating the efficacy of our Two-stage Retrieval-enhanced Mechanism.

- When comparing models of different scales, RetrievalPRM outperforms all evaluated language models, including Qwen2.5-72B-Instruct and Llama3.3-70B-Instruct, with the sole exception of QwQ-32B-Preview. Remarkably, RetrievalPRM achieves this with a model size of just 7B. This highlights that PRMs, being both lightweight and task-specific, maintain strong competitiveness and potential compared to LLMs as critics.

4.3 Ablation Study

We analyze two main components in the Two-stage Retrieval-enhanced Mechanism: *Question-level Retrieval* and *Step-level Retrieval*—through the following ablations:

RetrievalPRM (Ours): The complete version of our proposed method.

RetrievalPRM (w/o Step-level Retrieval): This variant retains only the Question-level Retrieval, removing Step-level Retrieval during both training and inference.

RetrievalPRM (w/o Question-level Retrieval): This variant retains only the Step-level Retrieval, removing Question-level Retrieval during both training and inference.

RetrievalPRM (w/o Question-level and Step-level Retrieval): In this variant, both Question-level and Step-level Retrieval are removed during training and inference.

The performance of these variants is presented in Table 2, from which we can draw the following observations:

- The performance of RetrievalPRM (w/o Step-level Retrieval) remains almost identical to that of RetrievalPRM on GSM8K and MATH but exhibits a slight decline on OlympiadBench and OmniMATH. This can be attributed to the fact that Step-level Retrieval information is partially absorbed by Question-level Retrieval. As a result, Question-level Retrieval alone may be sufficiently effective for relatively easy datasets, as the reference steps it provides contain adequate knowledge for step prediction. However, for more challenging datasets, Step-level Retrieval becomes significantly more crucial, as it offers finer-grained guidance essential for handling complex problem-solving processes.

- RetrievalPRM (w/o Question-level Retrieval) shows lower performance, as it relies solely on Step-level Retrieval. The model lacks knowledge of reference questions, which is useful to alleviate question OOD, restricting its overall performance.

- RetrievalPRM (w/o both Retrieval) performs the worst, which is expected, demonstrating the effectiveness of both question-level and Step-level Retrieval.

4.4 Hyperparameter Study

Figure 4 illustrates the impact of the number of retrieval questions on the model's performance. The findings are as follows:

Compared to Top-0, where no retrieval questions are used, models that incorporate retrieval questions show improved performance, highlighting the importance of Question-level Retrieval. It inspires us that Reference questions are important for PRM to get warmup, no matter how many reference questions there are.

The performance of Top-3 exhibits a slight decline, potentially due to two factors: (1) An excessive number of reference questions may lead

to an overly long input prompt, making it difficult for PRMs to comprehend or extract key information effectively. (2) A limited retrieval pool might result in later reference questions being less relevant than earlier ones, increasing the likelihood of misjudgments in the model's predictions.

Table 3: Performance across Different Step Numbers

step num	GSM8K	MATH	Olympiad-Bench	Omni-MATH
0	74.9	71.2	59.8	54.4
1	74.6	71.1	60.2	57.3
2	73.0	70.7	59.6	56.7
3	74.7	69.7	58.2	56.9

We have conducted additional experiments on the hyperparameter: step number. And the results are shown in Table 3. From the experimental results, we observe that changes in the number of steps have little impact on performance. These results suggest that our retrievalPRM exhibits strong robustness with respect to the number of reasoning steps.

4.5 Retrieval Strategy

Regarding the retrieval strategy, we experimented with simple information retrieval methods such as random selection and TF-IDF, and we also explored replacing Sentence-BERT with a larger language model like Qwen-1.5B to get question or step embeddings. The corresponding experimental results are presented in Table 4. From the experimental results, we have the following observations:

- As we move from no retrieval to increasingly advanced retrieval strategies—ranging from random selection to TF-IDF to Sentence-BERT—we observe that the model is better able to retrieve similar questions and steps. This leads to improved prediction accuracy of retrieval-enhanced PRM, demonstrating that the choice of retrieval strategy significantly impacts performance.

- Once the retrieval method reaches a certain level of quality, the performance of Sentence-BERT and Qwen-1.5B becomes comparable. We believe that, given the current retrieval pool, Sentence-BERT is already capable of retrieving highly relevant items, which explains the similar performance to Qwen-1.5B.

- However, as retrieval methods become more sophisticated, their computational cost also increases. Therefore, choosing a retrieval strategy involves a trade-off between efficiency and effectiveness. Under the current setting, Sentence-BERT offers a good balance, making it the preferable choice.

4.6 Results of Other Benchmarks

In order to evaluate the performances of retrieval-PRM completely, we conducted additional experiments employing two mainstream evaluation benchmarks: PRMBench and Best-of-N. Details are shown in Appendix E

4.7 Retrieval Pool Diversity

In this paper, our retrieval pool comprises both the MathShepherd and PRM800K datasets. To empirically examine the effect of the retrieval pool, we conducted supplementary experiments in which the retrieval pool was augmented with the recently released OpenReason dataset (OpenReason, 2024). From the results in Table 5. Detailed explanations are provided in Appendix F.

5 Related Works

5.1 Process Reward Models

Process reward models (PRMs) have demonstrated significant advantages over traditional outcome reward models (ORMs) (Cobbe et al., 2021) in enhancing process-level reasoning accuracy and improving long-process reasoning abilities in model training (Lightman et al., 2023; Uesato et al., 2022). A growing number of PRMs have been proposed for application in process-level reinforcement learning with human feedback (RLHF) (Wang et al., 2024b; Qin et al., 2024; Xia et al., 2025; o1 Team, 2024). For instance, Lightman et al. (2023) made a substantial contribution by releasing a large set of human-annotated data at the process level, opening up new research opportunities for multi-step reasoning.

Additionally, Wang et al. (2024b) introduces an automatic, self-supervised pipeline for generating process-level labels and training PRMs, enabling efficient data generation. Xia et al. (2025) employs PRMs as automatic evaluators to assess the accuracy of multi-step reasoning in language models (LMs). With the surge in PRM-focused research and data curation, numerous PRMs (o1 Team, 2024; Xiong et al., 2024; Sun et al., 2024; Gao et al.,

Table 4: Performance Comparison of Different Retrieval Strategies. Time consumption refers to the time required to retrieve 400 questions along with their corresponding steps.

Retrieval Strategy	GSM8K	MATH	OlympiadBench	OmniMATH	Time Consumption
Random Select	73.9	70.7	57.7	55.3	3s
Tf-idf	74.1	70.2	58.3	56.7	70s
Sentence-Bert	74.6	71.1	60.2	57.3	50min
Qwen-1.5B	73.1	71.7	58.9	57.3	12h

2024a; Wang et al., 2024a) have been proposed. Additionally, several studies focus on leveraging natural language feedback from large language models (LLMs) as rewards, which are termed critic models (McAleese et al., 2024; Zhang et al., 2024; Gao et al., 2024a).

However, most existing PRMs trained on math datasets such as GSM8K and MATH inevitably encounter Out-of-distribution issues, which can be divided into two categories: **question OOD**, where PRMs trained on simpler or medium-difficulty datasets lack understanding of questions from more challenging datasets, and **step OOD**, where different base models and model sizes in LLMs lead to different step distributions for the same question. This is reflected in differences in chain length, problem-solving approaches, and methods. To address these issues, we propose the RetrievalPRM framework to tackle the OOD problems encountered in the current PRM field, achieving promising results.

5.2 Retrieval-Augmented Generation

Retrieval-augmented generation (RAG) enhances language models by dynamically integrating external knowledge, pioneered by (Lewis et al., 2021) through their joint retrieval-generation architecture. Subsequent advances refined this paradigm. Guu et al. (2020) introduced REALM to co-train retrieval and generation modules via masked language modeling, while Izacard and Grave (2021) proposed Fusion-in-Decoder (FiD) to process multi-document contexts efficiently. Research further optimized retrieval precision through dense passage embeddings (Karpukhin et al., 2020) and scaled retrieval to web-level corpora (Borgeaud et al., 2022).

6 Conclusion

In this paper, we have addressed the significant out-of-distribution (OOD) challenges faced by Process Reward Models (PRMs), particularly step OOD and question OOD. By introducing the Retrieval Augmented Process Reward Model (RetrievalPRM), we propose an effective solution that leverages a Two-stage Retrieval-enhanced Mechanism to improve the generalization of PRMs across diverse models and problems. Extensive experiments on multiple real-world datasets have shown that RetrievalPRM consistently outperforms existing methods, highlighting its effectiveness in tackling OOD issues.

7 Limitation

RetrievalPRM has two main limitations. Firstly, the retrieval pool is only constructed from PRM800K and Math-Shepherd at present, which is relatively small and limits the diversity and breadth of the mathematical problems. Second, using Sentence-BERT to embed questions and steps struggles to capture the full complexity of mathematical problems as semantic similarity doesn't mean knowledge similarity in Math problems. As a result, the naive cosine similarity calculated through embeddings may fail to accurately reflect the true similarity between two questions.

Acknowledgments

The Shanghai Jiao Tong University team is partially supported by National Key RD Program of China (2022ZD0114804), Shanghai Municipal Science and Technology Major Project (2021SHZDZX0102) and National Natural Science Foundation of China (624B2096, 62322603, 62177033).

References

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin

Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae, Erich Elsen, and Laurent Sifre. 2022. Improving language models by retrieving from trillions of tokens. *Preprint*, arXiv:2112.04426.

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

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Bofei Gao, Zefan Cai, Runxin Xu, Peiyi Wang, Ce Zheng, Runji Lin, Keming Lu, Dayiheng Liu, Chang Zhou, Wen Xiao, Junjie Hu, Tianyu Liu, and Baobao Chang. 2024a. Llm critics help catch bugs in mathematics: Towards a better mathematical verifier with natural language feedback. *Preprint*, arXiv:2406.14024.

Bofei Gao, Feifan Song, Zhe Yang, Zefan Cai, Yibo Miao, Qingxiu Dong, Lei Li, Chenghao Ma, Liang Chen, Runxin Xu, Zhengyang Tang, Benyou Wang, Daoguang Zan, Shanghaoran Quan, Ge Zhang, Lei Sha, Yichang Zhang, Xuancheng Ren, Tianyu Liu, and Baobao Chang. 2024b. Omni-math: A universal olympiad level mathematic benchmark for large language models. *Preprint*, arXiv:2410.07985.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. *Preprint*, arXiv:2002.08909.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. 2024. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. *Preprint*, arXiv:2402.14008.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, Kai Dang, Yang Fan, Yichang Zhang, An Yang, Rui Men, Fei Huang, Bo Zheng, Yibo Miao, Shanghaoran Quan, Yunlong Feng, Xingzhang Ren, Xuancheng Ren, Jingren Zhou, and Junyang Lin. 2024. Qwen2.5-coder technical report. *Preprint*, arXiv:2409.12186.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. *Preprint*, arXiv:2007.01282.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen tau Yih. 2020. Dense passage retrieval for open-domain question answering. *Preprint*, arXiv:2004.04906.

Takio Kurita. 2021. Principal component analysis (pca). In *Computer vision: a reference guide*, pages 1013–1016. Springer.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2021. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Preprint*, arXiv:2005.11401.

Wendi Li and Yixuan Li. 2025. Process reward model with q-value rankings. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

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*.

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

Nat McAleese, Rai Michael Pokorny, Juan Felipe Ceron Uribe, Evgenia Nitishinskaya, Maja Trebacz, and Jan Leike. 2024. Llm critics help catch llm bugs. *Preprint*, arXiv:2407.00215.

Skywork o1 Team. 2024. Skywork-o1 open series. https://huggingface.co/Skywork.

OpenAI, :, Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, and Akila Welihinda et al. 2024. Gpt-4o system card. *Preprint*, arXiv:2410.21276.

R OpenAI. 2023. Gpt-4 technical report. arxiv 2303.08774. *View in Article*, 2(5).

OpenReason. 2024. Openreason-preview. https://huggingface.co/datasets/OpenReason/OpenReason-preview. Accessed: 2025-05-29.

Yiwei Qin, Xuefeng Li, Haoyang Zou, Yixiu Liu, Shijie Xia, Zhen Huang, Yixin Ye, Weizhe Yuan, Hector Liu, Yuanzhi Li, and Pengfei Liu. 2024. O1 replication journey: A strategic progress report – part 1. *Preprint*, arXiv:2410.18982.

N Reimers. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

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

Mingyang Song, Zhaochen Su, Xiaoye Qu, Jiawei Zhou, and Yu Cheng. 2025. Prmbench: A fine-grained and challenging benchmark for process-level reward models. *arXiv preprint arXiv:2501.03124*.

Zhiqing Sun, Longhui Yu, Yikang Shen, Weiyang Liu, Yiming Yang, Sean Welleck, and Chuang Gan. 2024. Easy-to-hard generalization: Scalable alignment beyond human supervision. *Preprint*, arXiv:2403.09472.

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

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. *Preprint*, arXiv:2211.14275.

Jun Wang, Meng Fang, Ziyu Wan, Muning Wen, Jiachen Zhu, Anjie Liu, Ziqin Gong, Yan Song, Lei Chen, Lionel M. Ni, Linyi Yang, Ying Wen, and Weinan Zhang. 2024a. Openr: An open source framework for advanced reasoning with large language models. *Preprint*, arXiv:2410.09671.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2024b. 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.

Shijie Xia, Xuefeng Li, Yixin Liu, Tongshuang Wu, and Pengfei Liu. 2025. Evaluating mathematical reasoning beyond accuracy. *Preprint*, arXiv:2404.05692.

Wei Xiong, Hanning Zhang, Nan Jiang, and Tong Zhang. 2024. An implementation of generative prm. `https://github.com/RLHFlow/RLHF-Reward-Modeling`.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. 2024a. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *Preprint*, arXiv:2409.12122.

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

Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. 2024. Generative verifiers: Reward modeling as next-token prediction. *Preprint*, arXiv:2408.15240.

Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2024. Processbench: Identifying process errors in mathematical reasoning. *arXiv preprint arXiv:2412.06559*.

Qihao Zhu, Daya Guo, Zhihong Shao, Dejian Yang, Peiyi Wang, Runxin Xu, Y Wu, Yukun Li, Huazuo Gao, Shirong Ma, et al. 2024. Deepseek-coder-v2: Breaking the barrier of closed-source models in code intelligence. *arXiv preprint arXiv:2406.11931*.

A Baselines

A.1 Open-source PRM

- Skywork-PRM (o1 Team, 2024) is a Qwen2.5-Math-based PRM published by KunLun.
- Qwen2.5-PRM (Zheng et al., 2024) is trained by fine-tuning the Qwen2.5-Math-7B-Instruct model on the PRM800K dataset.
- Math-Shepherd (Wang et al., 2024b) generates process labels for each step by estimating the empirical probability that a given step leads to the correct final answer and trains a PRM based on their published dataset.
- RLHFlow-PRM (Xiong et al., 2024) is an 8-billion-parameter reward model trained with process supervision.

A.2 Language Models as Critic

- Llama (Dubey et al., 2024) is an open-source model developed by Meta (formerly Facebook), designed for natural language understanding and generation tasks.
- Qwen2 (Yang et al., 2024b) is a large language model developed by Alibaba Cloud, offering multilingual support and strong capabilities in language understanding and generation.
- Qwen2.5 (Team, 2024) is an advanced iteration of the Qwen series, pretrained on 18 trillion tokens, enhancing knowledge retention, programming, and mathematical reasoning.
- Qwen2.5-MATH (Yang et al., 2024a) is a specialized model for mathematical problem-solving, trained on extensive math-focused data and incorporating Chain-of-Thought (CoT) and Tool-Integrated Reasoning (TIR).

System:
I want you to act as a math teacher. I will provide a mathematical question and several solution steps, and it will be your job to judge whether these steps are correct or not.

Input:
Question:
How many seconds are in 5.5 minutes?
Process:
Step 1 : 5.5 minutes is the same as 5 minutes and 0.5 minutes.
Step 2 : Since there are 60 seconds in a minute, then there are 300 seconds in 5 minutes.
Step 3 : And since there are 60 seconds in a minute, there are 30 seconds in 0.5 minutes.
Is that Step Correct? You should ONLY tell me + or -.

Output:
+.

Figure 5: The illustration of PRM input template.

- Qwen2.5-Coder (Hui et al., 2024) is a programming-oriented model trained on 5.5 trillion code-related tokens, excelling in code generation, debugging, and multilingual programming tasks.

- GPT-4o (OpenAI et al., 2024) is a multimodal AI model developed by OpenAI that processes and generates text, audio, and images in real-time, with enhanced speed and natural interaction capabilities.

B Implementation Details

B.1 Basemodel and Training hyperparameters

We selected Qwen-2.5-Math-7b-instruct(Team, 2024) as the foundational large language model (LLM) for our experiments. All computations were performed using H100 GPUs. To enhance training resource efficiency, we employed Parameter-Efficient Fine-tuning techniques LoRA. The LoRA configuration was set with a rank of 32, an alpha value of 64, and dropout set to 0.1. LoRA update matrices were specifically applied to the query and value projection matrices within the attention blocks.

We use PRM800K as our training data and both PRM800K and Math-Shepherd as our retrieval pool. The training process was carried out with batch sizes chosen from $\{64, 128, 256, 512\}$ and initial learning rates selected from $\{1 \times 10^{-3}, 1 \times 10^{-4}, 3 \times 10^{-4}, 1 \times 10^{-5}, 3 \times 10^{-5}\}$ using a linear scheduler.

B.2 Prompts

In this section, we show our training prompts for PRM in details as is shown in Figure 5 and Figure 6.

C Datasets

GSM8K (Cobbe et al., 2021): Grade School Math is a dataset for basic to intermediate math problems, covering arithmetic, algebra, geometry and other fields. Its difficulty is suitable for math problems in elementary to middle school.

MATH (Hendrycks et al., 2021): The MATH dataset contains a variety of math problems from basic to university level, covering multiple mathematical fields such as algebra, geometry, calculus, number theory, etc.

OlympiadBench (He et al., 2024): The Olympiadbench dataset contains questions from the Mathematical Olympiad. The questions are of high difficulty and involve complex combinatorial mathematics, number theory, geometry and other advanced mathematical fields.

Omni-MATH (Gao et al., 2024b): Omni-MATH is a general Olympiad-level mathematics benchmark dataset for large language models, covering multi-domain and high-difficulty mathematics problems, and is designed to evaluate the reasoning

System:

I want you to act as a math teacher. I will provide a mathematical question and several solution steps, and it will be your job to judge whether these steps are correct or not. First I will give you some similar questions and their steps for reference. For each step, if the step is correct, the step is labeled as +. If the step is wrong, the step is labeled as -. If there is no relevant or helpful information in the provided questions and steps, try to answer yourself.

Input:

Reference Question 1:
What is the equivalent number of seconds in 7.8 minutes?
Process:
Since there are 60 seconds in a minute, we can find the number of seconds by multiplying the number of minutes by 60. (+) So, 7.8 minutes is equal to 7.8 * 60 = 46 seconds.The answer is: 46 (-)
Reference Question 2:
...
Process:
...
Target Question:
How many seconds are in 5.5 minutes?
Process:
Step 1 : 5.5 minutes is the same as 5 minutes and 0.5 minutes.
Step 2 : Since there are 60 seconds in a minute, then there are 300 seconds in 5 minutes.

Reference Step1:
0.3 hours equal to 0.3 * 60 = 18 minutes. This reference step is correct.
Reference Step2:
...
Target Step 3 :
And since there are 60 seconds in a minute, there are 30 seconds in 0.5 minutes.
Is the Step Correct? You should ONLY tell me + or -.

Output:

+.

Figure 6: The illustration of RetrievalPRM input template.

ability of models in various mathematical fields.

Except for GSM8K, which focuses on grade school math problems, the other three datasets feature problems of competition or Olympiad-level difficulty.

D Supplementary Evaluation Results

In this section, we show the breakdown of our main results in Table 6 and ablation results in Table 7

E Results of Other Benchmarks

In order to evaluate the performances of retrievalPRM completely, we conducted additional experiments employing two mainstream evaluation benchmarks: PRMBench and Best-of-N.

PRMBench is a comprehensive benchmark designed to assess process-level reward models across multiple dimensions, including simplicity, soundness, and sensitivity (Song et al., 2025). It consists of 6,216 carefully curated problems and 83,456 step-level labels, enabling fine-grained evaluation of model robustness and generalization. Experimental results demonstrate that RetrievalPRM achieves the highest overall performance on PRMBench, outperforming significantly larger models such as O1-mini and Qwen-72B. Furthermore, RetrievalPRM excels in individual metrics, including soundness and simplicity, further underscoring the robustness and comprehensive effectiveness of our approach.

To further assess practical utility, we adopted the Best-of-N evaluation (Li and Li, 2025). In this setting, a policy model generates n candidate solutions for each problem, and the PRM assigns scores to each step within the candidate solutions. The final answer is selected by identifying the candidate with the highest trajectory score, which is computed using one of several aggregation strategies: min (using the minimal step score), $mean$ (average score), or $last$ (final step score). For each policy and PRM pair, we report the results using the best aggregation method. Our experiments leverage the publicly available solution steps released in the PQM paper to perform the Best-of-N assessment. RetrievalPRM consistently outperforms existing open-source PRMs across various policy models, providing further evidence that retrieval-augmented approaches can effectively mitigate the step-level out-of-distribution (Step-OOD) challenges arising from policy model variations.

These additional experiments in Table 8 and Table 9 further validate the effectiveness of RetrievalPRM and demonstrate its practical advantages.

F Retrieval Pool Diversity

In this paper, our retrieval pool comprises both the MathShepherd and PRM800K datasets. To empirically examine the effect of the retrieval pool, we conducted supplementary experiments in which the retrieval pool was augmented with the recently released OpenReason dataset (OpenReason, 2024). From the results in Table 5. Specifically, we observe that

- Compared to the original retrieval pool, expanding the pool leads to a noticeable improvement in overall performance. This supports our initial hypothesis that the composition of the retrieval pool has a significant impact on the results.
- The OpenReason dataset includes the challenging AIME benchmark. Adding solutions from this dataset into the retrieval pool results in larger performance gains on difficult test sets. This provides further evidence that the diversity of the retrieval pool plays a crucial role in enhancing model performance.
- Additionally, we conducted further experiments where the model was trained using only MathShepherd and PRM800K as the retrieval pool. At test time, however, we expanded the pool by including OpenReason. Even without retraining, the model achieved improved performance, which indicates: (1) RetrievalPRM has learned to effectively utilize retrieved knowledge in a generalizable way - even when the retrieval pool at test time differs from that at training time. (2) Leveraging this property, we can support retrievalPRM in solving harder or unseen problems simply by expanding the retrieval pool, without needing to retrain the model. This greatly enhances the reusability and scalability of retrievalPRM.

Table 5: Performance on Different Retrieval Pool

Train Ret-Pool	**Test Ret-Pool**	**GSM8K**	**MATH**	**OlympiadBench**	**OmniMATH**
MathShepherd+ PRM800K	MathShepherd+ PRM800K	74.6	71.14	60.23	57.33
MathShepherd+ PRM800K	MathShepherd+ PRM800K+ OpenReason	73.0	72.80	61.71	58.78
MathShepherd+ PRM800K+ OpenReason	MathShepherd+ PRM800K+ OpenReason	72.0	72.94	61.75	59.53

Table 6: Breakdown of evaluation results of different models on ProcessBench. The best result is given in bold, and the second-best value is underlined.

Model		GSM8k			MATH			OlympiadBench			OmniMATH		
		error	correct	F1	error	correct	F1	error	correct	F1	error	correct	F1
Open-source PRM	RetrievalPRM-7B(Ours)	64.7	88.1	**74.6**	67.2	75.6	**71.1**	56.0	65.2	**60.2**	52.8	62.65	**57.33**
	Qwen2.5-Math-7B-PRM800K	53.1	95.3	68.2	48.0	90.1	62.6	35.7	87.3	50.7	29.8	86.3	44.3
	Skywork-PRM-7B	61.8	82.9	70.8	43.8	69.2	53.6	17.9	31.9	22.9	14.0	41.9	21.0
	RLHFlow-PRM-Mistral-8B	33.8	99.0	50.4	21.7	72.2	33.4	8.2	43.1	13.8	9.6	45.2	15.8
	RLHFlow-PRM-Deepseek-8B	24.2	98.4	38.8	21.4	80.0	33.8	10.1	51.0	16.9	10.1	51.9	16.9
	Skywork-PRM-1.5B	50.2	71.5	59.0	37.9	65.3	48.0	15.4	26.0	19.3	13.6	32.8	19.2
	Math-Shepherd-PRM-7B	32.4	91.7	47.9	18.0	82.0	29.5	15.0	71.1	24.8	14.2	73.0	23.8
Language Models	QwQ-32B-Preview	81.6	95.3	**88.0**	78.1	79.3	**78.7**	61.4	54.6	**57.8**	55.7	68.0	**61.3**
	GPT-4o	70.0	91.2	79.2	54.4	76.6	63.6	45.8	58.4	51.4	45.2	53.5	61.9
	Qwen2.5-72B-Instruct	62.8	96.9	76.2	46.3	93.1	61.8	38.7	92.6	54.6	36.6	90.9	52.2
	Llama-3.3-70B-Instruct	72.5	96.9	82.9	43.3	94.6	59.4	31.0	94.1	46.7	28.2	90.5	43.0
	Qwen2.5-32B-Instruct	49.3	97.9	65.6	36.7	95.8	53.1	25.3	95.9	40.0	24.1	92.5	38.3
	Qwen2.5-14B-Instruct	54.6	94.8	69.3	38.4	87.4	53.3	31.5	78.8	45.0	28.3	76.3	41.3
	Qwen2.5-Coder-32B-Instruct	54.1	94.8	68.9	44.9	90.6	60.1	33.4	91.2	48.9	31.5	87.6	46.3
	Qwen2.5-Coder-14B-Instruct	33.8	96.4	50.1	25.4	92.4	39.9	20.7	94.1	34.0	15.9	94.2	27.3
	Qwen2.5-Coder-7B-Instruct	7.7	100.0	14.3	3.4	98.3	6.5	2.1	99.1	4.1	0.9	98.3	1.8
	Qwen2.5-Math-72B-Instruct	49.8	96.9	65.8	36.0	94.3	52.1	19.5	97.3	32.5	19.0	96.3	31.7
	Qwen2.5-Math-7B-Instruct	15.5	100.0	26.8	14.8	96.8	25.7	7.7	91.7	14.2	6.9	88.0	12.7
	Llama-3.1-70B-Instruct	64.3	89.6	74.9	35.4	75.6	48.2	35.1	69.9	46.7	30.7	61.8	41.0
	Meta-Llama-3-70B-Instruct	35.7	96.9	52.2	13.0	93.3	22.8	12.0	92.0	21.2	11.2	91.7	20.0
	Qwen2-72B-Instruct	57.0	82.9	67.6	37.7	70.9	49.2	34.0	55.2	42.1	32.3	53.1	40.2
	Qwen2.5-7B-Instruct	40.6	33.2	36.5	30.8	45.1	36.6	26.5	33.9	29.7	26.2	28.6	27.4
	Qwen2-7B-Instruct	40.6	4.7	8.4	30.5	13.8	19.0	22.4	10.9	14.7	20.0	8.7	12.1
	Llama-3.1-8B-Instruct	44.4	6.2	10.9	41.9	2.7	5.1	32.4	1.5	2.8	32.0	0.8	1.6
	Meta-Llama-3-8B-Instruct	42.5	7.8	13.1	28.6	9.1	13.8	27.1	2.7	4.8	26.1	8.3	12.6

Table 7: Breakdown of evaluation results of different variants of RetrievalPRM on ProcessBench. We remove different components of RetrievalPRM to evaluate the contribution of each part to the model. The best result is given in bold, and the second-best value is underlined.

Retrieval Components		GSM8k			MATH			OlympiadBench			OmniMATH			Avg.F1
Question-level	Step-level	error	correct	F1	error	correct	F1	error	correct	F1	error	correct	F1	
✓	✓	64.7	88.1	74.6	67.2	75.6	71.1	56.0	65.2	**60.2**	52.8	62.65	**57.33**	**65.8**
✓	×	61.8	94.8	**74.9**	62.1	83.3	**71.2**	48.7	77.3	59.8	43.2	73.4	54.4	65.0
×	✓	51.7	97.4	67.5	57.2	87.4	69.2	46.0	82.0	58.9	43.9	78.4	56.3	63.0
×	×	50.7	92.7	65.6	57.9	81.0	67.5	46.9	68.7	55.8	39.7	71.0	50.9	59.9

Table 8: Results of PRMBench

Model	Overall	NR.	NCL.	S1	ES.	SC.	DC.	CI.	S2	PS.	DR.	MS.	S3
RetrievalPRM(Ours)	68.9	51.9	58.6	55.3	76.2	74.8	69.5	79.5	75.0	62.3	73.0	99.2	78.2
o1-mini	68.8	65.6	63.7	64.6	74.5	67.7	73.8	72.3	72.1	61.8	64.8	100.0	75.5
Qwen2.5-Math-PRM-72B	68.2	50.4	58.8	54.6	73.7	71.1	72.2	78.6	73.9	60.3	71.2	99.4	77.0
GPT-4o	66.8	57.0	62.4	59.7	72.0	69.7	70.7	71.1	70.9	62.5	65.7	99.2	75.8
Qwen2.5-Math-PRM-7B	65.5	49.0	55.1	52.1	71.8	67.3	66.3	78.5	71.0	57.6	69.1	99.7	75.5
Skywork-PRM-7B	65.1	56.4	62.8	59.6	69.4	67.1	67.7	69.9	68.5	60.9	65.8	93.2	73.3
QwQ-Preview-32B	63.6	57.2	55.6	56.4	67.4	72.3	66.2	66.9	68.2	57.8	62.7	100.0	73.5
RLHFlow-PRM-Mistral-8B	54.4	46.1	47.3	46.7	56.6	55.1	54.4	63.8	57.5	51.5	56.2	97.9	68.5
Llemma-PRM800k-7B	52.0	49.3	53.4	51.4	56.4	47.1	46.7	53.3	50.9	51.0	53.5	93.6	66.0
MathShepherd-Mistral-7B	47.0	44.0	50.3	47.1	49.4	44.5	41.3	47.7	45.7	47.2	48.6	86.1	60.7

Table 9: Results of Best-of-n Search with different PRMs

Dataset	Policy Model	RetrievalPRM (Ours)	RLHFlow-PRM	Skywork-PRM	MathShepherd-Mistral
MATH	Llama-3-70B-Instruct	48.8	42.0	38.0	46.2
MATH	MetaMath-Mistral-7B	37.4	29.8	24.0	32.6
MATH	Muggle-Math-13B	33.8	23.6	19.0	28.4
GSM-Plus	Llama-3-70B-Instruct	74.5	70.8	67.7	72.6
GSM-Plus	MetaMath-Mistral-7B	63.7	53.5	48.0	60.3
GSM-Plus	Muggle-Math-13B	62.4	50.8	43.2	59.5