

Uncertainty-Based Methods for Automated Process Reward Data Construction and Output Aggregation in Mathematical Reasoning

Jiuzhou Han¹, Wray Buntine², Ehsan Shareghi¹

¹Department of Data Science & AI, Monash University

²College of Engineering and Computer Science, VinUniversity

jiuzhou.han@monash.edu, wray.b@vinuni.edu.vn, ehsan.shareghi@monash.edu

Abstract

Large language models have demonstrated remarkable capabilities in complex mathematical reasoning tasks, but they inevitably generate errors throughout multi-step solutions. Process-level Reward Models (PRMs) have shown great promise by providing supervision and evaluation at each intermediate step, thereby effectively improving the models' reasoning abilities. However, training effective PRMs requires high-quality process reward data, yet existing methods for constructing such data are often labour-intensive or inefficient. In this paper, we propose an uncertainty-driven framework for automated process reward data construction, encompassing both data generation and annotation processes for PRMs. Additionally, we identify the limitations of both majority vote and PRMs, and introduce two generic uncertainty-aware output aggregation methods: Hybrid Majority Reward Vote and Weighted Reward Frequency Vote, which combine the strengths of majority vote with PRMs. Extensive experiments on ProcessBench, MATH, and GSMPlus show the effectiveness and efficiency of the proposed PRM data construction framework, and demonstrate that the two output aggregation methods further improve the mathematical reasoning abilities across diverse PRMs.

Code — <https://github.com/Jiuzhouh/UnPRM>

1 Introduction

Inference-time scaling (Lightman et al. 2024) offers a practical approach to improving reasoning performance of Large Language Models (LLMs) by leveraging increased computational resources during inference. Within this framework, Process Reward Models (PRMs) (Uesato et al. 2022) have been introduced to assess the correctness of intermediate reasoning steps, providing a fine-grained mechanism for scoring and filtering candidate solutions and thereby supporting more robust answer selection. Unlike Outcome Reward Models (ORMs) (Cobbe et al. 2021), which focus solely on the final answer, PRMs provide fine-grained verification of the entire reasoning process. This granularity allows PRMs to identify solutions in which correct final answers are arrived at via flawed intermediate steps, revealing otherwise undetected errors in reasoning.

Copyright © 2026, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

A major challenge in developing effective PRMs lies in the annotation of high-quality step-level supervision data, which is typically expensive and time-consuming. Early approaches (Lightman et al. 2024) relied on human annotation to ensure label quality, but this method is costly and not scalable. To improve efficiency, recent research has focused on automated annotation techniques, including Monte Carlo (MC) methods that estimate step correctness based on the probability of reaching the correct final answer (Wang et al. 2024; Luo et al. 2024) and approaches that leverage strong LLMs as judges of step correctness (Tan et al. 2025; Gao et al. 2024). However, these methods can require substantial computational resources and often suffer from inefficiency.

To tackle these challenges, we propose an uncertainty-driven PRM data construction framework. Our approach first generates candidate solutions for annotation through uncertainty-guided sampling, then applies an uncertainty-driven automated annotation process to efficiently label the correctness of each reasoning step. This pipeline notably improves the quality and scalability of PRM training data.

While PRMs and Majority Vote (Wang et al. 2023) are commonly used to aggregate LLM outputs, both have their limitations. Majority Vote, which selects the most frequent answer among sampled outputs, can fail when answers are highly dispersed or the model confidently produces wrong solutions, yielding erroneous consensus. Conversely, PRMs may select suboptimal answers when faced with out-of-distribution or particularly challenging problems. To address the limitations and improve aggregation, we propose two uncertainty-aware hybrid strategies: Hybrid Majority Reward (HMR) Vote and Weighted Reward-Frequency (WRF) Vote. These methods combine the implicit confidence signals of Majority Vote with the explicit step-level feedback from PRMs, enabling more reliable answer selection.

We construct several variations of PRM training data and conduct extensive experiments on ProcessBench (Zheng et al. 2024), MATH (Hendrycks et al. 2021), and GSM-Plus (Li et al. 2024) to validate the efficiency and effectiveness of our uncertainty-driven PRM data construction framework. Additionally, we show that our proposed uncertainty-aware aggregation strategies generalise well across different PRMs and yield notable performance improvements compared to majority vote and traditional PRM methods, with WRF generally proving to be more robust.

2 Related Work

PRMs in Mathematical Reasoning

Mathematical reasoning tasks are among the most challenging for LLMs, as they demand rigorous multi-step logical thinking and precise manipulation of mathematical symbols, leaving little room for ambiguity or guesswork. A single error at any step can invalidate the entire solution, and the abstract, concise nature of mathematical language further compounds the difficulty. To assess the correctness of these intermediate steps, PRMs (Lightman et al. 2024; Uesato et al. 2022) have been introduced. PRMs assign a score to each reasoning step, reflecting its likelihood of correctness. By aggregating these step scores, a final score for the entire solution can be obtained, providing an overall assessment of its validity. Furthermore, PRMs enable the selection of the highest-scoring solution from multiple candidates, thereby further improving the reasoning performance of LLMs.

Mathematical Reasoning Step Verification

Although PRMs demonstrate significant potential for enhancing the performance of LLMs, their effectiveness relies heavily on the availability of high-quality labelled training data, particularly for evaluating the correctness of intermediate reasoning steps. There are three primary approaches to data annotation: (1) Human annotation (Lightman et al. 2024), which can yield highly accurate labels but is labour-intensive, expensive, and time-consuming; (2) LLM-as-a-Judge methods (Tan et al. 2025; Zhang et al. 2025a; Gao et al. 2024), which prompt LLMs to directly assess step correctness, offering automation but incurring high computational costs, especially when labelling large datasets, and inheriting the limitations of LLMs such as hallucination and inaccuracies; and (3) inferring step correctness from solution outcomes using techniques such as Monte Carlo estimation. The latter enables automated annotation by leveraging open-source LLMs (Wang et al. 2024), but often suffers from inefficiency. To address this, Luo et al. (2024) introduced a more efficient Monte Carlo Tree Search algorithm using binary search, while Sun et al. (2025) further improved efficiency with an adaptive binary search method. Building on these developments, we propose an uncertainty-driven search algorithm for automated step label annotation, further enhancing annotation efficiency without sacrificing quality.

Uncertainty Estimation in LLMs

Uncertainty estimation plays a crucial role in assessing the confidence of outputs generated by LLMs. Logits-based methods (Han, Buntine, and Shareghi 2024; Yang et al. 2023) quantify uncertainty by analysing the token-level logits produced by the model; however, these approaches are only feasible when direct access to output token logits is available. Alternatively, self-consistency (Wang et al. 2023) provides an implicit measure of model confidence by sampling multiple outputs for a given question—answers that appear frequently among the samples are considered high-confidence, while those with low frequency indicate greater uncertainty. Prior studies (Han, Buntine, and Shareghi 2024; Zhao et al. 2025) have found a strong correlation between

higher model confidence and answer correctness, with errors being more likely when the model is uncertain. Motivated by these findings, we propose leveraging uncertainty as a guiding signal to identify erroneous steps in mathematical reasoning solutions, thereby improving the automated annotation process for PRM training data.

3 Uncertainty-driven Automated Process Reward Data Construction

In this section, we first introduce the uncertainty estimation method employed in our approach. We then describe how this uncertainty estimation is leveraged to guide PRM data generation, and finally, we detail the design of our uncertainty-driven automated PRM data annotation process.

Uncertainty Estimation

To quantify the uncertainty associated with each candidate solution generated by the language model, we employ an entropy-based uncertainty estimation approach. Specifically, for a given solution consisting of a sequence of n tokens, we extract the log-probability assigned by the model to each generated token during the decoding process. Let p_i denote the token log-probability assigned to the i -th token in the solution, then we apply a softmax function to the token log-probabilities to obtain the probabilities $[z_1, z_2, \dots, z_n]$. The overall uncertainty u of the candidate solution is then computed as the entropy over the sequence of token probabilities, which captures the model’s average uncertainty throughout the solution. Formally, the uncertainty score is calculated as $u = -\sum_{i=1}^n z_i \cdot \log(z_i)$, where higher values of u indicate that the model was generally less confident across the sequence, while lower values indicate more confident and deterministic predictions. This entropy-based uncertainty metric supports our uncertainty-driven PRM data generation and automated step-level annotation.

Uncertainty-driven PRM Data Generation Process

To construct the training data for PRMs, given a set of questions, each paired with its ground-truth answer, we first sample k candidate solutions using an LLM, obtaining token-level log-probabilities for each generated solution. For each candidate solution, we compute an uncertainty score based on its token log-probabilities using an uncertainty function. We then verify each solution using the ground-truth final answer to categorise it as correct or incorrect. Among all correct solutions, we select the top m with the *highest* uncertainty scores, and similarly, among incorrect solutions, we select the top n most uncertain. The final candidate set comprises both the most uncertain correct and incorrect solutions, ensuring diversity and difficulty in the PRM training data. This targeted sampling encourages the PRM to learn more robustly from ambiguous or challenging reasoning trajectories, thereby improving its ability to identify and discriminate step-level correctness during inference. Full algorithm is outlined in technical appendix.

Uncertainty-driven PRM Data Annotation Process

To efficiently annotate the step-level correctness of candidate solutions, we propose an automated uncertainty-driven step label annotation process (Algorithm 1). Given a set of candidate solutions C partitioned into correct and incorrect solutions, we first assign `True` (correct) labels to all steps within each correct solution (LN3-7), assuming there are no mistakes in the intermediate steps. For a given incorrect solution composed of T steps, we compute the uncertainty $u(s_t)$ at each individual step s_t , and subsequently calculate the uncertainty delta $\Delta u(s_t) = u(s_t) - u(s_{t-1})$ for steps $t = 2, \dots, T$ (LN8-11). These deltas serve to identify steps where the model’s uncertainty exhibits the greatest increase, which are likely to correspond to points where reasoning errors occur (see Table 3 for supporting evidence). Steps are then ordered by descending uncertainty delta and placed in a list to prioritise annotation at the most uncertain transitions.

For each candidate step in this ordered list, we employ an adaptive sampling strategy following the prior work (Sun et al. 2025). Starting from the selected step s_i , we sample N new solution completions (LN12-18).¹ We refer to the set containing all these solutions for s_i as S_{all}^i and the set containing only the solutions ending with correct answer as S_{correct}^i . We refer to solution trajectory k in these sets as traj^k and its perplexity as $PPL(\text{traj}^k)$.² The Monte Carlo-based perplexity MC_{PPL} (Sun et al. 2025) is computed as,

$$MC_{PPL}(s_i) = \frac{\sum_{\text{traj}^k \in S_{\text{correct}}^i} \log PPL(\text{traj}^k)}{\sum_{\text{traj}^k \in S_{\text{all}}^i} \log PPL(\text{traj}^k)} \quad (1)$$

A threshold τ is defined as the MC perplexity corresponding to the initial problem state, where the input consists solely of the question. If the MC_{PPL} of the step s_i falls below the threshold τ , steps up to s_{i-1} are labelled as `True` (correct), and all subsequent steps from s_i onwards are labelled as `False` (incorrect) (LN19-23). In contrast to approaches that identify the first error step, our automated PRM data annotation process is designed to locate the most uncertain reasoning error within each incorrect solution. The annotated solutions, each with step-level correctness labels, are then aggregated to form the final training set \tilde{C} .

4 Uncertainty-aware Output Aggregation

In this section, we first discuss the limitations of commonly used output aggregation methods, such as Majority Vote and PRM-based approaches. We then propose two aggregation strategies that combine the strengths of both Majority Vote and PRMs to achieve more robust answer selection.

Limitations of Majority Vote and PRMs

LLMs often generate diverse solutions when prompted multiple times for the same problem, due to their inherent

¹The sample size N is dynamically increased until a minimum number of correct samples (N_{min}) is reached, or until a maximum sampling threshold (N_{max}) is met. The adaptive sampling strategy reduces computational overhead while ensuring annotation quality.

²Perplexity of a sequence X of length L is calculated as $\exp\left(-\frac{1}{L} \sum_{t=1}^L \log p(x_t | x_{<t})\right)$.

Algorithm 1: Uncertainty-driven Step Label Annotation

Input: Solution set C ; uncertainty function $u(\cdot)$

Param: $N_0, N_{\text{min}}, N_{\text{max}}, \tau$

Output: Labelled solution set \tilde{C}

```

1:  $\tilde{C} \leftarrow \emptyset$ 
2: for  $s \in C$  do
3:    $U \leftarrow \emptyset$ 
4:   if  $s \in C_{\text{correct}}$  then
5:     Label all steps in  $s$  as True
6:     Append labelled  $s$  to  $\tilde{C}$ ; continue
7:   end if
8:   for  $s_t \in s, t > 1$  do
9:     Compute  $\Delta u(s_t) = u(s_t) - u(s_{t-1})$ 
10:    Append  $(\Delta u(s_t), s_t)$  to  $U$ 
11:  end for
12:  Sort  $U$  in desc. order according to  $\Delta u$ 
13:  for  $s_i$  in  $U$  do
14:     $N \leftarrow N_0$ 
15:    repeat
16:      Sample  $N$  solutions from  $s_i$ ; count  $N_{\text{correct}}$ 
17:      Increase  $N$  dynamically if  $N_{\text{correct}} < N_{\text{min}}$ 
18:    until  $N_{\text{correct}} \geq N_{\text{min}}$  or reaching  $N_{\text{max}}$ 
19:    Compute  $MC_{PPL}$  over samples using Eq. (1)
20:    if  $MC_{PPL} < \tau$  then
21:      Label  $[s_1..s_{i-1}]$  as True,  $[s_i..s_T]$  as False
22:      Append labelled  $s$  to  $\tilde{C}$ ; break
23:    end if
24:  end for
25: end for
26: return  $\tilde{C}$ 

```

stochasticity and the complexity of mathematical reasoning tasks. Majority Vote (Wang et al. 2023) and PRMs (Lightman et al. 2024) are two commonly used and effective approaches to aggregate these multiple candidate outputs and identify the most likely correct answer from LLMs.

In Majority Vote, the answer that occurs most frequently among the sampled solutions is chosen, implicitly favouring the response that the model generates with the highest confidence. While this method is straightforward and often effective, it can fail when the candidate answers are highly dispersed or when the model is consistently confident in an incorrect solution, leading to erroneous consensus.

In contrast, PRMs provide a more explicit evaluation by predicting the correctness of each reasoning step within a solution, offering step-level signals to inform the answer selection process. This approach can more robustly identify subtle reasoning errors that majority vote may overlook. However, the effectiveness of PRM-based methods depends on the reward model’s ability to accurately assess a wide range of problem types; inaccuracies in the reward model can result in the selection of sub-optimal answers, especially for out-of-distribution or particularly challenging examples.

Given these complementary strengths and limitations, we propose two hybrid strategies: Hybrid Majority Reward (HMR) Vote and Weighted Reward-Frequency (WRF) Vote, which are designed to integrate the implicit confidence sig-

nals from majority vote with the explicit feedback signals provided by PRMs. By leveraging both consensus among candidate solutions and fine-grained reasoning evaluation, these methods aim to achieve more reliable and accurate answer aggregation for LLM-generated outputs.

Hybrid Majority Reward Vote

In Hybrid Majority Reward (HMR) vote strategy, given a set of N sampled candidate solutions, each comprising a sequence of reasoning steps and a final answer, the HMR method first extracts the answer from each solution to form a collection of candidate answers. Then, the most frequent answer through majority vote is determined. If the majority answer appears in at least half of the solutions ($f_{\text{maj}} \geq N/2$), it is directly selected as the final answer. However, if no answer achieves a strict majority ($f_{\text{maj}} < N/2$), indicating ambiguity or lack of consensus among candidates, the algorithm leverages the PRM to guide answer selection. Specifically, for each candidate solution, the PRM function is invoked to compute a list of step-wise scores, and the minimum score across the steps is used as the overall reward for this solution. The answer given by the solution with the highest reward is selected as the final answer. This hybrid strategy combines the robustness of majority vote with the fine-grained reasoning assessment provided by the PRM, ensuring both consensus and confidence inform the answer aggregation process.

Weighted Reward Frequency Vote

Weighted Reward-Frequency (WRF) vote strategy provides another answer aggregation method. Given a set of N sampled candidate solutions, each consisting of a series of reasoning steps and a final answer, the WRF algorithm integrates both the solution’s frequency and the quality of its reasoning steps as evaluated by the PRM. For each candidate solution, the PRM function computes step-wise scores, and the minimum score is taken as the overall reward for the solution. These rewards are grouped according to their corresponding final answers.

For each unique answer, the algorithm calculates the mean PRM reward and the frequency with which that answer appears among the candidates. Both metrics are then individually normalised across all answers to ensure comparability. Specifically, the normalised mean reward \hat{r}_a and the normalised frequency \hat{f}_a for answer a are computed using min-max normalisation. The final combined score for each answer is calculated as a weighted sum of its normalised mean reward and frequency, controlled by the weighting parameter $\alpha \in [0, 1]$. The answer with the highest combined score is selected as the output.

By integrating both the consensus among candidate solutions (frequency) and the confidence derived from step-level PRM rewards, the WRF vote approach offers a more nuanced and fine-grained mechanism for answer aggregation. In our experiments, we set $\alpha = 0.5$, giving equal weight to both frequency and reward components.

5 Experiments

PRM Training Data Construction

We utilise the MATH dataset (Hendrycks et al. 2021), which comprises 12,500 challenging competition-level mathematics problems, to construct our PRM training data. Following the EpicPRM (Sun et al. 2025) protocol, we adopt the same 3,500 randomly selected questions from the MATH training set as our seed questions. To enhance the diversity of sampled Chain-of-Thought (CoT) reasoning solutions for each question, we employ three different LLMs: Llama-3.1-8B-Instruct (Dubey et al. 2024), Qwen2.5-7B-Instruct (Yang et al. 2024a), and Mistral-7B-Instruct-v0.3 (Jiang et al. 2023). For each model, we set the sampling temperature to 0.8 and generate 32 solutions per mathematics question. Subsequently, we apply the uncertainty-driven PRM data generation method to select the 2 most uncertain correct solutions and 6 most uncertain incorrect solutions.

We then filter out solutions with undesired formats (e.g., answers not presented in the correct format) and split the remaining candidate solutions into intermediate steps for automated annotation. For step-level annotation, we employ our automated uncertainty-driven step label annotation method, which assigns True or False labels to each step of the candidate solutions from the three LLMs. This process yields 40K labelled training examples, referred to as UnPRM40K.

For comparison, prior work (Sun et al. 2025) uses similarity, rather than uncertainty, as the selection criterion. Specifically, it selects candidate solutions exhibiting the lowest cosine similarity scores. We follow this approach to generate candidate solutions and annotate them using the uncertainty-driven step label annotation method, resulting in another 40K training examples, denoted as SimPRM40K.

The uncertainty-driven step label annotation method identifies the most uncertain error step in the solution. To compare this with an alternative approach that labels data based on the first error step, we re-annotate the same 40K examples using the adaptive binary search method adopted in EpicPRM (Sun et al. 2025), referred to as EpicPRM40K.

To further investigate the effect of the error step’s location on model performance, we conduct an additional experiment in which the error step for incorrect candidate solutions is selected at random, denoted as RanPRM40K.

PRM Training

The objective of PRMs is to determine, at each step of the solution process, whether the reasoning trajectory remains correct. Given a problem q and a sequence of solution steps $s_1 \rightarrow s_2 \rightarrow \dots \rightarrow s_t$, the PRM model assigns a score y_t between 0 and 1 to indicate the correctness of the step. This formulation naturally leads to a binary classification framework, where the model outputs a probability reflecting whether the solution is still correct up to the current step.

To train the PRM, we employ supervised fine-tuning of a language model. The input consists of the problem statement concatenated with the intermediate reasoning steps, each separated by a special step tag (in our case, the Unicode character ‘§’). This tag is inserted between each step to

delineate step boundaries within the input sequence. For every step, the label is either correct ('+') or incorrect ('-'), and these labels are aligned with the tokens immediately following each step tag in the input sequence. This explicit tagging enables the model to attend to the step boundaries and precisely associate each predicted label with its corresponding reasoning step.

The model is optimised using the binary cross-entropy loss with logits, targeting the prediction of the correct token at each annotated step. The loss function used for training is given by:

$$\mathcal{L} = -\frac{1}{N} \sum_{i=1}^N [y_i \log \hat{p}_i + (1 - y_i) \log(1 - \hat{p}_i)] \quad (2)$$

where N is the number of step-level predictions in a batch, $y_i \in \{0, 1\}$ is the ground-truth label for the i -th step (1 for correct, 0 for incorrect), and \hat{p}_i is the model's predicted probability for the correctness of step i , obtained by applying the sigmoid function to the output logits.

During inference, the PRM predicts a step score for each intermediate step by extracting the logits associated with the candidate tokens at each step tag position and applying a softmax or sigmoid function. The resulting probability assigned to the correct token ('+') at each step reflects the model's confidence in the correctness of the current reasoning process. This allows the PRM to provide fine-grained scores for each step.

We use Qwen2.5-Math-7B-Instruct (Yang et al. 2024b) as the base model for PRM training. The PRMs are fine-tuned for three epochs on a single A100 GPU using the LoRA (Hu et al. 2022) technique for parameter-efficient adaptation.

PRM Best-of-N Evaluation

Best-of-N (BoN) evaluation, which selects the highest-scored solution from N candidates according to the PRM, is a widely-used approach for assessing PRM performance in prior research (Lightman et al. 2024; Wang et al. 2024; Zhang et al. 2025b). Following this approach, we sample up-to 128 candidate solutions for each problem, and use the minimum step score as the overall solution score.

Datasets We evaluate the PRMs on two mathematical reasoning datasets: MATH (Hendrycks et al. 2021) and GSMPlus (Li et al. 2024). For MATH, we use the same test set as in prior work (Lightman et al. 2024), which consists of 500 math problems uniformly sampled at random from all categories. GSMPlus is an augmented version of GSM8K (Cobbe et al. 2021) that introduces various mathematical perturbations, enabling a comprehensive assessment of LLMs' robustness in mathematical reasoning. As GSMPlus comprises eight types of questions, we uniformly select 50 questions from each type, resulting in 400 problems.

Models and Baselines We utilise two models as the reasoner: the black-box LLM GPT-4o and the open-source LLM Qwen2.5-Math-7B-Instruct. For baseline PRMs, we compare our UnPRM40K with SimPRM40K, EpicPRM40K, and RanPRM40K. In addition to PRMs

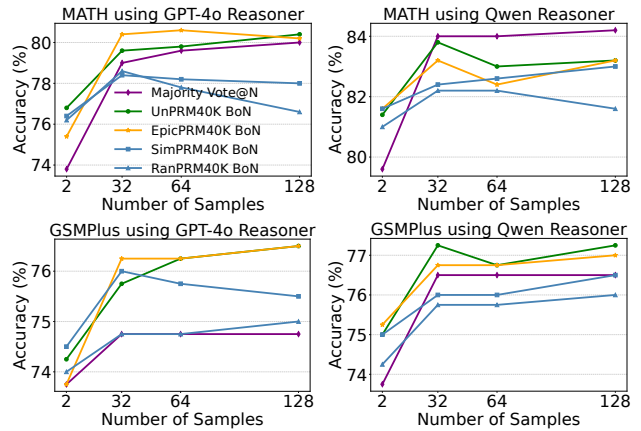


Figure 1: Evaluation results of different PRMs with BoN strategies on two datasets using two reasoners. Qwen Reasoner denotes Qwen2.5-Math-7B-Instruct Reasoner.

trained on datasets of equal size, we also compare UnPRM40K with two publicly available PRMs (Zhang et al. 2025b): Qwen2.5-Math-PRM-7B and Qwen2.5-Math-7B-PRM800K, which are trained on 1.8M and 264K examples, respectively, substantially more than the 40K examples used for UnPRM40K. For output aggregation methods, we compare our proposed HMR vote and WRF vote strategies with standard PRM vote and standard Majority Vote.

Results Figure 1 shows the accuracy of various PRM-based BoN aggregation strategies, alongside Majority Vote, on the MATH and GSMPlus datasets using both GPT-4o and Qwen2.5-Math-7B-Instruct reasoners. Across all configurations, UnPRM40K consistently outperforms SimPRM40K, demonstrating that uncertainty-driven PRM data generation is more effective than the similarity-driven approach. UnPRM40K also performs comparably to EpicPRM40K, which annotates incorrect solutions by identifying the first erroneous step, thereby validating the efficiency and effectiveness of our uncertainty-driven annotation method that locates the most uncertain error. As expected, RanPRM40K performs the worst, however, it still shows some improvement due to the correct labelling of correct solutions.

Figure 2 presents the results of various PRMs combined with different output aggregation strategies on the MATH and GSMPlus datasets, using the Qwen2.5-Math-7B-Instruct reasoner. Across all PRMs, both the WRF and HMR strategies consistently outperform standard Majority Vote and traditional PRM-based methods. Performance increases with the number of samples in every setting. Notably, when standard PRM methods underperform Majority Vote, employing HMR and WRF leads to substantial performance gains. Among the two uncertainty-aware aggregation strategies, WRF demonstrates greater robustness than HMR in most scenarios. These results highlight the effectiveness of uncertainty-aware output aggregation methods that integrate the complementary strengths of Majority Vote and PRMs. Notably, applying the WRF strategy to UnPRM40K yields the best performance among the three PRMs, indicating that our uncertainty-driven data generation approach is

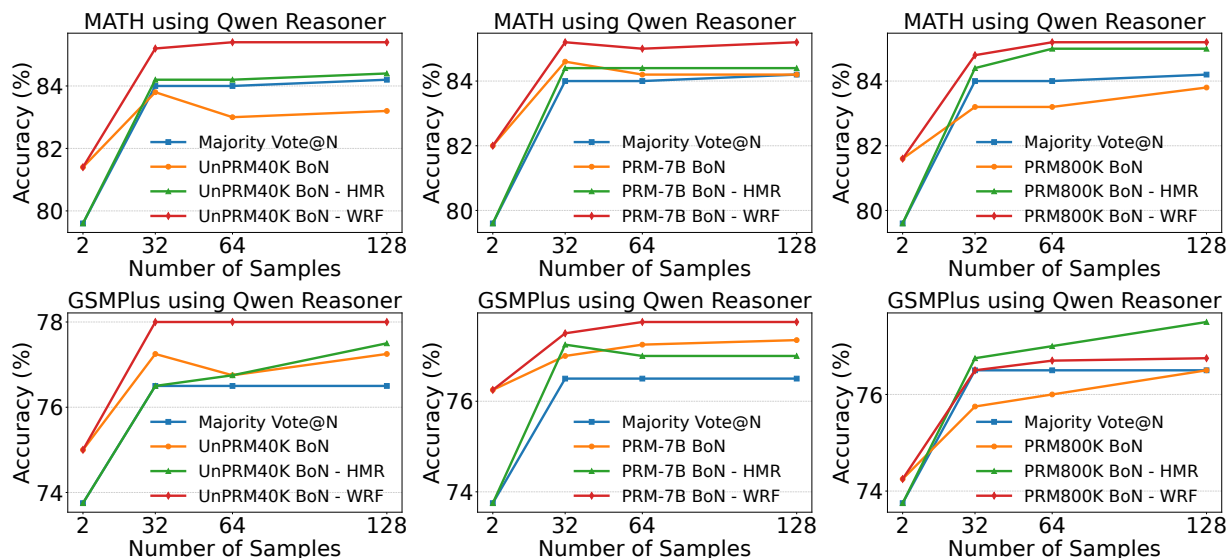


Figure 2: Evaluation results of different PRMs using diverse output aggregation strategies on MATH and GSMPlus datasets. Qwen Reasoner denotes Qwen2.5-Math-7B-Instruct Reasoner.

Model	Training Data Size	GSM8K	MATH	Olympiad Bench	Olympiad MATH	Average
Math-Shepherd-PRM-7B	445K	47.9	29.5	24.8	23.8	31.5
RLHFlow-PRM-Mistral-8B	273K	50.4	33.4	13.8	15.8	28.4
RLHFlow-PRM-Deepseek-8B	253K	38.8	33.8	16.9	16.9	26.6
EurusPRM-Stage2-7B	500K	47.3	35.7	21.2	20.9	31.3
Qwen2.5-Math-7B-RanPRM40K	40K	35.5	25.5	15.7	17.3	23.5
Qwen2.5-Math-7B-SimPRM40K	40K	51.2	38.5	29.5	27.4	36.7
Qwen2.5-Math-7B-UnPRM40K	40K	53.5	43.4	33.6	30.8	40.3
Qwen2.5-Math-7B-EpicPRM40K	40K	53.1	44.6	31.8	33.6	40.7
Qwen2.5-Math-7B-PRM800K	264K	68.2	62.6	50.7	44.3	56.5
Qwen2.5-Math-PRM-7B	1.8M	82.4	77.6	67.5	66.3	73.5

Table 1: Evaluation results on ProcessBench. We report the F1 score of the accuracies on erroneous and correct samples. Among these PRMs, only Qwen2.5-Math-7B-PRM800K is trained on the human annotation data.

particularly well-suited for enhancing uncertainty-aware aggregation methods like WRF vote.

PRM ProcessBench Evaluation

ProcessBench (Zheng et al. 2024) is a benchmark to measure the ability to identify erroneous steps in mathematical reasoning. It consists of 3,400 test cases, primarily focused on competition-level math problems. We test PRMs on it evaluate the step-level process errors identification ability.

Baselines. In addition to the above mentioned PRMs, we also compare with the following PRMs: Math-Shepherd-PRM-7B (Wang et al. 2024), RLHFlow-PRM-Mistral-8B (Dong et al. 2024), RLHFlow-PRM-Deepseek-8B (Dong et al. 2024), EurusPRM-Stage2-7B (Cui et al. 2025).

Results. The evaluation results on ProcessBench are presented in Table 1. UnPRM40k outperforms PRMs trained on automatically labelled datasets containing 200K–500K examples. In comparison to SimPRM40k, the results indicate that PRMs trained on similarity-driven data are less effective than those trained on uncertainty-driven data. As ex-

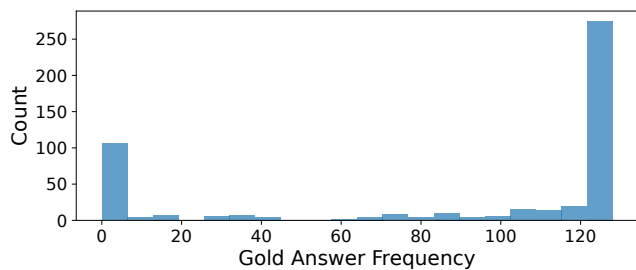
Algorithm	Verified Steps	Sampled Num.	Generated Tok.
Adaptive Binary Search	3144	104.98K	36.44M
Ours	1498 (-52%)	69.25K (-34%)	21.75M (-40%)

Table 2: Computational cost of two automated PRM data annotation algorithms when annotating the same 1500 solutions (460 correct solutions, 1040 incorrect solutions).

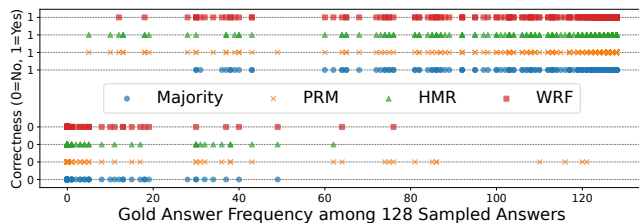
pected, RanPRM40k demonstrates limited ability to identify erroneous steps, resulting in a low F1 score, while EpicPRM40k, which is annotated using the first error step, performs slightly better than UnPRM40k. Qwen2.5-Math-7B-PRM800K, which benefits from high-quality human-annotated data, achieves strong performance with only 264K training examples, illustrating that data quality can substantially improve PRM performance. Scaling the training set size to 1.8M in Qwen2.5-Math-PRM-7B further boosts the F1 score on ProcessBench, highlighting the effectiveness of data scaling in PRM training.

6 Analysis

Computational Cost Analysis. Table 2 presents the computational cost analysis for two automated PRM data annotation algorithms applied to the same set of 1,500 solutions, comprising 460 correct and 1,040 incorrect cases. Annotation of correct solutions does not require any sampling, so the computational cost is primarily driven by the annotation of incorrect solutions. The Adaptive Binary Search method (Sun et al. 2025), used in our EpicPRM40K, annotates data by identifying the first erroneous step through a binary search process. In contrast, the Uncertainty-driven Search method, used in UnPRM40K and detailed in Algorithm 1, locates the most uncertain erroneous step for annotation. Both methods were run on a single A100 GPU.



(a) Visualisation of the distribution of gold answer frequencies in 128 outputs sampled from Qwen2.5-Math-7B-Instruct on MATH.



(b) Visualisation of correctness for four output aggregation methods (Majority, PRM, HMR, WRF) over 128 outputs across gold answer frequencies on MATH. Each method is shown in a separate lane, with points indicating whether predictions are correct (1) or incorrect (0) at each frequency. The reasoner is Qwen2.5-Math-7B-Instruct, and the PRM used is Qwen2.5-Math-7B-UnPRM40K.

Figure 3: (a) Distribution of gold answer frequencies and (b) correctness visualisation for aggregation methods.

The results demonstrate that our approach substantially reduces the number of verified steps, sampled instances, and generated tokens compared to the Adaptive Binary Search method. Not only is the Uncertainty-driven Search more cost-effective, but it also achieves comparable performance to the Adaptive Binary Search, as shown in Figure 1.

Output Aggregation Visualisation. Figure 3a displays the distribution of gold answer frequencies in 128 outputs sampled from Qwen2.5-Math-7B-Instruct on the MATH dataset. Among the 500 math questions, for more than half, the model consistently predicts the correct answer in all 128 samples. Conversely, for roughly 100 questions, the model fails to generate the correct answer even once within the 128 outputs. These results highlight a wide range of frequency distributions, suggesting a substantial proportion of questions where answer selection is non-trivial. In these cases, PRM-based methods are expected to help.

Figure 3b illustrates the accuracy of four output aggregation methods: Majority, PRM, HMR, and WRF—across varying gold answer frequencies. When the gold answer appears with high frequency (over 60 times), the Majority method reliably selects the correct answer, indicating strong model confidence. However, its performance deteriorates when the gold answer frequency drops below 20, often failing to recover the correct response. In contrast, the PRM method can still identify some correct answers even when the gold answer is infrequent (below 20), though it may make mistakes in high-frequency scenarios where the Majority method succeeds. The HMR and WRF strategies,

Sampling Model	Num of Samples	Avg Num Sample Step	Avg Error Step Uncertainty Rank
Llama-3.1-8B-Instruct	20,264	1.33	0.33
Qwen2.5-7B-Instruct	12,019	1.51	0.51
Mistral-7B-Instruct-v0.3	8,223	1.34	0.35

Table 3: Statistics of the annotated dataset UnPRM40K.

which integrate both Majority and PRM signals, notably reduce errors in the high-frequency regime where PRM alone underperforms. Moreover, WRF outperforms HMR in the mid-frequency range (between 20 and 40), yielding more correct predictions. These findings demonstrate the effectiveness of the two proposed uncertainty-aware aggregation methods in leveraging both model consensus and reasoning confidence to improve answer selection.

Error Step Uncertainty Analysis. UnPRM40k was generated and annotated using 3 different LLMs, with the dataset statistics summarised in Table 3. The average number of sampled steps reflects how many steps the uncertainty-driven search algorithm must verify on average to locate the most uncertain erroneous step, where a value of 1 represents optimal efficiency. Across all three models, the results are very close to 1, indicating that the uncertainty-driven search algorithm is highly efficient in pinpointing the most uncertain error. The average error step uncertainty rank indicates the uncertainty rank of the identified erroneous step, with 0 as the optimal value. Again, results are consistently near 0 across the three models, demonstrating that uncertainty serves as an effective proxy for locating errors. These findings are consistent with the intuition that LLMs are more likely to make mistakes where output is less certain.

7 Conclusion

This work presents an uncertainty-driven framework for constructing and annotating process reward data, addressing the key challenges in training effective PRMs for mathematical reasoning with LLMs. By leveraging uncertainty estimation, we efficiently generate and label high-quality step-level supervision data, greatly reducing annotation costs while maintaining performance. Furthermore, our proposed HMR and WRF aggregation strategies successfully combine the strengths of majority vote and PRM-based methods, leading to more robust and accurate answer selection. Extensive experiments demonstrate that our PRM data construction framework enhances both the efficiency and effectiveness of PRM training, and that the proposed output aggregation strategies are both effective and generalise well across different PRMs and mathematical reasoning tasks.

8 Limitations

While our uncertainty-aware aggregation methods incorporate answer frequency information, their performance can be influenced by the quality of the majority vote baseline. In scenarios where majority voting performs poorly, combining it with PRM-based approaches may not yield additional improvements and could potentially impact overall performance. However, we find that the proposed aggregation strategies are particularly effective when the majority vote performs better or comparable to the PRM.

References

- Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; Hesse, C.; and Schulman, J. 2021. Training Verifiers to Solve Math Word Problems. *CoRR*, abs/2110.14168.
- Cui, G.; Yuan, L.; Wang, Z.; Wang, H.; Li, W.; He, B.; Fan, Y.; Yu, T.; Xu, Q.; Chen, W.; Yuan, J.; Chen, H.; Zhang, K.; Lv, X.; Wang, S.; Yao, Y.; Han, X.; Peng, H.; Cheng, Y.; Liu, Z.; Sun, M.; Zhou, B.; and Ding, N. 2025. Process Reinforcement through Implicit Rewards. *CoRR*, abs/2502.01456.
- Dong, H.; Xiong, W.; Pang, B.; Wang, H.; Zhao, H.; Zhou, Y.; Jiang, N.; Sahoo, D.; Xiong, C.; and Zhang, T. 2024. RLHF Workflow: From Reward Modeling to Online RLHF. *Trans. Mach. Learn. Res.*, 2024.
- Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Yang, A.; Fan, A.; Goyal, A.; Hartshorn, A.; Yang, A.; Mitra, A.; Sra-vankumar, A.; Korenev, A.; Hinsvark, A.; Rao, A.; Zhang, A.; Rodriguez, A.; Gregerson, A.; Spataru, A.; Rozière, B.; Biron, B.; Tang, B.; Chern, B.; Caucheteux, C.; Nayak, C.; Bi, C.; Marra, C.; McConnell, C.; Keller, C.; Touret, C.; Wu, C.; Wong, C.; Ferrer, C. C.; Nikolaidis, C.; Allonsius, D.; Song, D.; Pintz, D.; Livshits, D.; Esiobu, D.; Choudhary, D.; Mahajan, D.; Garcia-Olano, D.; Perino, D.; Hupkes, D.; Lakomkin, E.; AlBadawy, E.; Lobanova, E.; Dinan, E.; Smith, E. M.; Radenovic, F.; Zhang, F.; Synnaeve, G.; Lee, G.; Anderson, G. L.; Nail, G.; Mialon, G.; Pang, G.; Cucurell, G.; Nguyen, H.; Korevaar, H.; Xu, H.; Touvron, H.; Zarov, I.; Ibarra, I. A.; Kloumann, I. M.; Misra, I.; Evtimov, I.; Copet, J.; Lee, J.; Geffert, J.; Vranes, J.; Park, J.; Mahadeokar, J.; Shah, J.; van der Linde, J.; Billock, J.; Hong, J.; Lee, J.; Fu, J.; Chi, J.; Huang, J.; Liu, J.; Wang, J.; Yu, J.; Bitton, J.; Spisak, J.; Park, J.; Rocca, J.; Johnstun, J.; Saxe, J.; Jia, J.; Alwala, K. V.; Upasani, K.; Plawiak, K.; Li, K.; Heafield, K.; Stone, K.; and et al. 2024. The Llama 3 Herd of Models. *CoRR*, abs/2407.21783.
- Gao, B.; Cai, Z.; Xu, R.; Wang, P.; Zheng, C.; Lin, R.; Lu, K.; Lin, J.; Zhou, C.; Xiao, W.; Hu, J.; Liu, T.; and Chang, B. 2024. LLM Critics Help Catch Bugs in Mathematics: Towards a Better Mathematical Verifier with Natural Language Feedback. *CoRR*, abs/2406.14024.
- Han, J.; Buntine, W. L.; and Shareghi, E. 2024. Towards Uncertainty-Aware Language Agent. In Ku, L.; Martins, A.; and Srikumar, V., eds., *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, 6662–6685. Association for Computational Linguistics.
- Hendrycks, D.; Burns, C.; Kadavath, S.; Arora, A.; Basart, S.; Tang, E.; Song, D.; and Steinhart, J. 2021. Measuring Mathematical Problem Solving With the MATH Dataset. In Vanschoren, J.; and Yeung, S., eds., *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*.
- Hu, E. J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; and Chen, W. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.
- Jiang, A. Q.; Sablayrolles, A.; Mensch, A.; Bamford, C.; Chaplot, D. S.; de Las Casas, D.; Bressand, F.; Lengyel, G.; Lample, G.; Saulnier, L.; Lavaud, L. R.; Lachaux, M.; Stock, P.; Scao, T. L.; Lavril, T.; Wang, T.; Lacroix, T.; and Sayed, W. E. 2023. Mistral 7B. *CoRR*, abs/2310.06825.
- Li, Q.; Cui, L.; Zhao, X.; Kong, L.; and Bi, W. 2024. GSM-Plus: A Comprehensive Benchmark for Evaluating the Robustness of LLMs as Mathematical Problem Solvers. In Ku, L.; Martins, A.; and Srikumar, V., eds., *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, 2961–2984. Association for Computational Linguistics.
- Lightman, H.; Kosaraju, V.; Burda, Y.; Edwards, H.; Baker, B.; Lee, T.; Leike, J.; Schulman, J.; Sutskever, I.; and Cobbe, K. 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.
- Luo, L.; Liu, Y.; Liu, R.; Phatale, S.; Lara, H.; Li, Y.; Shu, L.; Zhu, Y.; Meng, L.; Sun, J.; and Rastogi, A. 2024. Improve Mathematical Reasoning in Language Models by Automated Process Supervision. *CoRR*, abs/2406.06592.
- Sun, W.; Du, Q.; Cui, F.; and Zhang, J. 2025. An Efficient and Precise Training Data Construction Framework for Process-supervised Reward Model in Mathematical Reasoning. *CoRR*, abs/2503.02382.
- Tan, X.; Yao, T.; Qu, C.; Li, B.; Yang, M.; Lu, D.; Wang, H.; Qiu, X.; Chu, W.; Xu, Y.; and Qi, Y. 2025. AU-RORA: Automated Training Framework of Universal Process Reward Models via Ensemble Prompting and Reverse Verification. *CoRR*, abs/2502.11520.
- Uesato, J.; Kushman, N.; Kumar, R.; Song, H. F.; Siegel, N. Y.; Wang, L.; Creswell, A.; Irving, G.; and Higgins, I. 2022. Solving math word problems with process- and outcome-based feedback. *CoRR*, abs/2211.14275.
- Wang, P.; Li, L.; Shao, Z.; Xu, R.; Dai, D.; Li, Y.; Chen, D.; Wu, Y.; and Sui, Z. 2024. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. In Ku, L.; Martins, A.; and Srikumar, V., eds., *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, 9426–9439. Association for Computational Linguistics.
- Wang, X.; Wei, J.; Schuurmans, D.; Le, Q. V.; Chi, E. H.; Narang, S.; Chowdhery, A.; and Zhou, D. 2023. Self-Consistency Improves Chain of Thought Reasoning in Language Models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.
- Yang, A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Li, C.; Liu, D.; Huang, F.; Wei, H.; Lin, H.; Yang, J.; Tu, J.; Zhang, J.; Yang, J.; Yang, J.; Zhou, J.; Lin, J.; Dang, K.; Lu, K.; Bao, K.; Yang, K.; Yu, L.; Li, M.; Xue, M.; Zhang, P.; Zhu, Q.; Men, R.; Lin, R.; Li, T.; Xia, T.; Ren, X.; Ren, X.; Fan, Y.; Su, Y.; Zhang, Y.; Wan, Y.; Liu, Y.; Cui, Z.; Zhang,

Z.; and Qiu, Z. 2024a. Qwen2.5 Technical Report. *CoRR*, abs/2412.15115.

Yang, A.; Zhang, B.; Hui, B.; Gao, B.; Yu, B.; Li, C.; Liu, D.; Tu, J.; Zhou, J.; Lin, J.; Lu, K.; Xue, M.; Lin, R.; Liu, T.; Ren, X.; and Zhang, Z. 2024b. Qwen2.5-Math Technical Report: Toward Mathematical Expert Model via Self-Improvement. *CoRR*, abs/2409.12122.

Yang, Y.; Li, H.; Wang, Y.; and Wang, Y. 2023. Improving the Reliability of Large Language Models by Leveraging Uncertainty-Aware In-Context Learning. *CoRR*, abs/2310.04782.

Zhang, L.; Hosseini, A.; Bansal, H.; Kazemi, M.; Kumar, A.; and Agarwal, R. 2025a. Generative Verifiers: Reward Modeling as Next-Token Prediction. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Zhang, Z.; Zheng, C.; Wu, Y.; Zhang, B.; Lin, R.; Yu, B.; Liu, D.; Zhou, J.; and Lin, J. 2025b. The Lessons of Developing Process Reward Models in Mathematical Reasoning. *CoRR*, abs/2501.07301.

Zhao, X.; Kang, Z.; Feng, A.; Levine, S.; and Song, D. 2025. Learning to Reason without External Rewards. *CoRR*, abs/2505.19590.

Zheng, C.; Zhang, Z.; Zhang, B.; Lin, R.; Lu, K.; Yu, B.; Liu, D.; Zhou, J.; and Lin, J. 2024. ProcessBench: Identifying Process Errors in Mathematical Reasoning. *CoRR*, abs/2412.06559.