🔍 *TrajSelector*: Harnessing Latent Representations for Efficient and Effective Best-of-N in Large Reasoning Model

Anonymous ACL submission

Abstract

Large language models (LLMs) have shown remarkable progress in complex reasoning tasks, largely enabled by test-time scaling (TTS) paradigms that allocate additional compute during inference. Among these, external TTS (particularly the Best-of-N selection paradigm) yields scalable performance improvements by selecting from multiple independently generated reasoning trajectories. However, this approach faces a key limitation: the underutilization of the LLM's intrinsic latent representations. We introduce ***TrajSelector***, an efficient and effective Best-of-N framework that exploit the hidden states in the sampler LLM for process-level scoring. A lightweight verifier (with only 0.6B parameters) evaluates the quality of step-wise trajectory, and then aggregates these scores to identify the optimal reasoning trajectory. Our framework employs a fully data-driven, end-to-end training recipe that eliminates reliance on massive step-level annotations. Experiential results across five benchmarks demonstrate that *TrajSelector* delivers consistent performance gains. In Best-of-32 settings, it surpasses majority voting by 4.61% accuracy and outperforms existing 7B-scale process reward models by 4.31% to 12.21%, while using only 0.6B parameters and requiring lower GPU memory footprint.

1 Introduction

Large language models (LLMs) have achieved remarkable progress in domains such as mathematical reasoning over the past two years (Comanici et al., 2025; Shao et al., 2024; Wang et al., 2025b). A key driver of these advancements is the emergence of the test-time scaling paradigm (Guo et al., 2025a; Muennighoff et al., 2025; Xia et al., 2025), which boosts model performance by allocating additional computational resources during the inference phase. Test-time scaling (TTS) can be categorized into two complementary forms: (i) **Internal TTS** (Wei et al., 2023; Yu et al., 2025a) achieves this by extending the model's reasoning of longer chain-of-thoughts; (ii) **External TTS** (Zheng et al., 2025; Wang et al., 2025c; Fu et al., 2025; Chollet, 2024) does so by having the model exploring multiple reasoning solutions in parallel.

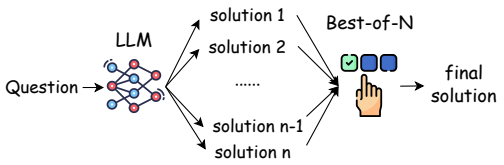


Figure 1: Best-of-N selection method illustration.

Our work centers on harnessing external TTS to enhance model performance. Although generating multiple independent solutions enables parallelized computation, the key challenge lies in answer aggregation, formally modeled as a **Best-of-N** selection problem (Figure 1). Existing approaches fall into two categories: (i) the use of independent process reward models (PRMs) as external verifiers to select among reasoning trajectories (Xia et al., 2024; Zou et al., 2025; Cui et al., 2025); (ii) the exploitation of intrinsic model states for correctness evaluation (Wang et al., 2023; Zuo et al., 2025; Wei et al., 2025; Wang et al., 2025a). However, both methods encounter key limitations: standalone PRMs often require computationally expensive deployments at the 7B scale (Zhang et al., 2025c; Zou et al., 2025), while state-based methods suffer from inconsistent performance and reliability issues across diverse tasks (Zhang et al., 2025b; Zhao et al., 2025).

We identify two primary bottlenecks in existing Best-of-N methods: (i) high-quality verifiers are typically large and computationally intensive, and their internal representations are often misaligned with those of the sampler LLM; (ii) training PRMs requires generally relies on costly step-level

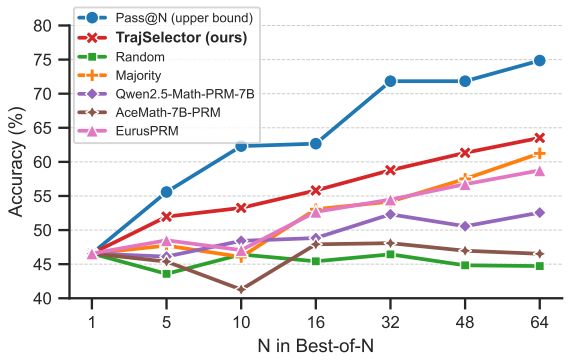


Figure 2: Best-of-N Scaling Curve. The y-axis represents the average accuracy of different methods across the 5 benchmarks in the experimental section. ***TrajSelector*** achieves robust performance improvement as N increases.

annotations. These constraints hinder the practicality of deploying process verifiers in real-world settings. Overcoming these challenges requires a lightweight verifier that effectively exploit the sampler LLM's intrinsic latent representation, coupled with a training strategy that eliminates the need for step-level supervision.

Motivated by these challenges, we propose TrajSelector, an efficient and effective Best-of-N framework that explot the latent representations inherent in the sampler LLM for solution selection. Our framework requires only a minimal-parameter LLM to function as a process verifier. The core idea is to repurpose the last hidden states from the sampler—rich in introspective signals—for step-level scoring. By coupling generation and evaluation at the representation level, TrajSelector facilitates accurate reasoning assessment with a minimal number of additional parameters, eliminating the need for a large standalone verifier.

We construct the training dataset using OpenR1-Math-220K (Hugging Face, 2025) and DeepMath-103K (He et al., 2025) as data sources. To eliminate reliance on process-level annotations, we adopt a data-driven training recipe for *TrajSelector*. During training, the sampler LLM is kept frozen, and only the lightweight process verifier is updated. Compared to full-scale PRM training, this approach demands significantly fewer computational resources while achieving superior Best-of-N performance. As illustrated in Figure 2, TrajSelector delivers consistent performance improvements across a range of Best-of-N settings ($N \in [1, 64]$). In the Best-of-32 setting, our method surpasses Majority Voting by 4.61% in accuracy and outperforms other process reward models by 4.31% to 12.21%, using only a 0.6B-parameter verifier.

Our main contributions are as follows:

- We empirically reveal that existing mainstream process reward models exhibit unstable performance in Best-of-N evaluation, often showing performance gaps compared to even the naive Majority Voting baseline. This finding motivates our design of a more reliable selection mechanism.
- We present ***TrajSelector***, which reuses the sampler model's step-final hidden states as self-reflective signals for a lightweight verifier to perform step-wise scoring and pooled trajectory evaluation, enabling efficient Best-of-N selection. This design achieves test-time gains with minimal parameter overhead.
- We successfully apply ***TrajSelector*** to two downstream applications (*offline data selection* and *RL reward modeling*). Positive experimental results further validate the practical utility of our approach.

2 Related Work

2.1 Test-Time Scaling (TTS)

Test-Time Scaling improves model reasoning by allocating additional compute at inference, and has recently become central with the success of OpenAI o1 (Jaech et al., 2024). Internal TTS scales the depth of reasoning within the model via extended chain-of-thought (CoT). Systems like OpenAI o1 (Jaech et al., 2024) and DeepSeek-R1 (Guo et al., 2025a) explicitly integrate structured reasoning traces, enabling decomposition and self-correction (Jin et al., 2024; Yeo et al., 2025). Complementarily, external TTS scales the breadth of inference by generating and evaluating multiple reasoning trajectories. The most prevalent paradigm among these involves parallel sampling from the model, followed by Best-of-N aggregation of solutions (Wang et al., 2023; Ma et al., 2025; Wang et al., 2025c; Zheng et al., 2025). Wang et al. (2023) and Zuo et al. (2025) employ a majority voting scheme to select the final solution from candidates. (Ma et al., 2025) demonstrates that multiple sampling without explicit reasoning outperforms a single reasoning process augmented with long chain-of-thought. Wang et al. (2025c) employs two linear layers for process scoring; however, it requires

reinforcement learning to dynamically adjust the parameters of the sampler LLM during training, which may induce catastrophic forgetting in the policy model. In contrast, our method does not require parameter modifications to the sampler LLM, thereby avoiding training failures arising from issues in the quality and distribution of post-training data.

2.2 Process Reward Model (PRM)

For selecting the most likely correct answer from multiple candidates, the mainstream approach employs PRMs for scoring and selection. Representative open efforts include Qwen2.5-Math-PRM-7B (Zhang et al., 2025c), which reports step-level gains over strong baselines at a comparable scale, and the PRM800K-tuned PRM Qwen2.5-Math-7B-PRM800K (Zheng et al., 2024), both emphasizing the value of large, process-labeled corpora for math reasoning. To reduce human labeling, works such as Math-Shepherd (Wang et al., 2024b) and ReasonEval-7B (Xia et al., 2024) automate stepwise assessment by external tools. Beyond stepwise scoring, ReasonFlux-PRM-7B (Zou et al., 2025) introduces trajectory-aware supervision for improving Best-of-N. Further, EurusPRM (Cui et al., 2025) advances implicit and online process rewards aiming to scalable PRMs for external TTS.

However, the aforementioned PRM methods are primarily designed to assess readability, coherence, and other qualitative aspects of CoT reasoning. In contrast, our approach is specifically tailored for Best-of-N selection, scoring reasoning processes by evaluating their impact on final answer correctness. Moreover, to ensure practical deployability, we implement this functionality using a lightweight 0.6B-parameter model, significantly reducing computational overhead while maintaining effectiveness.

3 Problem Statement

We consider the task of reasoning-aware answer generation using LLMs. Given a natural language query x, the model $\mathcal{M}$ is required to generate a final answer $\hat{r}$ accompanied by a sequence of T intermediate reasoning steps $\tau = (s_1, s_2, \ldots, s_T)$ that reflect the model's internal decision-making trajectory:

$$\tau \sim \mathcal{M}(\cdot|x), \quad \hat{r} \sim \mathcal{M}(\cdot|\tau, x) \tag{1}$$

Each reasoning step s_t corresponds to a discrete cognitive operation—such as deduction, retrieval, transformation, or hypothesis refinement—that cumulatively constructs the path from the query to the final answer.

The external test-time scaling (TTS) problem we address involves improving LLM performance post-training by leveraging multiple independently generated responses at inference time. Each data point is represented as a tuple (x, r), consisting of a query and its corresponding ground-truth answer. To assess the correctness of a generated response $\hat{r}$, we define a binary label:

$$y = \mathbb{I}[r = g(\hat{r})] \tag{2}$$

where $\mathbb{I}[\cdot]$ is the indicator function, returning 1 if the model's response exactly matches the ground truth r, and 0 otherwise. $g(\cdot)$ is used to extract the answer to the query from the response $\hat{r}$, often implemented in a rule-based manner.

The Best-of-N strategy serves as a foundational technique in external TTS. Rather than relying on a single output, the model generates N independent responses $\hat{r}_1, \hat{r}_2, \ldots, \hat{r}_N$, each evaluated for correctness with labels $y_1, \ldots, y_N$. Under this paradigm, the model $\mathcal{M}$ is considered to have answered the query x correctly if at least one $y_n = 1$.

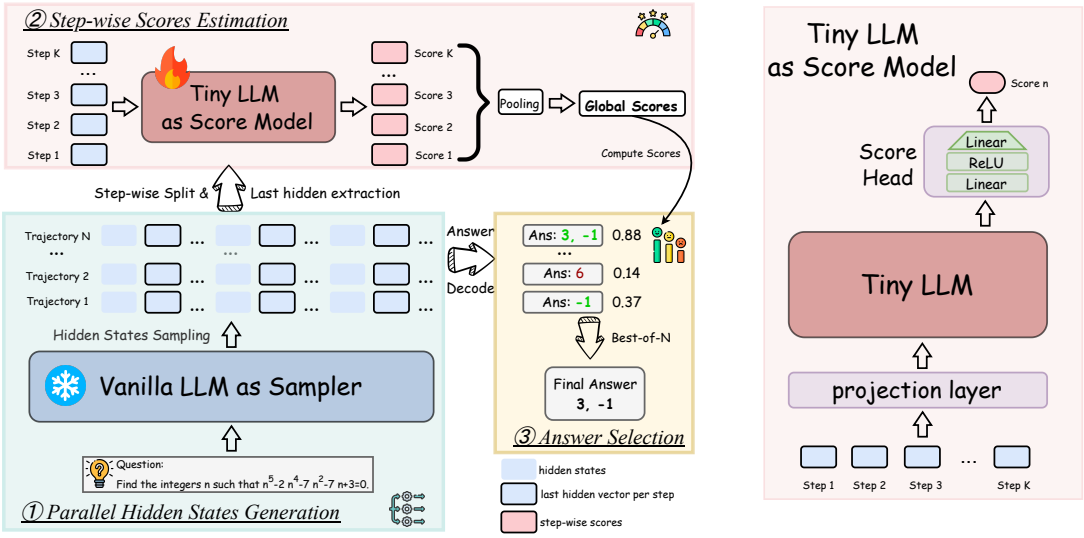
4 Method

Key motivations. We identify three key challenges in existing external TTS approaches: **(i)** insufficient utilization of self-reflective semantic representation in sampler models; **(ii)** high computational costs of large-scale PRMs; **(iii)** label noise from automated step-level annotations.

Our primary objective is to develop a compact yet effective response selection method for Best-of-N paradigm. The proposed approach trains a lightweight model (e.g., Qwen3-0.6B-Base) to score step-level latent reasoning, while keeping the primary reasoning model (e.g., Qwen3-8B) frozen. The step-level scores are then aggregated to derive a final response-level score. To improve training stability and reduce the impact of label noise, we introduce a customized classification loss coupled with pseudo-label crafting.

4.1 Overview

In this section, we propose a unified framework comprising a sampler model $\mathcal{M}_\phi$ for response generation, a lightweight process score model f_θ for



(a) Overview of method **TrajSelector** (b) Process Score Model Architecture

Figure 3: Overview Architecture

evaluation. This design exploits the sampler's intrinsic latent representations—specifically, its hidden states—to inform the scoring process. Another advantage of this approach lies in its efficiency: Best-of-N is achieved with minimal additional parameters, as the large sampler model remains frozen during training.

As illustrated in Figure 3(a), the framework ***TrajSelector*** operates in a three-stage pipeline. First, the sampler model generates multiple candidate responses in parallel for a given query, while the last hidden states are extracted as latent representations. Second, the reasoning trajectory is segmented into discrete steps, the hidden states of which are then passed to the compact process score model, which outputs a scalar estimate of reasoning quality. These step-level scores are then aggregated to yield a global score representing the overall quality of the response. Finally, the candidate with the highest global score is selected as the optimal output.

4.2 Process Score Model

The process score model assigns a score between 0 and 1 to each reasoning step within a reasoning trajectory, as shown in Figure 3(b). Architecturally, it comprises a tiny LLM and a score head. The LLM component is a 0.6B-parameter language model (Qwen3-0.6B-Base). As observed by Guo et al. (2025b); Fu et al. (2025), LLMs encode capabilities such as self-rewarding and self-reflection inherent in their hidden states. Accordingly, rather than relying on the generated tokens after a classification head for each reasoning step, the process score model takes the last hidden states as input. The score head is a shallow neural network consisting of two linear layers, producing a three-class classification output: *wrong*, *neutral*, and *right*. The *neutral* class serves as a noise-absorbing buffer, which will be elaborated upon in the following sections. The model's final score for a reasoning step is the predicted probability assigned to the *right* class by the scoring head.

From a runtime perspective, the process involves three components: (i) segmenting the reasoning trajectory into discrete steps; (ii) scoring each step using the process score model; and (iii) aggregating the individual scores to compute an overall response score.

Segmentation of reasoning steps. The reasoning trajectory, enclosed between the `<think>` and `</think>` tags in each response, is extracted for step segmentation. We divide this trajectory into discrete steps using the delimiter `'\n\n'`, thereby avoiding the need to introduce additional step-specific tokens or fine-tune the LLM to produce step-formatted outputs. After segmentation, the final token of each step is identified, and its last hidden state from the sampler model is used for scoring.

Step-wise scoring. For each reasoning step, the hidden states of its step token serves as the self-reflective signal of that step. These representations are concatenated into a sequence and input into the process score model. The model analyzes the sequence and produces a score between 0 and 1 for

each step via its score head.

Score aggregation. Step-wise scores are combined through a pooling operation to compute a global score representing the overall quality of the reasoning trajectory. Specifically, we employ an arithmetic mean as the pooling function.

Following aggregation, each response is assigned a global trajectory score, with higher scores reflecting higher-quality reasoning. The response with the highest global score is selected as the final Best-of-N output.

4.3 Training

A central challenge in training the process score model lies in the scarcity of high-quality labeled data for intermediate reasoning steps. To address this, we adopt the strategy introduced in FreePRM (Sun et al., 2025), casting the training process as a standard classification task that employs only the final outcome label as weak supervision. To further alleviate label noise, we incorporate an auxiliary mechanism designed to absorb uncertainty in the supervision signal. This approach obviates the need for labor-intensive, manually annotated intermediate steps and supports a data-driven paradigm in which the model learns to evaluate process quality autonomously.

Given a reasoning trajectory $\tau = (s_1, s_2, \cdots, s_T)$ consisting of T steps, and a ground truth label $y \in \{0, 1\}$ indicating whether the final answer is correct, we create a pseudo-label $\tilde{y}_t$ for each step $s_t \in \tau$ as defined in Equation 3.

$$\tilde{y}_t = y, \quad t = 1, 2, \ldots, T \tag{3}$$

However, this pseudo-labeling strategy introduces step-level noise, as not all steps within a trajectory that leads to a correct final answer are necessarily high-quality. To mitigate this, we extend the binary classification task by introducing a third class as a buffer (Sun et al., 2025). Accordingly, for each reasoning step s_t, the process score model predicts a probability distribution over three classes: right (p_t^r), wrong (p_t^w), and buffer (p_t^b), subject to the constraint defined in Equation 4.

$$\begin{aligned} (p_t^r, p_t^w, p_t^b) &= f_\theta(s_t) \\ s.t.\ p_t^r + p_t^w + p_t^b &= 1 \end{aligned} \tag{4}$$

where the distribution is obtained with a softmax transformation on the output embedding of the scoring head.

Accordingly, for a pseudo-label $\tilde{y}_t$, the training objective is formulated to encourage the behavior specified in Equation 5:

$$\begin{cases} p_t^r + p_t^b = 1, & \text{if } \tilde{y}_t = 1, \\ p_t^w + p_t^b = 1, & \text{if } \tilde{y}_t = 0. \end{cases} \tag{5}$$

This formulation allows the model to route ambiguous or noisy reasoning steps through the buffer class, reducing overfitting to potentially incorrect labels. Based on this objective, the final training loss is defined in Equation 6, where T denotes the number of reasoning steps in the trajectory τ:

$$\mathcal{L}(\theta|\tau) = -\frac{1}{T}\sum_{t=1}^{T}[\tilde{y}_t \log(p_t^r + p_t^b) + (1 - \tilde{y}_t)\log(p_t^w + p_t^b)] \tag{6}$$

During the training procedure, the sampler LLM $\mathcal{M}_\phi$ remains frozen.

5 Experiment

5.1 Experiment Settings

Training Dataset We construct the training corpus from two public datasets: OpenR1-Math-220k (Hugging Face, 2025) and DeepMath-103K (He et al., 2025). Each example in the public datasets contains a question and a ground-truth answer. We employ Qwen3-8B (Team, 2025) to perform reasoning and generate the corresponding thinking trajectory and response for each question. Each response is automatically labeled by comparing its final answer with the ground truth via `Math-Verify` (Kydlíček). Given the substantial imbalance where correct samples outnumber incorrect ones, we retain all incorrect samples as negatives and downsample correct samples to a 1:1 ratio for positives. The resulting dataset is then shuffled to form the final training set which contains 133K examples. Statistical information can be found in Appendix B.

Network Architecture. We employ Qwen3-8B as the frozen sampler model, initialize the weights of the process score model using Qwen3-0.6B-Base, implement hidden states mapping between the two LLMs through a linear layer, and construct the score head with two linear layers and a ReLU activation function (Agarap, 2019). Appendix E illustrates the network architecture pseudo-code.

Training Strategy. We employ DeepSpeed (Rajbhandari et al., 2020) and HuggingFace transformers (Wolf et al., 2020) to train the ***TrajSelector***. The hyper-parameters are shown in Appendix D.

Method	AMC-23 (/40)	AIME-24 (/30)	AIME-25 (/30)	BeyondAIME (/100)	HMMT-25 (/30)	BRUMO-25 (/30)	Avg (%)
			Best-of-32				
Pass@32 (Oracle)	**38**	**24**	**22**	**46**	**15**	**23**	**71.83**
Random Selection	34	17	12	17	8	16	46.44
Majority Voting	36	20	17	25	8	**18**	54.17
ReasonFlux-PRM-7B	35	19	16	21	8	16	50.86
Qwen2.5-Math-PRM-7B	35	**21**	16	23	8	16	52.31
Qwen2.5-Math-7B-PRM800K	36	19	15	20	7	15	49.44
ReasonEval-7B	35	20	15	20	7	**18**	51.25
Math-Shepherd	34	11	16	23	7	14	46.67
AceMath-7B-RM	35	19	13	21	6	16	48.08
EurusPRM	36	19	17	28	10	17	54.47
TrajSelector (ours)	**38**	**21**	**18**	**31**	**11**	**18**	**58.78**
			Best-of-16				
Pass@16 (Oracle)	**38**	**24**	**22**	**41**	**15**	**21**	**62.67**
Random Selection	33	19	13	20	6	13	45.42
Majority Voting	36	**20**	**16**	25	9	16	53.06
ReasonFlux-PRM-7B	35	17	13	22	9	**19**	50.47
Qwen2.5-Math-PRM-7B	36	17	13	23	6	18	48.83
Qwen2.5-Math-7B-PRM800K	35	17	14	19	9	18	49.97
ReasonEval-7B	36	**20**	13	22	7	16	49.97
Math-Shepherd	34	18	12	22	5	16	46.17
AceMath-7B-RM	35	18	13	20	7	16	47.91
EurusPRM	36	19	13	26	**10**	18	52.67
TrajSelector (ours)	**37**	**20**	**16**	**29**	**10**	18	**55.81**

Table 1: Experimental Results of Best-of-32 & Best-of-16

Benchmarks. We conduct Best-of-N experiments on the following benchmarks: AMC23, AIME24, AIME25, HMMT25, BRUMO25 (Balunović et al., 2025) and BeyondAIME (Seed et al., 2025). We prompt the sampler model to generate N candidate responses in parallel for a given question, then employ different methods to select one response from the candidates as the final answer, and subsequently evaluate the correctness of this final answer to compute the accuracy metric. The user instruction used for generating candidate responses is shown in Appendix A.

Baselines. We compare our method with the following baseline methods: (1) **Random Selection**: Select a response randomly from the N candidates. (2) **Majority Voting** (Wang et al., 2023): Select the answer that appears most frequently as the final answer. (3) **Process Reward Model**: Select the response with the highest score computed by process reward model from the N candidates. We selected the following strong process reward models as baselines: Qwen2.5-Math-PRM-7B (Zhang et al., 2025c), Qwen2.5-Math-7B-PRM800K (Zheng et al., 2024), ReasonFlux-PRM-7B (Zou et al., 2025), ReasonEval-7B (Xia et al., 2024), Math-Shepherd (Wang et al., 2024b), AceMath-7B-RM (Liu et al., 2024) and EurusPRM (Cui et al., 2025; Sun et al., 2025). (4) **Pass@N**: A test is considered passed if at least one of the N candidate responses is correct. The results of this method represent the theoretical upper bound for Best-of-N selection.

5.2 Experiment Results

The main experimental results are presented in Table 1 and Table 2.

Effective. As observed from the experimental results, our method TrajSelector demonstrates superior performance and stronger robustness compared to other baselines across multiple well-known benchmarks, and it can exhibit consistent performance improvements under various Best-of-N settings. Specifically, in the Best-of-32 settings, the average accuracy of our method is 4.61% higher than that of Majority Voting, and 4.31% to 12.21% higher than that of other process reward model-based methods. Similar performance improvements have also been validated by experimental results under other Best-of-N settings.

Efficient. Beyond performance gains, our method achieves efficiency improvements in two key aspects: (i) **Reduced GPU memory usage**. Compared with the 7B PRM model in the baseline, our method reduces the memory occupied by the PRM from 4,330M to 565.5M. (ii) **Reduced inference consumption.** Compared with the 7B PRM model in the baseline, when running the evaluation set consisting of 30 questions (e.g. AIME24 and

Method	AMC-23 (/40)	AIME-24 (/30)	AIME-25 (/30)	BeyondAIME (/100)	HMMT-25 (/30)	BRUMO-25 (/30)	Avg (%)
			Best-of-1				
Pass@1	34	16	11	21	8	17	46.56
			Best-of-5				
Pass@5 (Oracle)	**37**	**21**	**14**	**31**	**11**	**17**	**55.58**
Random Selection	33	17	9	19	8	14	43.58
Majority Voting	36	15	**13**	23	7	**17**	47.72
ReasonFlux-PRM-7B	36	17	12	20	5	16	46.11
Qwen2.5-Math-PRM-7B	36	17	12	20	5	16	46.11
Qwen2.5-Math-7B-PRM800K	36	17	**13**	20	5	16	46.67
ReasonEval-7B	34	18	12	24	**9**	16	48.72
Math-Shepherd	36	16	11	20	5	15	44.44
AceMath-7B-RM	36	17	**13**	19	4	15	45.38
EurusPRM	36	18	**13**	21	7	16	48.50
TrajSelector (ours)	**37**	**19**	**13**	**26**	**9**	**17**	**51.97**
			Best-of-10				
Pass@10 (Oracle)	**37**	**21**	**19**	**38**	**12**	**21**	**62.31**
Random Selection	35	16	13	21	4	18	46.42
Majority Voting	36	**18**	12	23	4	15	46.06
ReasonFlux-PRM-7B	35	**18**	12	23	5	**19**	48.42
Qwen2.5-Math-PRM-7B	35	**18**	13	23	5	18	48.42
Qwen2.5-Math-7B-PRM800K	**37**	17	13	22	6	18	49.08
ReasonEval-7B	36	16	**15**	19	6	15	47.06
Math-Shepherd	35	17	11	21	5	17	45.86
AceMath-7B-RM	34	15	12	16	4	13	41.28
EurusPRM	36	17	13	19	4	18	47.06
TrajSelector (ours)	**37**	**18**	**15**	**27**	**9**	18	**53.25**

Table 2: Experimental Results of Best-of-1, Best-of-5 and Best-of-10

AIME25), the inference time of our method on the PRM is reduced from 54.99s to 7.02s.

5.3 Ablations & Analysis

Ablation Study on Loss Function. To absorb the noise information in pseudo-labels, our method modifies the standard BCELoss (Mao et al., 2023) function to design the score head as a three-class classification network. To validate the effectiveness of this design, we replace the score head with a binary classification network and employ the standard BCELoss function for training. The experimental results presented in Figure 4 demonstrate that the additional incorporation of a buffer probability into the design of the loss function can effectively mitigate noise in the training data and achieve enhanced performance.

Ablation Study on Sampler Model. To demonstrate the generalizability of our method across different model sizes, in addition to using Qwen3-8B as the frozen sampler model, we tested the performance of Qwen3-4B and Qwen3-14B models. Appendix H presents detailed experimental results, and these experimental findings are consistent with our conclusions.

Ablation Study on Score Model. Our method employs Qwen3-0.6B-Base as a tiny LLM to initialize the score model, which has not undergone

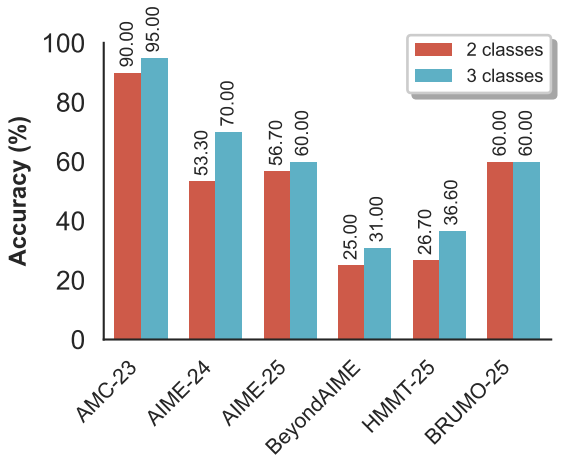


Figure 4: Ablation Study on Loss Function

human alignment training and thus possesses generalizability across a broader range of downstream tasks (Wang et al., 2024c). Here, we attempt to replace it with the Qwen3-0.6B model to demonstrate its effectiveness. The results presented in Figure 5 demonstrate that models not trained and aligned via RLHF (Bai et al., 2022) exhibit superior performance.

Larger N in Best-of-N. Although the range of N value selection in our main experimental results has already covered common parallel sampling quantities, we attempted to increase the N

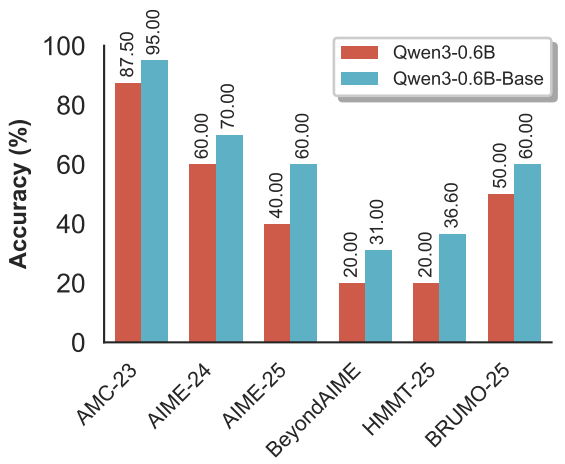


Figure 5: Ablation Study on Score Model

value to a larger extent in the Best-of-N experiments. The experimental results in Appendix I demonstrate that the effectiveness of our proposed method remains valid as the value of N increases.

More Model Family. To further validate the generalizability of our method across different LLM families, we attempted to replace the Qwen3 series models with the DeepSeek-R1-Distill-Qwen-7B and DeepSeek-R1-Distill-Llama-8B (Guo et al., 2025a). The experimental results in Appendix J demonstrate that our method remains effective when applied to different LLM families.

5.4 Downstream Application

Offline Data Selection. To demonstrate that the trained process score model can effectively distinguish between high-quality and low-quality thinking trajectory processes, we attempted to apply the process score model to the scenario of offline data selection. More details can be found in Appendix F.

RL reward modeling. We explore the application of TrajSelector as a lightweight verifier to provide reward signals for LLM reinforcement learning. Specifically, we treat the process scores generated by TrajSelector as step-level reward signals and integrate them as a plugin into the GRPO algorithm (Guo et al., 2025a) to address its inherent reward sparsity issue. This approach yields promising results, with detailed experimental setup and analysis provided in Appendix G.

6 Discussion

Advantages over Other PRMs Our process score model has a 0.6B parameter overhead, substantially smaller than that of mainstream 7B PRMs. Unlike conventional PRMs—where token-level information flow with the policy model requires tokenizer conversion—our approach uses hidden states for information exchange, effectively preserving adequate self-reflective signals from the sampler model.

Why Does the Method Work? We attribute TrajSelector's effectiveness to three key factors: (1) Correctness checking is simpler than problem-solving and requires no model as large as the sampler; (2) It fully exploits the sampler model's latent representations as self-reflective signals, enabling the score model to reuse part of its capabilities (detailed earlier); (3) Recent studies (Wang et al., 2025c; Park et al., 2025; Guo et al., 2025c; Fu et al., 2025; Zhao et al., 2025) confirm that such self-reflective signals reflect answer correctness, laying the foundation for our method.

Why not use BERT? While the heavily pre-trained encoder-only BERT (Devlin et al., 2019) is well-suited for numerical regression, our experiments showed that step counts in LLM-generated thinking trajectories often exceed its 512-token context limit—leading us to abandon this approach. Instead, the growing maturity of LLM-based regression research (Zhang et al., 2025c,a; Wang et al., 2024a) inspired us to adopt a tiny LLM as the score head.

7 Conclusion

In this paper, we introduced ***TrajSelector***, an efficient and effective framework for Best-of-N selection in large reasoning models. By exploiting the sampler LLM's intrinsic latent representations and integrating a lightweight process verifier, our approach enables step-wise scoring and trajectory aggregation without relying on costly step-level annotations or heavy standalone PRMs. The data-driven end-to-end training recipe, incorporating a noise-absorbing three-class loss, ensures robust learning and mitigates label noise. *TrajSelector* advances external TTS by highlighting the potential of latent representations for accessible inference. Future work could explore hybrid integration with internal TTS, extensions to non-math domains, and adaptive N based on query difficulty.

Limitations

Our research primarily focuses on the Best-of-N setting within external Test-time Scaling, which represents the most common scenario. However,

the potential for further integration with Monte Carlo search to unlock the full effectiveness of process scoring remains underexplored. To facilitate rigorous answer verification, we concentrate on training and evaluation in the mathematical domain; extending the benefits of this external Test-time Scaling to open-ended question answering domains warrants additional research. Furthermore, this paper only makes preliminary attempts at leveraging TrajSelector for selecting high-quality reasoning data, with more refined recipes awaiting future investigations.

Ethics Statement

The datasets (OpenR1-Math-220K (Hugging Face, 2025) and DeepMath-103K (He et al., 2025)) and models (Qwen series (Team, 2025)) employed in this study are all open-source, thereby incurring no risks associated with licensing. Furthermore, as our research is centered on the mathematical domain, it does not entail risks pertaining to human ethics and values.

References

Abien Fred Agarap. 2019. Deep learning using rectified linear units (relu). *Preprint*, arXiv:1803.08375.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, and 12 others. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *Preprint*, arXiv:2204.05862.

Mislav Balunović, Jasper Dekoninck, Ivo Petrov, Nikola Jovanović, and Martin Vechev. 2025. Matharena: Evaluating llms on uncontaminated math competitions. *Preprint*, arXiv:2505.23281.

François Chollet. 2024. Openai o3 breakthrough high score on arc-agi-pub. `https://arcprize.org/blog/oai-o3-pub-breakthrough`. Accessed: 2024-12-20.

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

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, Luke Marris, Sam Petulla, Colin Gaffney, Asaf Aharoni, Nathan Lintz, Tiago Cardal Pais, Henrik Jacobsson, Idan Szpektor, Nan-Jiang Jiang, and 3290 others. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *Preprint*, arXiv:2507.06261.

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Yichao Fu, Xuewei Wang, Yuandong Tian, and Jiawei Zhao. 2025. Deep think with confidence. *arXiv preprint arXiv:2508.15260*.

Etash Guha, Ryan Marten, Sedrick Keh, Negin Raoof, Georgios Smyrnis, Hritik Bansal, Marianna Nezhurina, Jean Mercat, Trung Vu, Zayne Sprague, Ashima Suvarna, Benjamin Feuer, Liangyu Chen, Zaid Khan, Eric Frankel, Sachin Grover, Caroline Choi, Niklas Muennighoff, Shiye Su, and 31 others. 2025. Openthoughts: Data recipes for reasoning models. *Preprint*, arXiv:2506.04178.

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

Jizhou Guo, Zhaomin Wu, Hanchen Yang, and Philip S. Yu. 2025b. Mining intrinsic rewards from llm hidden states for efficient best-of-n sampling. *Preprint*, arXiv:2505.12225.

Jizhou Guo, Zhaomin Wu, Hanchen Yang, and Philip S. Yu. 2025c. Mining intrinsic rewards from llm hidden states for efficient best-of-n sampling. *Preprint*, arXiv:2505.12225.

Nathan Habib, Clémentine Fourrier, Hynek Kydlíček, Thomas Wolf, and Lewis Tunstall. 2023. Lighteval: A lightweight framework for llm evaluation.

Zhiwei He, Tian Liang, Jiahao Xu, Qiuzhi Liu, Xingyu Chen, Yue Wang, Linfeng Song, Dian Yu, Zhenwen Liang, Wenxuan Wang, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. 2025. Deepmath-103k: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning.

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. *Preprint*, arXiv:2103.03874.

Hugging Face. 2025. Open r1: A fully open reproduction of deepseek-r1.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, and 1 others. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Mingyu Jin, Qinkai Yu, Dong Shu, Haiyan Zhao, Wenyue Hua, Yanda Meng, Yongfeng Zhang, and Mengnan Du. 2024. The impact of reasoning step length on large language models. *arXiv preprint arXiv:2401.04925*.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, SOSP '23, page 611–626, New York, NY, USA. Association for Computing Machinery.

Hynek Kydlíček. Math-Verify: Math Verification Library.

Zihan Liu, Yang Chen, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. 2024. Acemath: Advancing frontier math reasoning with post-training and reward modeling. *arXiv preprint*.

Wenjie Ma, Jingxuan He, Charlie Snell, Tyler Griggs, Sewon Min, and Matei Zaharia. 2025. Reasoning models can be effective without thinking. *Preprint*, arXiv:2504.09858.

Anqi Mao, Mehryar Mohri, and Yutao Zhong. 2023. Cross-entropy loss functions: Theoretical analysis and applications. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 23803–23828. PMLR.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. 2025. s1: Simple test-time scaling. *Preprint*, arXiv:2501.19393.

Young-Jin Park, Kristjan Greenewald, Kaveh Alim, Hao Wang, and Navid Azizan. 2025. Know what you don't know: Uncertainty calibration of process reward models. *Preprint*, arXiv:2506.09338.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. Zero: memory optimizations toward training trillion parameter models. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '20. IEEE Press.

ByteDance Seed, :, Jiaze Chen, Tiantian Fan, Xin Liu, Lingjun Liu, Zhiqi Lin, Mingxuan Wang, Chengyi Wang, Xiangpeng Wei, Wenyuan Xu, Yufeng Yuan, Yu Yue, Lin Yan, Qiying Yu, Xiaochen Zuo, Chi Zhang, Ruofei Zhu, Zhecheng An, and 255 others. 2025. Seed1.5-thinking: Advancing superb reasoning models with reinforcement learning. *Preprint*, arXiv:2504.13914.

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

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

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

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

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024a. Improving text embeddings with large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11897–11916, Bangkok, Thailand. Association for Computational Linguistics.

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.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. 2025a. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. *Preprint*, arXiv:2506.01939.

Xinming Wang, Jian Xu, Aslan H Feng, Yi Chen, Haiyang Guo, Fei Zhu, Yuanqi Shao, Minsi Ren, Hongzhu Yi, Sheng Lian, and 1 others. 2025b. The hitchhiker's guide to autonomous research: A survey of scientific agents. *Authorea Preprints*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*.

Zhichao Wang, Bin Bi, Shiva Kumar Pentyala, Kiran Ramnath, Sougata Chaudhuri, Shubham Mehrotra, Zixu, Zhu, Xiang-Bo Mao, Sitaram Asur, Na, and Cheng. 2024c. A comprehensive survey of llm alignment techniques: Rlhf, rlaif, ppo, dpo and more. *Preprint*, arXiv:2407.16216.

Zixiao Wang, Yuxin Wang, Xiaorui Wang, Mengting Xing, Jie Gao, Jianjun Xu, Guangcan Liu, Chenhui Jin, Zhuo Wang, Shengzhuo Zhang, and Hongtao Xie. 2025c. Test-time scaling with reflective generative model. *Preprint*, arXiv:2507.01951.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models. *Preprint*, arXiv:2201.11903.

Lai Wei, Yuting Li, Chen Wang, Yue Wang, Linghe Kong, Weiran Huang, and Lichao Sun. 2025. Unsupervised post-training for multi-modal llm reasoning via grpo. *arXiv preprint arXiv:2505.22453*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Shijie Xia, Xuefeng Li, Yixin Liu, Tongshuang Wu, and Pengfei Liu. 2024. Evaluating mathematical reasoning beyond accuracy. *arXiv preprint arXiv:2404.05692*.

Shijie Xia, Yiwei Qin, Xuefeng Li, Yan Ma, Run-Ze Fan, Steffi Chern, Haoyang Zou, Fan Zhou, Xiangkun Hu, Jiahe Jin, Yanheng He, Yixin Ye, Yixiu Liu, and Pengfei Liu. 2025. Generative ai act ii: Test time scaling drives cognition engineering. *Preprint*, arXiv:2504.13828.

Edward Yeo, Yuxuan Tong, Morry Niu, Graham Neubig, and Xiang Yue. 2025. Demystifying long chain-of-thought reasoning in llms. *arXiv preprint arXiv:2502.03373*.

Bin Yu, Hang Yuan, Haotian Li, Xueyin Xu, Yuliang Wei, Bailing Wang, Weizhen Qi, and Kai Chen. 2025a. Long-short chain-of-thought mixture supervised fine-tuning eliciting efficient reasoning in large language models. *Preprint*, arXiv:2505.03469.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, and 16 others. 2025b. Dapo: An open-source llm reinforcement learning system at scale. *Preprint*, arXiv:2503.14476.

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

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. 2025a. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*.

Yanzhi Zhang, Zhaoxi Zhang, Haoxiang Guan, Yilin Cheng, Yitong Duan, Chen Wang, Yue Wang, Shuxin Zheng, and Jiyan He. 2025b. No free lunch: Rethinking internal feedback for llm reasoning. *Preprint*, arXiv:2506.17219.

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

Wenting Zhao, Pranjal Aggarwal, Swarnadeep Saha, Asli Celikyilmaz, Jason Weston, and Ilia Kulikov. 2025. The majority is not always right: Rl training for solution aggregation. *Preprint*, arXiv:2509.06870.

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*.

Tong Zheng, Hongming Zhang, Wenhao Yu, Xiaoyang Wang, Runpeng Dai, Rui Liu, Huiwen Bao, Chengsong Huang, Heng Huang, and Dong Yu. 2025. Parallel-r1: Towards parallel thinking via reinforcement learning. *Preprint*, arXiv:2509.07980.

Jiaru Zou, Ling Yang, Jingwen Gu, Jiahao Qiu, Ke Shen, Jingrui He, and Mengdi Wang. 2025. Reasonflux-prm: Trajectory-aware prms for long chain-of-thought reasoning in llms. *Preprint*, arXiv:2506.18896.

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

A Evaluation Prompt Template

The prompt template used is sourced from LightEval (Habib et al., 2023).

Evaluation Prompt Template

Solve the following math problem efficiently and clearly. Please reason step by step, and put your final answer within \boxed{}.

B Training Dataset Statistics

The size of the training dataset is shown in the Table 3. As can be seen, the ratio of positive to negative samples we selected is maintained at 1:1.

Data Source	Positive	Negative	Total
DeepMath-103K	11,592	11,592	23,184
OpenR1-Math-220k	55,258	55,258	110,516
Overall	**66,850**	**66,850**	**133,700**

Table 3: Training Dataset Size

C Sampling Parameters

Table 4 presents the sampling parameters used during the policy model's inference time in the dataset construction process and benchmark evaluation. These parameter values are all derived from the official Best Practices recommended by Qwen3-8B (Team, 2025). The inference service for the LLM is deployed using vLLM (Kwon et al., 2023) as the foundation infrastructure.

Parameter	Value
Enable Thinking	True
Temperature	0.6
Top-p	0.95
Max tokens	10,000
Top-k	20
Min-p	0

Table 4: Sampling Parameters

D Training Parameters

Table 5 presents the training hyper-parameters used in the training process.

Parameter	Value
Samples	133K
Trainable Part	Full model
Learning Rate	1×10^{-4}
Epoch	3
Optimizer	AdamW
DeepSpeed	Zero2
Weight Decay	0.1
Max Seq.Length	10,000 tokens
Per-device Batch Size	2
Gradient Accumulation	4
Max tokens	10,000
Mixed Precision	bfloat16
GPU Nums	8× NVIDIA H100

Table 5: Training Parameters

E Pseudo Code of Network Architecture

Figure 6 shows the PyTorch implementation pseudo code for the TrajSelector network architecture.

```
class TrajSelector(PreTrainedModel):
  # this is the annotation
  def __init__(self, config):
    ...
    self.policy_model = load(model_name="Qwen3-8B")
    self.prm = load(model_name="Qwen3-0.6B-Base")
    self.projection = nn.Linear(
      policy_hidden_states_dim,
      prm_hidden_states_dim
    )
    self.score_head = nn.Sequential(
      nn.Linear(prm_hidden_states_dim, d),
      nn.ReLU(),
      nn.Linear(d, num_labels),
    )
    ...
```

Figure 6: PyTorch style code of TrajSelector.

F Offline Data Selection

We used OpenThoughts-114K (Guha et al., 2025) as the source dataset, from which 1K samples were selected as the supervised fine-tuning (SFT) dataset via different data selection strategies. Subsequently, we performed SFT on the Qwen2.5-14B-Instruct (Team, 2024) model; the performance of the fine-tuned model was evaluated to assess the effectiveness of the various data selection strategies. Additionally, we employed two datasets for comparison: s1K (Muennighoff et al., 2025) (a carefully human-curated dataset) and a randomly selected dataset. All training strategies and parameter selections follow the settings in the s1 (Muennighoff et al., 2025), and the evaluation methods strictly adhere to the default strategy of the open-r1 (Hugging Face, 2025) evaluation suite. The rationale for this approach is that these experimental protocols have been explored in recent related studies (Zou et al., 2025; Yu et al., 2025a). The experimental results are presented in Table 6.

Dataset	MATH500	AIME24	AIME25	GPQA
random	71.6	16.7	20.0	34.8
s1K	78.8	40.0	33.3	41.4
Math-Shepherd-PRM-7B	67.8	13.3	6.7	33.3
Qwen2.5-Math-PRM-7B	73.2	26.7	20.0	39.4
ReasonFlux-PRM-7B	84.8	40.0	33.3	47.5
TrajSelector (our)	**86.4**	**43.3**	**43.3**	**53.5**

Table 6: Offline Data Selection Evaluation

As observed from the experimental results, the training samples selected by TrajSelector are more effective in improving model performance, which reflects TrajSelector's capability to assess the quality of thinking trajectories.

G Application in Reinforcement Learning

We apply TrajSelector as a lightweight process reward model in LLM reinforcement learning. Specifically, we employ Qwen3-8B as the policy model and modify the GRPO algorithm (Guo et al., 2025a) to integrate the original outcome-based reward (r_{out}) with the process-level rewards ($\hat{r}$) generated by our method, yielding a more fine-grained token-level reward signal:

$$r_t = \beta \cdot \hat{r} + (1 - \beta) \cdot r_{\text{out}}, \tag{7}$$

where $\hat{r}$ denotes the process reward at token position t produced by TrajSelector, r_{out} represents the outcome reward from GRPO, and $\beta = 0.5$ is a weighting hyperparameter that balances the two reward components.

G.1 Experimental Setup

We initialize the policy model from Qwen3-8B and employ GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021) as training data. The model is trained using the modified GRPO (Shao et al., 2024) and DAPO (Yu et al., 2025b) algorithm with our hybrid reward function. Other experimental settings follow those in SimpleRL-Zoo (Zeng et al., 2025).

G.2 Results

Table 7 presents the performance comparison between the baselines and our modified version integrated with TrajSelector. We evaluate the models on four mathematical reasoning benchmarks: AIME24, AIME25, BeyondAIME, and BRUMO25. The evaluation metric is the average accuracy over 32 runs, denoted as $avg@32$.

Table 7: Performance comparison on reinforcement learning benchmarks.

Method	AIME24	AIME25	BeyondAIME	BRUMO25
Qwen3-8B	64.4	58.4	38.0	49.4
GRPO	77.6	64.2	39.0	56.9
DAPO	79.0	66.5	40.6	58.4
GRPO + **ours**	**80.2**	**67.8**	**41.5**	**59.4**
DAPO + **ours**	**82.3**	**68.7**	**42.8**	**60.8**

The results demonstrate that incorporating fine-grained process rewards from TrajSelector into the reinforcement learning algorithm leads to improved performance, addressing the reward sparsity issue inherent in outcome-only reinforcement learning approaches.

H Ablation Study on Sampler Model

To explore the feasibility of using models with more diverse scales as frozen sampler models, we conducted experimental evaluations using Qwen3-4B and Qwen3-14B respectively as the models for response generation. The experimental results are presented in the Table 8. From the experimental results, the adoption of TrajSelector still yields benefits compared to Majority Voting method.

I Larger N in Best-of-N

The experimental results for Best-of-48 and Best-of-64 are presented in the Table 9. The evaluation setup is consistent with Section 5.1 in the main experiment.

From the experimental results, TrajSelector maintains stable performance and continues to yield benefits when N takes larger values, demonstrating a scaling trend where its effectiveness improves as N increases.

J Generalization to More Model Families

To address concerns regarding the generalizability of our architecture across different model families, we conducted two supplementary experiments following the training and evaluation protocols described in this paper. Specifically, we evaluate TrajSelector on Best-of-64 selection using two additional sampler models:

- **DeepSeek-R1-Distill-Qwen-7B**: A 7B model based on the Qwen2.5 architecture.
- **DeepSeek-R1-Distill-Llama-8B**: An 8B model based on the Llama3 architecture.

These experiments demonstrate that TrajSelector is not constrained to specific model architectures, tokenizers, or pre-trained corpora, and consistently achieves superior performance compared to baseline methods.

J.1 Results on DeepSeek-R1-Distill-Qwen-7B

Table 10 presents the Best-of-64 results using DeepSeek-R1-Distill-Qwen-7B as the sampler model. Our method achieves the best performance across all five benchmarks, substantially outperforming both Majority Voting and existing 7B-scale PRMs.

Method	AMC-23 (/40)	AIME-24 (/30)	AIME-25 (/30)	BeyondAIME (/100)	HMMT-25 (/30)	BRUMO-25 (/30)	Avg (%)
Best-of-32 + Qwen3-4B as Sampler							
Pass@32 (Oracle)	**39**	**24**	**21**	**50**	**17**	**22**	**71.25**
Majority Voting	**38**	21	16	26	12	17	56.83
TrajSelector (ours)	**38**	**22**	**17**	**30**	**12**	**18**	**59.17**
Best-of-32 + Qwen3-14B as Sampler							
Pass@32 (Oracle)	**39**	**24**	**22**	**53**	**17**	**24**	**73.42**
Majority Voting	39	21	14	34	13	21	60.25
TrajSelector (ours)	**39**	**23**	**18**	**33**	**13**	**19**	**67.86**

Table 8: Experimental Results of Best-of-32 on Qwen3-4B and Qwen3-14B

Method	AMC-23 (/40)	AIME-24 (/30)	AIME-25 (/30)	BeyondAIME (/100)	HMMT-25 (/30)	BRUMO-25 (/30)	Avg (%)
Best-of-48							
Pass@48 (Oracle)	**38**	**24**	**24**	**46**	**16**	**23**	**71.83**
Random Selection	32	17	13	19	7	14	44.83
Majority Voting	36	22	17	25	11	19	57.50
ReasonFlux-PRM-7B	36	18	15	21	8	16	50.17
Qwen2.5-Math-PRM-7B	35	19	15	23	7	16	50.08
Qwen2.5-Math-7B-PRM800K	36	18	15	20	9	16	50.56
ReasonEval-7B	35	21	14	20	8	16	50.69
Math-Shepherd	34	11	14	23	7	13	43.00
AceMath-7B-RM	35	20	11	21	5	16	46.97
EurusPRM	36	20	17	27	12	18	56.72
TrajSelector (our)	**38**	**22**	**19**	**33**	**12**	**19**	**61.33**
Best-of-64							
Pass@64 (Oracle)	**39**	**25**	**24**	**55**	**17**	**23**	**74.86**
Random Selection	36	15	13	25	4	14	44.72
Majority Voting	37	23	17	35	13	19	61.25
ReasonFlux-PRM-7B	34	20	17	29	9	17	54.00
Qwen2.5-Math-PRM-7B	34	20	16	27	9	16	52.56
Qwen2.5-Math-7B-PRM800K	35	19	16	28	9	16	52.58
ReasonEval-7B	35	18	15	22	7	16	49.36
Math-Shepherd	33	17	17	22	7	15	48.53
AceMath-7B-RM	33	16	14	20	5	18	46.52
EurusPRM	37	21	18	30	11	19	58.75
TrajSelector (our)	**39**	**23**	**19**	**37**	**12**	**20**	**63.52**

Table 9: Experimental Results of Best-of-48 & Best-of-64

Method	AIME24	AIME25	BeyondAIME	HMMT25	BRUMO25
+ Majority Voting	83.3	70.0	48.0	46.7	73.3
+ ReasonFlux-PRM-7B	66.7	53.3	25.0	23.3	53.3
+ Qwen2.5-Math-PRM-7B	63.3	50.0	29.0	30.0	46.7
+ Qwen2.5-Math-7B-PRM800K	63.3	53.3	28.7	29.0	50.0
+ TrajSelector (ours)	**90.0**	**76.7**	**51.0**	**53.3**	**73.3**

Table 10: Best-of-64 results on DeepSeek-R1-Distill-Qwen-7B across mathematical reasoning benchmarks.

Method	AIME24	AIME25	BeyondAIME	HMMT25	BRUMO25
+ Majority Voting	80.0	63.3	45.0	46.7	63.3
+ ReasonFlux-PRM-7B	56.7	46.7	22.0	20.0	46.7
+ Qwen2.5-Math-PRM-7B	53.3	43.3	22.0	23.3	50.0
+ Qwen2.5-Math-7B-PRM800K	56.7	43.3	23.0	26.7	50.0
+ TrajSelector (ours)	**86.7**	**73.3**	**48.0**	**46.7**	**67.7**

Table 11: Best-of-64 results on DeepSeek-R1-Distill-Llama-8B across mathematical reasoning benchmarks.

J.2 Results on DeepSeek-R1-Distill-Llama-8B

Table 11 shows the Best-of-64 results using DeepSeek-R1-Distill-Llama-8B as the sampler model. Results are reported as accuracy percentages. Despite the different architecture (Llama3 vs. Qwen2.5), TrajSelector maintains its effectiveness and achieves the best performance across all benchmarks.

J.3 Discussion

The experimental results demonstrate several key findings:

Architecture agnostic. TrajSelector successfully generalizes across different model architectures (Qwen2.5 and Llama3), indicating that our approach is not tied to specific architectural choices or tokenization schemes.

Consistent superiority. Across both model families and all five benchmarks, TrajSelector consistently outperforms all baseline methods, including the widely-adopted Majority Voting and multiple strong 7B-scale PRMs.