EFFICIENT TEST-TIME SCALING OF MULTI-STEP REASONING BY PROBING INTERNAL STATES OF LARGE LANGUAGE MODELS

Jingwei Ni[1 ◇], **Ekaterina Fadeeva**[1 ◇], **Tianyi Wu**[2◇]
Mubashara Akhtar[1], **Jiaheng Zhang**[2], **Elliott Ash**[1], **Markus Leippold**[4]
Timothy Baldwin[3 5], **See-Kiong Ng**[1], **Artem Shelmanov**[3 †], **Mrinmaya Sachan**[1 †]
[1]ETH Zurich, [2]National University of Singapore, [3]MBZUAI,
[4]University of Zurich, [5]The University of Melbourne
◇ Equal contribution. † Equal supervision.

ABSTRACT

LLMs can solve complex tasks by generating long, multi-step reasoning chains. Test-time scaling (TTS) can further improve LLM performance by sampling multiple variants of intermediate reasoning steps, verifying their correctness, and strategically choosing the best steps for continuation. However, existing verification approaches, such as Process Reward Models (PRMs), are computationally expensive, limited to specific domains, and require large-scale human or model-generated annotations. We propose a lightweight alternative for step-level reasoning verification based on probing the internal states of LLMs. We train a transformer-based probe that uses the internal states of the frozen LLM to estimate the credibility of its reasoning steps during generation. Annotation can be generated either by another larger LLM (e.g., DeepSeek-R1) or in a self-supervised manner by the original model itself. The probes are both effective and lightweight, containing fewer than 10M parameters. Across multiple domains, including mathematics, planning, and general knowledge question answering, our probes match or even exceed the performance of PRMs that are up to 810× larger. Our findings suggest that the internal states of LLMs encode their confidence in reasoning processes and can serve as reliable signals for reasoning step verification, offering a promising direction towards scalable and generalizable TTS and introspective LLMs.[1]

1 INTRODUCTION

Chain-of-thought (CoT) prompting has proven highly effective in eliciting the reasoning capabilities of Large Language Models (LLMs) to solve complex tasks (Wei et al., 2022). Recent post-training approaches further enhance this capability through reinforcement learning, where models are rewarded for producing responses that demonstrate chain-of-thought reasoning prior to emitting the final answer, leading to the range of latest Language Reasoning Models (LRMs) (DeepSeek-AI, 2025; Yang et al., 2025; Abdin et al., 2025).

Even the best LRMs generate incorrect reasoning trajectories and ultimately erroneous outputs. A single flawed reasoning step can derail the entire solution. Empirical evidence suggests, however, that allowing the model to attempt the task multiple times or exploring multiple candidate continuations at intermediate steps can increase the likelihood of reaching a correct answer – a paradigm known as **test-time scaling (TTS)** (Yao et al., 2023; Snell et al., 2024). Many TTS strategies rely on a scorer that evaluates the quality of individual reasoning steps and provides guidance to steer the reasoning process towards better solutions.

The contemporary approach to reasoning step verification is through process reward models (PRMs: Lightman et al., 2024; Zhang et al., 2025c). Originally designed to guide RL post-training, PRMs

[1]Code and data: `https://github.com/cant-access-rediska0123/uncertainty4reasoning`

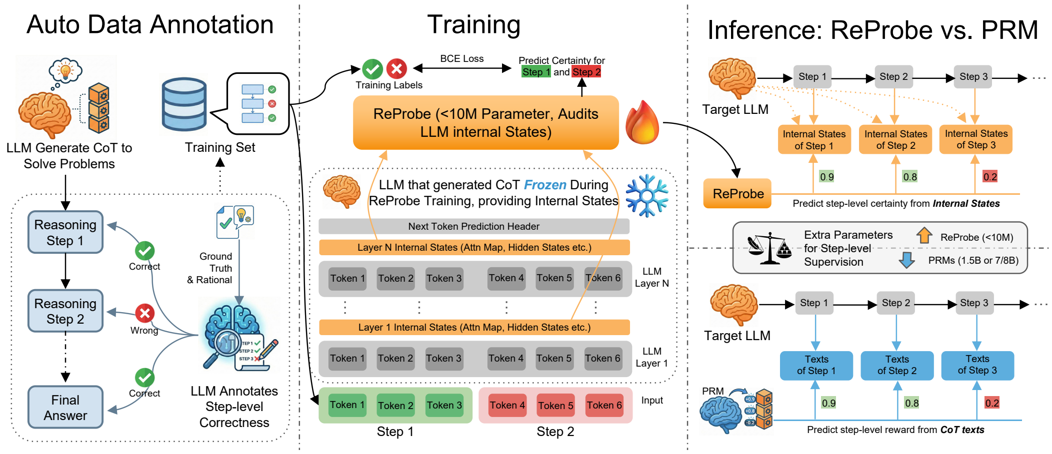

Figure 1: **Left**: Given a problem set, target LLM generates CoTs and an annotator LLM labels step-level correctness. **Middle**: Training ReProbe with target LLM providing internal states while frozen. **Right**: at inference, ReProbe and PRM monitors internal states and output text correspondingly. ReProbe has significantly fewer parameters.

were later found useful for TTS. However, PRM-based verification has key drawbacks. First, it requires expensive Monte-Carlo rollouts for training annotation and substantial computation for fine-tuning. Constructing such data is also challenging when step-level correctness cannot be inferred from the final result, such as in mathematical proofs (Azerbayev et al., 2023) and planning (Zheng et al., 2024). Second, deploying PRMs incurs substantial inference-time overhead, as they require an additional LLM, increasing GPU memory and compute cost. Third, PRMs are typically fine-tuned on narrow domains, such as mathematics, limiting generalization.

In another interesting line of work, the correctness of reasoning steps is assessed through uncertainty quantification (UQ) methods (Gal et al., 2016; Malinin & Gales, 2021). Unlike PRMs, which represent external knowledge, UQ assumes that a model's outputs and internal states provide information about the reliability of its generations (Fu et al., 2025; Yan et al., 2025). While computationally lightweight, unsupervised UQ methods typically exhibit limited effectiveness in related tasks such as hallucination detection (Vazhentsev et al., 2025; Shelmanov et al., 2025).

In this work, we ask: can the cost-efficiency of UQ methods be combined with the performance of PRMs? Prior work in UQ shows that LLM internal states encode rich credibility signals that can be efficiently extracted using learned classifiers for hallucination detection (Azaria & Mitchell, 2023; Su et al., 2024; Shelmanov et al., 2025). Building on this insight, we introduce a lightweight training-based **Reasoning Probe (ReProbe)** that uses LLM internal states to determine the credibility of generated reasoning steps. Unlike PRMs, which rely on external knowledge to evaluate generated reasoning traces, ReProbe operates directly on LLM internal signals produced during generation – hidden states, attention weights, and logits – making step-level reasoning verification substantially more efficient (see Fig 1).

Extensive in-domain (ID) and out-of-domain (OOD) evaluations highlight the advantages of our approach in three key aspects: **(1) Strong generalizable performance**: ReProbes achieve competitive or superior performance compared to PRMs in step-level verification and test-time scaling, especially when dealing with OOD reasoning tasks; **(2) Computational efficiency**: ReProbes outperform PRMs that are up to $150\times$ larger and remain competitive with PRMs up to $800\times$ larger. While typical PRMs contain 1.5–8B parameters, the probes trained in our work require fewer than 10M parameters; **(3) Training data efficiency**: unlike PRMs, which often rely on proprietary datasets (He et al., 2024), costly human annotations (Lightman et al., 2024), or consensus-filtering involving both Monte Carlo rollouts and LLM judgments (Zhang et al., 2025c), ReProbes can be trained on data annotated automatically in a self-supervised and cost-efficient manner.

2 Background

2.1 Test-Time Scaling

LLMs solve complex tasks by generating CoT reasoning traces (Guo et al., 2025). Given an input question $\mathbf{x}$, models produce a sequence of reasoning steps $\mathbf{r} = \{r_1, r_2, \ldots, r_T\}$ followed by a final answer $\mathbf{y} = g(\mathbf{r}, \mathbf{x})$, and we assume that we can effectively extract individual steps and the final answer. Although linear CoT reasoning improves final task performance, locally incorrect steps r_t can propagate and cause incorrect answers.

Test-time scaling (Yao et al., 2023; Snell et al., 2024) aims to further improve the performance of LLMs without retraining the entire model by allocating additional computation or using more sophisticated inference strategies. By leveraging multiple candidate solutions and verification signals, TTS facilitates the selection of a correct answer. Common approaches include best-of-N (BoN) sampling and beam search (Xie et al., 2023; Yao et al., 2023; Snell et al., 2025) (Fig 2).

Best-of-N (BoN) assumes that we sample not one, but N reasoning trajectories $\{\mathbf{r}^{(1)}, \mathbf{r}^{(2)}, \dots, \mathbf{r}^{(N)}\}$. Each trajectory $\mathbf{r}^{(j)}$ is evaluated using a scoring function $Q_{\text{BoN}}\left(\mathbf{r}^{(j)}\right)$, and the final answer is derived from the best-scoring trajectory:

$$\mathbf{r}^* = \arg\max_{1 \le j \le N} Q_{\text{BoN}}\left(\mathbf{r}^{(j)}\right), \quad \mathbf{y} = g(\mathbf{x}, \mathbf{r}^*).$$

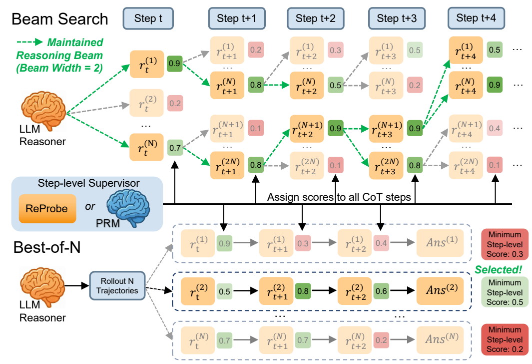

Figure 2: Illustration of the two prevalent test-time scaling methods: Best-of-N and Beam Search.

Beam Search (BS) (Xie et al., 2023; Snell et al., 2025), also known as the tree-of-thought with the breadth first search (Yao et al., 2023), assumes that we keep $B \ge 1$ best reasoning trajectories so far; and for each of them, at a new step, we sample N variants of continuations: $\{r_t^{(i,1)}, r_t^{(i,2)}, \dots, r_t^{(i,N)}\}, 1 \le i \le B$. Each of the continuations is assessed using a quality function $Q_{\text{bs}}\left(\mathbf{r}_{1:t}^{(i,k)}\right)$, and reasoning trajectories are expanded with the best continuations $\mathbf{r}_{1:t}^{(i)} = \mathbf{r}_{1:t-1}^{(i)} \circ r_t^{(i,*)}$. After reaching the final step T, the answer is obtained from the best reasoning trajectory.

$$S'_t = \left\{ \mathbf{r}_{1:t-1}^{(i)} \circ r_t^{(i,k)} \;\middle|\; \mathbf{r}_{1:t-1}^{(i)} \in S^*_{t-1},\ 1 \le k \le N \right\},$$

$$S^*_t = \arg\max_{S \subseteq S'_t,\ |S| = B} \sum_{\mathbf{r}_{1:t}^{(i,k)} \in S} Q_{\text{bs}}\left(\mathbf{r}_{1:t}^{(i,k)}\right),$$

$$\mathbf{r}^* = \arg\max_{\mathbf{r}_{1:T}^{(i)} \in S^*_T} Q_{\text{bs}}\left(r_{1:T}^{(i)}\right), \ \mathbf{y} = g(\mathbf{x}, \mathbf{r}^*).$$

2.2 Verifying Reasoning Steps

The most prominent approach for assessing reasoning steps in both BoN and beam search test-time scaling to date has been PRMs (Lightman et al., 2024; Luo et al., 2024; Wang et al., 2024). PRMs are critics designed to estimate the quality of each partial reasoning state $\mathbf{r}_{1:t}$ by assigning a reward $R_{\text{PRM}}(\mathbf{r}_{1:t})$ that reflects the likelihood of the chain eventually leading to a correct solution. PRMs are typically implemented by a separate LLM trained to evaluate the plausibility and correctness of intermediate reasoning steps. The annotation for training PRMs comes from various sources, including crowdsourcing, self-consistency checking, and synthetic data generation pipelines.

In BS, the reward from a PRM is used as a step-wise quality function. In BoN, the score for the complete chain $\mathbf{r}^{(j)}$ can be obtained via a temporal aggregation of process rewards, e.g., minimum step score (Zhang et al., 2025c):

$$Q_{\text{BoN}}\left(\mathbf{r}^{(j)}\right) = \min_{1 \le t \le T^{(j)}} R_{\text{PRM}}\left(\mathbf{r}_{1:t}^{(j)}\right),$$

$$Q_{\text{bs}}\left(\mathbf{r}_{1:t}^{(i,k)}\right) = R_{\text{PRM}}\left(\mathbf{r}_{1:t}^{(i,k)}\right).$$

PRMs come with several limitations: (1) they are relatively large models, typically containing 1.5B-8B parameters, which leads to additional computational and memory overhead during inference;

(2) PRMs are often domain-specific, being trained for tasks such as mathematical reasoning, and exhibit limited generalization to unseen domains.

Recent work also suggests using confidence scores for the verification of reasoning steps or entire reasoning trajectories as a drop-in replacement for PRMs (Fu et al., 2025): the lower the uncertainty, the more trustworthy is the step. This is motivated by their efficiency and generalization.

3 Verifying Reasoning Steps via Probing Internal States of LLMs

ReProbe model. We construct a reasoning probe as a *plug-and-play* module on top of a frozen LLM that outputs a probability of how likely the current reasoning step is correct. While PRMs rely solely on generated tokens, our probe introspectively leverages richer per-token features extracted from the base LLM's internal states, which allows it to be both efficient and effective (see Fig 1).

The feature extractor $F(\mathbf{x}, \mathbf{r}_{1:t-1}, r_t)$ can leverage various internal signals at each token position. Specifically, we investigate two sets of features: (1) **Attn+Logit**: attention weights over the 5 preceding tokens from all layers and logits of the top-K candidate generations; and (2) **Hidden states** from all layers. We select these feature sets because they have proven effective for factual hallucination detection (Shelmanov et al., 2025; Azaria & Mitchell, 2023). However, their role in reasoning supervision has been unexplored.

The architecture of ReProbe is as follows:

1. **Linear feature projection layer** that adjusts the dimension of token-level features:
$$\mathbf{h}_0 = \mathrm{Linear}_{\mathrm{proj}}(F(\mathbf{x}, \mathbf{r}_{1:t-1}, r_t)).$$
2. **Transformer layers:** a stack of L layers that capture contextual dependencies:
$$\mathbf{h}_l = \mathrm{Transformer}_l(\mathbf{h}_{l-1}), \quad l = 1, \ldots, L.$$
3. **Aggregation** of token-level features h_L^τ into a step-level vector:
$$h_r = \frac{1}{|\mathbf{r}_t|} \sum_{\tau \in \mathbf{r}_t} h_L^\tau.$$
4. **Two-layer classification NN** that outputs a final logit $\mathbf{z}$:
$$\mathbf{z} = \mathrm{Linear}_2\Big(\sigma\big(\mathrm{Dropout}(\mathrm{Linear}_1(h_r))\big)\Big),$$
where $\sigma(\cdot)$ is a GeLU activation function.

Reasoning step extraction is done in two ways. LLMs in non-thinking mode are instructed to generate each CoT step on a separate, self-contained line (see §F.1). LLMs reliably follow this format and produce well-formatted CoT steps, so they can be easily extracted during training and inference via simple pattern matching. LLMs operating in native thinking mode cannot be explicitly instructed to produce reasoning trajectories with clearly separated steps. To mitigate this problem, we treat each sentence as a reasoning step.

Constructing training data. To construct the training data for ReProbe, we use 10.8K problems (prompts) from the training subset of PRM800K. This dataset is an established resource for training and evaluating LLM reasoning capabilities. As most PRMs are already trained on LLM generations derived from it, it enables a fair comparison (Wang et al., 2024; Zhang et al., 2025c).

We generate 3 reasoning trajectories per problem across 10.8K mathematical problems, yielding ~32K samples. We use nucleus sampling (top-k=50, top-p=0.95, temperature=1.0). For annotation and training efficiency, we restrict the length of the generation during training. This restriction is removed during evaluation to ensure that our results are free of length bias.

We annotate the correctness of each reasoning step using an LLM as a judge. The judge is provided with the input question, the target LLM's CoT steps and final answer, as well as the ground-truth answer, and is prompted to assess the correctness of each reasoning step. The annotation prompts

Table 1: PR-AUC↑ for detecting incorrect reasoning steps (Qwen3-8B). Best scores are shown in **bold**. Other competitive scores show clear advantages are underlined. # Sample indicates the number of training samples; each sample corresponds to a reasoning trajectory with step-level labels.

Method	# Sample	Math (ID)			Planning (OOD)			QA (OOD)		Average		
		MATH	GSM8k	ProofNet	Trips	Meetings	Calendar	StrQA	SciQA	ID	OOD	Overall
Unsupervised Uncertainty Quantification (UQ)												
Random	-	.173	.061	.153	.524	.588	.486	.116	.125	.129	.368	.278
MaxProb	-	.221	.106	.194	.578	.655	.483	.114	.259	.174	.418	.326
MaxEntropy	-	.212	.122	.185	.545	.618	.443	.119	.227	.173	.390	.309
Perplexity	-	.205	.099	.176	.519	.572	.418	.110	.228	.160	.369	.291
Self-Certainty	-	.213	.101	.155	.516	.643	.482	.120	.243	.156	.401	.309
CCP	-	.250	.090	.168	.584	.645	.452	.119	.235	.169	.407	.318
P(True)	-	.164	.059	.172	.535	.608	.490	.126	.263	.132	.404	.302
Semantic Entropy	-	.257	.116	.173	.565	.610	.492	.111	.265	.182	.409	.324
Lexical Similarity	-	.250	.119	.170	.569	.603	.490	.120	.254	.180	.407	.322
Degree Matrix	-	.227	.089	.147	.534	.597	.484	.107	.258	.154	.396	.305
*PRMs **150**× Larger than ReProbes*												
Skywork-PRM-1.5B	Unk	.283	.412	.147	.433	.532	.502	.254	.408	.281	.426	.371
H4-Qwen2.5-PRM-1.5B-0.2	369K	.259	.171	.159	.597	.633	.495	.213	.228	.196	.433	.344
*PRMs **750**× to **810**× Larger than ReProbes*												
Math-Shepherd-PRM-7B	440K	.380	.405	.147	.662	.660	.657	.284	.415	.311	.536	.451
RLHFlow-PRM-Deepseek-8B	253K	.289	.540	.136	.583	.579	.504	.390	.518	.322	.515	.442
RLHFlow-PRM-Mistral-8B	273K	.233	.537	.118	.523	.555	.499	.349	.415	.296	.468	.404
Universal-PRM-Qwen2.5-Math-7B	690K	.534	.624	**.329**	.730	.753	.691	.328	.330	.496	.566	.540
Qwen2.5-Math-7B-PRM800k	265K	**.586**	.613	.301	.708	.768	.727	.362	.404	.500	.594	.559
Qwen2.5-Math-PRM-7B	860K	.531	**.702**	.310	.711	.757	.745	.334	.429	**.514**	.595	.565
Reasoning Probes (ReProbes)												
★ ReProbe, Attn+Logit, Self-anno (Ours)	**32K**	.529	.594	.260	.735	.779	.779	.394	.404	.461	.618	.559
★ ReProbe, Attn+Logit, DeepSeek-anno (Ours)	**32K**	.465	.616	.243	.740	.802	.786	**.395**	.361	.441	.617	.551
★ ReProbe, Hidden States, Self-anno (Ours)	**32K**	.558	.673	.264	**.799**	**.819**	.785	.381	**.553**	.498	**.667**	**.604**
★ ReProbe, Hidden States, DeepSeek-anno (Ours)	**32K**	.534	.650	.281	.793	.817	**.789**	.344	.450	.488	.639	.582

are provided in §F.2. We consider two approaches to step annotation: (1) an external verification setting, in which a larger LLM evaluates the reasoning steps – following Zheng et al. (2025), we use DeepSeek-R1; and (2) a self-supervised setting, in which the same LLM annotates its own generated CoT steps. Other details for reproducing the training data construction can be found in §B.1.

Training ReProbe. We train the probes using standard binary cross-entropy loss. Due to severe class imbalance, we adjust the class weights. Only the parameters of ReProbe are trained; the entire LLM body remains frozen. We further note that training ReProbes is highly cost-efficient: the total annotation cost is only $200, and the required compute amounts to just 4 GH200 GPU-hours (see §B.3).

4 Experimental Setup

4.1 LLMs for Reasoning, ReProbe Training, Evaluation Datasets, and Baselines

LLMs for reasoning used in our experiments include three state-of-the-art models: `Qwen3-8B` (Yang et al., 2025) in non-thinking CoT mode, `Qwen3-1.7B` and `Qwen3-32B` in native thinking mode, and `Phi-4` (Abdin et al., 2025).

Probe training settings. We use Shelmanov et al. (2025)'s framework to train ReProbes. We conducted a vast hyperparameter search; the best training hyperparameters are detailed in §B.2.

Baselines. We benchmark against a broad set of baselines, covering (1) two small 1.5B PRMs; (2) six large state-of-the-art 7-8B PRMs; (3) basic and state-of-the-art UQ methods implemented in the LM-Polygraph framework (Fadeeva et al., 2023; Vashurin et al., 2025). More details about the baselines are presented in §C.1.

Evaluation datasets span three domains: mathematical reasoning (in-domain), planning (OOD), and general knowledge QA (OOD). We select datasets that demand non-trivial reasoning and include reasoning steps that are unambiguously verifiable, enabling reliable evaluation by both LLMs and human annotators. Test dataset details are given in §§ C.2 and C.3.

Method	# Sample	Math (ID)			Planning (OOD)		QA (OOD)		Average		
		MATH	GSM8k	ProofNet	Trips	Calendar	StrQA	SciQA	ID	OOD	Overall
PRMs 750× to 810× Larger than ReProbes											
Qwen2.5-Math-7B-PRM800k	263K	89.8	80.4	95.2	13.1	45.0	86.7	91.2	88.5	59.0	71.6
Qwen2.5-Math-PRM-7B	860K	88.1	95.4	93.6	13.1	41.5	79.2	83.6	92.4	54.4	70.7
Reasoning Probes (ReProbes)											
★ ReProbe, Attn+Logit, Self-anno (Ours)	**32K**	**90.3**	95.4	95.1	15.0	41.5	94.8	96.2	93.6	61.9	75.5
★ ReProbe, Attn+Logit, DeepSeek-anno (Ours)	**32K**	81.8	93.3	79.1	11.8	42.4	90.7	90.8	84.7	58.9	70.0
★ ReProbe, Hidden States, Self-anno (Ours)	**32K**	84.1	97.3	90.6	**15.6**	40.5	91.2	92.8	90.7	60.0	73.2
★ ReProbe, Hidden States, DeepSeek-anno (Ours)	**32K**	86.8	**98.8**	**95.6**	13.4	**48.0**	**96.7**	**96.7**	**93.7**	**63.7**	**76.6**

Table 2: Beam search decoding accuracy across datasets (Qwen3-8B).

4.2 Evaluation Settings

We conduct experiments in three settings: (1) step-level correctness assessment; (2) best-of-N TTS; and (3) beam search TTS.

(1) Step-level correctness. For each question in the test set, we generate a CoT trace using the Qwen or Phi models and evaluate the correctness of their reasoning steps with DeepSeek-R1. To ensure accurate judgments, DeepSeek-R1 is provided with the ground-truth answer, reasoning steps, and supporting evidence (see the prompt in §F.2). To validate the reliability of the judge, we evaluate it against (1) the human annotations from a random subset of PRM800k and (2) a manually annotated set of 1000 steps spanning QA, planning, and ProofNet tasks. DeepSeek-R1 achieves 95% acc. on PRM800k and ~90% on other datasets (see Table 16 and §D for details). Based on the judges' assessments, we compute the PR-AUC evaluation metric.

(2) Best-of-N TTS setting assumes that we generate N reasoning trajectories per problem (N=10 for the mathematical and QA datasets, N=5 for the planning datasets, temperature=1.0). The evaluation metric in this setting is the accuracy of the final solution. For GSM8K, we use an exact match against the gold-standard final answer. For other datasets, where final answers may be open-ended or structurally complex, we use DeepSeek-R1 to assign binary correctness labels (1 for correct, 0 for incorrect) based on both the problem statement and the reference solution. The grading prompt is provided in §F.2.

(3) Beam search TTS setting assumes that we maintain a beam of $B = 5$ partial reasoning trajectories at each step. From each partial trajectory, we sample $N = 5$ candidate continuations, resulting in $B \cdot N$ potential candidates at each step. We assign a step-level correctness score to each candidate and retain the top-B expanded trajectories with the highest scores. The correctness of the final solution is evaluated following the BoN setting, using overall accuracy as the metric.

5 Main Results

Step-level correctness assessment results for Qwen3-8B are presented in Table 1, and for Phi-4 and Qwen3-1.7B/32B in native thinking mode in Tables 6, 8 and 9 correspondingly in §A.

Unsupervised UQ methods, while providing valuable signals for detecting reasoning errors, substantially underperform other approaches. Small PRMs show clear gains over unsupervised UQ methods in the mathematical in-domain setting, but yield limited improvements on OOD datasets.

The best large PRMs considered in our work, such as Qwen2.5-Math-7B-PRM800k and Qwen2.5-Math-PRM-7B, substantially outperform all unsupervised UQ methods and smaller PRMs on all tasks.

Despite using 750-810× fewer parameters than PRMs, *all ReProbe variants perform on par with or even surpass the best PRMs*. For ID mathematical datasets, ReProbes substantially outperform all PRM baselines except for the two strongest Qwen2.5-Math-based PRMs. Even in these cases, ReProbe performance remains highly competitive and closely matches the best PRMs.

It is not surprising that parameter-heavy PRMs achieve slightly better results on ID mathematical datasets, as they tend to strongly overfit to this domain. In contrast, ReProbe, with far fewer

Method	# Sample	Math (ID)			Planning (OOD)			QA (OOD)		Average		
		MATH	GSM8k	ProofNet	Trips	Meetings	Calendar	StrQA	SciQA	ID	OOD	Overall
Pass@N, Majority Voiting, or Larger LLM												
Qwen3-8B pass@1 (Lower Bound)	-	92.4	95.6	74.1	8.1	5.5	23.5	86.8	92.7	87.4	43.3	59.8
Qwen3-8B pass@N (Upper Bound)	-	99.3	99.2	97.8	36.2	16.0	60.5	98.3	99.3	98.8	56.1	72.1
Qwen3-14B pass@1	-	93.4	97.6	76.0	38.7	5.0	40.5	91.9	95.6	89.0	54.3	67.3
Majority Voting	-	–	97.6	–	–	–	–	86.6	92.5	–	–	–
*PRMs **150**× Larger than ReProbes*												
Skywork-PRM-1.5B	Unk	94.4	97.6	76.5	6.9	6.0	23.0	86.6	96.3	89.5	43.8	60.9
H4-Qwen2.5-PRM-1.5B-0.2	369K	91.7	95.1	71.4	15.6	4.5	22.0	84.6	94.7	86.1	44.3	59.9
*PRMs **750**× to **810**× Larger than ReProbes*												
Math-Shepherd-PRM-7B	440K	93.0	95.5	72.8	9.1	4.0	30.0	87.3	95.8	87.1	45.2	60.9
RLHFlow-PRM-Deepseek-Data	253K	92.7	96.4	71.7	8.7	3.5	25.5	87.6	94.1	86.9	43.9	60.0
RLHFlow-PRM-Mistral-Data	273K	93.7	96.3	71.7	8.4	3.5	30.0	87.8	94.1	87.2	44.8	60.7
Universal-PRM-Qwen2.5-Math-7B	690K	**95.7**	97.5	76.0	9.7	4.0	24.0	87.8	**97.1**	**89.7**	44.5	61.5
Qwen2.5-Math-7B-PRM800k	263K	92.7	97.3	74.4	5.9	6.0	27.5	87.1	96.9	88.1	44.7	61.0
Qwen2.5-Math-PRM-7B	860K	93.7	**97.8**	76.0	7.2	5.5	26.5	88.1	96.9	89.2	44.8	61.5
Reasoning Probes (ReProbes)												
★ ReProbe, Attn+Logit, Self-anno (Ours)	**32K**	94.4	97.5	73.6	9.4	6.5	**31.0**	**88.6**	**97.1**	88.5	46.5	62.3
★ ReProbe, Attn+Logit, DeepSeek-anno (Ours)	**32K**	92.7	**97.8**	76.5	**17.2**	**7.0**	26.0	**88.6**	96.9	89.0	**47.1**	**62.8**
★ ReProbe, Hidden States, Self-anno (Ours)	**32K**	94.4	96.1	**78.5**	13.1	6.0	26.0	88.1	95.8	**89.7**	45.8	62.3
★ ReProbe, Hidden States, DeepSeek-anno (Ours)	**32K**	92.7	96.1	74.1	15.0	5.5	26.0	**88.6**	96.9	87.6	46.4	61.9

Table 3: Best-of-N decoding accuracy across datasets (Qwen3-8B). For datasets with verifiable final answers, we also provide the majority voting baseline.

Table 4: PR-AUC of ReProbes trained on the diverse vs. homogeneous subsets of 2K questions (with three trajectories sampled per question). Increased data diversity consistently improves ReProbe's overall performance.

Method	# Sample	Math (ID)			Planning (OOD)			QA (OOD)		Average		
		MATH	GSM8k	ProofNet	Trips	Meetings	Calendar	StrQA	SciQA	ID Avg.	OOD Avg.	Overall Avg.
ReProbe, Attn+Logit, Indiverse	6K	.308	.549	**.205**	.626	.687	.685	**.377**	.251	.354	.525	.461
ReProbe, Attn+Logit, Diverse	6K	**.409**	**.575**	.180	**.707**	**.793**	**.792**	.271	**.325**	**.388**	**.578**	**.507**

parameters, avoids such domain-specific overfitting. This advantage becomes evident in the average OOD PR-AUC, where all ReProbes significantly outperform the best PRMs. For all OOD planning and QA datasets, the best PR-AUC scores are consistently achieved by ReProbes. In summary, *ReProbes slightly lag behind the strongest PRMs on ID tasks but outperform them on OOD tasks*. Similar results are observed for Phi-4 and for Qwen3-1.7B/32B in the native thinking mode.

Notably, the self-supervised ReProbe achieves comparable average performance to the externally supervised variant, offering an *efficient self-supervised solution for reasoning step verification* that is particularly effective in OOD settings.

Given the poor performance of unsupervised UQ in identifying incorrect steps, we exclude it from subsequent test time scaling experiments.

Test time scaling. Best-of-N results for Qwen3-8B are presented in Table 3, and supplementary results are in §A. ReProbes achieve the best performance on all datasets except MATH, where ReProbe ranks second best. Notably, ReProbe-based TTS enables Qwen3-8B to outperform its larger sibling – Qwen3-14B on multiple benchmarks: MATH, GSM8K, ProofNet, Meeting Planning, and ScienceQA. As in the step assessment setting, ReProbe exhibits strong generalization. While best PRMs (Universal-PRM-7B, Qwen2.5-MATH-7B-PRM, and Qwen2.5-Math-PRM-7B) reach parity with ReProbe on in-domain datasets MATH and GSM8K, they often fall behind on OOD tasks, such as planning and StrategyQA.

Beam search TTS results with Qwen3-8B are reported in Table 2. Across all in-domain and out-of-domain datasets, ReProbe achieves the strongest performance, outperforming state-of-the-art PRMs.

Overall, *ReProbe provides stable gains in test time scaling across tasks of varying difficulty, from relatively simple datasets (MATH, GSM8K, ScienceQA) to complex planning benchmarks.*

6 Analysis

Synergy of PRMs and ReProbes. PRMs and ReProbes capture complementary aspects of reasoning quality: PRMs rely on their own knowledge and textual cues from the generated rationale, whereas probes exploit internal signals of the LLM that encode model confidence. To explore whether these two perspectives can be synergistic, we train a logistic regression model that takes both the PRM score and the ReProbe score as input to predict step-level correctness labels. The model is trained on a random subset of 200 questions from the training set. Table 5 reports the step-level PR-AUC, showing that this combination yields additional improvements. This finding suggests that integrating confidence-based signal and the PRM external knowledge is a promising direction for future work.

Table 5: Synergy of PRMs with ReProbes. Step-level PR-AUC, Qwen3-8B.

Method	**MATH**	**GSM8k**	**ProofNet**
PRM1 (Qwen2.5-Math-7B-PRM800k)	.586	.613	.301
PRM2 (Qwen2.5-Math-7B)	.531	.702	.310
ReProbe, Attn+Logit, DeepSeek-anno	.529	.594	.260
ReProbe + PRM1	**.613**	.674	.318
ReProbe + PRM2	.573	**.710**	**.327**

Training data quantity and diversity. We investigate how data quantity and diversity influence ReProbe performance. Fig 3 (left) shows that larger training sets, more questions, and sampled reasoning trajectories consistently improve ReProbe performance on both ID and OOD tasks. In practice, however, high-quality questions are often scarce, making it difficult to scale by prompt expansion alone. Alternatively, it is possible to sample multiple reasoning trajectories per question. Fig 3 (right) compares scaling strategies, demonstrating that trajectory sampling is an effective way to boost ReProbe performance.

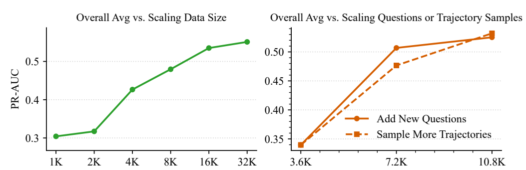

Figure 3: Left: PR-AUC of ReProbes with increasing training set size (x-axis). Right: scaling training data either by adding new unique questions or by sampling additional trajectories. Average PR-AUC across all datasets is reported. The results for each individual dataset are in Fig 5, §A.

To study the impact of data diversity, we train ReProbes on two 2K-question subsets of training data: one highly homogeneous and one highly diverse. For similarity assessment, questions are embedded using Qwen3-Embedding-8B (Zhang et al., 2025b). The homogeneous subset is constructed by selecting the 2K nearest neighbors to the median embedding, whereas the diverse subset is obtained via farthest-first traversal (Gonzalez, 1985). Table 4 shows that increased data diversity leads to consistently improved ReProbe performance.

Architecture. We compare three architectures for ReProbe: *(1) ReProbe (step-level)*: leverages a transformer encoder and predicts step-level correctness; *(2) Linear Probe (token-level)*: replacing transformer-based ReProbe with a linear layer to predict token-level labels (all tokens of incorrect step should be labeled incorrect); and *(3) ReProbe (token-level)*: uses a transformer encoder and predicts token-level labels. All variants use the same training setup (hidden states as features, DeepSeek-anno). Table 11 shows that step-level ReProbe outperforms both token-level variants.

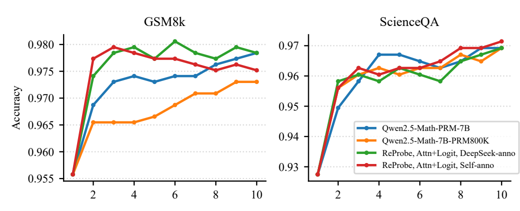

Figure 4: Best-of-N accuracy on GSM8k and ScienceQA for different N values (Qwen3-8B).

Impact of reasoning length on the performance of ReProbes and PRMs is investigated in §A.4. We show that the performance of both PRMs and ReProbes degrades with increasing reasoning length, but only slightly.

Scaling the number of samples N**.** Fig 4 analyzes best-of-N performance on GSM8k and ScienceQA for $1 \leq N \leq 10$. As expected, accuracy improves with increasing N across methods on both datasets.

On GSM8k, both ReProbe variants consistently outperform PRMs; on ScienceQA all methods exhibit comparable performance.

ReProbe efficiency. The theoretical analysis of the efficiency of ReProbe compared to PRMs is presented in §E. We also empirically illustrate the runtime efficiency of ReProbe in Table 13. ReProbe achieves a 2.6×–25× speedup over state-of-the-art PRMs.

7 Related Work

PRMs. Research in PRMs has advanced by scaling and refining step-level annotations: from manual labeling (Uesato et al., 2022; Lightman et al., 2024), to MC-based automatic labeling (Wang et al., 2024; Luo et al., 2024), and consensus methods combining LLM-as-a-Judge and MC estimation (Zhang et al., 2025c; Zhao et al., 2025). Generative PRMs further extend step-level judgment with long CoT (Xiong et al., 2025; Zhao et al., 2025). Our approach differs from PRMs in that ReProbe is not a language model and operates directly on internal LLM states rather than textual inputs.

Uncertainty quantification methods recently have been employed for test-time scaling and improving reasoning performance (Mo & Xin, 2024; Yin et al., 2024; Zhang et al., 2025a; Fu et al., 2025; Yan et al., 2025; Kang et al., 2025). However, so far, only unsupervised UQ methods have been used as step scorers. We propose a training-based probe and show that it achieves much better performance than unsupervised UQ.

Formal verification has recently been used to verify LLM reasoning steps (Zhou et al., 2024; Hu et al., 2025; Liu et al., 2025; Zhou & Zhang, 2025). However, these methods often require specialized autoformalization data for training and are limited to narrow domains (e.g., math proofs). ReProbes are much more general; our experiments show that they can generalize to OOD.

8 Conclusion

We introduced ReProbe, a lightweight step-level verifier that reads an LLM's own internal states to guide reasoning process. ReProbe can be trained fully self-supervised without human labels, verifiable final answers, or Monte-Carlo rollouts. Across math, planning, and QA, ReProbe delivers strong in- and out-of-domain results and is competitive with, or better than, far larger PRMs, making it a practical building block for resource-efficient reasoning systems. Beyond replacing PRMs, ReProbe complements them: combining PRM scores with ReProbe's score consistently improves performance, indicating the two signals capture complementary aspects of reasoning quality. This synergy suggests a promising path toward strong hybrid verifiers that marry introspective confidence with process rewards. Looking ahead, our findings pave the way for more efficient test-time scaling for LLMs in reasoning tasks.

Ethics Statement

We use publicly available datasets (MATH, GSM8K, ProofNet, ScienceQA, and StrategyQA), which have no data privacy issues. All artifacts we use are under licenses allowing research usage. Human annotations were conducted by the authors of this paper. We do not identify any other ethical risks associated with this study.

Reproducibility Statement

We open-source our trained ReProbes, code, prompts, human annotations, and processed datasets to ensure reproducibility. For training, evaluation, and sampling we fix random seeds to 1 or 42 (as specified in the codebase).

One limitation for exact reproduction is the use of API-based DeepSeek-R1, which is inherently non-deterministic even with fixed prompts and temperature. To mitigate this, we release all DeepSeek-R1 annotations used for training and evaluation. Running our codebase from scratch reproduces the same trends and observations, even if exact numbers vary slightly.

REFERENCES

Marah Abdin, Sahaj Agarwal, Ahmed Awadallah, Vidhisha Balachandran, Harkirat Behl, Lingjiao Chen, Gustavo de Rosa, Suriya Gunasekar, Mojan Javaheripi, Neel Joshi, et al. Phi-4-reasoning technical report. *arXiv preprint arXiv:2504.21318*, 2025.

AURORA. Aurora: Automated training framework of universal process reward models via ensemble prompting and reverse verification, 2025. URL `https://auroraprm.github.io/`.

Amos Azaria and Tom Mitchell. The internal state of an LLM knows when it's lying. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 967–976, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.68. URL `https://aclanthology.org/2023.findings-emnlp.68/`.

Zhangir Azerbayev, Bartosz Piotrowski, Hailey Schoelkopf, Edward W. Ayers, Dragomir Radev, and Jeremy Avigad. Proofnet: Autoformalizing and formally proving undergraduate-level mathematics, 2023. URL `https://arxiv.org/abs/2302.12433`.

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. Training verifiers to solve math word problems, 2021. URL `https://arxiv.org/abs/2110.14168`.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL `https://arxiv.org/abs/2501.12948`.

Ekaterina Fadeeva, Roman Vashurin, Akim Tsvigun, Artem Vazhentsev, Sergey Petrakov, Kirill Fedyanin, Daniil Vasilev, Elizaveta Goncharova, Alexander Panchenko, Maxim Panov, Timothy Baldwin, and Artem Shelmanov. LM-polygraph: Uncertainty estimation for language models. In Yansong Feng and Els Lefever (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 446–461, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-demo.41. URL `https://aclanthology.org/2023.emnlp-demo.41/`.

Ekaterina Fadeeva, Aleksandr Rubashevskii, Artem Shelmanov, Sergey Petrakov, Haonan Li, Hamdy Mubarak, Evgenii Tsymbalov, Gleb Kuzmin, Alexander Panchenko, Timothy Baldwin, et al. Fact-checking the output of large language models via token-level uncertainty quantification. In *Findings of the Association for Computational Linguistics ACL 2024*, pp. 9367–9385, 2024.

Yichao Fu, Xuewei Wang, Yuandong Tian, and Jiawei Zhao. Deep think with confidence, 2025. URL `https://arxiv.org/abs/2508.15260`.

Yarin Gal et al. Uncertainty in deep learning, 2016.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. Did Aristotle Use a Laptop? A Question Answering Benchmark with Implicit Reasoning Strategies. *Transactions of the Association for Computational Linguistics (TACL)*, 2021.

Teofilo F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38:293–306, 1985. ISSN 0304-3975. doi: https://doi.org/10.1016/0304-3975(85)90224-5. URL `https://www.sciencedirect.com/science/article/pii/0304397585902245`.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025.

Jujie He, Tianwen Wei, Rui Yan, Jiacai Liu, Chaojie Wang, Yimeng Gan, Shiwen Tu, Chris Yuhao Liu, Liang Zeng, Xiaokun Wang, Boyang Wang, Yongcong Li, Fuxiang Zhang, Jiacheng Xu, Bo An, Yang Liu, and Yahui Zhou. Skywork-o1 open series. `https://huggingface.co/Skywork`, November 2024. URL `https://huggingface.co/Skywork`.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset, 2021. URL https://arxiv.org/abs/2103.03874.

Xiaolin Hu, Qinghua Zhou, Bogdan Grechuk, and Ivan Y. Tyukin. Stepproof: Step-by-step verification of natural language mathematical proofs, 2025. URL https://arxiv.org/abs/2506.10558.

HuggingFaceH4. Qwen2.5-math-1.5b-instruct-prm-0.2. Hugging Face Model Hub, 2025. URL https://huggingface.co/HuggingFaceH4/Qwen2.5-Math-1.5B-Instruct-PRM-0.2. Fine-tuned version of Qwen/Qwen2.5-Math-1.5B-Instruct on the prm800k-trl-dedup dataset, trained with PRM and TRL. Safetensors format. arXiv:2211.14275.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, Scott Johnston, Sheer El-Showk, Andy Jones, Nelson Elhage, Tristan Hume, Anna Chen, Yuntao Bai, Sam Bowman, Stanislav Fort, Deep Ganguli, Danny Hernandez, Josh Jacobson, Jackson Kernion, Shauna Kravec, Liane Lovitt, Kamal Ndousse, Catherine Olsson, Sam Ringer, Dario Amodei, Tom Brown, Jack Clark, Nicholas Joseph, Ben Mann, Sam McCandlish, Chris Olah, and Jared Kaplan. Language models (mostly) know what they know, 2022. URL https://arxiv.org/abs/2207.05221.

Zhewei Kang, Xuandong Zhao, and Dawn Song. Scalable best-of-n selection for large language models via self-certainty. In *2nd AI for Math Workshop @ ICML 2025*, 2025. URL https://openreview.net/forum?id=nddwJseiiy.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=VD-AYtP0dve.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=v8L0pN6EOi.

Zhen Lin, Shubhendu Trivedi, and Jimeng Sun. Generating with confidence: Uncertainty quantification for black-box large language models, 2024. URL https://openreview.net/forum?id=XJiN1VkgA0.

Chengwu Liu, Ye Yuan, Yichun Yin, Yan Xu, Xin Xu, Zaoyu Chen, Yasheng Wang, Lifeng Shang, Qun Liu, and Ming Zhang. Safe: Enhancing mathematical reasoning in large language models via retrospective step-aware formal verification. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12171–12186, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.594. URL https://aclanthology.org/2025.acl-long.594/.

Pan Lu, Swaroop Mishra, Tony Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. In *The 36th Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

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

Andrey Malinin and Mark J. F. Gales. Uncertainty estimation in autoregressive structured prediction. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*, 2021. URL https://openreview.net/forum?id=jN5y-zb5Q7m.

Shentong Mo and Miao Xin. Tree of uncertain thoughts reasoning for large language models. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 12742–12746. IEEE, 2024.

Artem Shelmanov, Ekaterina Fadeeva, Akim Tsvigun, Ivan Tsvigun, Zhuohan Xie, Igor Kiselev, Nico Daheim, Caiqi Zhang, Artem Vazhentsev, Mrinmaya Sachan, Preslav Nakov, and Timothy Baldwin. A head to predict and a head to question: Pre-trained uncertainty quantification heads for hallucination detection in LLM outputs. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 35700–35719, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.1809. URL https://aclanthology.org/2025.emnlp-main.1809/.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling parameters for reasoning. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Representation Learning*, volume 2025, pp. 10131–10165, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/1b623663fd9b874366f3ce019fdfdd44-Paper-Conference.pdf.

Weihang Su, Changyue Wang, Qingyao Ai, Yiran Hu, Zhijing Wu, Yujia Zhou, and Yiqun Liu. Unsupervised real-time hallucination detection based on the internal states of large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 14379–14391, Bangkok, Thailand, 2024. doi: 10.18653/v1/2024.findings-acl.854. URL https://aclanthology.org/2024.findings-acl.854.

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

Roman Vashurin, Ekaterina Fadeeva, Artem Vazhentsev, Lyudmila Rvanova, Akim Tsvigun, Daniil Vasilev, Rui Xing, Abdelrahman Boda Sadallah, Kirill Grishchenkov, Sergey Petrakov, Alexander Panchenko, Timothy Baldwin, Preslav Nakov, Maxim Panov, and Artem Shelmanov. Benchmarking uncertainty quantification methods for large language models with lm-polygraph. *Transactions of the Association for Computational Linguistics*, 2025.

Artem Vazhentsev, Ekaterina Fadeeva, Rui Xing, Gleb Kuzmin, Ivan Lazichny, Alexander Panchenko, Preslav Nakov, Timothy Baldwin, Maxim Panov, and Artem Shelmanov. Unconditional truthfulness: Learning unconditional uncertainty of large language models. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 35661–35682, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.1807. URL https://aclanthology.org/2025.emnlp-main.1807/.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9426–9439, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.510. URL https://aclanthology.org/2024.acl-long.510/.

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

Yuxi Xie, Kenji Kawaguchi, Yiran Zhao, James Xu Zhao, Min-Yen Kan, Junxian He, and Michael Qizhe Xie. Self-evaluation guided beam search for reasoning. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, pp. 41618–41650, 2023.

Wei Xiong, Hanning Zhang, Nan Jiang, and Tong Zhang. An implementation of generative prm, 2024.

Wei Xiong, Wenting Zhao, Weizhe Yuan, Olga Golovneva, Tong Zhang, Jason Weston, and Sainbayar Sukhbaatar. Stepwiser: Stepwise generative judges for wiser reasoning, 2025. URL https://arxiv.org/abs/2508.19229.

Hang Yan, Fangzhi Xu, Rongman Xu, Yifei Li, Jian Zhang, Haoran Luo, Xiaobao Wu, Luu Anh Tuan, Haiteng Zhao, Qika Lin, et al. Mur: Momentum uncertainty guided reasoning for large language models. *arXiv preprint arXiv:2507.14958*, 2025.

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

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

Zhangyue Yin, Qiushi Sun, Qipeng Guo, Zhiyuan Zeng, Xiaonan Li, Junqi Dai, Qinyuan Cheng, Xuanjing Huang, and Xipeng Qiu. Reasoning in flux: Enhancing large language models reasoning through uncertainty-aware adaptive guidance. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2401–2416, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.131. URL https://aclanthology.org/2024.acl-long.131/.

Jinghan Zhang, Xiting Wang, Fengran Mo, Yeyang Zhou, Wanfu Gao, and Kunpeng Liu. Entropy-based exploration conduction for multi-step reasoning. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 3895–3906, Vienna, Austria, July 2025a. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.201. URL https://aclanthology.org/2025.findings-acl.201/.

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. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*, 2025b.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. The lessons of developing process reward models in mathematical reasoning, 2025c. URL https://arxiv.org/abs/2501.07301.

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

Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. ProcessBench: Identifying process errors in mathematical reasoning. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1009–1024, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.50. URL https://aclanthology.org/2025.acl-long.50/.

Huaixiu Steven Zheng, Swaroop Mishra, Hugh Zhang, Xinyun Chen, Minmin Chen, Azade Nova, Le Hou, Heng-Tze Cheng, Quoc V. Le, Ed H. Chi, and Denny Zhou. Natural plan: Benchmarking llms on natural language planning, 2024. URL https://arxiv.org/abs/2406.04520.

Jin Zhou, Charles Staats, Wenda Li, Christian Szegedy, Kilian Weinberger, and Yuhuai Wu. Don't trust: Verify – grounding llm quantitative reasoning with autoformalization. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Representation Learning*, volume 2024, pp. 2544–2563, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/0a79ecda13603817de4cdfc68b417e89-Paper-Conference.pdf.

Kuo Zhou and Lu Zhang. Step-wise formal verification for llm-based mathematical problem solving, 2025. URL https://arxiv.org/abs/2505.20869.

A Additional Experimental Results

A.1 Results with Phi-4

To verify that our framework works for models of different sizes, families, and post-training approaches, we conduct step-level correctness and BoN evaluation on a Phi-4 ReProbe trained on Qwen3-8B annotated training data. The step-level correctness prediction and BoN results are presented in Table 6 and Table 7 correspondingly. On step-level correctness, ReProbe achieves the best performance in Meeting and Calendar Planning, and the best average performance on OOD tasks. On StrategyQA and overall average, it ranks second.

For BoN, the Qwen3-8B–annotated ReProbe outperforms two strong Qwen2.5-Math PRMs on MATH and matches the strongest PRMs on GSM8K. On ProofNet and StrategyQA, it also outperforms some much larger PRMs.

A.2 ReProbe under Native Thinking Mode

Native LRM reasoning often occurs within <think> tags without explicit step separations. To evaluate ReProbe in this setting, we train a ReProbe model for Qwen3-1.7B and Qwen3-32B in native thinking mode. We modify the experimental setup from §4 in two ways: (1) Since free-form reasoning from LRMs is much longer than formatted CoT, we replace the DeepSeek-R1 API with a locally deployed `GPT-OSS-120B` for step-level correctness annotation to reduce API costs. This model outperforms the January version of DeepSeek-R1 on multiple benchmarks (e.g., GPQA Diamond, AIME 2024). (2) Without format-based step separations, we treat each sentence as a step and prompt `GPT-OSS-120B` to identify errors conditioned on all previous sentences (prompts in §F). Results in Tables 8 and 9 show that ReProbe outperforms PRMs on ID and OOD average scores in native thinking mode.

A.3 Ablation Study on Data Quantity and ReProbe Architectures

Table 11 shows that the linear probe and token-level ReProbe are worse than the proposed step-level ReProbe. Fig 5 shows the per-dataset performance improvement by scaling the data quantity.

Method	# Sample	Math (ID)			Planning (OOD)			QA (OOD)		Average		
		MATH	GSM8k	ProofNet	Trips	Meetings	Calendar	StrQA	SciQA	ID Avg.	OOD Avg.	Overall
Unsupervised Uncertainty Quantification (UQ)												
Random	-	.106	.038	.082	.552	.463	.324	.172	.086	.075	.319	.228
MaxProb	-	.127	.084	.123	.618	.548	.380	.252	.158	.111	.391	.286
MaxEntropy	-	.112	.079	.107	.585	.533	.362	.248	.135	.099	.373	.270
Perplexity	-	.117	.066	.099	.557	.508	.323	.228	.143	.094	.352	.255
PRMs 150× Larger than ReProbes												
Skywork-PRM-1.5B	Unk	.219	.181	.185	.408	.467	.327	.237	<u>.415</u>	.195	.371	.305
H4-Qwen2.5-PRM-1.5B-0.2	369K	.174	.061	.105	.534	.476	.434	.212	.116	.113	.354	.264
PRMs 750× to 810× Larger than ReProbes												
Math-Shepherd-PRM-7B	440K	.248	.188	.188	.747	.584	.489	.249	.327	.208	.479	.378
RLHFlow-PRM-Deepseek-8B	253K	.200	.263	.109	.558	.455	.343	.315	**.440**	.191	.422	.335
RLHFlow-PRM-Mistral-8B	273K	.141	.195	.093	.462	.424	.309	.261	.311	.143	.353	.274
Universal-PRM-Qwen2.5-Math-7B	690K	**.485**	.213	**.263**	.741	.559	.497	.321	.252	.320	.474	.416
Qwen2.5-Math-7B-PRM800k	265K	<u>.474</u>	**.406**	.238	**.825**	<u>.599</u>	<u>.568</u>	**.355**	.329	**.373**	<u>.535</u>	**.474**
Qwen2.5-Math-PRM-7B	860K	.427	<u>.377</u>	<u>.240</u>	<u>.791</u>	.594	.555	.333	.310	<u>.348</u>	.517	<u>.453</u>
Reasoning Probes (ReProbes)												
★ ReProbe, Attn+Logit, Qwen3-8B-anno (Ours)	**32K**	.404	.340	.155	.756	**.646**	**.592**	<u>.347</u>	.347	.300	**.538**	<u>.448</u>

Table 6: PR-AUC for detecting incorrect reasoning steps for Phi-4. Best scores are shown in **bold**, and other competitive scores are <u>underlined</u>. # Sample indicates the number of training samples, where each sample corresponds to a reasoning trajectory with step-level annotations. ‡ Qwen2.5-Math-PRM-7B's training data is filtered from an 860K-sample dataset; the exact size after filtering is not specified in their paper.

A.4 Impact of Reasoning Length on ReProbe Performance

Figure 6 compares the performance of ReProbe with two selected PRM models across different bins of total reasoning steps. The results indicate that ReProbe consistently outperforms the PRM baselines on shorter generations (i.e., ≤ 8 reasoning steps), while achieving comparable performance as the reasoning chains become longer.

Method	# Sample	Math (In-domain)			Reasoning QA	
		MATH	GSM8k	ProofNet	StrQA	SciQA
Pass@N and Majority Voting						
Phi-4 pass@1 (Lower Bound)	-	90.5	97.9	86.8	93.0	95.7
Phi-4 pass@N (Upper Bound)	-	98.2	100.	97.2	99.1	99.8
Majority Voting	-	–	**100.**	–	93.2	94.6
PRMs 150× Larger than ReProbes						
Skywork-PRM-1.5B	Unk	95.9	99.5	**95.7**	92.5	96.6
H4-Qwen2.5-PRM-1.5B-0.2	369K	93.5	**100.**	89.2	91.8	96.4
PRMs 750× to 810× Larger than ReProbes						
Math-Shepherd-PRM-7B	440K	92.9	98.4	95.2	93.2	96.6
RLHFlow-PRM-Deepseek-Data	253K	94.1	98.9	89.6	**94.8**	**98.0**
RLHFlow-PRM-Mistral-Data	273K	92.3	98.4	90.3	94.1	97.3
Universal-PRM-Qwen2.5-Math-7B	690K	**96.4**	**100.**	93.5	92.5	96.8
Qwen2.5-Math-7B-PRM800k	263K	93.5	**100.**	92.4	93.9	97.5
Qwen2.5-Math-PRM-7B	860K	93.5	**100.**	94.4	94.1	97.5
Reasoning Probes (ReProbes)						
★ ReProbe, Attn+Logit, Qwen3-8B-anno (Ours)	**32K**	95.9	**100.**	93.1	93.0	95.7

Table 7: Best-of-N decoding accuracy across datasets for Phi-4 model. For datasets with verifiable final answer, we also provide Majority Voting.

Method	# Sample	Math (ID)			Planning (OOD)			QA (OOD)		Average		
		MATH	GSM8k	ProofNet	Trips	Meetings	Calendar	StrQA	SciQA	ID	OOD	Overall
*PRMs **750**× to **810**× Larger than ReProbes*												
Math-Shepherd-PRM-7B	440K	.049	.248	.234	.219	.481	.338	.265	.313	.177	.323	.268
RLHFlow-PRM-Deepseek-Data	253K	.063	.267	.228	.169	.498	.322	**.449**	.368	.186	.361	.296
RLHFlow-PRM-Mistral-Data	273K	.047	.184	.164	.141	.457	.275	.273	.279	.132	.285	.228
Universal-PRM-Qwen2.5-Math-7B	690K	.175	.331	.299	.272	.708	.446	.335	.329	.268	.418	.362
Qwen2.5-Math-7B-PRM800K	263K	.217	.392	**.505**	.270	.716	.587	.391	.336	.371	.460	.427
Qwen2.5-Math-PRM-7B	860K	.293	.270	.465	.328	**.732**	**.605**	.378	.293	.343	.467	.421
Reasoning Probes (ReProbes)												
ReProbe, Hidden States, GPT-OSS-anno (Ours)	**10.8K**	**.312**	**.433**	.465	**.334**	.505	.604	.426	**.451**	**.403**	**.474**	**.447**

Table 8: PR-AUC↑ for detecting incorrect reasoning steps for Qwen3-1.7B in native thinking mode.

A.5 Token-Level Feature Importance Analysis

To better understand where the uncertainty signal captured by ReProbe arises, we conducted a token-level feature importance analysis.

For each token in a reasoning step, we replace that token's feature vector with the mean hidden feature vector over the entire generation and measure the absolute change in the predicted uncertainty score. This change is treated as the token's importance.

We find that the signal is not uniformly distributed across tokens. Instead, it is often concentrated near step boundaries. In approximately 40% of reasoning steps the most important token is the final token, while in 12.5% of steps it is the first token.

Moreover, the most influential tokens frequently correspond to sentence boundary markers, especially punctuation tokens. The token "." is the most important token in approximately 27% of reasoning steps. Interestingly, the token "Wait" appears as the most important token in about 2.5% of reasoning steps, consistent with its role as a marker of self-correction or uncertainty during chain-of-thought reasoning.

Figure 7 shows the distribution of the relative position of the most important token within each reasoning step. The results indicate that the uncertainty signal often appears toward the end of reasoning steps, suggesting that the model's internal confidence is updated when a step is completed.

A.6 Cascaded ReProbe–PRM Verification

Table 5 in the main paper suggests that ReProbe and PRMs capture complementary signals: PRMs rely primarily on external knowledge and textual cues, while ReProbe leverages internal model confidence signals.

Method	# Sample	Math (ID)			Planning (OOD)			QA (OOD)		Average		
		MATH	GSM8k	ProofNet	Trips	Meetings	Calendar	StrQA	SciQA	ID	OOD	Overall
*PRMs **750**× to **810**× Larger than ReProbes*												
Skywork-PRM-1.5B	Unk	.303	.886	.289	.420	.565	.508	**.527**	.265	.493	.457	.470
H4-Qwen2.5-PRM-1.5B-0.2	369K	.133	.464	.149	.470	.325	.386	.129	.221	.249	.306	.285
Universal-PRM-Qwen2.5-Math-7B	690K	.578	.903	.568	.737	.634	.637	.189	.303	.683	.500	.569
Qwen2.5-Math-7B-PRM800K	263K	.409	.894	.454	.573	.591	.594	.282	**.350**	.586	.478	.518
Qwen2.5-Math-PRM-7B	860K	.668	**.930**	.646	.696	.660	.676	.313	.315	**.748**	.532	**.613**
Reasoning Probes (ReProbes)												
ReProbe, Hidden States, GPT-OSS-anno (Ours)	**10.8K**	**.676**	.535	**.682**	**.752**	**.734**	**.698**	.302	.303	.631	**.558**	.585

Table 9: PR-AUC↑ for detecting incorrect reasoning steps for Qwen3-32B in native thinking mode.

Method	# Sample	Math (ID)			QA (OOD)		Average		
		MATH	GSM8k	ProofNet	StrQA	SciQA	ID	OOD	Overall
Pass@N									
Qwen3-1.7B pass@1 (Lower Bound)	–	54.6	70.1	47.7	43.2	48.3	57.5	45.8	52.8
Qwen3-1.7B pass@N (Upper Bound)	–	83.1	93.3	77.4	74.0	89.2	84.6	81.6	83.4
Unsupervised Uncertainty Quantification (UQ)									
MaxProb	–	55.6	84.0	52.9	50.9	59.7	64.2	**55.3**	60.6
MaxEntropy	–	62.2	83.6	47.1	49.4	**60.0**	64.3	54.7	60.5
Perplexity	–	55.6	84.0	52.9	50.9	59.7	64.2	**55.3**	60.6
*PRMs **150**× Larger than ReProbes*									
H4-Qwen2.5-PRM-1.5B-0.2	369K	63.7	78.3	52.9	45.4	47.2	65.0	46.3	57.5
*PRMs **750**× to **810**× Larger than ReProbes*									
Math-Shepherd-PRM-7B	440K	**67.3**	84.6	56.2	**54.6**	47.2	69.4	50.9	62.0
RLHFlow-PRM-Deepseek-Data	253K	64.9	82.8	49.6	50.0	45.6	65.8	47.8	58.6
RLHFlow-PRM-Mistral-Data	273K	62.6	82.4	48.8	51.2	49.8	64.6	50.5	59.0
Universal-PRM-Qwen2.5-Math-7B	690K	**67.3**	85.0	**57.0**	48.5	57.4	**69.8**	53.0	**63.0**
Qwen2.5-Math-PRM-7B	860K	62.6	**85.4**	52.9	44.8	51.5	67.0	48.2	59.4
Reasoning Probes (ReProbes)									
ReProbe, Hidden States, GPT-OSS-anno (Ours)	**10.8K**	62.2	85.0	**57.9**	51.8	45.9	68.4	48.9	60.6

Table 10: Best-of-N=10 decoding accuracy across datasets for Qwen3-1.7B in native thinking mode.

This motivates a cascaded verification strategy, where ReProbe acts as a lightweight first-pass filter and a computationally expensive PRM is invoked only for uncertain reasoning steps.

We evaluate a simple hybrid strategy:

1. Apply ReProbe to each reasoning step.
2. If the predicted uncertainty score exceeds a threshold t, the step is passed to a PRM.
3. Otherwise, the ReProbe prediction is used directly.

By tuning the threshold t, we can trade off verification cost and performance.

Importantly, the cascaded strategy can match the PRM's in-domain performance while significantly reducing the number of PRM evaluations. In our experiments, we skip approximately 56% of PRM evaluations on MATH (correspondning to $t = 0.3$) and 31% on ProofNet ($t = 0.1$) while maintaining comparable higher or equal (Table 12).

These results suggest that cascaded verifiers combining lightweight introspective probes with heavier reward models provide a promising direction for efficient reasoning verification.

A.7 Computational Efficiency

To assess computational efficiency, we benchmarked the runtime of ReProbe and several PRM models on 500 test samples from the MATH dataset. The runtime excludes LLM generation and step-extraction overhead. For ReProbe, we measure only the feature-extraction stage together with the forward pass of the ReProbe classifier; for PRMs, we measure solely the inference time of each model. PRM runtimes are based on the official implementations provided in their respective Hugging

Method	# Sample	Math (ID)			Planning (OOD)			QA (OOD)		Average		
		MATH	GSM8k	ProofNet	Trips	Meetings	Calendar	StrQA	SciQA	ID	OOD	Overall
ReProbe, Step-Level (Proposed)	**32K**	.534	.650	.281	.793	.817	.789	.344	.450	.488	.639	.582
ReProbe, Token-Level	**32K**	–.046	–.049	–.072	–.129	–.070	–.017	–.021	–.075	–.055	–.063	–.060
Linear Probe, Token-Level	**32K**	–.112	–.036	–.063	–.157	–.224	–.422	+.011	+.009	–.070	–.157	–.124

Table 11: Replacing the proposed ReProbe with other architectures. Change in PR-AUC relative to the original step-level ReProbe.

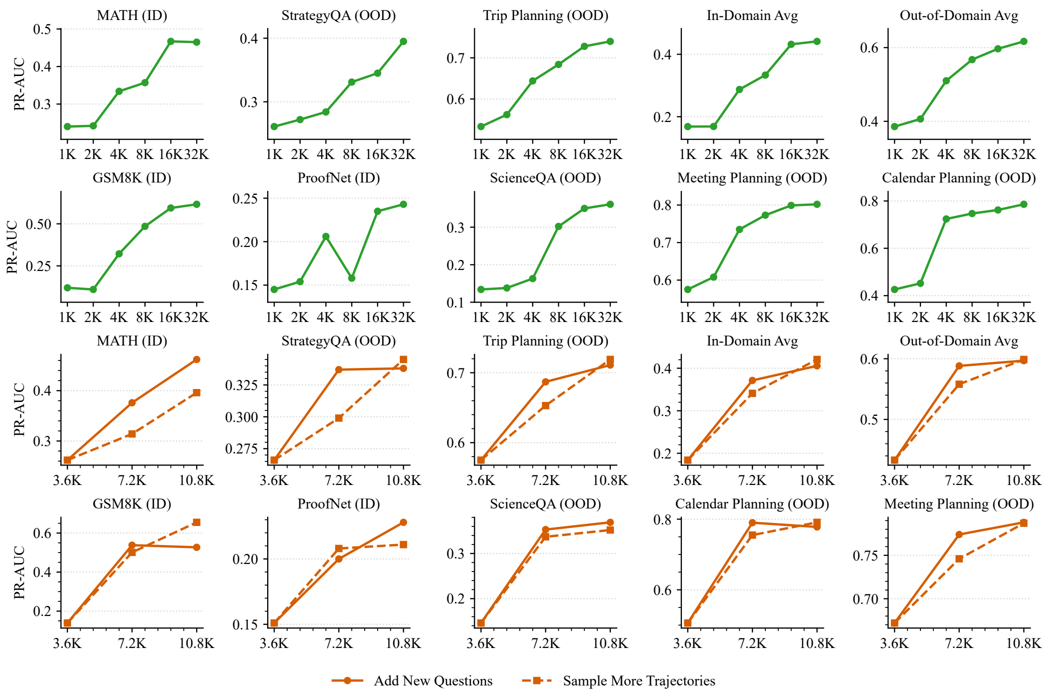

Figure 5: Top 2 rows: PR-AUC of ReProbes with increasing training set size (x-axis). Bottom 2 rows: scaling training data either by adding new unique questions or by sampling additional trajectories.

Face repositories, while ReProbe uses our own implementation of feature extraction and inference. All evaluations were performed with a batch size of 1. Table 13 presents the results.

The results show that ReProbe is by far the fastest method, which is expected given its lightweight architecture and small parameter footprint. In contrast, PRM models incur substantially higher inference costs, often by one to two orders of magnitude. Minor non-monotonicities in runtime across PRMs (e.g., smaller models being slower than larger ones) are attributable to differences in engineering and implementation rather than model size. A theoretical analysis of time and memory complexity can be found in §E.

B ReProbe Training Details

B.1 Training Data Annotation Details

In our experiments, we employ two LLM judges for the annotation of ReProbe training data. For DeepSeek-R1, we follow the officially recommended inference setting, using a temperature of 0.6. Similarly, when using Qwen3-8B as a training data annotator, we also follow the recommended inference hyperparameter, using a temperature of 0.7, a top_k of 20, and a top_p of 0.95. We access Qwen3-8B through vLLM local deployment and access DeepSeek-R1 through the DeepSeek API. The annotation prompts are detailed in Fig 10. For annotation efficiency, we restrict the generation of Qwen3 8B to 256 tokens during training.

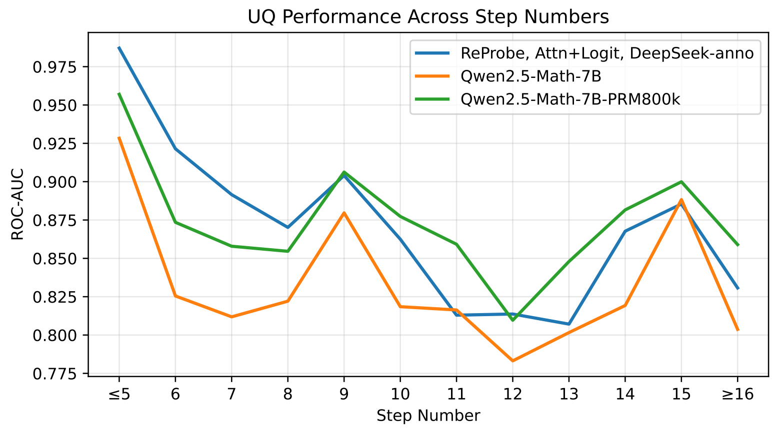

Figure 6: ROC-AUC on the step-level MATH benchmark using Qwen3-8B, evaluated across bins of reasoning chain length (number of reasoning steps), for the proposed ReProbe and two PRM models.

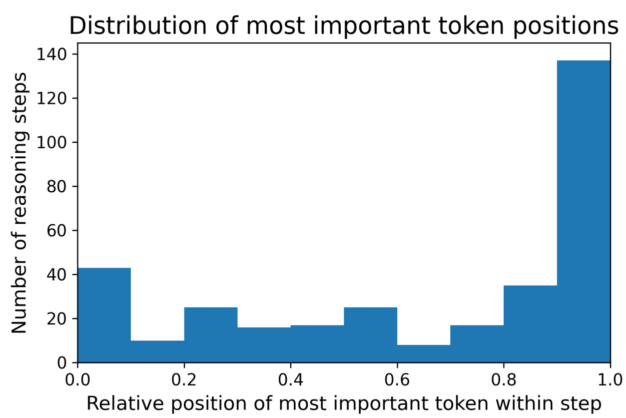

Figure 7: Distribution of the relative position of the most important token within each reasoning step.

Qwen3, with native thinking enabled, generates long CoTs enclosed within <think> tags, without explicit step separations. To train ReProbe for native reasoning, we modify the experimental setup from in two ways: (1) since the native thinking trajectory is much longer than the formatted CoT, we replace the DeepSeek-R1 API with the locally deployed `GPT-OSS-120B` for step-level correctness annotation to reduce API costs. This model outperforms the January version of DeepSeek-R1 on multiple benchmarks (e.g., GPQA Diamond, AIME 2024). (2) Without format-based step separations, we treat each sentence as a step and prompt `GPT-OSS-120B` to identify errors conditioned on all previous sentences. For the native thinking mode, during training, we restrict the generation length to 1024 tokens.

B.2 Training Hyperparameter Details

For all experiments on `Phi-4` and `Qwen3-8B`, we use the same set of hyperparameters. We use a learning rate of 5e-4, a batch size of 128, and a positive class weight of 3. For ReProbes, we use one layer of the transformer encoder with a hidden size of 512 and 16 attention heads. All training continues for 5 epochs. We use 10% of the training data and a 200-sample set of GSM8K, ScienceQA, and StrategyQA for validation and best checkpoint selection.

Method	Math	ProofNet
ReProbe, Hidden States, DeepSeek-anno	.514	.260
Qwen2.5-Math-7B-PRM800k	.585	.301
Hybrid cascade ($t = 0.1$)	**.601**	**.303**
Hybrid cascade ($t = 0.2$)	.590	.273
Hybrid cascade ($t = 0.3$)	.585	.234

Table 12: Hybrid cascaded verifier combining ReProbe and PRM. We report PR-AUC↑ on 2 datasets: Math and ProofNet.

Method	**Runtime**
*PRMs **150**× Larger than ReProbes*	
Skywork-PRM-1.5B	17 s
H4-Qwen2.5-PRM-1.5B-0.2	8 min 20 s
*PRMs **750**× to **810**× Larger than ReProbes*	
Qwen2.5-Math-7B-PRM800k	37 s
Qwen2.5-Math-PRM-7B	34 s
Math-Shepherd-PRM-7B	5 min 51 s
RLHFlow-PRM-Mistral-Data	5 min 19 s
RLHFlow-PRM-Deepseek-Data	5 min 33 s
Universal-PRM-Qwen2.5-Math-7B	4 min 22 s
Reasoning Probes (ReProbes)	
★ ReProbe, Hidden States, Self-anno (Ours)	14 s
★ ReProbe, Hidden States, DeepSeek-anno (Ours)	**13 s**

Table 13: Runtime comparison of ReProbe and PRMs on 500 MATH test samples (batch size = 1).

B.3 Training Cost

All experiments are carried out on cluster nodes with 4 GH200 GPUs and another cluster node with 2 H100 GPUs. Training one ReProbe on 32K datapoints with Hidden States as features takes ~4 GH200 GPU hours, while the variant with Attn+Logit as features takes ~32 GH200 GPU hours. The training of the Attn+Logit ReProbe is slower since extracting attention maps prohibits the use of efficient attention implementations (e.g., Flash Attention).

The total annotation cost for the training dataset (32K samples drawn from PRM800k) was approximately $200 per model. In addition, around $300 was spent on evaluation across eight datasets (three for math, three for planning, and two for QA), including the annotation of both step-level correctness and complete reasoning chains for the evaluated decoding strategies.

C Additional Details of Evaluation Setup

C.1 Baselines

We benchmark against a broad set of baselines, covering PRMs of different sizes and UQ methods.

Small PRMs (1.5B): we use 2 PRMs fine-tuned from Qwen2.5-Math-1.5B: `Skywork-PRM-1.5B` (He et al., 2024) and `H4-Qwen2.5-PRM-1.5B-0.2` (HuggingFaceH4, 2025).

Large PRMs (7–8B): we use (1) `Math-Shepherd-PRM-7B` (Wang et al., 2024), which determines the process labels for each step by estimating the empirical probability of reaching the correct final answer (MC estimation); (2) `RLHFlow-PRM-8B-DeepSeek/Mistral` models (Xiong et al., 2024) trained with MC-estimated labels from DeepSeek/Mistral rollouts; (3) `Universal-PRM-7B` (AURORA, 2025) trained using ensemble prompting and reverse verification; (4) `Qwen2.5-Math-7B-PRM800k` trained on the PRM800k dataset (Lightman et al., 2024); and (5) `Qwen2.5-Math-PRM-7B` (Zhang et al., 2025c), which combines MC estimation with LLM-as-a-Judge consensus and currently achieves the best result on ProcessBench compared to PRMs of similar scale and computation (Zhao et al., 2025).

UQ methods evaluated in our experiments fall into two categories, differing in computational cost: (1) *lightweight scores that use only single generation*: `Maximum Sequence Probability`, `Mean Token Entropy`, `Perplexity` (Fadeeva et al., 2023), `P(True)` (Kadavath et al., 2022), `CCP` (Fadeeva et al., 2024), and `Self-Certainty` (Kang et al., 2025), which show significant advantages in reasoning chain selection (Fu et al., 2025); and (2) *sampling-based scores*: `Semantic Entropy`, `Lexical Similarity` (Kuhn et al., 2023), and `Degree Matrix` (Lin et al., 2024). To compute the latter, we draw $M = 10$ alternative steps per step position, making them much less computationally efficient. All methods are implemented using the LM-Polygraph framework (Fadeeva et al., 2023; Vashurin et al., 2025).

C.2 Test Dataset Details

The mathematical domain includes `MATH` (high school and competition-style math problems; Hendrycks et al., 2021), `GSM8K` (grade-school math word problems; Cobbe et al., 2021), and `ProofNet` (natural language proofs of undergraduate-level math problems; Azerbayev et al., 2023). Planning domain includes `NaturalPlan` (Zheng et al., 2024) (spans three real-world tasks – trip planning, meeting planning, and calendar scheduling). The QA domain includes `ScienceQA` without multi-modal context (Lu et al., 2022) (covers 26 science subjects from elementary to high school) and `StrategyQA` (Geva et al., 2021) – a general knowledge QA benchmark that requires implicit multi-step reasoning. Due to computational limitations, we evaluate on randomly sampled subsets of the test sets.

Table 14 presents the statistics of the test datasets used for the step-level benchmark. Importantly, we rely on DeepSeek-R1 for step-level evaluation, which is an expensive API-based LLM. We thus take a subset of test sets for this evaluation. While there are only a few hundred questions, there are a substantial number of reasoning steps, which allows us to draw representative insights.

Dataset	Model	#Questions	#Steps	Mean #Steps	%Correct Steps	Answer Mean Len (Tokens)
MATH	Qwen3-8B	200	1203	6.2	85.5%	204
	Phi-4	200	2296	11.5	89.2%	417
GSM8k	Qwen3-8B	200	1056	5.1	94.0%	139
	Phi-4	200	1286	6.4	95.7%	174
ProofNet	Qwen3-8B	186	1211	6.5	81.9%	236
	Phi-4	186	2227	12.0	91.1%	515
Trips	Qwen3-8B	320	5492	17.2	46.9%	641
	Phi-4	320	4486	14.0	44.87%	584
Meetings	Qwen3-8B	200	2569	12.8	40.9%	739
	Phi-4	200	3350	16.75	52.3%	584
Calendar	Qwen3-8B	200	1616	8.1	52.5%	365
	Phi-4	200	2172	10.86	66.9%	428
StrategyQA	Qwen3-8B	500	2865	5.73	81.1%	121
	Phi-4	500	3288	6.6	82.6%	182
ScienceQA	Qwen3-8B	500	2621	5.24	88.7%	110
	Phi-4	500	3415	6.8	91.9%	172

Table 14: Test dataset statistics used for step-level benchmark.

C.3 Test Dataset Examples

Table 15 presents example datapoints randomly picked from our benchmarking datasets.

Dataset	Question	Answer
GSM8k	Natalia sold clips to 48 of her friends in April, and then she sold half as many clips in May. How many clips did Natalia sell altogether in April and May?	- Step 1: Natalia sold 48 clips in April. - Step 2: Natalia sold half as many clips in May, which is $48 \div 2 = 24$. - Step 3: Add them: $48 + 24 = 72$. <Answer>: 72
ProofNet	If r is rational ($r \neq 0$) and x is irrational, prove that $r + x$ is irrational.	- Step 1: Assume for contradiction that $r + x$ is rational. - Step 2: Then $x = (r + x) - r$ is rational. - Step 3: This contradicts x being irrational. - Step 4: Therefore $r + x$ is irrational. <Answer>: $r + x$ is irrational
MATH	Between what two consecutive integers is $\sqrt{30}+\sqrt{50}$ located? Answer in the form "m and n".	- Step 1: $\sqrt{30} \approx 5.477$. - Step 2: $\sqrt{50} \approx 7.071$. - Step 3: Sum ≈ 12.548. - Step 4: Lies between 12 and 13. <Answer>: 12 and 13
Trips	You plan to visit 3 European cities for 14 days. Stay in Istanbul for 5 days, Tallinn for 4 days, Zurich for 7 days. From day 1-7, there is a show in Zurich. Direct flights exist: Istanbul–Tallinn, Zurich–Tallinn, Zurich–Istanbul. Find a valid trip plan.	- Step 1: Start in Zurich for days 1-7 (show). - Step 2: Fly Zurich $\rightarrow$ Tallinn, stay days 8-11. - Step 3: Fly Tallinn $\rightarrow$ Istanbul, stay days 12-14. <Answer>: Day 1-7: Zurich, Day 8-11: Tallinn, Day 12-14: Istanbul
Meetings	You arrive at Financial District at 9:00AM. Richard: Marina District 3:30-5:30PM (need 90 min). Andrew: Alamo Square 5:00-10:00PM (need 120 min). Travel: FD-Marina 15min, FD-Alamo 17min, Marina-Alamo 15min. Plan a schedule to meet both.	- Step 1: Leave FD 2:45PM, arrive Marina 3:00PM. - Step 2: Meet Richard 3:30-5:00PM. - Step 3: Leave Marina 5:15PM, arrive Alamo 5:30PM. - Step 4: Meet Andrew 5:30-7:30PM. <Answer>: FD $\rightarrow$ Marina $\rightarrow$ Alamo schedule works
Calendar	Schedule a 30min meeting for Laura and Paul on Monday between 9:00-17:00. Laura busy: 11:30-12:30, 14:30-15:00, 16:00-16:30. Paul busy: 9:30-10:00, 11:00-14:30, 15:00-17:00. Paul prefers not after 9:30.	- Step 1: Laura free 9:00-11:30. - Step 2: Paul free 9:00-9:30 and before 9:30 only. - Step 3: Intersection is 9:00-9:30. <Answer>: Meeting 9:00-9:30
StrategyQA	Did Donatello use a smartphone?	- Step 1: Donatello lived 1386-1466. - Step 2: Smartphones invented 21st century (iPhone 2007). - Step 3: Impossible for him to have used one. <Answer>: No, he did not
ScienceQA	Which word is not like the others? A. horse B. goat C. squirrel D. leg	- Step 1: Horse, goat, squirrel are animals. - Step 2: Leg is a body part. - Step 3: Outlier is D. <Answer>: D. leg

Table 15: Examples of question-answer pairs from each dataset, ordered by domain: Math (GSM8k, ProofNet, MATH), Planning (Trips, Meetings, Calendar), and QA (StrategyQA, ScienceQA).

D Human Validation of LLM-as-a-Judge

DeepSeek-R1's judgement for step-level correctness is compared against human annotations. The results are shown in Table 16. Besides the 17K PRM800K labels, we manually annotate 1000 steps spanning all studied domains. Here are the sampling details.

We first randomly shuffle the DeepSeek-R1 annotated test set. The human annotation then starts from the first row of the shuffled dataset. Human annotators are not provided with DeepSeek-R1's rationales (i.e., contents within the <think> tag) and must independently assess the correctness of each reasoning step, following the step-level correctness definition in Fig 10.

Regardless of dataset difficulty, annotators are requested to annotate at least 100 steps from each dataset, with the option to annotate more depending on their availability.

Dataset	# Steps	Acc.
PRM800K	17,067	95.29
ProofNet	193	87.05
Trips	102	93.06
Meetings	101	95.70
Calendar	102	91.09
StrQA	313	95.85
SciQA	245	99.28

Table 16: Accuracy of the DeepSeek-R1-based step-level evaluation pipeline relative to human labels.

E THEORETICAL ANALYSIS OF REPROBE EFFICIENCY

Let the target LLM generate a reasoning trace of total length T tokens, segmented into S reasoning steps with token lengths $\{m_s\}_{s=1}^{S}$ such that $\sum_{s=1}^{S} m_s = T$, and define $m_{\max} := \max_s m_s$. A PRM is a separate Transformer critic with L_{prm} layers, hidden width d_{prm}, and parameter count P_{prm}, which scores (possibly growing) text prefixes. With standard dense self-attention, evaluating the PRM over a length-T sequence incurs time at least $\Omega\big(L_{\text{prm}}\, d_{\text{prm}}\, T^2\big)$ (quadratic attention) and memory $\mathcal{O}\big(P_{\text{prm}} + L_{\text{prm}}\, T\, d_{\text{prm}}\big)$ (parameters plus KV-cache).

Hidden-states ReProbe instead taps per-token hidden representations from the frozen target LLM. Let f be the probe's per-token feature dimension. ReProbe is a lightweight Transformer with L_{rp} layers, hidden width d_{rp}, and parameter count P_{rp}, applied *within each step* to the length-m_s feature sequence, preceded by a token-wise linear projection $\mathbb{R}^f \to \mathbb{R}^{d_{\text{rp}}}$. ReProbe's total time complexity (TC_{rp}) over the full trace:

$$\begin{aligned} TC_{rp} &= \mathcal{O}\Big(\sum_{s=1}^{S}[m_s f d_{\text{rp}} + L_{\text{rp}}(m_s^2 d_{\text{rp}} + m_s d_{\text{rp}}^2)]\Big) \\ &= \mathcal{O}\Big(T f d_{\text{rp}} + L_{\text{rp}}(d_{\text{rp}} \textstyle\sum_s m_s^2 + d_{\text{rp}}^2 T)\Big). \end{aligned} \tag{1}$$

where $m_s f d_{\text{rp}}$ is the cost of the per-token feature projection, $m_s^2 d_{\text{rp}}$ is the quadratic self-attention mixing cost within a step, and $m_s d_{\text{rp}}^2$ accounts for the token-wise dense linear maps inside each Transformer block (e.g., Q/K/V and output projections, as well as the MLP). Its memory is $\mathcal{O}\big(P_{\text{rp}} + m_{\max}(f + L_{\text{rp}} d_{\text{rp}})\big)$ when step-local features/KV-cache are discarded after scoring each step. Using $\sum_s m_s^2 \leq m_{\max} T$, we obtain the upper bound

$$TC_{rp} \leq \mathcal{O}\big(T f d_{\text{rp}} + L_{\text{rp}}(d_{\text{rp}} m_{\max} T + d_{\text{rp}}^2 T)\big)\,,$$

which is near-linear in T whenever $m_{\max} \ll T$ (equivalently, S grows with T). By contrast, PRM-based verification remains at least quadratic in T due to attention over length-T prefixes and additionally incurs substantially larger memory complexity (typically $P_{\text{rp}} \ll P_{\text{prm}}$).

F PROMPTS

F.1 PROMPTS FOR REASONING

For LLMs in non-thinking mode, we use a domain-agnostic, format-enforcing prompt to elicit structured step-by-step reasoning, detailed in Fig 8. For thinking mode, we use a slightly different prompt specified in Fig 9.

F.2 TRAINING DATA ANNOTATION PROMPTS

For step-level annotation, we use the 2-stage prompts shown in Fig 10, while chain-level correctness annotations are obtained using the prompt in Fig 11. Since the native thinking mode generates much longer CoT than in the non-thinking mode, we use a different annotation prompt. Specifically, rather than annotating all steps in one LLM call, we annotate one step per LLM call, using the question, ground truth, and previously generated steps as input. Fig 12 shows the annotation prompt template.

```
<|im_start|>user
You will be presented with a <Question>. Before providing the [Answer], you should first think step-by-step
      carefully.

Your response format:
<start of response>
Reasoning Steps:
- Step 1: [Your first reasoning step]
- Step 2: [Your second reasoning step]
- Step 3: [Next step, and so on...]
...
- Step N: [Final reasoning step]
<Answer>: [Your final answer]
<end of response>

Strict Requirements:
- DO NOT include any text outside the specified format.
- Each reasoning step MUST be written on a **single line only**: NO line breaks, bullet points, or substeps
      within a step.
- Each step should express one precise and **self-contained** logical operation, deduction, calculation, or
      fact application.
- Steps MUST provide explicit result of the step or concrete reasoning outcomes. Avoid vague explanations or
      meta-descriptions of the reasoning process.
    - For example:
        - Good: "- Step 1: Multiply 5 by 4, which equals 20."
        - Bad: "- Step 1: Multiply 5 by 4." (no result of the step or concrete reasoning outcome)
- Continue writing steps until the problem is solved.
- Violating ANY requirement above is NOT acceptable.

Now answer:
<Question>: {q}<|im_end|>
<|im_start|>assistant
<think>

</think>
Reasoning Steps:
```

Figure 8: Prompt template for Phi-4 and Qwen3-8B in non-thinking mode. The prompt is designed to elicit structured step-by-step reasoning.

```
<|im_start|>user
Answer the following question. Enclose your answer in <answer></answer>.
<Question>: {q}<|im_end|>
<|im_start|>assistant
```

Figure 9: Prompt template for Qwen3-1.7B and Qwen3-32B in native thinking mode.

G THE USAGE OF LLMS

In this paper, we mainly use LLMs as objects of study. We also use DeepSeek-R1 and Qwen3-8B as training data annotators, as detailed in §3 and §F.1. We also use DeepSeek-R1 as a judge to grade answer correctness, as detailed in §4. DeepSeek-R1's accuracy in these tasks is manually validated through human annotation (see Table 16). In coding and writing, we use LLM assistants (e.g., ChatGPT) to identify grammar errors and debug. Such usage is under careful human supervision.

H LIMITATIONS

§5 shows that ReProbe performance increases with larger training data and benefits from data diversity. Curves in Fig 3 and Fig 5, although show diminishing marginal gains, do not seem to reach the top for tasks like StrategyQA. Therefore, it seems possible to further unleash the potential of ReProbe with further data scaling – sampling more reasoning trajectories per question or adding new questions beyond the PRM800K training set. In this work, we do not involve questions outside the PRM800K training set to establish a fair comparison with PRMs trained on data derived from this set (e.g., Qwen2.5-Math-7B-PRM800K). Due to a limited budget for annotation (annotation of 32K reasoning trajectories with DeepSeek-R1 cost >2000 USD), we also do not annotate more reasoning trajectories per question. We leave this promising scaling to future work, including the integration of diverse, high-quality questions outside the math domain.

Step-Level Annotation – Stage 1 Prompt:

```
You are given a problem, a ground-truth solution, and a step-by-step student solution. Your task is to
      analyze each step in the student's solution to determine whether it is both logically correct and
      relevant.

Instructions:
- Carefully examine each student step for logical errors or unnecessary/redundant reasoning.
- If all steps are correct and they lead to the same final answer as the ground-truth solution, conclude that
       there are no errors.
- If any step contains an error that would prevent the student from reaching the correct solution, identify
      and report those specific steps with an explanation.

PROBLEM:
{problem}

GROUND-TRUTH SOLUTION:
{answer}

STUDENT'S SOLUTION STEPS:
{steps}

Now, please evaluate whether the student's steps are correct and logical.
```

Step-Level Annotation – Stage 2 Prompt (Postprocessing):

```
You are given:
- A problem
- A student's step-by-step solution (as a Python list of string steps)
- An assessment of student's solution

Your task:
Output a single Python list where each element is:
- 1 if the corresponding step is correct
- 0 if the step is incorrect

Important:
- Output only the list, nothing else.
- The list must have the same length as the number of steps.

PROBLEM:
{problem}

STUDENT'S SOLUTION STEPS:
{steps}

ASSESSMENT OF STUDENT SOLUTION STEPS:
{reply}

OUTPUT LIST:
```

Figure 10: Two-step prompting procedure for step-level correctness annotation. The first stage evaluates the solution and identifies flaws, while the second converts this into binary correctness labels.

ReProbes need to be trained on the internal states of the target LLMs they supervise. Therefore, contrary to PRMs, they cannot be directly applied to another LLM since they depend on model-specific internal states. However, ReProbes are highly efficient in parameter size, training data, and training compute, making their training relatively inexpensive. Moreover, once trained, ReProbes can significantly reduce inference costs compared to PRMs. In practice, applications may focus on a small number of widely used models. If ReProbes for these models are shared online (e.g., on HuggingFace), practitioners could readily download and apply them without training from scratch. Future work may also investigate fine-tuning ReProbes to adapt to customized or fine-tuned versions of target LLMs.

```
You will be given a <Problem> and its proposed <Solution>. Your task is to assess whether the solution is **
      correct** or **incorrect**.

Respond using the **exact format** below, do not include any text outside this template.
Output format:
<start of response>
Solution comments:
... your comments on the solution, explaining reasoning, pointing out any errors or confirming correctness
      ...
<Grade>: (Correct|Incorrect)
<end of response>

<Problem>: {problem}

<Solution>: {solution}
```

Figure 11: The prompt used for annotating chain-level correctness by evaluating the full reasoning trace.

```
You are given a problem, its ground-truth solution, and a student's (incomplete) reasoning process so far.
Your task is to determine whether the **latest step** in the student's reasoning is correct.

Instructions:
- A step is **wrong** if it contains explicit logical or computational errors, or if it contradicts any
      previous steps.
- Redundant, unnecessary, or non-informative steps are **not** considered wrong.
- If the latest step is correct, output **1**. If it is wrong, output **0**.
- Respond with the number (0 or 1) **only**, with no extra text or explanation.

PROBLEM:
{problem}

GROUND-TRUTH SOLUTION:
{answer}

STUDENT'S REASONING PROCESS SO FAR:
{steps}

THE LATEST STEP TO EVALUATE:
{latest_step}
```

Figure 12: Prompt template for annotating correctness of reasoning steps in native thinking mode.