TR-ICRL: Test-Time Rethinking for In-Context Reinforcement Learning

Anonymous ACL submission

Abstract

In-Context Reinforcement Learning (ICRL) enables Large Language Models (LLMs) to learn online from external rewards directly within the context window. However, a central challenge in ICRL is reward estimation, as models typically lack access to ground-truths during inference. To address this limitation, we propose **Test-Time Rethinking for In-Context Reinforcement Learning (TR-ICRL)**, a novel ICRL prompting framework that leverages Test-Time Scaling to generate reward-like feedback during inference. TR-ICRL operates by first retrieving the most relevant instances from an unlabeled evaluation set for a given query. During each ICRL iteration, LLM generates a set of candidate answers for every retrieved instance. Next, a pseudo-label is derived from this set through majority voting. This label then serves as a proxy to give reward-like guidance and generate formative feedback, guiding LLM learn in-context and online, instead of supervised data. In the end, this self-improvement context is integrated with the original query to form a comprehensive prompt, with the answer determining through a final round of majority voting. TR-ICRL is evaluated on mainstream reasoning and knowledge-intensive tasks, where it demonstrates significant performance gains. Remarkably, TR-ICRL improves Qwen2.5-7B by 21.23% on average on MedQA and even **137.59%** on AIME2024. Extensive ablation studies and analyses further validate the effectiveness and robustness of our approach. Our code is available at https://anonymous.4open.science/r/TR-ICRL.

1 Introduction

Large Language Models (LLMs) (Yan et al., 2025; Guo et al., 2025; Jaech et al., 2024) have demonstrated remarkable advancements across a wide range of domains. Recently, OpenAI's o1 (Jaech et al., 2024) and Deepseek-R1 (Guo et al., 2025) have shown that Test-Time Scaling (TTS) (Zhang et al., 2025b; Balachandran et al., 2025) can significantly enhance the reasoning capabilities of LLMs by leveraging additional computational resources during inference.

Following the breakthrough of in-context supervised learning (Brown et al., 2020), there is growing interest in formalizing In-Context Reinforcement Learning (ICRL) (Moeini et al., 2025; Monea et al., 2024; Ye et al., 2026). While early theoretical instantiations were predominantly confined to simulated environments such as multi-armed bandits (Krishnamurthy et al., 2024), ICRL has rapidly evolved into a versatile paradigm for sophisticated reasoning. Recent empirical evidence (Song et al., 2025) demonstrates that in-context reinforcement learners exhibit a powerful emergent capability to autonomously refine their strategies across diverse, high-stakes domains. These applications now range from orchestrating complex scientific experiments and open-ended creative writing to solving olympiadlevel mathematics. By bypassing the need for explicit gradient updates (Brown et al., 2020), ICRL enables LLMs to adapt to the nuances of these knowledge-intensive tasks in real-time, purely through contextual interaction.

Despite these advancements, this raises a fundamental question: **how can Test-Time Scaling be optimally interleaved with in-context reinforcement learning to facilitate autonomous strategy refinement?** A primary bottleneck in deploying ICRL within open-ended environments is its traditional dependence labeled data. In the absence of external supervision, a fundamental challenge arises: how to derive robust reward signals during test time to facilitate self-improvement without access to ground-truth labels.

To mitigate these challenges, we introduce **Test-Time Rethinking for In-Context Reinforcement Learning (TR-ICRL)**, a novel ICRL prompting framework that leverages Test-Time Scaling to gen-

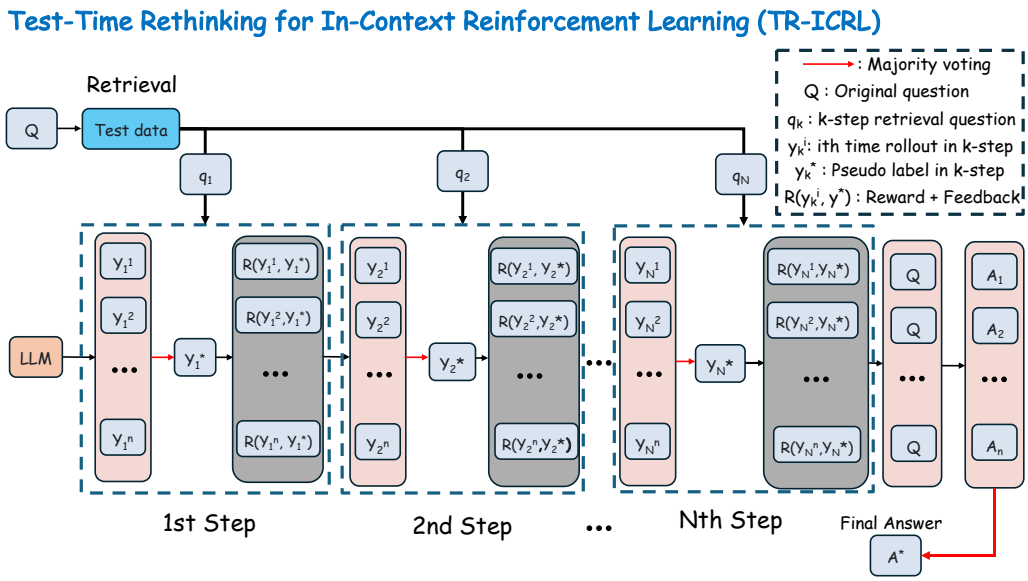

Figure 1: **TR-ICRL** combines both Test-Time Scaling (TTS) and In-Context Reinforcement Learning (ICRL).

erate reward-like feedbacks, thereby bolstering the performance of In-Context Reinforcement Learning during inference, as illustrated in Figure 1.

TR-ICRL begins by retrieving the N most similar instances from the unlabeled evaluation set for a given input query. For each retrieved instance, the model executes K parallel rollouts. A pseudo-reward signal is then constructed via majority voting, which is further transformed into natural language feedback to guide **rethinking** stage. In this stage, the LLM processes a reward message to generate formative feedback, specifically identifying and rectifying latent reasoning fallacies. Once these N iterations are complete, the model synthesizes the full contextual history—integrating retrieved predictions and their corrective feedback—to address the original query. Through iterative ICRL updates, the model progressively builds a self-improvement context that integrates retrieved experiences, pseudo-reward signals, and corrective feedback. Finally, an aggregation phase compiles all candidate answers, using majority voting to determine the final output.

In our experiments, we evaluate TR-ICRL across a diverse suite of models, including two instruction-tuned models and two Large Reasoning Models (LRMs). We evaluate TR-ICRL on 5 reasoning benchmarks and 2 knowledge-intensive benchmarks to assess the its versatility. Notably, integrating TR-ICRL with Qwen2.5-7B yields a substantial average improvement of 58.92% (increasing from 34.80 to 55.30) on AMC, alongside a remarkable **137.59%** gain (from 7.9 to 18.7) on the more challenging AIME2024. For knowledge-intensive tasks, Llama3.1-8B exhibits a marked improvement of 21.22% on MedQA and a significant 36.68% increase on MedXpertQA. Our preliminary findings demonstrate that TR-ICRL is highly effective across diverse model scales and task domains. Furthermore, we conducted ablation studies to evaluate the retrieval quality and the ranking of the selected questions. Notably, the framework exhibits robust scalability, suggesting a high performance ceiling as model capacity increases. In conclusion, our contributions can be summarized as follows:

- We introduce **Test-Time Rethinking for In-Context Reinforcement Learning (TR-ICRL)**, a novel ICRL prompting framework designed for both intensive reasoning and knowledge-intensive tasks.

- We evaluate TR-ICRL on multiple LLMs across five mathematical reasoning and two knowledge-intensive benchmarks, demonstrating consistent and significant improvements over the base models.

- Extensive experiemntal results demonstrate that TR-ICRL enables efficient and stable ICRL in a fully unsupervised manner, effectively eliminating the requirement for ground-truth labels during inference.

2 Related Work

2.1 In-Context Reinforcement Learning

Reinforcement Learning (Zhang et al., 2025a) has emerged as a primary catalyst for augmenting the reasoning capabilities of LLMs. Building upon this foundation, In-Context Reinforcement Learning (Moeini et al., 2025) has been formalized to describe models that adapt their behavior dynamically without the need for gradient updates. A central tenet of ICRL involves conditioning a policy π_θ on both the current state s_t and a dynamic context C_t, where actions are sampled according to $\pi_\theta(a_t \mid s_t, C_t)$ (Duan et al., 2016). While C_t can be instantiated through various mechanisms, a prevalent convention defines it via the historical trajectory τ_t. The efficacy of ICRL rests on the hypothesis that the forward pass of a static neural network θ implicitly executes an RL procedure, enabling the policy to generalize to out-of-distribution Markov Decision Processes at test-time (Laskin et al., 2022). This phenomenon often termed in-context improvement—posits that LLM performance scales monotonically with context length, provided C_t remains semantically relevant to the underlying task.

2.2 Test-Time Scaling

Test-Time Scaling (Zhang et al., 2025b; Balachandran et al., 2025) enhances the reasoning capabilities of LLMs during inference by leveraging additional computational resources without altering model weights. A foundational technique is CoT (Wei et al., 2022), which encourages models to "think step by step" (Lightman et al., 2023), significantly improving performance on complex tasks. More structured approaches include Best-of-N (BoN) sampling (Brown et al., 2024), beam search (Snell et al., 2024), and Monte Carlo Tree Search (Zhang et al., 2024). These methods generate multiple candidate solutions, often applying majority voting (Stiennon et al., 2020), PRM (Yuan et al., 2024) as verifier, or LLM-as-a-judge (Zheng et al., 2023) for greater accuracy.

3 TR-ICRL

We introduce our prompting framework, illustrated in Algorithm 1, which formulates test-time reasoning as an iterative ICRL process over retrieved contextual instances. Specifically, the framework operates in three parts: (1) Context Construction: Given a target question, we retrieve the most semantically similar questions from the unlabeled evaluation set, which serve as contextual instances for ICRL (Section 3.1). (2) Test-Time In-Context Rethinking: For each retrieved instance, the model generates an initial reasoning trajectory. A pseudo-reward signal is constructed via majority voting over multiple rollouts. Conditioned on this signal, the model produces the natural language feedback that is used to iteratively refine its in-context reasoning state, effectively updating its self-improvement context (Section 3.2). (3) Final Prediction: The reasoning outputs induced from all retrieved contexts are combined, and the final prediction is selected via majority voting (Section 3.3).

3.1 Context Construction

Unlike traditional reinforcement learning, which improves a policy through parameter updates, TR-ICRL operates entirely in the in-context space without modifying model weights. This enables test-time reasoning through iterative refinement of contextual states rather than optimization in parameter space, making it a novel ICRL prompting framework. Under this paradigm, unlabeled instances retrieved from the evaluation set serve as contextual reasoning experiences that form the basis for in-context learning dynamics. These experiences enable the model to simulate policy improvement through pseudo-rewarded feedback signals, without requiring ground-truth supervision or explicit gradient-based learning.

To ensure high task-relevance, we employ an embedding model to vectorize the input queries. We then compute the cosine similarity between the target question and the unlabeled set to retrieve the most semantically similar instances. These retrieved cases serve as the initial prompts for the Test-Time In-Context Rethinking process.

3.2 Test-Time In-Context Rethinking

Inspired by self-consistency with CoT (Wang et al., 2022), which shows that correct answers tend to form dense and consistent clusters among multiple model outputs, we generate multiple predictions for the retrieved questions and then apply majority voting to get the pseudo-labels. These labels serve as a ground-truth proxy during the rethinking stage and generates reward messages based on its correctness. By incorporating these rewards back into the context, the LLM learns from its prior outputs. This allows the model to systematically identify reasoning flaws and develop a more comprehensive

Algorithm 1 Test-Time Rethinking for In-Context Reinforcement Learning (TR-ICRL)

Input: Query x_{test}, unlabeled evaluation set $\mathcal{D}$, LLM π_θ, Step N, Rollout number K, Retrieval $\mathcal{R}$.
Output: Final prediction y^*.

$\mathcal{Q} = \{q_1, q_2, \ldots, q_N\} \leftarrow \mathcal{R}(x_{\text{test}}, \mathcal{D})$
Initialize K message buffers $\mathcal{M}^{(0)} = \{m_1^{(0)}, \ldots, m_K^{(0)}\}$ with system prompt.
for $i = 1$ **to** N **do**
  $\{a_i^{(1)}, \ldots, a_i^{(K)}\} \leftarrow \pi_\theta(m_k^{(i-1)} \oplus q_i)$ for $k = 1 \ldots K$
  $\hat{y}_i \leftarrow \text{Vote}(\{\text{Extract}(a_i^{(k)})\}_{k=1}^K)$
  $\{r_i^{(1)}, \ldots, r_i^{(K)}\} \leftarrow \text{Reward}(\text{Extract}(a_i^{(k)}), \hat{y}_i)$ for $k = 1 \ldots K$ // Reward calculations.
  $\{f_i^{(1)}, \ldots, f_i^{(K)}\} \leftarrow \pi_\theta(m_k^{(i-1)} \oplus q_i \oplus a_i^{(k)} \oplus r_i^{(k)})$ for $k = 1 \ldots K$ // Generate feedbacks.
  $m_k^{(i)} \leftarrow m_k^{(i-1)} \oplus (q_i, a_i^{(k)}, r_i^{(k)}, f_i^{(k)})$ for $k = 1 \ldots K$ // Update in-context memory.
end for
Construct final prompts: $\mathcal{P} \leftarrow \{m_k^{(N)} \cup \{x_{\text{test}}\}\}_{k=1}^K$
$\{\hat{a}_*^{(1)}, \ldots, \hat{a}_*^{(K)}\} \leftarrow \text{Parallel}(\pi_\theta(p)$ for $p \in \mathcal{P})$
$y^* \leftarrow \text{Vote}(\{\text{Extract}(\hat{a}_*^{(k)})\}_{k=1}^K)$
return y^*

understanding of the problem at inference time. We provide a detailed description of the overall process below.

Rollout Given the retrieved questions from the context construction stage, LLM generates the multiple predictions, in the format of zero-shot CoT reasoning (Kojima et al., 2022).

For each retrieved question $\hat{x}_i$, the LLM input α_i is formatted as:

$$\alpha_i = \text{Q: } \hat{x}_i. \text{ A: } [\texttt{Z}], \tag{1}$$

where i represents the i-th step, and [Z] represents zero shot trigger (Guo et al., 2025). More details about the triggers used for different benchmarks are described in Appendix A. Subsequently, based on the input α_i, LLM is instructed to generate the prediction for the retrieved question, obtaining the y_i. Furthermore, recognizing that some LLMs may not fully comply with the instructions when answering a query (e.g., by refusing to answer), TR-ICRL incorporates an additional filtering step to exclude abnormal responses that deviate from the given instructions.

Rethinking The rethinking stage in TR-ICRL consists of two distinct processes: **reward** and **feedback**. During the reward process, we use majority voting to get a pseudo-label y^*, *i.e.*,

$$y^* = \arg\max_{c \in \mathcal{C}} \sum_{i=1}^{i} \mathbf{1}\{y_i = c\}. \tag{2}$$

Majority voting serves as a pseudo-reward construction mechanism that supports iterative contextual refinement in TR-ICRL. Then the prediction y_i is evaluated against the corresponding the pseudo-label y^*, which can be given as:

$$R_i = \begin{cases} R_{\text{correct}}, & \text{if } y^* = y_i, \\ R_{\text{wrong}}, & \text{otherwise.} \end{cases} \tag{3}$$

Specifically, we assign a binary reward based on correctness (e.g., 'Well done! Your answer is correct.')."

While existing studies (Dai et al., 2022) establishes a theoretical duality between in-context learning and fine-tuning, a critical functional discrepancy remains: fine-tuning explicitly computes gradients based on the divergence between predictions and ground-truth labels.

To bridge this gap, we introduce a feedback process wherein the LLM generates a corrective response f_i, conditioned on a reward message. This process is designed to emulate the weight-adjustment logic of a gradient update; specifically, it lets the model rethink its prior output through the reward message. For correct predictions, this feedback reinforces the established logical trajectory. Conversely, for incorrect instances, it facilitates error diagnosis and rectifies flawed intermediate reasoning steps.

3.3 Final Prediction

The original question is appended to the context messages, and the combined messages are presented to the model. The model generates the final responses from the enriched context messages, enabling reasoning guided by the prediction to similar questions and the associated reward messages. Each generated response can be given as:

$$A_i = \text{LLM}(D_i, x). \tag{4}$$

In this context, D_i refers to the context messages in i-th time sample.

Finally, the model generates N corresponding responses to the original question. Each generated response is considered a candidate answer, $A = \{A_1, A_2, \ldots, A_N\}$. The final answer A^* is determined by majority voting, selecting the candidate that appears most frequently.

4 Experiments

4.1 Experimental Setup

Models To evaluate the generality of TR-ICRL across different backbones, we conduct experiments using Qwen2.5-7B-Instruct (Yang et al., 2024) and Llama3.1-8B-Instruct (Grattafiori et al., 2024) as instruct-tuned models. For large reasoning models, we employ Qwen3-8B (Yang et al., 2025) and DeepSeek-R1-0528-Qwen3-8B (Guo et al., 2025).

Benchmarks To evaluate the applicability of TR-ICRL across reasoning and knowledge-intensive tasks of varying difficulty, we assess its performance on three widely used reasoning benchmarks: MATH500 (Hendrycks et al., 2021), AMC (Li et al., 2024), and GSM8K (Cobbe et al., 2021), as well as on two more challenging reasoning benchmarks, AIME2024[1] and AIME2025[2]. To assess performance on knowledge-intensive tasks, we evaluate TR-ICRL using MedQA (Jin et al., 2021), a standard medical benchmark. Furthermore, we extend our evaluation to MedXpertQA[3], a challenging medical knowledge benchmark.

Implementation Details We employ vLLM (Kwon et al., 2023) for online inference, deploying the model on 2×NVIDIA A100 (80GB) GPUs. For context retrieval, we utilize Qwen3-8B-Embedding (Zhang et al., 2025c) to generate vector representations. During inference, we sample responses using a temperature of 0.6 and $top_p = 0.8$, with the maximum number of generated tokens to 8192. For each retrieval question, we sample 8 times to get pseudo-label via majority voting. In the majority voting, we select the answer that appears most frequently as the final prediction. If there are multiple options with the same frequency, we randomly select one as the final answer. We use accuracy as the evaluation metric.

[1] https://huggingface.co/datasets/HuggingFaceH4/aime_2024

[2] https://huggingface.co/datasets/opencompass/AIME2025

[3] https://huggingface.co/datasets/TsinghuaC3I/MedXpertQA/tree/main/Text

4.2 Main Results

TR-ICRL performs well on most tasks and models TR-ICRL achieves robust and significant improvements across various benchmarks. The experimental results, detailed from step 1 through step 10 in Figure 2, demonstrate the scalability and consistency of our approach. on common reasoning benchmarks, Qwen2.5-7B with TR-ICRL improves performance by 16.72% on MATH500 and 58.91% (from 34.80 to 55.30) on AMC, demonstrating consistent gains on standard mathematical reasoning tasks. More notably, on the more challenging reasoning benchmarks, TR-ICRL leads to dramatic relative improvements ranging from **137.59%** (from 7.90 to 18.77) on AIME2024 and **109.84%** (from 6.88 to 14.43) on AIME2025 for Qwen2.5-7B. These results indicate that TR-ICRL is particularly effective at enhancing complex, multi-step reasoning capabilities where base models struggle most.

Beyond reasoning tasks, TR-ICRL also exhibits strong generalization to knowledge-intensive tasks. On the MedQA, LLaMA3.1-8B equipped with TR-ICRL outperform their respective backbones by 26.43%. On more challenging medical benchmark MedXpertQA, Llama3.1-8B with TR-ICRL surpasses the backbone by 36.68% (from 13.20 to 18.04). These results underscore the broad applicability and robustness of TR-ICRL, demonstrating its effectiveness across both reasoning and knowledge-intensive tasks.

However, we observe a performance degradation during the latter stages of TR-ICRL. In the iterative rethinking process, the context serves as a dynamic buffer that accumulates task instructions, multiple rollouts, reward signals, and feedback. As this sequence length expands, the model becomes increasingly susceptible to informational interference. Specifically, the model struggles to maintain contextual focus when reconciling contradictory or noisy data pairs within a long-form sequence. This distraction from accumulating history can dilute the signal of the most recent rewards, leading to a breakdown in the model's ability to internalize the optimal reasoning path.

TR-ICRL Performs well on LRMs LRMs are increasingly becoming central to contemporary research and applications. We thus conducted experiments to evaluate the effectiveness of TR-ICRL on

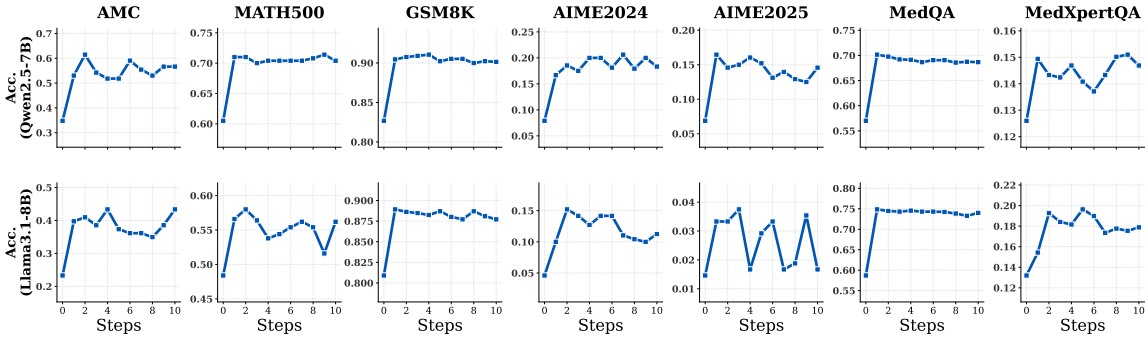

Figure 2: We evaluate **TR-ICRL** across a range of 1 to 10 ICRL steps within both reasoning and knowledge-intensive tasks. Performance at step 0 serves as the experimental baseline.

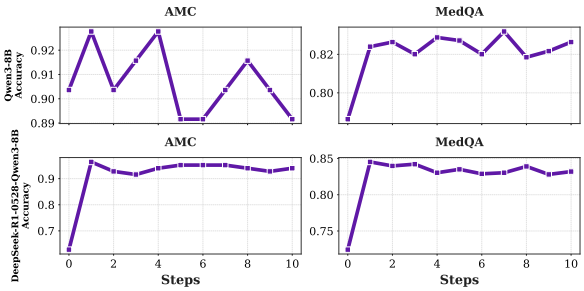

Figure 3: The evolution of LRM performance.

LRMs. Our results demonstrate that LRMs achieve significant improvements by integrating their inherent reasoning capabilities with TR-ICRL. As illustrated in Figure 3, DeepSeek-R1-0528-Qwen3-8B exhibits a remarkable **49.79%** (from 62.82 to 94.09) improvement on the AMC benchmark. Furthermore, this approach extends effectively to the medical domain; on the MedQA benchmark, the same model yields a significant 15.28% increase in accuracy, underscoring the robustness of the proposed method across diverse reasoning tasks. On Qwen3-8B, contextual interference is markedly more severe; specifically, performance at steps 6 and 10 on the AMC falls below the baseline.

4.3 Ablation study

To evaluate the specific impact of the retrieved question distribution, we conduct ablation studies using three alternative selection strategies: (1) Random: We select questions by randomly sampling from the unlabeled evaluation set, rather than relying on contextual similarity. (2) Min-Similarity: We intentionally retrieve questions with the lowest similarity scores to test the boundaries of relevance. (3) Cross-Domain: We select questions from unrelated fields to assess the robustness of TR-ICRL when retrieved examples originate from a different distribution. For instance, we utilize MedQA (medical domain) as the retrieval source for mathematical problems and MATH500 (math domain) as the source for medical queries. The results are summarized in Figure 4, with the detailed descriptions are provided in Appendix B.

Min-similarity and random configurations consistently underperformed. This performance drop underscores the critical role of context relevance in ICRL. Cross-domain configuration highlights a clear limitation in TR-ICRL regarding out-of-distribution generalization; specifically, the guidance provided by cross-domain cases is demonstrably less effective than that derived from in-domain.

5 Analysis

5.1 Contextual coherence in TR-ICRL

To evaluate the impact of contextual relevance on model performance, we conducted a series of experiments varying the presentation order of retrieved content. We compared three distinct configurations—increasing similarity, decreasing similarity, and randomized ordering, as illustrated in Figure 5.

Across all three datasets, the **increasing** configuration consistently outperformed both the decreasing and random strategies. On Math500, the increasing strategy maintained a significant margin, achieving a peak accuracy of 71.4% at step 9. This suggests that the model's reasoning efficacy is enhanced when highly relevant information is presented in closer proximity to the query prompt. Conversely, the decreasing strategy exhibited lower accuracy, most notably on the AMC dataset where performance reached a nadir of approximately 0.46 at step 5. The performance fluctuations observed in the decreasing configuration across datasets indicate that distancing highly relevant context from the final instruction introduces noise and degrades

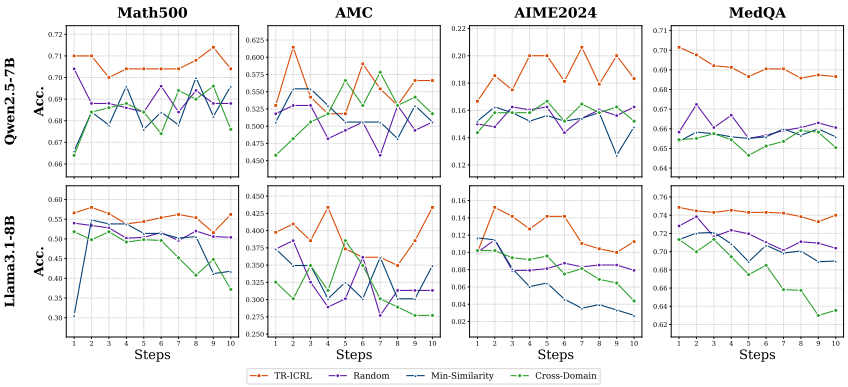

Figure 4: Ablation study of retrieved question distribution in TR-ICRL.

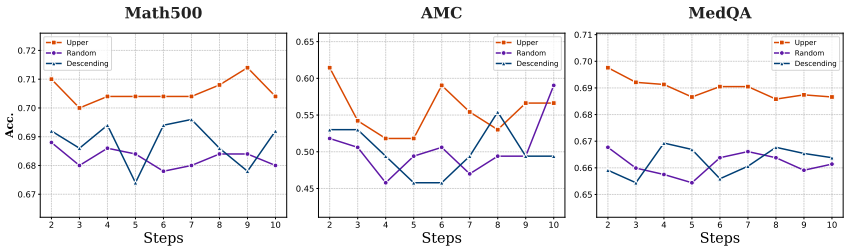

Figure 5: Performance impact of contextual sequence ordering across diverse benchmarks.

reasoning quality. These findings demonstrate that for TR-ICRL, prioritizing the proximity of relevant information to the final reasoning step is a critical design choice for maximizing performance.

5.2 Why TR-ICRL work in challenging benchmarks?

We observed a remarkable phenomenon: on challenging benchmarks such as AIME 2024, the base model initially achieves a score of only **7.9**. However, without any additional fine-tuning, it reaches significantly higher performance through TR-ICRL. We attribute this leap to a reward mechanism based on majority voting, a phenomenon we define as the '**Lucky reward**'.

For an incorrectly predicted answer, even if the estimated label does not match the ground truth label, as long as it differs from the predicted answer, we will still output a negative reward. Namely, it is sufficient that the estimated label differs from the predicted answer for us to assign the correct negative reward. Reward messages are denser than labels, allowing for more opportunities to get useful reward messages even when the estimated label is inaccurate.

Therefore, we find that the majority voting rewards in TR-ICRL remain remarkably accurate even as model capability decreases. As shown in Figure 6, on AIME2024, for instance, while majority voting achieves a raw accuracy of only **16.56%**, the resulting reward accuracy reaches **71.71%**. A similar trend appears in AIME2025, where a low accuracy of **20.00%** still yields a reward accuracy of **73.33%**.

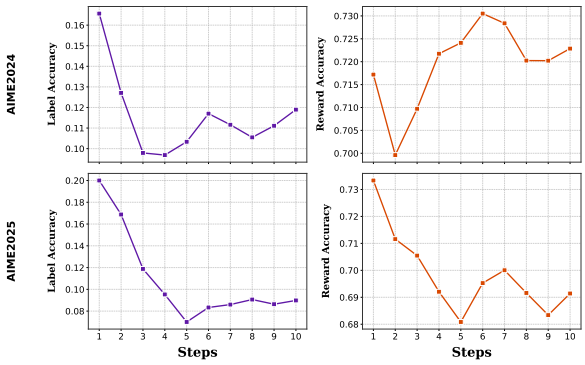

Figure 6: Comparison of label accuracy and reward accuracy on AIME2024 and AIME2025. Even with low label accuracy, reward accuracy remains high due to "lucky reward".

'Lucky reward' ensures that most outputs receive correct reinforcement despite inaccurate label estimation. While poorer model performance leads to more frequent mistakes, it also raises the likelihood that an estimated label will accurately flag those mistakes as incorrect. This phenomenon ensures a consistently high reward accuracy, creating a reliable signal that supports effective learning during the test time.

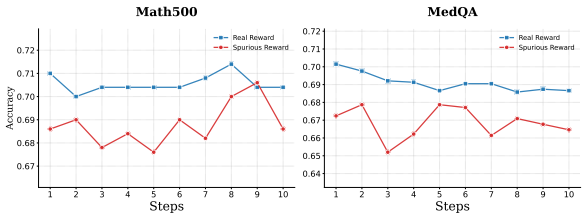

Figure 7: Results of spurious rewards on Qwen2.5-7B.

5.3 Do reward messages help?

Inspired by (Shao et al., 2025), we wanted to explore the effectiveness of reward messages in ICRL. We also conducted a spurious reward experiment on Qwen2.5-7B.

As shown in Figure 7, we observe a consistent performance gap between TR-ICRL and the spurious rewards setting across all configurations. Under the real reward setting, the model maintains a relatively high accuracy, fluctuating between 0.70 and 0.715. Conversely, the spurious reward consistently results in lower accuracy, generally staying below 0.69. Notably, at step 9, the spurious reward performance briefly converges with the real reward, before sharply declining again at step 10. The performance gap is even more pronounced in the knowledge-intensive tasks.

These results highlight the critical role of reward message correctness. When rewards aligned with the majority consensus, the model is encouraged to consolidate reliable reasoning trajectories. In contrast, spurious reward introduces systematic inconsistencies between correct answers and feedback signals, thereby distorting the learning signal and limiting performance improvements.

Interestingly, the spurious reward setting still outperforms the vanilla baseline. We attribute this phenomenon to several factors. Integrating the feedback into the context messages facilitates a more effective dual descent gradient. This approach establishes a more precise dual formulation that bridges In-Context Learning and fine-tuning. For relatively simple questions, the model often derives the correct answer based on prior knowledge or commonsense reasoning. Even when prompted to revise its reasoning, the model tends to converge to the same conclusion, meaning that spurious rewards have minimal impact on the final prediction.

In contrast, for sufficiently difficult questions where the model consistently fails to produce correct answers, spurious rewards actually tell them that they answered incorrectly and help the model reconsider its reasoning trajectory.

Table 1: Comparison of TR-ICRL with BoN, Self-Refine and Reflexion.

Setting	Acc.(MATH)	Acc.(AMC)
TR-ICRL (Step 1)	**71.00**	53.01
BoN (N=16)	69.60	43.37
MCTS	65.00	36.14
Self-Refine (Iteration=8)	61.20	42.17
Reflexion	66.60	**55.42**

5.4 Comparisons with other methods

To evaluate its efficiency, we compare the performance of TR-ICRL (Step 1) against several established baselines, such as Best-of-N (Brown et al., 2024), Self-Refine (Madaan et al., 2023) and Reflexion (Shinn et al., 2023), under a **fixed token** budget, as well as MCTS under a maximum token constraint, as shown in Table 1 and more detailed comparisons are shown in appendix D.

TR-ICRL consistently outperforms most baselines, especially BoN, by using a more structured and feedback-driven reasoning process rather than relying on diverse sampling. Compared to Self-Refine, it avoids limitations from relying only on internal critiques, which can be weak for complex math problems. Instead, it incorporates contextual signals like past predictions and feedback to guide reasoning more effectively. It also remains competitive with strong methods like Reflexion on benchmarks such as AMC, while showing better robustness on harder math tasks. Overall, the method improves accuracy and produces more coherent reasoning paths without increasing computational cost, making it a strong and practical approach for complex reasoning problems.

6 Conclusion

In this paper, we propose **Test-Time Rethinking for In-Context Reinforcement Learning (TR-ICRL)**, a novel ICRL prompting framework on test time without access to ground-truth labels. A central innovation of TR-ICRL is its rethinking stage, which is driven by two integrated processes: reward estimation and feedback generation. In the reward process, we leverage majority voting to derive a reliable reward. Then, the model refines its reasoning by incorporating reward messages, achieving autonomous optimization without requiring external intervention. Our experiments demonstrate the strong potential of TR-ICRL, achieving consistent improvements. As a result, TR-ICRL presents itself as a promising method for ICRL prompting.

Limitations

One limitation of TR-ICRL is its dependence on the quality and domain alignment of retrieved instances. When retrieval quality deteriorates or when retrieved examples originate from substantially different domains, the resulting pseudo-labels and feedback signals may become noisy or misleading, which can in turn weaken the effectiveness of iterative refinement. As a result, performance may degrade under cross-domain retrieval settings or in scenarios where suitable in-domain examples are unavailable. Despite its effectiveness, TR-ICRL's reliance on a simple majority vote in reward estimation may be overly reductive. Moreover, this reward estimation generates a purely binary reward signal, it lacks the nuance required for complex reasoning tasks. To enhance decision making robustness, it is essential to differentiate between various inference paths. Incorporating uncertainty metrics, such as Perplexity (PPL) or Entropy would allow the framework to quantify the confidence of each trajectory. Furthermore, replacing binary signals with a rubric-based reward system would enable more granular scoring.

References

Vidhisha Balachandran, Jingya Chen, Lingjiao Chen, Shivam Garg, Neel Joshi, Yash Lara, John Langford, Besmira Nushi, Vibhav Vineet, Yue Wu, and 1 others. 2025. Inference-time scaling for complex tasks: Where we stand and what lies ahead. *arXiv preprint arXiv:2504.00294*.

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V Le, Christopher Ré, and Azalia Mirhoseini. 2024. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Jierun Chen, Shiu-hong Kao, Hao He, Weipeng Zhuo, Song Wen, Chul-Ho Lee, and S-H Gary Chan. 2023. Run, don't walk: chasing higher flops for faster neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12021–12031.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Damai Dai, Yutao Sun, Li Dong, Yaru Hao, Shuming Ma, Zhifang Sui, and Furu Wei. 2022. Why can gpt learn in-context? language models implicitly perform gradient descent as meta-optimizers. *arXiv preprint arXiv:2212.10559*.

Yan Duan, John Schulman, Xi Chen, Peter L Bartlett, Ilya Sutskever, and Pieter Abbeel. 2016. RL^2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

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

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

Chan-Jan Hsu, Davide Buffelli, Jamie McGowan, Feng-Ting Liao, Yi-Chang Chen, Sattar Vakili, and Da-shan Shiu. 2025. Group think: Multiple concurrent reasoning agents collaborating at token level granularity. *arXiv preprint arXiv:2505.11107*.

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*.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. 2021. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14):6421.

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

Akshay Krishnamurthy, Keegan Harris, Dylan J Foster, Cyril Zhang, and Aleksandrs Slivkins. 2024. Can large language models explore in-context? *Advances in Neural Information Processing Systems*, 37:120124–120158.

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*, pages 611–626.

Michael Laskin, Luyu Wang, Junhyuk Oh, Emilio Parisotto, Stephen Spencer, Richie Steigerwald, DJ Strouse, Steven Hansen, Angelos Filos, Ethan Brooks, and 1 others. 2022. In-context reinforcement learning with algorithm distillation. *arXiv preprint arXiv:2210.14215*.

Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Huang, Kashif Rasul, Longhui Yu, Albert Q Jiang, Ziju Shen, and 1 others. 2024. Numinamath: The largest public dataset in ai4maths with 860k pairs of competition math problems and solutions. *Hugging Face repository*, 13(9):9.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Jia Liu, ChangYi He, YingQiao Lin, MingMin Yang, FeiYang Shen, and ShaoGuo Liu. 2025. Ettrl: Balancing exploration and exploitation in llm test-time reinforcement learning via entropy mechanism. *arXiv preprint arXiv:2508.11356*.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, and 1 others. 2023. Self-refine: Iterative refinement with self-feedback. *Advances in neural information processing systems*, 36:46534–46594.

Amir Moeini, Jiuqi Wang, Jacob Beck, Ethan Blaser, Shimon Whiteson, Rohan Chandra, and Shangtong Zhang. 2025. A survey of in-context reinforcement learning. *arXiv preprint arXiv:2502.07978*.

Giovanni Monea, Antoine Bosselut, Kianté Brantley, and Yoav Artzi. 2024. Llms are in-context bandit reinforcement learners. *arXiv preprint arXiv:2410.05362*.

Rulin Shao, Shuyue Stella Li, Rui Xin, Scott Geng, Yiping Wang, Sewoong Oh, Simon Shaolei Du, Nathan Lambert, Sewon Min, Ranjay Krishna, and 1 others. 2025. Spurious rewards: Rethinking training signals in rlvr. *arXiv preprint arXiv:2506.10947*.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652.

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

Kefan Song, Amir Moeini, Peng Wang, Lei Gong, Rohan Chandra, Shangtong Zhang, and Yanjun Qi. 2025. Reward is enough: Llms are in-context reinforcement learners. *arXiv preprint arXiv:2506.06303*.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. *Advances in neural information processing systems*, 33:3008–3021.

Jun Wang, Meng Fang, Ziyu Wan, Muning Wen, Jiachen Zhu, Anjie Liu, Ziqin Gong, Yan Song, Lei Chen, Lionel M Ni, and 1 others. 2024a. Openr: An open source framework for advanced reasoning with large language models. *arXiv preprint arXiv:2410.09671*.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2024b. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9426–9439.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

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

Hao Wen, Yifan Su, Feifei Zhang, Yunxin Liu, Yunhao Liu, Ya-Qin Zhang, and Yuanchun Li. 2025. Parathinker: Native parallel thinking as a new paradigm to scale llm test-time compute. *arXiv preprint arXiv:2509.04475*.

Siyu Yan, Long Zeng, Xuecheng Wu, Chengcheng Han, Kongcheng Zhang, Chong Peng, Xuezhi Cao, Xunliang Cai, and Chenjuan Guo. 2025. Muse: Mcts-driven red teaming framework for enhanced multi-turn dialogue safety in large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 21293–21314.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, and 1 others. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Yaoqi Ye, Yiran Zhao, Keyu Duan, Zeyu Zheng, Kenji Kawaguchi, Cihang Xie, and Michael Qizhe Shieh. 2026. In-context reinforcement learning for tool use in large language models. *arXiv preprint arXiv:2603.08068*.

Lifan Yuan, Wendi Li, Huayu Chen, Ganqu Cui, Ning Ding, Kaiyan Zhang, Bowen Zhou, Zhiyuan Liu, and Hao Peng. 2024. Free process rewards without process labels. *arXiv preprint arXiv:2412.01981*.

Dan Zhang, Sining Zhoubian, Ziniu Hu, Yisong Yue, Yuxiao Dong, and Jie Tang. 2024. Rest-mcts*: Llm self-training via process reward guided tree search. *Advances in Neural Information Processing Systems*, 37:64735–64772.

Kaiyan Zhang, Yuxin Zuo, Bingxiang He, Youbang Sun, Runze Liu, Che Jiang, Yuchen Fan, Kai Tian, Guoli Jia, Pengfei Li, and 1 others. 2025a. A survey of reinforcement learning for large reasoning models. *arXiv preprint arXiv:2509.08827*.

Qiyuan Zhang, Fuyuan Lyu, Zexu Sun, Lei Wang, Weixu Zhang, Zhihan Guo, Yufei Wang, Irwin King, Xue Liu, and Chen Ma. 2025b. What, how, where, and how well? a survey on test-time scaling in large language models. *arXiv preprint arXiv:2503.24235*.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, and 1 others. 2025c. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, and 1 others. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623.

A TR-ICRL Implementation Details

A.1 Question prompt template

In MedQA, we use zero-shot CoT template adapted from Deepseek-R1:

```
Q: {question}\nA: Please reason step by step, and put your final answer (selected from options A to D) within \boxed{}.
```

In MedXpertQA, because the range of answer is different, this is another template:

```
Q: {question}\nA: Please provide a step-by-step explanation, followed by your final answer (selected from options A to J) within \boxed{}.
```

In reasoning benchmark, we will use the following template to guide the responses:

```
Q: {question}\nA: Please reason step by step, and put your final answer within \boxed{}.
```

A.2 Reward template

After the model generates a prediction for a retrieval question, it is verified against the corresponding the pseudo-label. If the prediction is correct, a positively reward label is appended to the context, affirming the validity of the reasoning process. In cases of incorrect predictions, instead of explicitly pointing out the error, a supportive and constructive reward label is introduced. This approach encourages further reflection and exploration without directly identifying the mistake. When the prediction is correct:

```
User: Well done! Your answer is correct.
```

When the prediction is wrong:

```
User: Unfortunately, your answer is wrong! Review your previous answer. Find the reason for the mistake.
```

B A Detail ablation study

In ablation analysis, we selected MATH500, AMC and AIME2024 as the reasoning benchmarks, and MedQA as the knowledge-intensive benchmark.

The ablation study reveals a significant performance gap when the quality of the retrieved context is compromised. Both the random and min-similarity configurations consistently underperform relative to the standard TR-ICRL framework across all benchmarks. For a model to effectively utilize in-context reinforcement, the retrieved questions must be mathematically or logically relevant to the target problem to provide meaningful guidance during the iterative thinking process. While random sampling bypasses similarity entirely, min-similarity intentionally selects the least relevant cases; both strategies fail to match the performance gains achieved when contexts are selected based on high similarity scores.

The cross-domain strategy consistently yields the lowest performance across nearly all benchmarks, often trailing significantly behind the standard TR-ICRL. This performance gap is particularly pronounced in the AIME2024 and MedQA tasks, where accuracy drops sharply when the model is provided with out-of-distribution contexts. These results highlight a clear limitation in OOD generalization; the reasoning patterns inherent in one domain (e.g., medical knowledge) do not effectively translate to provide helpful guidance for another (e.g., competitive mathematics). The data demonstrates that guidance derived from cross-domain cases is demonstrably less effective than in-domain examples, underscoring that the benefits of TR-ICRL are highly dependent on domain-aligned context.

C TR-ICRL Is comparable to or Outperforms Large Parameter LLMs

To ensure a fair and rigorous comparison between the performance of smaller models (7B) empowered by **TR-ICRL** and larger baseline models (72B), we conducted a controlled-variable analysis focusing on total computational overhead (FLOPs) (Chen et al., 2023).

For a single forward pass, the total FLOPs can be approximated as $C \approx 2 \cdot P \cdot N_{\text{tokens}}$, where P is the parameter count and N_{tokens} is the sequence length. We define the **Relative Compute Cost (RCC)** as:

$$RCC = P \times \text{Avg. Tokens per Response} \quad (5)$$

By fixing the computational budget, we demonstrate that TR-ICRL enhanced small models not only outperform larger models but do so with high

Model	Params	Avg. Tokens (k)		Relative Cost (units)		Accuracy (%)	
	(B)	MATH	AMC	MATH	AMC	MATH	AMC
Qwen2.5-72B	72	0.62	0.92	89.28	**132.48**	62.10	41.11
TR-ICRL (step 1)	7	**0.48**	**0.81**	**80.64**	136.08	**71.00**	**53.01**

Table 2: Resource vs. Performance Comparison on MATH and AMC Benchmarks. We report the total computational overhead using RCC. For the Qwen2.5-72B baseline, cost is calculated based on **Best-of-2** majority voting. In contrast, TR-ICRL (Step 1) achieves superior accuracy using a single iterative step within Qwen2.5-7B.

resource efficiency. As illustrated in Table 2, empirical results reveal that TR-ICRL yields substantial absolute accuracy gains of 14.3% on MATH and 28.9% on AMC over the much larger 72B baseline. These findings validate that our iterative framework effectively bridges the performance gap between model scales, enabling a 7B-parameter model to surpass a 72B-parameter model through ICRL.

D Comparison with Related Test-Time Methods

D.1 Test-Time Scaling Methods

We compared TR-ICRL with some common test-time scaling methods, as shown in Table 3. Overall, TR-ICRL consistently achieves the best performance across all evaluated tasks, substantially outperforming both the base model and existing TTS baselines such as BoN and MCTS. These improvements suggest that TR-ICRL more effectively guides the model toward high-quality reasoning trajectories, rather than relying solely on increased deeper search. In particular, while BoN benefits from diversified sampling, its gains on AMC remain limited, and MCTS exhibits weaker overall performance under the same computational budget. By contrast, TR-ICRL leverages informative contexts that predictions and reward messages, which th model internalizes through In-Context Learning, resulting in more reliable inference trajectories. In summary, TR-ICRL's ability to substantially enhance accuracy while maintaining high efficiency highlights its practical value and strong potential for reasoning tasks.

D.2 Test-Time training methods

In contrast to prior approaches that rely on explicit model training (Hsu et al., 2025; Wen et al., 2025; Liu et al., 2025), TR-ICRL operates entirely at inference time and does not require parameter updates, resulting in substantially lower computational cost, as shown in Table 4.

The central novelty of TR-ICRL lies in its reward-conditioned in-context update mechanism. Rather than merely selecting among sampled candidates, the framework iteratively revises and improves generated reasoning based on evaluated outcomes. Experimental comparisons further show that TR-ICRL achieves competitive or superior performance relative to training-based approaches while requiring substantially fewer computational resources.

Method	MATH500	AMC	MedQA
Baseline	60.50	34.80	57.00
BoN	69.60	43.37	64.02
MCTS	65.00	36.14	62.13
TR-ICRL	**71.00**	**53.01**	**70.15**

Table 3: Comparison of TR-ICRL with related test-time scaling methods on Qwen2.5-7B.

E The importance of majority voting

We conducted a comparative experiment, replacing majority voting with random voting as the pseudo-label, as shown in Table 5.

We found that removing the majority vote reduced the model's performance, but it was still better than the baseline. So we emphasize that majority voting in our method is not used as an isolated decoding trick, but as a pseudo-label construction mechanism within a broader iterative feedback loop. In TR-ICRL, voting does not directly produce the final answer; instead, it generates a reward proxy that conditions the subsequent rethinking and context-updating process. This distinguishes our approach from standard majority voting or self-consistency methods, where voting is the terminal decision rule.

Second, the performance gains cannot be attributed to majority voting alone, since the core mechanisms of TR-ICRL: (i) reward-conditioned reasoning updates, (ii) iterative in-context memory refinement, and (iii) feedback-driven correction of

Method	Acc. (%)	Training time	Inference time	Price/h	Resource	Cost ($)
1.5B Models (MATH500)						
DeepSeek-R1-Distill-Qwen-1.5B (w TR-ICRL)	73.49	0h	0.7h	$1.49	2 × A100	2.086
ParaThinker-1.5B (Wen et al., 2025)	73.10	1.8h	0.3h	$1.49	8 × A100	25.032
7B Models (GPQA)						
Qwen2.5-7B (w TR-ICRL)	42.42	0h	1h	$1.49	2 × A100	2.98
Qwen2.5-7B (TTRL*)	33.6	-	-	-	-	-
8B Models (AIME2024)						
LLaMA3.1-8B (w TR-ICRL)	15.21	0h	1.2h	$1.49	2 × A100	3.576
LLaMA3.1-8B (ETTRL) (Liu et al., 2025)	16.90	11.25h	0.5h	$1.49	8 × A100	140.06
32B Models (MATH500)						
Qwen2.5-32B (w TR-ICRL)	86.40	0h	1.1h	$1.49	2 × A100	3.278
Qwen2.5-32B (Group Think) (Hsu et al., 2025)	87.00	3.5h	0.4h	$2.12	8 × B200	66.144

Table 4: Comparison of TR-ICRL with related training methods. *: We were unable to find training script for GPQA in TTRL, and therefore are temporarily unable to report the training time and required computational resources. The results are from the original report.

Models	MATH500	AMC	MedQA
Qwen2.5-7B	60.50	34.80	57.00
Qwen2.5-7B (w TR-ICRL)	**71.00**	**53.01**	**70.15**
Qwen2.5-7B (w TR-ICRL random voting)	69.40	50.60	67.48

Table 5: Ablation study on the effect of majority voting in TR-ICRL.

Dataset	Train Num	Test Num	Options Num
MATH500	0	500	N/A
AMC	0	83	N/A
GSM8K	7473	1319	N/A
AIME2024	0	30	N/A
AIME2025	0	30	N/A
MedQA	10178	1273	4
MedXpertQA	5	2450	10

Table 6: The Statistics of benchmarks.

intermediate trajectories. These components are essential in transforming static ensemble aggregation into a dynamic inference-time optimization process.

F Additional Experiments Details

F.1 Baseline Models

For all baseline models, we use zero-shot to inference. For Large reasoning Models, we follow the corresponding recommended prompting guidelines to remove the system prompt.
The zero-shot template is :

```
Q: {question}\nA: Put your final
answer within\boxed{}.
```

F.2 Data Statistics

The detailed benchmark statistics are shown in Table 6.

F.3 Evaluation Metrics

We employ accuracy as our evaluation metric. To ensure statistical robustness on the AIME 2024 and AIME 2025 datasets, we report the mean accuracy across **16** independent trials. Performance on all remaining datasets is reported based on a single experimental trial.

F.4 Answer cleaning

As we guide the model in generating answers, we use the `\boxed{}` format to standardize the final answer output. However, due to differences across models, we apply various regular expressions to extract the final answer accurately, as displayed in Listing 1 below.

F.5 Majority voting

For each reasoning step, we extract the final answer from all K rollouts. We use a majority voting strategy to select the consensus answer. To ensure robustness in cases of parity, ties are resolved via random selection among the most frequent candidates. This ensures that the reward signal is grounded in the most probable collective hypothesis of the model, as displayed in Listing 2 below.

F.6 Details in baseline Methods

Best-of-N is implemented using OpenR (Wang et al., 2024a). For reasoning tasks, we employ Math-Shepherd-Mistral-7B-PRM (Wang et al., 2024b) as the process reward model (PRM).

Listing 1: Implementation of the boxed answer extraction function.

```
def extract_boxed_answer_r1(text):
    if text is None or len(text) == 0:
        return None
    if len(text) == 1:
        return text
    match = re.search(r'\\boxed{((?:[^{}]|\{[^{}]*\})*)}', text)
    if match:
        inner_text = match.group(1)
        if len(inner_text) == 0:
            return None
        elif len(inner_text) != 1:
            text_match = re.search(r'\\text\{([A-Za-z])\}', inner_text)
            if text_match:
                return text_match.group(1)
            else:
                return inner_text
        else:
            return inner_text
    else:
        match = re.search(r'\\boxed{(.*)}', text)
        if match:
            inner_text = match.group(1)
            if inner_text.startswith('(') and inner_text.endswith(')'):
                inner_text = inner_text[1:-1]
            return inner_text

    answer_match = re.search(
        r'(?:Final\s+)?Answer\s*:\s*\(?([A-Z])\)?', text, re.IGNORECASE)
    if answer_match:
        return answer_match.group(1)
    return None
```

For Best-of-N, we set the temperature to 0.6, generate 16 candidate sequences with a maximum of 4096 new tokens, and select the final prediction via majority voting.

For Reflexion, we evaluate the model's output by performing multiple sampling and applying a majority voting mechanism.

Listing 2: Implementation of majority voting.

```
def vote(choices) -> str:
    if not choices:
        return None
    frequency = Counter(choices)
    max_count = max(frequency.values())
    candidates = [key for key, value in frequency.items() if value == max_count]
    result = random.choice(candidates)
    return result
```