R^2-LLMs: Enhancing Test-Time Scaling of Large Language Models with Hierarchical Retrieval-Augmented MCTS

Anonymous ACL submission

Abstract

Test-time scaling has emerged as a promising paradigm in language modeling, leveraging additional computational resources at inference time to enhance model performance. In this work, we introduce **R^2-LLMs**, a novel and versatile hierarchical retrieval-augmented reasoning framework designed to improve test-time scaling in large language models (LLMs) without requiring distillation from more advanced models to obtain chain-of-thought (CoT) training data. **R^2-LLMs** enhances inference-time generalization by integrating dual-level retrieval-based in-context learning: (1) At the **coarse-level**, our approach extracts abstract templates from complex reasoning problems and retrieves similar problem-answer pairs to facilitate high-level in-context learning; (2) At the **fine-level**, during Monte Carlo Tree Search (MCTS), **R^2-LLMs** efficiently retrieves analogous intermediate solution steps from reference mathematical problem datasets, refining step-wise reasoning with the aid of a process reward model (PRM) for scoring. **R^2-LLMs** is a robust hierarchical reasoning-augmentation method that enhances in-context-level reasoning while seamlessly integrating with step-level tree search methods. Utilizing PRM, it refines both candidate generation and decision-making for improved reasoning accuracy. Empirical evaluations on the **MATH500, GSM8K, and OlympiadBench-TO** datasets achieve relative substantial improvement with an increase up to **16%** using LLaMA-3.1-8B compared to the baselines, showcasing the effectiveness of our approach in complex mathematical reasoning tasks.

1 Introduction

Emergent abilities of Large Language Models (LLMs) have traditionally relied on increased training-time computation through large-scale generative pretraining (Kaplan et al., 2020; Hoffmann et al., 2022; Wei et al., 2022a). Recently, Test-Time Scaling (TTS) has emerged as a complementary paradigm, enhancing reasoning capabilities by allocating extra computational resources at inference (Snell et al., 2024), as validated by DeepSeek-R1 (Guo et al., 2025) and OpenAI's O1 (OpenAI, 2024).

Existing TTS approaches are mainly: (1) **Self-evolution TTS**, which improves reasoning by generating extended Chain-of-Thought (CoT) sequences via large-scale reinforcement learning (RL), exemplified by DeepSeek-R1; and (2) **Search-based TTS**, which leverages pre-trained models using inference-time search strategies like Best-of-N (Brown et al., 2024), beam search (Snell et al., 2024), and Monte Carlo Tree Search (MCTS)(Zhang et al., 2025b; Guan et al., 2025). Search-based methods have gained traction for their efficiency and flexibility, often incorporating Process Reward Models (PRMs) to evaluate intermediate reasoning steps and guide the search effectively(Snell et al., 2024; Wu et al., 2024b; Face, 2024; Wang et al., 2023a).

Among search-based TTS methods, MCTS demonstrates notable advantages, as mathematical multi-step reasoning tasks inherently involve complex search processes that necessitate systematic exploration of diverse reasoning paths. MCTS excels in managing extensive search spaces by effectively balancing exploration with exploitation, efficiently prioritizing promising candidate paths, and iteratively refining solutions towards optimality (Guan et al., 2025). However, conventional PRM+MCTS approaches primarily rely on the information learned during pre-training, which can lead to local optima or exploration blind spots when encountering highly diverse or underrepresented problem distributions (Zhang et al., 2025b). Moreover, these methods depend solely on the PRM to evaluate steps within MCTS, which may fail to capture global problem-solving strategies and semantic

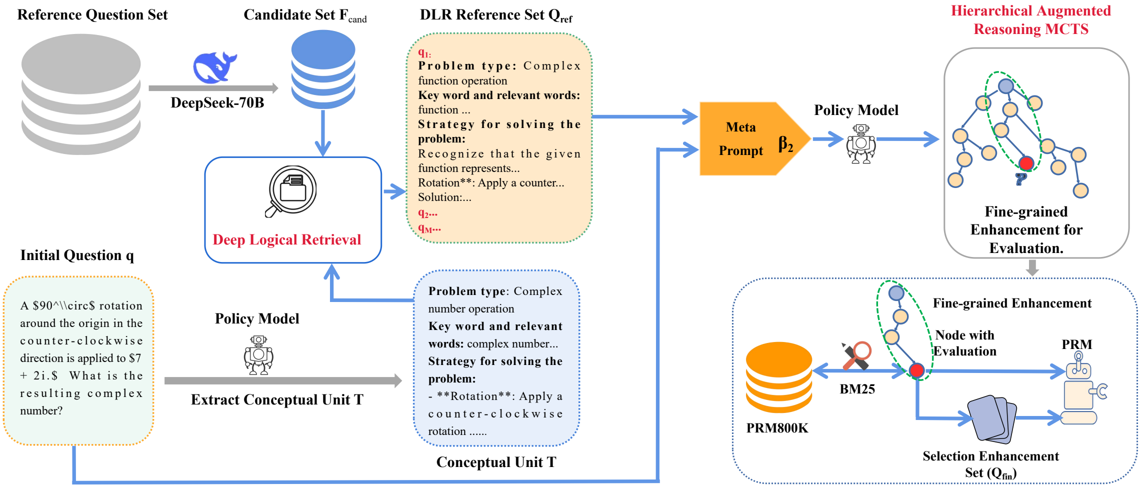


Figure 1: Illustration of the reasoning process of R^2-LLMs. R^2-LLMs employ Hierarchical Augmented Reasoning MCTS to answer the initial question, utilizing two enhancement methods: logical enhancement and fine-grained enhancement.

relationships. As a result, the reward signals guiding the search process can be sparse or suboptimal, reducing overall efficiency and accuracy. This limitation increases the risk of deepening the search along incorrect trajectories, ultimately leading to failure in complex reasoning tasks. These challenges underscore the necessity for a more effective and generalizable inference scaling approach—one that enhances reasoning capabilities without requiring extensive additional training while offering a plug-and-play search strategy to improve robustness and adaptability across diverse problem settings.

To enhance the precision of reasoning path exploration, we propose **R^2-LLMs** that leverages external retrieval to enhance inference-time generalization through a dual-level retrieval-based in-context learning mechanism. **For coarse-level**, we propose Deep Logical Retrieval in section 3.3. Our approach retrieves analogous problem-answer pairs via abstract problem templates to provide diverse exemplars, enabling the model to capture underlying patterns and variability in problem structures. This facilitates more effective in-context learning, enhancing the model's adaptability to unseen problems. **For fine-level**, we further introduce Hierarchical Augmented Reasoning MCTS in section 3.4, . During MCTS, R^2-LLMs dynamically retrieves relevant intermediate solution steps from external mathematical problem datasets, enriching the reasoning process with similar prior knowledge. By incorporating these retrieved steps, PRM can provide more informed and contextually consistent evaluations, reducing the risk of inefficient exploration.

Empirical results demonstrate that the proposed retrieval-augmented steps enable R^2-LLMs to generalize more effectively to complex and unseen problems by leveraging diverse problem-solving strategies from reference datasets. This mitigates the limitations of relying solely on the immediate problem context and significantly enhances reasoning performance. Our approach is evaluated on policy models LLaMA 3.1-8B (Dubey et al., 2024) and Qwen 2-7B (Yang et al., 2024a), outperforming ICL-based and tree-based baselines on MATH500 (Hendrycks et al., 2021), GSM8K (Cobbe et al., 2021), and OlympiadBench-TO (He et al., 2024).

2 Related Works

Test Time Scaling for LLMs. Scaling inference-time compute has emerged as a compelling paradigm for enhancing the performance of LLMs (OpenAI, 2024; Guo et al., 2025). Early work in this area explored techniques such as majority voting (Wang et al., 2023b) and best-of-N methods (Brown et al., 2024; Li et al., 2023), which generate multiple candidate solutions and select the most frequent or highest-scoring output. More advanced approaches have leveraged search-based strategies, including Monte Carlo Tree Search (MCTS) (Choi et al., 2023; Zhang et al., 2023; Liu et al., 2024; Zhou et al., 2023), to systematically explore the reasoning space and improve accuracy. To further enhance search efficiency, recent studies have integrated Process Reward Models (PRMs) to guide the selection of high-quality reasoning paths (Setlur et al., 2024; Snell et al., 2024; Lightman et al., 2023; Luo et al., 2024; Wang et al., 2023a). These models provide refined, step-wise evaluations, particularly beneficial in complex reasoning tasks. Additionally, methods such as BoT (Yang et al., 2024b) employ historical thought templates to steer exploration, achieving notable improvements in inference efficiency. ReasonFlux (Yang et al., 2025) adaptively scales fundamental and essential thought templates for simplifying the search space of complex reasoning. In contrast, our proposed R^2-LLMs framework employs a hierarchical retrieval-augmented strategy that leverages external reference data at both coarse and fine levels, enriching in-context learning and refine intermediate solution steps during MCTS to enhance PRM evaluations.

Mathematical Reasoning. Mathematical reasoning has long been one of the most challenging tasks in artificial intelligence. Early efforts relied on rule-based methods (Feigenbaum et al., 1963; Fletcher, 1985), but the advent of large language models has shifted the focus toward enhancing reasoning capabilities both during training—via fine-tuning with high-quality mathematical data (Shao et al., 2024; Yang et al., 2024a; Lewkowycz et al., 2022; Yue et al., 2023)—and at inference time through prompt engineering (Wei et al., 2022b) and self-refinement techniques (Madaan et al., 2024; Gou et al., 2023; Ke et al., 2024). More recently, researchers have advanced stepwise reasoning by decomposing complex problems into individual reasoning steps. Approaches such as Tree of Thoughts (Yao et al., 2023) and Monte Carlo Tree Search (MCTS) (Zhang et al., 2024; Chen et al., 2024; Feng et al., 2023; Zhu et al., 2022) explore multiple solution paths, with Process Reward Models (PRMs) (Lightman et al., 2023; Luo et al., 2024) providing real-time verification to prune suboptimal paths. While these methods improve accuracy, they often depend on internal model knowledge and struggle with diverse or unseen problems. In contrast, our proposed R^2-LLMs framework uses a hierarchical retrieval-augmented approach to boost test-time scaling for mathematical reasoning by integrating external reference data. Unlike previous methods that depend solely on internal reasoning and risk local optima, our approach enriches the process with diverse, contextually relevant examples and intermediate steps.

In-context learning with relevant samples In-context learning is a cost-effective guidance approach that enhances model output quality by leveraging similar examples, eliminating the need for fine-tuning (Zhou et al., 2024; Dong et al., 2022). Specifically, CoT (Wei et al., 2022b; Kojima et al., 2022) guides the model's reasoning process. Self-Consistency (SC) (Wang et al., 2023b) enhances performance by generating multiple reasoning paths and selecting the most consistent outcome. In addition, Buffer of thought (BoT) (Liu et al., 2024) enhances large language model reasoning by utilizing high-level thought templates, shifting the focus beyond problem-level in-context learning. Different from BoT's template matching, R^2-LLMs employs a hierarchical retrieval-augmented framework to dynamically integrate global strategies and local reasoning for enhanced problem solving.

3 Method

Overview of R^2-LLMs. In this section, we present a detailed overview of R^2-LLMs, with the specific process illustrated in Figure 1. In Section 3.1, we briefly introduce the preliminary of MCTS. When solving an initial mathematical question q, we efficiently extract the **conceptual unit** T (Section 3.2), which captures the core information of q and serves as the basis for retrieving a relevant **DLR reference set** Q_{ref} (Section 3.3). Leveraging Q_{ref}, we employ MCTS to conduct hierarchical augmented reasoning MCTS for q (Section 3.4).

3.1 MCTS Preliminary

MCTS is a heuristic search algorithm that incrementally builds a tree using stochastic simulations. Unlike Minimax, it selects actions via statistical sampling and refines estimates with more simulations. In this paper, we define the MCTS as $MCTS(\cdot)$. The MCTS algorithm consists of four main phases:

Selection. Starting from the root node, the algorithm recursively selects child nodes until it reaches a leaf node. A policy guides the selection process, often the Upper Confidence Bound for Trees (UCT), which balances exploration (trying less-visited nodes) and exploitation (favoring nodes with higher rewards).

$$\mathrm{UCT}(v) = \frac{Q(v)}{N(v)} + c\sqrt{\frac{\ln N(\mathrm{parent}(v))}{N(v)}}, \qquad (1)$$

where $Q(v)$ is the total reward of node v, $N(v)$ is the visit count of node v, and c is a constant controlling the balance between exploration and exploitation.

Expansion. If the selected leaf node is not terminal, the algorithm expands it by adding child nodes for possible actions, progressively exploring new parts of the search space.

Simulation (Rollout). A simulation starts from the expanded node, taking random or policy-driven actions until reaching a terminal state. This *Rollout* estimates the node's reward.

Backpropagation. The simulation results update node statistics (e.g., total reward, visit count) from the expanded node to the root, refining node quality estimates iteratively.

3.2 Conceptual Unit Extraction

Some studies (Zhang et al., 2025a; Yang et al., 2024b) indicate that highly relevant questions, along with their reasoning steps or solution templates related to the initial question, can enhance policy models' reasoning abilities and improve their problem-solving accuracy. However, mathematical questions often involve various types, logical conditions, and constraints, forming intricate logical structures. Relying solely on surface-level semantic information makes it challenging to directly determine the correlation between different problems. Therefore, it is essential to extract generalization representations from these questions, allowing for effective categorization and the identification of connections across different questions types. Doing so promotes deeper analysis and a more comprehensive understanding.

Inspired from previous works (Yang et al., 2024b; Wu et al., 2024a), we extract the generalization features of the initial question from three key perspectives: **problem types**, **key terms**, and **relevant solution strategies**. We collectively refer to this triplet as the conceptual unit, denoted by $T = \left(t_{\text{type}}, t_{\text{key}}, t_{\text{strategy}}\right)$, where t_{type} denotes the problem type, t_{key} represents the key terms within the problem, and t_{strategy} corresponds to the associated solution strategy. The inference factor T can be obtained as:

$$T = LLM(\beta_1(x)), \tag{2}$$

where $\beta_1(\cdot)$ denotes the meta prompt used for extracting the conceptual unit and x is original question. The detailed extraction process is provided in the Appendix B.2.

3.3 Deep Logical Retrieval

In this section, we propose deep logical retrieval (DLR) to assist the policy model in effective reasoning. Given an initial question q and its conceptual unit T, DLR aims to retrieve several questions with similar inference logic along with their corresponding reasoning steps to serve as references for the policy model. These similar questions and reasoning steps help the model better understand the reasoning path, thereby enhancing its reasoning capability and efficiency.

Given a set of reference questions, we use DeepSeek-70B (Guo et al., 2025) to generate a conceptual unit using Eq. 2 and further construct the candidate set $F_{cand} = \{(q_i, T_i, s_i)\}_{i=1}^{n}$, where s_i represents the solution steps for the i-th question. For a candidate set F_{cand} consisting of n questions, solution steps and their corresponding conceptual units T_i, we implemented a two-stage selection process—Preliminary Filtering followed by Refined Selection—to identify questions that exhibit deep logical relevance to the initial question q. We describe this two-stage retrieval process as follows:

Preliminary Filtering. In the preliminary filtering stage, we employ the BM25 (Robertson and Jones, 1976) algorithm to retrieve the most relevant questions from the candidate set F_{cand}, based on the given query q and its corresponding problem type t_{type}. Specifically, we construct a query pair (q, t_{type}) of initial questions and compute its similarity with subset of conceptual unit from the candidate set to construct $\{(q_i, t_{\text{type}})_i\}_{i=1}^{n}$ using BM25. The candidate questions are then ranked by their BM25 scores, and the top N most relevant ones are selected as the coarse-level selection set Q_{ref}^{coa}. The coarse-level set is constructed by filtering candidate questions based on the semantic similarity of the query q and its corresponding problem type t_{type}. The selected questions not only belong to the same category as the initial question but also share a similar knowledge foundation in the problem-solving process, ensuring greater accuracy and relevance for subsequent matching.

Refined Selection. After obtaining coarse-level set Q_{ref}^{coa}, we further perform fine-grained selection among the these questions. Specifically, we utilize $(t_{\text{key}}, t_{\text{strategy}})$ from the conceptual unit T of the initial question q and compare them with the set $\{(t_{\text{key}_i}, t_{\text{strategy}_i})\}_{i=1}^{N} \in Q_{\text{ref}}^{\text{coa}}$ to compute the semantic similarity score as:

$$e_i = E\left(t_{\text{key}_i}, t_{\text{strategy}_i}\right), e = E\left(t_{\text{key}}, t_{\text{strategy}}\right)) \tag{3}$$

$$S_{\text{ref},i} = Cosine(e_i, e), \tag{4}$$

Table 1: Comparative performance of reasoning methods across three benchmark datasets. The best results in each box are highlighted in bold for clarity.

Model	Method	Dataset			Average
		MATH500	**GSM8K**	**OlympiadBench**	
LLaMA-3.1-8B-Instruct	Zero-shot CoT (Kojima et al., 2022)	18.0	61.5	15.4	31.6
	Few-shot CoT (Wei et al., 2022b)	47.2	76.6	16.3	46.7
	CoT+SC@4 (Wang et al., 2023b)	44.2	80.5	16.5	47.1
	R^2-LLMs	**52.5**	**87.4**	**23.7**	**54.5**
Qwen2-7B-instruct	Zero-shot CoT (Kojima et al., 2022)	36.9	76.6	21.3	44.9
	Few-shot CoT (Wei et al., 2022b)	52.9	85.7	21.6	53.4
	CoT+SC@4 (Wang et al., 2023b)	55.6	87.7	21.7	55.0
	R^2-LLMs	**60.6**	**89.1**	**28.5**	**59.4**

where $E(\cdot)$ is an LM encoder, specifically a pre-trained SentenceBERT (Reimers, 2019). $S_{\text{ref},i}$ is used to measure the cosine similarity between the encoded representations of question keywords and question-solving strategies. By integrating the model's predicted solving strategies and the keywords extracted from the problem, it further assesses the relationship between the initial question q and identifies candidate questions $q_i \in Q_{\text{ref}}^{coa}$ that share similar problem-solving knowledge and reasoning approaches. Subsequently, we selected the M most relevant questions and compiled their corresponding solution processes and strategy into the DLR reference set $Q_{\text{ref}} = \{(q_i, s_i, t_{strategy_i}) \mid i \in \arg\max_{i} {}_M S_{\text{ref},i}, i \in [1, N]\}$.

3.4 Hierarchical augmented reasoning MCTS

After obtaining the DLR reference set Q_{ref} for the initial question q, we designed a hierarchical augmented reasoning MCTS approach to solve the problem. This method divides the reasoning process into two main components: Logical Reasoning Enhancement and Fine-grained Enhancement. Logical Reasoning Enhancement is tasked with refining the generation of high-quality reasoning steps, whereas Fine-grained Enhancement aims to deliver more accurate evaluations for every node within the MCTS, thereby boosting the precision and efficiency of the decision-making process as a whole. Next, we will delve into a comprehensive explanation of these two enhancement approaches.

Logical Reasoning Enhancement. In Logical Reasoning Enhancement, we utilize a logic-driven guidance mechanism to enable MCTS to produce high-quality and coherent solution paths. Specifically, during the MCTS reasoning process, we utilize Q_{ref} as a reference to steer the policy model $P(\cdot)$ in producing the subsequent node v_i at i-th state, leveraging the preceding set of node states $\mathrm{V}_{i-1} = \{v_1, \ldots, v_{i-1}\}$ using meta prompt $\beta_2(\cdot)$:

$$v_i = P\left(\beta_2\left((q, Q_{\text{ref}}), \mathrm{V}_{i-1}\right)\right). \tag{5}$$

Logical Reasoning Enhancement empowers the policy model to draw insights from logically analogous problems, thereby enhancing its ability to generate high-quality candidate solutions. By leveraging established logical patterns and structures, this approach guides the model in delivering more precise and contextually relevant answers.

Fine-grained Enhancement. At the i-th state, the policy model generates U candidate nodes $\mathrm{V}_i^{cand} = \{v_{i,j}\}_{j=1}^{U}$ based on the previous state node V_{i-1}. Among these candidates, the node with the highest Q value is selected as the final node for the current state, i.e., $v_i = \arg\max_{v \in V_i^{cand}} Q(v)$. It ensures that at each step, the most optimal successor node is chosen to efficiently construct the path.

Aligned with previous research (Zhang et al., 2025b), we use PRM to approximate each node values $Q(v_{i,j})$, where $v_{i,j} \in \mathrm{V}_i^{cand}$. To further enhance the accuracy of PRM's value estimation, we propose a fine-grained (FG) enhancement evaluation approach. We begin by using $\hat{\mathrm{V}}_{i,j} = (q, v_1, ..., v_i, v_{i,j})$, where $v_1, ..., v_i$ represent previous steps, as a query to perform BM25 retrieval within a fine-grained set F_{FG}, retrieving a selection enhancement set $Q_{\text{fin}_{i,j}}$ with size K that contains questions and reasoning steps relevant to q. Each reasoning step is assigned a relevance score $R(v_{i,j})$, which aids in evaluating the values of the node $v_{i,j}$:

$$R(v_{i,j}) = R_{\text{PRM}}\left(\beta_3(\hat{\mathrm{V}}_{i,j}, Q_{\text{fin}_{i,j}})\right), \tag{6}$$

where $R_{\text{PRM}}(\cdot)$ denotes the evaluation score generated by PRM, and $\beta_3(\cdot)$ represents the meta prompt that assists PRM in enhancing the scoring process.

4 Experiment

4.1 Experiment setting

Policy and Reward Models. We use three LLMs as policy models: LLaMA 3-8B, LLaMA 3.1-8B (Dubey et al., 2024), and Qwen 2-7B (Yang et al., 2024a). For PRM, we adopt Mistral-7B (Tang et al., 2024), trained on PRM800K[1]. Notably, we use a logit-based PRM approach rather than step-wise prompting.

Evaluation Benchmark. We test our method on three challenging open source mathematical benchmarks: MATH500 (Hendrycks et al., 2021), focused on high school-level competition mathematics; GSM8K (Cobbe et al., 2021), covering middle school to early high school level problems; and OlympiadBench-TO (He et al., 2024), designed for problems at the level of international mathematics olympiads.

Candidate Set Selection. For MATH500, we randomly select 2,500 questions and reasoning steps from PRM800K due to its rich and complex mathematical reasoning, which aligns well with their characteristics. For GSM8K, we chose the same number of questions from MAWPS (Koncel-Kedziorski et al., 2016) and MATHQA (Amini et al., 2019), as MAWPS offers various application questions and AQuA includes multiple choice questions based on reasoning, covering GSM8K's real-world math scenarios. For all tests, the candidate set size is 2500. DeepSeek-70B (Guo et al., 2025) generated all conceptual units within this set, including the reasoning steps for questions sourced from MAWPS and MATHQA. Regarding OlympiadBench-TO, we choose OpenThoughts [2].

Numbers of DLR Reference Set Q_{ref} and Selection Enhancement Set Q_{fin}. The DLR reference set maintains consistency in both question selection and candidate set. By default, the DLR reference set consists of 4 samples. Additionally, for the selection enhancement set, we selected samples from PRM800K, with a set size of 3. We show the sensitively analysis in Section 4.6.

Baseline. We primarily compare our approach against three traditional example-based ICL methods: zero-shot CoT (Kojima et al., 2022), few-shot CoT (Wei et al., 2022b), and SC+CoT (Wang et al., 2023b). For SC, we conduct four sampling iterations, referred to as CoT+SC@4. Additionally, we compare our approach with various tree-based structures, including ToT (Yao et al., 2023), RAP (Hao et al., 2023), ReST-MCTS* (Zhang et al., 2025b) and LiteSearch (Wang et al., 2024).

Evaluation metrics. We use accuracy (%) as the evaluation metric, where a solution is correct only if the model's final answer exactly matches the ground truth.. A solution is deemed correct only if the final reasoning process is fully aligned with the ground truth.

In addition, the more experiment can be seen in Appendix A.

4.2 Performance on Various Reasoning Benchmarks

Table 1 compares the reasoning performance of LLaMA-3.1-8B-Instruct and Qwen2-7B-instruct across three datasets (MATH, GSM8K, OlympiadBench) using four reasoning methods. R^2-LLMs achieves the highest scores, with Qwen2-7B-instruct outperforming LLaMA-3.1-8B-instruct in all settings. For example, on MATH, Qwen2-7B-Instruct reaches 60.6% with the proposed method, compared to LLaMA-3.1-8B-Instruct's 52.5%. Few-shot CoT and CoT+SC@4 show notable improvements over Zero-shot CoT; for instance, LLaMA-3.1-8B-Instruct's GSM8K score rises from 61.5% (Zero-shot CoT) to 80.5% (CoT+SC@4). Meanwhile, our method also shows a significant improvement on OlympiadBench.

4.3 Comparison with Other Tree-based Methods

To further assess the effectiveness of R^2-LLMs, we conducted comparative experiments on the MATH and GSM8K datasets against leading tree-based approaches. Specifically, we selected ToT, RAP, ReST-MCTS*, and LiteSearch as baselines and utilized the widely adopted Llama-3-8B-Instruct as the backbone model for inference, ensuring fairness and comparability in our evaluation.

Table 2 compares tree-based methods on GSM8K and MATH. R^2-LLMs achieve the best results—84.6% on GSM8K and 34.7% on MATH—outperforming all baselines. Notably, it surpasses LiteSearch by 2.3% and RAP by 4.1% on GSM8K, and leads ReST-MCTS by 3.3% on MATH, showing strong performance on both arithmetic and complex reasoning tasks. Compared with other methods, R^2-LLMs outperforms existing approaches by combining coarse-level retrieval

[1] The Mistral-7B PRM model is open-source: https://huggingface.co/peiyi9979/math-shepherd-mistral-7b-prm

[2] https://huggingface.co/open-thoughts/OpenThinker-7B

Table 2: Comparison of tree-based methods on GSM8K and MATH. **Bold** indicates the best performance.

Model	Method	Dataset	
		GSM8K	**MATH500**
LLaMA-3-8B-Instruct	ToT	69.0	13.6
	RAP	80.5	18.8
	ReST-MCTS*	-	31.4
	LiteSearch	82.3	-
	R^2-LLMs	**84.6**	**34.7**

for global strategy, which aids in handling rare problems, with fine-grained retrieval that enriches node evaluation using external steps, effectively mitigating sparse rewards in standard MCTS and other tree-based methods.

4.4 Domain Impact on the DLR Reference Set

When selecting the DLR reference set, we generally ensure it is in-domain with the test set. To assess the model's ability to generalize to out-of-domain scenarios, we evaluate its performance on datasets such as GSM8K and MATH. Furthermore, we apply the DLR reference set across different domains to test their adaptability. Table 3 presents the impact of the domain relationship between the DLR reference set and the test questions on accuracy. The experiment is based on the LLaMA-3.1-8B-Instruct model and is conducted on two mathematical problem datasets, GSM8K and MATH. In-domain means that the DLR reference set and the test questions come from the same domain, whereas out-of-domain indicates that they belong to different domains. The experimental results reveal that the model achieves its best performance on in-domain inference sets, where the domain of the questions aligns closely with the DLR reference set. For instance, on the GSM8K dataset, the model attains a score of 87.4%, demonstrating strong generalization capabilities within the same domain. However, when evaluated using out-of-domain DLR reference set, where the question domain differs significantly, R^2-LLMs's performance declines noticeably. For example, on the MATH dataset, the score drops to 43.5%, indicating a substantial performance gap compared to in-domain tasks. From the results mentioned above, it can be concluded that although the performance degrades across different domains, in most cases, it still helps the model to enhance the overall results.

4.5 Ablation Analysis

To assess the impact of each component on the performance of R^2-LLMs, we conducted a series of ablation experiments. The baseline MCTS method is compared against three variants: $\text{MCTS}_{\text{w/DLR}}$, which incorporates logical reasoning enhancements, $\text{MCTS}_{\text{w/FG}}$, which integrates fine-grained enhancement, and $\text{MCTS}_{\text{w/DLR+FG}}$ is equal to R^2-LLMs, which combines both improvements.

Table 4 presents the results of an ablation study evaluating the impact of different components in R^2-LLMs on the MATH and GSM8K datasets. We conduct experiments using two instruction-tuned language models, LLaMA-3.1-8B-Instruct and Qwen2-7B-Instruct, under different approaches. The results demonstrate that each individual enhancement contributes to performance gains, with the combination of both ($\text{MCTS}_{\text{w/DLR+FG}}$) yielding the best results across both datasets. Specifically, LLaMA-3.1-8B-Instruct achieves 52.5% on MATH and 87.4% on GSM8K, while Qwen2-7B-Instruct reaches 60.6% and 89.1%, respectively. In addition, for LLaMA-3.1-8B-Instruct, incorporating logical reasoning enhancements ($\text{MCTS}_{\text{w/DLR}}$) leads to an absolute gain of +3.7% on MATH (46.6% $\rightarrow$ 50.3%) and +4.1% on GSM8K (82.5% $\rightarrow$ 86.6%), while adding fine-grained enhancement ($\text{MCTS}_{\text{w/FG}}$) provides a smaller improvement of +0.9% on MATH (46.6% $\rightarrow$ 47.5%) and +0.4 on GSM8K (82.5% $\rightarrow$ 82.9%). When both components are combined ($\text{MCTS}_{\text{w/DLR+FG}}$), the performance further increases to 52.5% (+5.9%) on MATH and 87.4% (+4.9%) on GSM8K, demonstrating a synergistic effect. These findings highlight the effectiveness of our proposed method in improving mathematical reasoning performance.

4.6 Sensitively Analysis

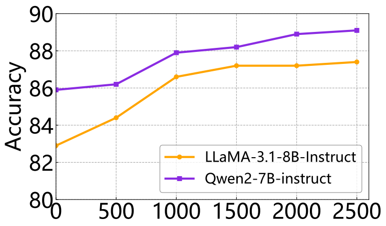
In this section, we examine how the sample size of the candidate set F_{cand}, the DLR reference set Q_{ref}, and the selection enhancement set Q_{fin} affects the results, using Qwen2-7B-Instruct and LLaMA-3.1-8B-Instruct as policy models on the GSM8K and MATH datasets.

Impact of Candidate Set Size. Figure 2a and Figure 2d illustrate the impact of candidate set size on accuracy for the GSM8K and MATH datasets, respectively. Both figures reveal a positive correlation between candidate set size and accuracy. Notably, accuracy increases sharply as the sample size grows from 1000 to 1500, but after reaching 2000, the overall improvement plateaus.

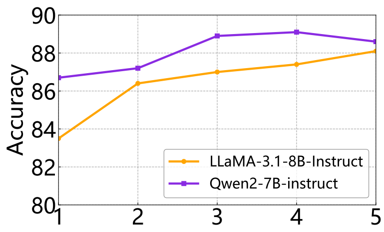
Impact of DLR Reference Set Size. Figure 2b and Figure 2e depict the effect of DLR reference set size on accuracy for the GSM8K and MATH datasets. Like the candidate set size, a larger DLR

Table 3: Performance evaluation of the impact of DLR reference sets Q_{ref} on the reasoning capabilities of R^2-LLMs, tested on the GSM8K and MATH datasets. **Bold** indicates the best performance.

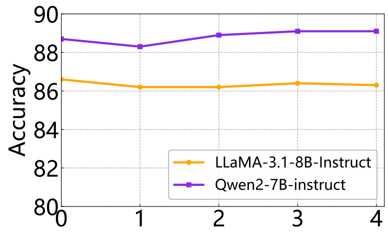
Model	Datasets	Deep Logical Retrieval (DLR) Reference Set			
		w/o	In Domain	Out of Domain	
LLaMA-3.1-8B-Instruct	**GSM8K**	-	**MAWPS & MATHQA**	**PRM800K**	**AMC 12**
		82.9	87.4	86.2	83.15
	MATH	-	**PRM800K**	**MATHQA**	**AMC12**
		47.5	52.5	43.5	49.9



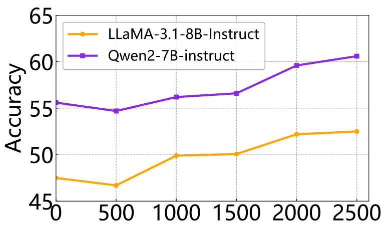
(a) Candidate Set (F_{cand}) Size N



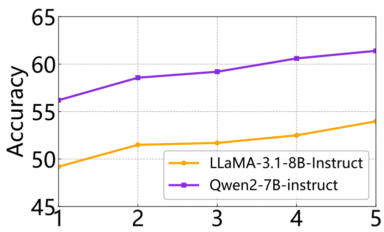
(b) DLR Reference Set (Q_{ref}) Size M



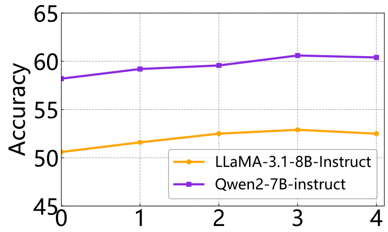
(c) Selection Enhancement Set (Q_{fin})



(d) Candidate Set (F_{cand}) Size N



(e) DLR Reference Set (Q_{ref}) Size M



(f) Selection Enhancement Set (Q_{fin}) Size K

Figure 2: Sensitivity Analysis on GSM8K (top row) and MATH500 (bottom row).

Table 4: Ablation analysis of R^2-LLMs. The best results in each box are highlighted in **bold** for clarity. DLR denotes Deep Logical Retrieval and FG denotes Fine-grained Enhancement.

Model	Method	Dataset	
		MATH	**GSM8K**
LLaMA-3.1-8B-Instruct	MCTS	46.6	82.5
	$\text{MCTS}_{\text{w/DLR}}$	50.3	86.6
	$\text{MCTS}_{\text{w/FG}}$	47.5	82.9
	$\text{MCTS}_{\text{w/DLR+FG}}$	**52.5**	**87.4**
Qwen2-7B-instruct	MCTS	53.7	85.9
	$\text{MCTS}_{\text{w/DLR}}$	58.2	88.7
	$\text{MCTS}_{\text{w/FG}}$	55.6	84.8
	$\text{MCTS}_{\text{w/DLR+FG}}$	**60.6**	**89.1**

reference set leads to a noticeable improvement in accuracy. When the size reaches 4, accuracy increases by 5% on MATH and 4.5% on GSM8K with LLaMA-3.1-8B-Instruct. This indicates that expanding the DLR reference set size can effectively improve the reasoning quality of MCTS, thereby enhancing accuracy.

Impact of Selection Enhancement Set Size. Figure 2c and Figure 2f present the influence of selection enhancement set size on the GSM8K and MATH datasets, respectively. While there is still a positive correlation between set size and accuracy, its impact is less pronounced compared to the effect of candidate set size. Specifically, for the MATH dataset, increasing the size to 3 results in only a modest 2.4% improvement in accuracy.

5 Conclusion

In this work, we presented R^2-LLMs, a hierarchical retrieval-augmented framework that enhances test-time scaling for LLMs by leveraging both Deep Logical Retrieval at the coarse level and Hierarchical Augmented Reasoning MCTS at the fine level. Our approach integrates external reference data to enrich in-context learning and employs a process reward model to refine candidate generation and decision-making. Empirical results, with improvements up to 16% on key benchmarks, validate the effectiveness of R^2-LLMs in tackling complex mathematical reasoning tasks.

6 Limitation

Our current evaluations have focused primarily on mathematical reasoning benchmarks, leaving its effectiveness in other domains—such as general knowledge, symbolic logic, and multimodal tasks—less explored. Besides, most experiments have been conducted using relatively modest models, and further investigation is needed to understand the performance and scalability of R^2-LLMs on larger, potentially closed-source models.

7 Potential Risks

Our work makes clear contributions by enhancing LLMs' reasoning abilities. It enables more accurate and trustworthy AI support in complex reasoning tasks such as education, scientific analysis, and decision-making. However, improved reasoning capabilities also pose risks—such as producing persuasive yet inaccurate outputs—especially when reasoning chains are poorly guided. Therefore, responsible and transparent use of such enhanced reasoning frameworks is essential to ensure positive societal impact.

References

Aida Amini, Saadia Gabriel, Peter Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. *arXiv preprint arXiv:1905.13319*.

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*.

Guoxin Chen, Minpeng Liao, Chengxi Li, and Kai Fan. 2024. Step-level value preference optimization for mathematical reasoning. *arXiv preprint arXiv:2406.10858*.

Sehyun Choi, Tianqing Fang, Zhaowei Wang, and Yangqiu Song. 2023. Kcts: knowledge-constrained tree search decoding with token-level hallucination detection. *arXiv preprint arXiv:2310.09044*.

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

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Tianyu Liu, et al. 2022. A survey on in-context learning. *arXiv preprint arXiv:2301.00234*.

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

Hugging Face. 2024. Scaling test-time compute: A key factor in large model inference. Accessed: 2025-02-14.

Edward A Feigenbaum, Julian Feldman, et al. 1963. *Computers and thought*, volume 37. New York McGraw-Hill.

Xidong Feng, Ziyu Wan, Muning Wen, Stephen Marcus McAleer, Ying Wen, Weinan Zhang, and Jun Wang. 2023. Alphazero-like tree-search can guide large language model decoding and training. *arXiv preprint arXiv:2309.17179*.

Charles R Fletcher. 1985. Understanding and solving arithmetic word problems: A computer simulation. *Behavior Research Methods, Instruments, & Computers*, 17(5):565–571.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. 2023. Critic: Large language models can self-correct with tool-interactive critiquing. *arXiv preprint arXiv:2305.11738*.

Xinyu Guan, Li Lyna Zhang, Yifei Liu, Ning Shang, Youran Sun, Yi Zhu, Fan Yang, and Mao Yang. 2025. rstar-math: Small llms can master math reasoning with self-evolved deep thinking. *arXiv preprint arXiv:2501.04519*.

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

Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. 2023. Reasoning with language model is planning with world model. *arXiv preprint arXiv:2305.14992*.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. 2024. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. *arXiv preprint arXiv:2402.14008*.

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

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Pei Ke, Bosi Wen, Andrew Feng, Xiao Liu, Xuanyu Lei, Jiale Cheng, Shengyuan Wang, Aohan Zeng, Yuxiao Dong, Hongning Wang, et al. 2024. Critiquellm: Towards an informative critique generation model for evaluation of large language model generation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13034–13054.

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.

Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. 2016. Mawps: A math word problem repository. In *Proceedings of the 2016 conference of the north american chapter of the association for computational linguistics: human language technologies*, pages 1152–1157.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. 2022. Solving quantitative reasoning problems with language models. *Advances in Neural Information Processing Systems*, 35:3843–3857.

Yifei Li, Zeqi Lin, Shizhuo Zhang, Qiang Fu, Bei Chen, Jian-Guang Lou, and Weizhu Chen. 2023. Making language models better reasoners with step-aware verifier. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5315–5333.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *arXiv preprint arXiv:2305.20050*.

Jiacheng Liu, Andrew Cohen, Ramakanth Pasunuru, Yejin Choi, Hannaneh Hajishirzi, and Asli Celikyilmaz. 2024. Don't throw away your value model! generating more preferable text with value-guided monte-carlo tree search decoding. In *First Conference on Language Modeling*.

Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, Jiao Sun, et al. 2024. Improve mathematical reasoning in language models by automated process supervision. *arXiv preprint arXiv:2406.06592*.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. 2024. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36.

OpenAI. 2024. Learning to reason with llms. Accessed: 2025-02-14.

N Reimers. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *EMNLP*.

Stephen E Robertson and K Sparck Jones. 1976. Relevance weighting of search terms. *Journal of the American Society for Information science*, 27(3):129–146.

Amrith Setlur, Chirag Nagpal, Adam Fisch, Xinyang Geng, Jacob Eisenstein, Rishabh Agarwal, Alekh Agarwal, Jonathan Berant, and Aviral Kumar. 2024. Rewarding progress: Scaling automated process verifiers for llm reasoning. *arXiv preprint arXiv:2410.08146*.

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

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*.

Zhengyang Tang, Xingxing Zhang, Benyou Wang, and Furu Wei. 2024. Mathscale: Scaling instruction tuning for mathematical reasoning. *arXiv preprint arXiv:2403.02884*.

Ante Wang, Linfeng Song, Ye Tian, Baolin Peng, Dian Yu, Haitao Mi, Jinsong Su, and Dong Yu. 2024. Litesearch: Efficacious tree search for llm. *arXiv preprint arXiv:2407.00320*.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Y.Wu, and Zhifang Sui. 2023a. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. *ArXiv*, abs/2312.08935.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023b. Self-consistency improves chain of thought reasoning in language models. *NeurIPS*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022a. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.

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

Jinyang Wu, Mingkuan Feng, Shuai Zhang, Feihu Che, Zengqi Wen, and Jianhua Tao. 2024a. Beyond examples: High-level automated reasoning paradigm in in-context learning via mcts. *arXiv preprint arXiv:2411.18478*.

Yangzhen Wu, Zhiqing Sun, Shanda Li, Sean Welleck, and Yiming Yang. 2024b. Inference scaling laws: An empirical analysis of compute-optimal inference for problem-solving with language models. *arXiv preprint arXiv:2408.00724*.

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

Ling Yang, Zhaochen Yu, Bin Cui, and Mengdi Wang. 2025. Reasonflux: Hierarchical llm reasoning via scaling thought templates.

Ling Yang, Zhaochen Yu, Tianjun Zhang, Shiyi Cao, Minkai Xu, Wentao Zhang, Joseph E Gonzalez, and Bin Cui. 2024b. Buffer of thoughts: Thought-augmented reasoning with large language models. *NeurIPS 2024*.

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

Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. 2023. Mammoth: Building math generalist models through hybrid instruction tuning. *arXiv preprint arXiv:2309.05653*.

Beichen Zhang, Yuhong Liu, Xiaoyi Dong, Yuhang Zang, Pan Zhang, Haodong Duan, Yuhang Cao, Dahua Lin, and Jiaqi Wang. 2025a. Booststep: Boosting mathematical capability of large language models via improved single-step reasoning. *arXiv preprint arXiv:2501.03226*.

Dan Zhang, Sining Zhoubian, Ziniu Hu, Yisong Yue, Yuxiao Dong, and Jie Tang. 2025b. Rest-mcts*: Llm self-training via process reward guided tree search. *Advances in Neural Information Processing Systems*, 37:64735–64772.

Di Zhang, Jianbo Wu, Jingdi Lei, Tong Che, Jiatong Li, Tong Xie, Xiaoshui Huang, Shufei Zhang, Marco Pavone, Yuqiang Li, et al. 2024. Llama-berry: Pairwise optimization for o1-like olympiad-level mathematical reasoning. *arXiv preprint arXiv:2410.02884*.

Shun Zhang, Zhenfang Chen, Yikang Shen, Mingyu Ding, Joshua B Tenenbaum, and Chuang Gan. 2023. Planning with large language models for code generation. *arXiv preprint arXiv:2303.05510*.

Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. 2023. Language agent tree search unifies reasoning acting and planning in language models. *arXiv preprint arXiv:2310.04406*.

Yuxiang Zhou, Jiazheng Li, Yanzheng Xiang, Hanqi Yan, Lin Gui, and Yulan He. 2024. The mystery of in-context learning: A comprehensive survey on interpretation and analysis. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 14365–14378.

Xinyu Zhu, Junjie Wang, Lin Zhang, Yuxiang Zhang, Ruyi Gan, Jiaxing Zhang, and Yujiu Yang. 2022. Solving math word problems via cooperative reasoning induced language models. *arXiv preprint arXiv:2210.16257*.

A Extra Experiments

A.1 Comparison with other TTS baselines

To ensure a fair comparison with other TTS-based methods, we select BoT (Liu et al., 2024) as the baseline. As shown in Table 5, we evaluate its performance on two popular base models (LLaMA-3.1-Instruct and Qwen-2.5-Instruct) across the GSM8K and MATH500 datasets, comparing it with the existing TTS-based method BoT. The results demonstrate that **R^2-LLMs** consistently outperforms BoT in all settings. Notably, the improvement is particularly significant on the more challenging MATH500 dataset (e.g., from 25.7 to 52.5, or from 34.5 to 60.6), indicating that our method not only generalizes well across different backbones but also significantly boosts the model's performance on complex mathematical reasoning tasks.

Table 5: Comparison with another TTS-based method (BoT). The best results in each box are highlighted in **bold** for clarity.

Model	Method	GSM8K	MATH500
LLaMA-3.1-Instruct	BoT	62.5	25.7
	R^2-LLMs	87.4	52.5
Qwen-2.5-Instruct	BoT	80.4	34.5
	R^2-LLMs	89.1	60.6

A.2 Time consumption analysis

Table 6: The computational overhead incurred by **R^2-LLMs**.

Method	MATH500	GSM8k
Plain MCTS	7.00h	3.20h
R^2-LLMs	7.35h	3.45h

To evaluate the computational efficiency of our method, we compare the runtime of **R^2-LLMs** with the baseline Plain MCTS across two benchmark datasets, as shown in Table 6. On the MATH500 dataset, **R^2-LLMs** completes the task in 7.35 hours using 4 A100 GPUs, compared to 7.00 hours required by the baseline. Similarly, on GSM8K, the runtime increases modestly from 3.20 to 3.45 hours. These results demonstrate that **R^2-LLMs** achieves significant performance gains with only a marginal increase in computational overhead—approximately 5% on MATH500 and 7.8% on GSM8K—validating the practicality and scalability of our method for real-world deployment.

A.3 Performance with different PRM model

Table 7: Performance comparision with different PRM models. The best results in each box are highlighted in **bold** for clarity.

Method	MATH500	GSM8k
MCTS w/ Mistral-7B	46.6	82.5
R^2-LLMs w/ Mistral-7B	52.5	87.4
MCTS w/Qwen2.5-7B Math PRM	47.9	84.1
R^2-LLMs w/Qwen2.5-7B Math PRM	**53.2**	**89.7**

To further validate the effectiveness of our approach, we compare **R^2-LLMs** with different state-of-the-art PRM (Policy Retrieval Module) backbones, as shown in Table 7. When using Mistral-7B as the PRM, **R^2-LLMs** achieves substantial improvements over the MCTS baseline, with scores rising from 46.6 to 52.5 on MATH500 and from 82.5 to 87.4 on GSM8K. Similarly, when adopting the more advanced Qwen2.5-7B Math PRM, our method further boosts performance, reaching 53.2 on MATH500 and 89.7 on GSM8K. These results confirm that **R^2-LLMs** consistently enhances performance across PRM choices, and benefits even more from stronger PRMs, highlighting the flexibility and scalability of our framework. Due to time constraints, we focus on evaluation using LLaMA 3.1 8B for fair and efficient comparison.

A.4 Ablation analysis on abstrct template

Table 8: Ablation analysis on abstract template using MATH500. The best results in each box are highlighted in **bold** for clarity.

Method	MATH500
Without anything	47.5
Only problem types	48.7
Only key terms	49.7
Only solution strategies	50.7
Problem types + key terms	49.7
Problem types + solution strategies	50.9
Key terms + solution strategies	52.2
Problem types + solution strategies + key terms (**R^2-LLMs**)	52.5

To investigate the effectiveness of each component in the abstract template, we conduct an ablation study on the MATH500 dataset using the LLaMA-3.1-8B Instruct model. As shown in Table 8, the abstract template consists of three elements: problem types, key terms, and relevant solution strategies. Removing all components results in the lowest accuracy of 47.5%. Adding only problem types slightly improves performance to 48.7%, while including only key terms or only solution strategies leads to further gains of 49.7% and 50.7%, respectively. Combining problem types with key terms does not yield additional benefits (49.7%), but combining problem types with solution strategies improves the score to 50.9%. Notably, the combination of key terms and solution strategies achieves a stronger result of 52.2%. The best performance of 52.5% is obtained when all three components are included, as used in **R^2-LLMs**, confirming that each part of the abstract template contributes incrementally to overall reasoning performance.

A.5 Results on larger policy model

Table 9: Performance using larger policy model (Qwen-2.5 14B) with MATH500 and GSM8K.

Method	MATH500	GSM8K
Plain MCTS	88.5	58.4
R^2-LLMs	91.6	62.6

To address the suggestion of evaluating our method on a larger language model, we conduct additional experiments using **Qwen-2.5 14B** as the policy model. As shown in Table 9, our method **R^2-LLMs** achieves strong performance improvements over the baseline Plain MCTS on both benchmarks. Specifically, on the challenging MATH500 dataset, accuracy increases from 88.5% to **91.6%**, and on GSM8K, from 58.4% to **62.6%**. These results confirm that **R^2-LLMs** consistently enhance performance even with larger-scale models, demonstrating its robustness and scalability across model sizes.

B Example Appendix

B.1 Example of DLR set samples

Initial questions: A 90° rotation around the origin in the counter-clockwise direction is applied to $7+2i$. What is the resulting complex number?

Sample 1

Related question 1: Let $z = 2+\sqrt{2}-(3+3\sqrt{2})i$, and let $c = 2-3i$. Let w be the result when z is rotated around c by $\frac{\pi}{4}$ counter-clockwise. Find w.

Problem type: Complex number operation.

Key words and relevant words: complex number, rotation, counter-clockwise, center of rotation, angle.

Problem solving strategy: The transformation involves translating the system so that the center of rotation aligns with the origin, applying a complex exponential rotation to achieve the desired angular displacement, and then translating back to the original coordinate system. This process ensures that the rotated point maintains its relative position to the center while undergoing the specified rotation. The final result is expressed in terms of its real and imaginary components to provide a complete representation in the complex plane.

Sample 2

Related question 2:

The function $f(z) = \dfrac{(-1+i\sqrt{3})z + (-2\sqrt{3}-18i)}{2}$ represents a rotation around some complex number c.

Find c.

Problem type: Complex number operation.

Key words and relevant words: function, rotation, complex number, transformation.

Problem solving strategy: The transformation follows a structured process, beginning with a translation to align the rotation center with the origin, followed by the application of a complex linear mapping that encodes both rotation and translation. The fixed point of this transformation, obtained by solving $f(c) = c$, determines the invariant center around which the system rotates. By decomposing this result into its real and imaginary components, a complete representation of the transformation in the complex plane is achieved.

B.2 Meta Prompt

Meta Prompt β_1

As an expert in solving mathematical problems, you excel at extracting key information from users' mathematical queries for analysis. You can skillfully transform the extracted information into a format that is suitable for handling the problem. If the problem can be generalized to a higher level to address multiple issues, you will provide further analysis and explanation in your next response.

Please categorize and extract the crucial information required to solve the problem from the user's input query. Combine these two elements to generate distilled information. Subsequently, pass this distilled information to the downstream meta planner based on the problem type. The problem type should belong to one of the six categories mentioned above, and the distilled information should include:

Extract the problem type in the given range and key words from the user's input, which will be handed over to the respective expert for task resolution, ensuring that all essential information required to solve the problem is provided. The objective of the problem and corresponding constraints. Propose a meta problem based on the issue to address the user's query, and handle more input and output variations. At the same time, provide similar terms based on the key words and relevant words.

Additionally, based on the extracted information, provide a strategy or possible solution for addressing the problem. Your task is to extract the key information, and you do not need to provide the final result in your response.Please follow the format below for extraction and stop responding after outputting the distilled information.

Problem type:

Key word and relevant words:

Your relevant abstract strategy for solving the problem:

Figure 3: Meta Prompt β_1

Meta Prompt β_2

Related Problem:

...

Distilled Results for the Related Problem:
Problem type: Complex function transformation : ⋯.
Key word and relevant words: ⋯
Your relevant strategy for solving the problem: ⋯
Existing Steps:
...

Based on the above steps, and related problem and their relevant strategy as reference, the possible current step-by-step solution is:

Figure 4: Meta Prompt β_2

Meta Prompt β_3

Related Problem: ⋯
Related Problem Existing Steps and corresponding score is:⋯
Our existing Steps and its score:
Based on the above steps, the possible each step-by-step solution score is:

Figure 5: Meta Prompt β_3

Figures 3, 4, and 5 illustrate three distinct meta prompts, each designed to assist the large model in a specific task: extracting conceptual units, enhancing DLR reasoning, and refining fine-grained details.

B.3 Preliminary Filtering & Refined Selection example

Initial Question

"A 90° rotation around the origin transforms the complex number (7+2i). What is the result?"

Step 1: Preliminary Filtering (BM25)

- **Input:** Query = (problem text, type="complex number rotation").
- **Candidate Questions Retained (Top 3 by BM25):**
  1. "Rotate (3+4i) by 180° around the origin."
  2. "Find the result of rotating (1+i) by 45° counter-clockwise."
  3. "Let (z=2+3i). Rotate (z) by 90° around (1+i)."
- **Rationale:** BM25 prioritizes questions with overlapping keywords (e.g., "rotate", "complex number") and matching problem types.

Step 2: Refined Selection (SentenceBERT)

- **Conceptual Unit of Initial Question:**
  - (T_{key}): ["origin", "counter-clockwise", "complex number"]
  - (T_{strategy}): "Apply rotation matrix to complex coordinates."
- **Scores ($S_{\textbf{ref},i}$):**
  - Candidate 1: 0.82 (high, shares "origin" and strategy).
  - Candidate 2: 0.45 (low, angle differs).
  - Candidate 3: 0.12 (discarded, rotation center mismatch).
- **Output (DLR Reference Set):** Includes Candidate 1's solution steps (e.g., "Multiply by $e^{i\pi}$" for 180° rotation).