

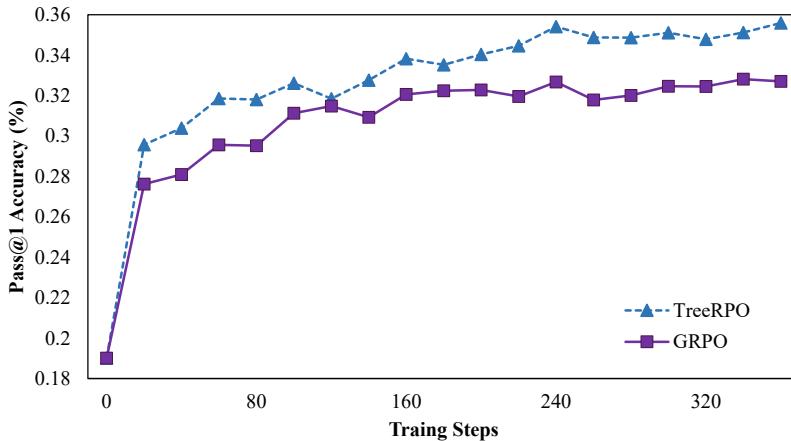
000 TREERPO: TREE RELATIVE POLICY OPTIMIZATION

001
002
003 **Anonymous authors**

004 Paper under double-blind review

005 006 007 ABSTRACT

008
009 Large Language Models (LLMs) have shown remarkable reasoning capabilities
010 through Reinforcement Learning with Verifiable Rewards (RLVR) methods. How-
011 ever, a key limitation of existing approaches is that rewards defined at the full tra-
012 jectory level provide insufficient guidance for optimizing the intermediate steps of
013 a reasoning process. To address this, we introduce **TREERPO**, a novel method
014 that estimates the mathematical expectations of rewards at various reasoning steps
015 using tree sampling. Unlike prior methods that rely on a separate step reward
016 model, TREERPO directly estimates these rewards through this sampling process.
017 Building on the group-relative reward training mechanism of GRPO, TREERPO
018 innovatively computes rewards based on step-level groups generated during tree
019 sampling. This advancement allows TREERPO to produce fine-grained and dense
020 reward signals, significantly enhancing the learning process and overall perfor-
021 mance of LLMs. Experimental results demonstrate that our TREERPO algorithm
022 substantially improves the average Pass@1 accuracy of Qwen-2.5-Math on test
023 benchmarks, increasing it from 19.0% to 35.5%. Furthermore, TREERPO signifi-
024 cantly outperforms GRPO by 2.9% in performance while simultaneously reducing
025 the average response length by 18.1%, showcasing its effectiveness and efficiency.



040
041 Figure 1: The average Pass@1 accuracy of TREERPO and GRPO with Qwen-2.5-Math-1.5b on
042 four mathematical benchmarks: MATH-500, OlympiadBench, Minerva, and AIME.

043 044 1 INTRODUCTION

045 Recent advancements in test-time scaling with reinforcement learning methods bring milestone
046 progress to Large Language Models (LLMs). Reasoning models such as OpenAI O1 (OpenAI,
047 2024), DeepSeek R1 (Guo et al., 2025), and QwQ (Qwen, 2024) have demonstrated significantly
048 superior performance in complex reasoning tasks. Reinforcement Learning with Verifiable Rewards
049 (RLVR) plays a pivotal role in this progress, which enhances the model’s performance by continu-
050 ously exploring reasoning paths toward correct answers on verifiable problems.

051 In the realm of LLM-RL integration for complex reasoning, existing approaches can be broadly
052 categorized into two paradigms: *reward model-based* methods (Ouyang et al., 2022; Shen et al.,

054 2025; Schulman et al., 2017) and *reward model-free* methods (Rafailov et al., 2023; Shao et al.,
 055 2024; Zeng et al., 2025; Luo et al., 2025b). Among reward model-based methods, reward models are
 056 typically divided into outcome reward models (ORMs; Cobbe et al. 2021; Yu et al. 2023) and process
 057 reward models (PRMs; Lightman et al. 2023; Wang et al. 2024; Lu et al. 2024a; Chen et al. 2025).
 058 ORMs provide a single scalar reward for the entire generation sequence, while PRMs offer step-
 059 wise evaluations of the reasoning path. The fine-grained, dense reward signals from PRMs generally
 060 yield superior RL performance compared to ORMs. However, acquiring high-quality training data
 061 for PRMs remains challenging, as accurately annotating the correctness of intermediate reasoning
 062 steps requires substantial domain expertise. This data scarcity significantly hinders the scalability of
 063 PRM-based approaches.

064 Recent breakthroughs in enhancing LLM reasoning capabilities, such as GRPO (Shao et al., 2024)
 065 and its variants (Yu et al., 2025; Yue et al., 2025), have adopted a reward model-free paradigm.
 066 These methods leverage verifiable reward functions trained on complex reasoning datasets, where
 067 rewards are determined by whether the model’s final output matches the ground-truth numerical
 068 answer or passes predefined unit tests in programming tasks. This approach achieves remarkable
 069 scalability by eliminating the need for human annotations or reward models. However, similar to
 070 ORMs, these rule-based methods only provide trajectory-level rewards, offering limited guidance for
 071 optimizing intermediate reasoning steps. Consequently, the question of how to deliver dense, fine-
 072 grained reward signals without relying on reward models presents an important research direction.

073 To address this challenge, we propose **TREERPO**, a novel approach that estimates the mathematical
 074 expectations of rewards at various reasoning steps through tree sampling. Unlike previous methods
 075 that require explicit step-level reward models, TREERPO employs a tree-based sampling mech-
 076 anism to approximate these expectations. Building upon GRPO’s group-relative reward training
 077 framework, TREERPO innovatively computes rewards based on step-level groups within the sam-
 078 pled tree structure. This design enables the generation of fine-grained, dense reward signals that
 079 guide the model’s reasoning process more effectively while maintaining the scalability advantages
 080 of verifiable reward functions. Through this approach, TREERPO achieves more efficient and ef-
 081 fective optimization of LLM reasoning capabilities.

082 To summarize, our main contributions are as follows:

- 083 • To the best of our knowledge, TREERPO is the first reward model-free method that pro-
 084 vides dense process reward signals through tree sampling and group relative reward com-
 085 putation, significantly enhancing the efficiency of RL-based reasoning optimization.
- 086 • Through extensive experimentation, TREERPO was found to significantly increase Qwen-
 087 2.5-Math-1.5B’s Pass@1 accuracy on various benchmarks from 19.0% to 35.5%, including
 088 a **2.9%** lead over GRPO.
- 089 • Detailed analysis demonstrates that TREERPO achieves higher accuracy and reduces to-
 090 ken consumption. Specifically, it cuts the average response length on test benchmarks by
 091 **18.1%** compared to GRPO, showcasing more efficient and precise reasoning.

093 2 RELATED WORKS

095 2.1 ELICITING COMPLEX REASONING ABILITY

098 Complex reasoning tasks (Hendrycks et al., 2021; He et al., 2024; Lewkowycz et al., 2022; Zeng
 099 et al., 2024; Yang et al., 2025; Xiang et al., 2025) such as mathematical problem solving are one of
 100 the most challenging tasks for LLMs. Various methods are proposed to elicit the reasoning ability
 101 of LLMs. These approaches can be divided into two groups:

- 102 1) *In-context learning*: These methods aim to improve the reasoning ability of LLMs by design-
 103 ing various prompting strategies and frameworks without updating the model parameters. Chain-
 104 of-thought (CoT; Wei et al. 2022) prompting shows that intermediate reasoning steps can greatly
 105 improve model performance. Subsequent research (Zhang et al., 2023; Yao et al., 2023; Bi et al.,
 106 2023; Yang et al., 2024b) has further enhanced CoT through various methods.
- 107 2) *Fine-tuning*: This line of approaches (Yang et al., 2022; Yu et al., 2024; Lu et al., 2024b; Huang
 108 et al., 2024; Tong et al., 2024) involve finetuning on extensive and high-quality datasets to improve
 109 reasoning capabilities. The core of these methods is to construct high-quality question-response

108 pairs with chain-of-thought reasoning processes. MetaMath (Yu et al., 2024) focuses on data aug-
 109mentation for both questions and answer texts. MathGenie (Lu et al., 2024b) collects a vast amount
 110of data through open-source language models. DART-Math (Tong et al., 2024) generates diverse
 111solutions with the difficulty-aware rejection sampling. Recent studies (Shao et al., 2024; Hu et al.,
 1122025; Zeng et al., 2025; Luo et al., 2025b; Yu et al., 2025; Yue et al., 2025) have explored reinforce-
 113ment learning in complex reasoning tasks and have acquired great achievements. Inspired by recent
 114successes in reinforcement learning for complex reasoning tasks, we propose TREERPO, an innova-
 115tive reinforcement learning method that leverages tree sampling to further enhance LLM reasoning
 116ability.

117 2.2 REINFORCEMENT LEARNING WITH LLMs

120 Reinforcement Learning from Human Feedback (RLHF; Ouyang et al. 2022) has been widely used
 121 in LLM alignments. Direct Preference Optimization (DPO; Rafailov et al. 2023) is further proposed
 122 to simplify the training pipeline of RLHF, which directly uses pair-wise preference data for model
 123 optimization. Recent studies (OpenAI, 2024; Guo et al., 2025; XAI, 2024; DeepMind, 2024; Qwen,
 124 2024; Team et al., 2025) have shown that reinforcement learning can significantly improve the rea-
 125 soning ability of models. This type of work can roughly be divided into two categories:

126 1) *Reward model-based*: There are two primary types of reward models: the Outcome Reward
 127 Model (ORM) and the Process Reward Model (PRM). Prior effort (Lightman et al., 2023) sug-
 128 gests that PRM outperforms ORM due to the fine-grained step-by-step reward signals. Math-
 129 Shepherd (Wang et al., 2024) trains a PRM by estimating the potential for a given reasoning step.
 130 However, training a reward model requires extensive, high-quality annotated data, especially for
 131 PRMs. This hinders the scaling of reward models in the field of complex reasoning.

132 2) *Reward model-free*: DPO is one of these methods, but it requires the elaborate construction of
 133 pairwise data for training. Step-DPO (Lai et al., 2024) constructs a pipeline to generate pair-wise
 134 step-level data and surpasses the performance of DPO. The other line of research (Shao et al., 2024;
 135 Hu et al., 2025; Zeng et al., 2025; Luo et al., 2025b) has shown that verification functions are ef-
 136 fective in improving the reasoning capabilities of LLMs. They avoid the need for reward models,
 137 offering a simple yet effective approach. The typical methods are GRPO (Shao et al., 2024) and its
 138 variants DAPO (Yu et al., 2025) and VAPO (Yue et al., 2025). However, rule-based reward is similar
 139 to ORM, providing trajectory-level reward signals rather than fine-grained process reward signals.
 140 Unlike existing efforts, TREERPO achieves fine-grained, dense reward signals without relying on
 141 a separate reward model. TREERPO can offer a more scalable solution for obtaining dense reward
 142 signals in complex reasoning tasks.

143 3 TREERPO: METHODOLOGY

144 In this section, we elaborate on the proposed TREERPO. First, we present tree sampling in Sec-
 145 tion 3.1, which is designed to construct step step-level group to enhance long-chain reasoning abil-
 146 ities with GRPO. Next, in Section 3.2, we introduced the pruning strategy to improve the sampling
 147 and training efficiency in TREERPO. In Section 3.3, we discuss the numerical influence of standar-
 148 dized binary rewards and continuous rewards on advantage computation and propose a new advantage
 149 computation method for continuous rewards.

151 3.1 TREE SAMPLING

153 While GRPO has been proven to be effective and suitable for scaling in complex reasoning tasks
 154 with verifiable reward, it only provides the trajectory-level reward by evaluating the final answer of
 155 the generated sequences. Instead, to provide step-level reward estimation without using a reward
 156 model, we designed tree sampling.

158 Given an input question q , the language model generates an N -ary tree through iterative sampling,
 159 governed by the following constraints:

160 161 • **Branching Factor:** At each decoding step, the model samples N candidate continuations,
 162 expanding N new branches from the current node.

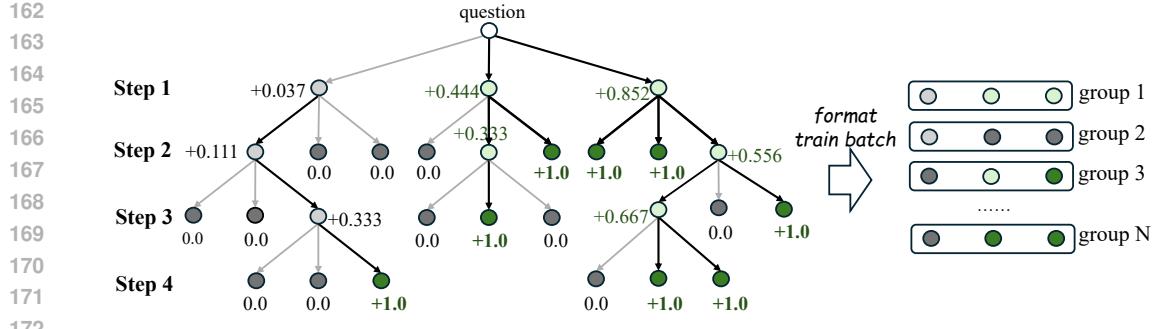


Figure 2: The sampling process of our TREERPO. TREERPO starts from the question, sampling N nodes at each step until generation is completed or the maximum depth D is reached. Then, a verifiable reward function is used to evaluate all leaf nodes and then back-propagates the rewards to their parent nodes, thereby obtaining intermediate step rewards, which achieves process reward signaling. We traverse each node and aggregate all children steps of a node into a group to compute advantages, which are finally formatted into the training batch.

- **Depth Limit:** The tree expansion terminates when any path reaches the maximum depth D , ensuring tractability.
- **Step Segmentation:** We directly divide the steps according to the token length. Each step produces at most L_{step} tokens per branch. Generation halts for a branch if a stop token is generated, or the branch violates reaches depth limit. A more precise step division method is our future work.

As shown in Figure 2, the tree’s reward computation follows a bottom-up recursive expectation scheme, where:

- **Leaf Evaluation:** For each leaf node v_{leaf} , the verification function ϕ takes the *entire path* $P = [v_{\text{root}}, \dots, v_{\text{leaf}}]$ as input and computes the reward:

$$r_{\text{leaf}} = \phi(P) = \phi([v_{\text{root}}, \dots, v_{\text{leaf}}]),$$

- **Parent Propagation:** Non-leaf nodes aggregate rewards from their children:

$$r_{\text{node}} = \mathbb{E}_{c \in \text{Children}(v_{\text{node}})} [r_c].$$

This propagates bottom-up, weighting all viable completion paths.

In conclusion, our tree sampling framework estimates the reward of each step as its potential to deduce the correct final answer.

3.2 DATA PRUNING

Similar to the Dynamic Sampling strategy of DAPO, we filter out the samples to keep all data samples in the training batch with effective gradients.

In the data construction pipeline of TREERPO, a **group** \mathcal{G} is formally defined as the set of child nodes c_1, \dots, c_n originating from a common parent node p , as illustrated in Figure 2. Adopting a strategy analogous to the dynamic sampling approach in DAPO, we perform group-level filtering based on reward distribution characteristics.

$$\Delta R_{\mathcal{G}} = \max_{c_i \in \mathcal{G}} R(c_i) - \min_{c_j \in \mathcal{G}} R(c_j) \quad (1)$$

where $R(c_i)$ denotes the reward associated with child node c_i . We introduce a variance threshold τ such that a group \mathcal{G} is included in the training batch \mathcal{B} if and only if:

$$\mathcal{G} \in \mathcal{B} \iff \Delta R_{\mathcal{G}} > \tau \quad (2)$$

216 The threshold τ operates as a hyperparameter controlling the trade-off between sample diversity and
 217 learning signal strength in the batch construction process.
 218

219 This data selection criterion ensures all samples in the batch with effective gradients and improves
 220 the efficiency of the training process.
 221

222 3.3 ADVANTAGE COMPUTATION

223 In the vanilla GRPO framework, the advantage estimation is derived by normalizing binary rewards:
 224

$$225 \quad \hat{A}_{i,t} = \frac{r_i - \text{mean}(\{R_i\}_{i=1}^G)}{\text{std}(\{R_i\}_{i=1}^G)}. \quad (3)$$

228 However, when applied to continuous rewards, this approach introduces significant bias. For in-
 229 stance, two reward sequences, $\mathbf{R}_1 = [0, 0, 1, 1]$ and $\mathbf{R}_2 = [0.49, 0.49, 0.51, 0.51]$, produce identical
 230 normalized advantages $[-1, -1, 1, 1]$, despite their distinct reward distributions. While \mathbf{R}_1 exhibits
 231 a clear bimodal separation, \mathbf{R}_2 contains only minor variations (a maximal difference of 0.02). This
 232 indicates that standard normalization fails to properly scale advantages for continuous rewards, lead-
 233 ing to misleading policy updates.
 234

235 To mitigate this bias, we propose an alternative advantage computation that preserves the statistical
 236 properties of binary reward normalization while accommodating continuous rewards. Instead of
 237 computing the empirical variance from \mathbf{R} , we define the normalization factor as $\sigma = \mu(1 - \mu)$,
 238 where μ is the mean reward. This formulation maintains consistency with the variance of Bernoulli-
 239 distributed rewards ($\text{Var}[R] = \mu(1 - \mu)$) while generalizing to continuous settings.
 240

241 For a given reward sequence $\mathbf{R} = [R_1, R_2, \dots, R_n]$, the advantage is computed as:
 242

$$243 \quad \mu = \frac{1}{n} \sum_{i=1}^n R_i \quad \sigma = \mu(1 - \mu) \quad (4)$$

$$244 \quad A_i = \frac{R_i - \mu}{\sigma}$$

245 By fixing the variance term σ to $\mu(1 - \mu)$, we ensure that advantage values remain interpretable and
 246 stable, avoiding the overamplification of small differences in continuous rewards. This approach
 247 bridges the gap between binary and continuous reward normalization while maintaining the original
 248 scaling behavior of GRPO.
 249

250 3.4 OBJECTIVE OF TREERPO

251 We adopt the clipped objective of GRPO, together with a directly imposed KL penalty term: Ad-
 252 ditionally, the KL-regularization between current policy π_θ and the reference policy π_{ref} is directly
 253 added to the loss function:
 254

$$255 \quad \mathcal{J}_{\text{TreeRPO}}(\theta) = \mathbb{E}_{(q \sim \mathcal{D}, \{o_i\}_{i=1}^G \sim \pi_{\theta_{\text{old}}}(q)} \left[\frac{1}{G} \sum_{i=1}^G \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \left(\min \left(r_{i,t}(\theta) \hat{A}_{i,t}, \text{clip} \left(r_{i,t}(\theta), 1 - \varepsilon, 1 + \varepsilon \right) \hat{A}_{i,t} \right) - \beta D_{\text{KL}}(\pi_\theta || \pi_{\text{ref}}) \right) \right], \quad (5)$$

256 where
 257

$$258 \quad r_{i,t}(\theta) = \frac{\pi_\theta(o_{i,t} | q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t} | q, o_{i,<t})}. \quad (6)$$

259 4 EXPERIMENTS

260 **Datasets.** We conduct the evaluation of our experiments using 4 widely used mathematical rea-
 261 soning benchmarks: MATH-500 (Lightman et al., 2023), OlympiadBench (He et al., 2024), Minver-
 262 vaMath (Lewkowycz et al., 2022), and AIME24. Among them, Math-500 are 500 items screened
 263 out from the original MATH test split. The subset consists of 500 representative problems, and the
 264

270 evaluation produces similar results to the full-set evaluation. In the training scenario, we use the
 271 training split of MATH dataset, which contains 7.5k high-quality training samples. In the future,
 272 we will extend the experiment to the DeepScaler (Luo et al., 2025b) training data, which is a more
 273 challenging dataset for mathematical reasoning.
 274

275 **Parameter Setting.** Our experiments are based on Qwen2.5-Math series language models (Yang
 276 et al., 2024a). In the evaluation procedure, we set the temperature as 0.6 to sample 8 candidate re-
 277 sponses for each question. In the reinforcement learning training procedure, we set the temperature
 278 as 0.6 and roll out 8 responses for each question. The learning rate is 1e-6 for both GRPO and
 279 TREERPO. The coefficients for KL divergence and entropy loss are $\beta = 0.001$ and $\alpha = -0.001$,
 280 respectively. For GRPO, the training batch size is 128 and the mini-batch size is 64. For our
 281 TREERPO, the training batch size is 128. Since the training data size of each step of TREERPO is
 282 floating, the size of our mini-batch is obtained as half of the training data size. By default, the max-
 283 imum prompt length is 512, and the maximum response length is 1152 for GRPO. For TREERPO,
 284 the maximum prompt length is 512, the maximum step length L_{step} is 384, the maximum depth D
 285 of tree sampling is set as 3, and the N -ary is set as 8. For better efficiency, we set the data pruning
 286 coefficient τ to 0.1 as described in Sec. 3.2.
 287

288 **Implementation Details.** We follow the rllm (Luo et al., 2025a;b) framework which is derived
 289 from the verl (Sheng et al., 2024) pipeline. Both rllm and verl integrate the vilm (Kwon et al., 2023)
 290 framework for efficient inference of models. All of our experiments are conducted on A800 GPUs.
 291 At present, the LLM of our experiment is the Qwen2.5-Math series. Due to the limitations of time
 292 and computation resources, we have only reported the 1.5b model. We plan to conduct experiments
 293 on 7b and 32b as soon as possible
 294

295 **Metrics.** We use the same verification function in rllm to evaluate the performance of LLMs.
 296 Compared with other repositories, the reward function implemented by rllm is more complete and
 297 systematic. For the test results, the accuracy rate we report is **pass@1(avg@8)** performance for all
 298 tested benchmarks.
 299

300 Table 1: Overall performance of *Pass@1 (Avg@16)* performance of Qwen2.5-Math series.
 301

Method	AIME24	MATH500	Olympiad	Minerva	Macro Accuracy
<i>Qwen2.5-Math-1.5B as the Base Model</i>					
GRPO Baseline	13.8	67.9	28.5	20.5	32.7
TreeRPO	16.8 (↑+3.0)	70.7 (↑+2.8)	30.9 (↑+2.6)	24.0 (↑+3.5)	35.6 (↑+2.9)
<i>Qwen2.5-Math-7B as the Base Model</i>					
GRPO Baseline	26.7	74.3	34.7	27.1	40.7
TreeRPO	26.7	75.5 (↑+1.2)	35.4 (↑+0.7)	28.1 (↑+1.0)	41.4 (↑+0.7)

310
 311 4.1 MAIN RESULTS
 312

313 We show the performance comparison of GRPO baseline and our TreeRPO on Qwen2.5-Math-
 314 1.5/7B in four selected benchmarks: AIME24, MATH-500, Olympiad Benchmarks, and Minerva
 315 Math. As illustrated in Table 1, for Qwen2.5-Math-1.5B, the Macro Accuracy has improved by
 316 2.9%. Furthermore, we consider that the reason why the improvement and repetition of Qwen2.5-
 317 Math-7B is not as significant as that of Qwen2.5-Math-1.5B lies in the fact that the MATH training
 318 data for Qwen2.5-Math-7B is too simple, resulting in the improvement of the algorithm not being
 319 significantly reflected. In general, our **TreeRPO** has achieved a consistency improvement compared
 320 to GRPO baseline.
 321

322 **TREERPO demonstrates significant performance improvements.** We conduct TREERPO and
 323 GRPO on Qwen2.5-Math-1.5b model with the training split of the MATH dataset, and conduct the
 324 evaluation on four selected benchmarks: Math-500, MinervaMath, OlympiadBench, and AIME. As

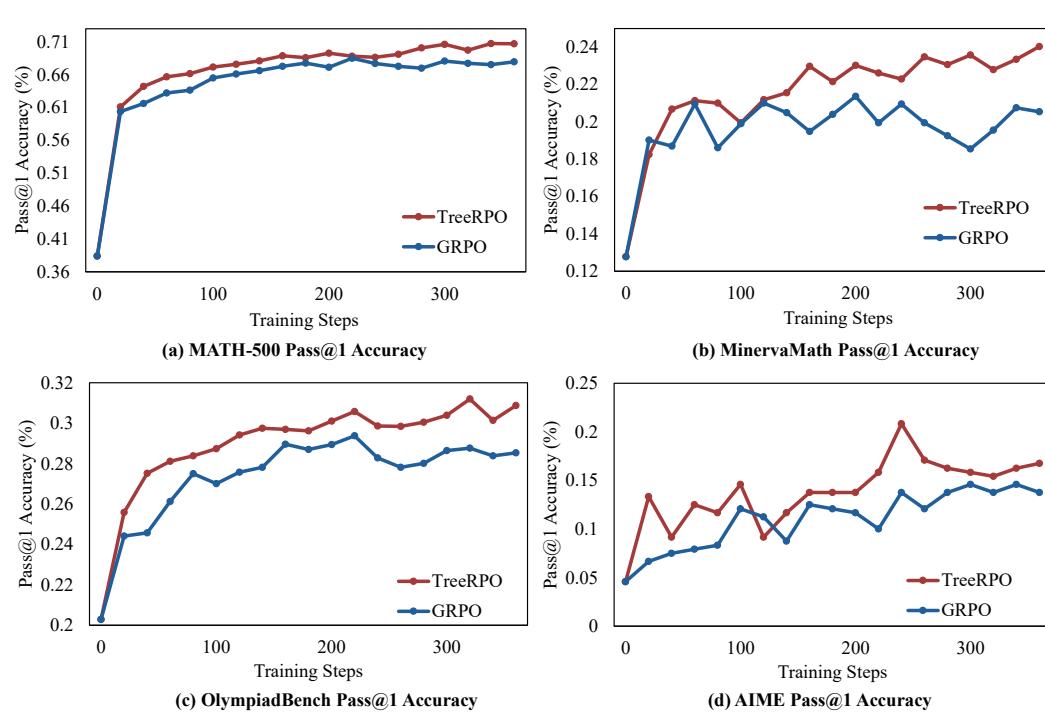


Figure 3: Performance comparison of our TRERPO and GRPO on the four selected benchmarks: Math-500, MinervaMath, OlympiadBench, and AIME. The experiments are conducted with Qwen2.5-Math-1.5b, an LLM pretrained with a large amount of mathematical corpus.

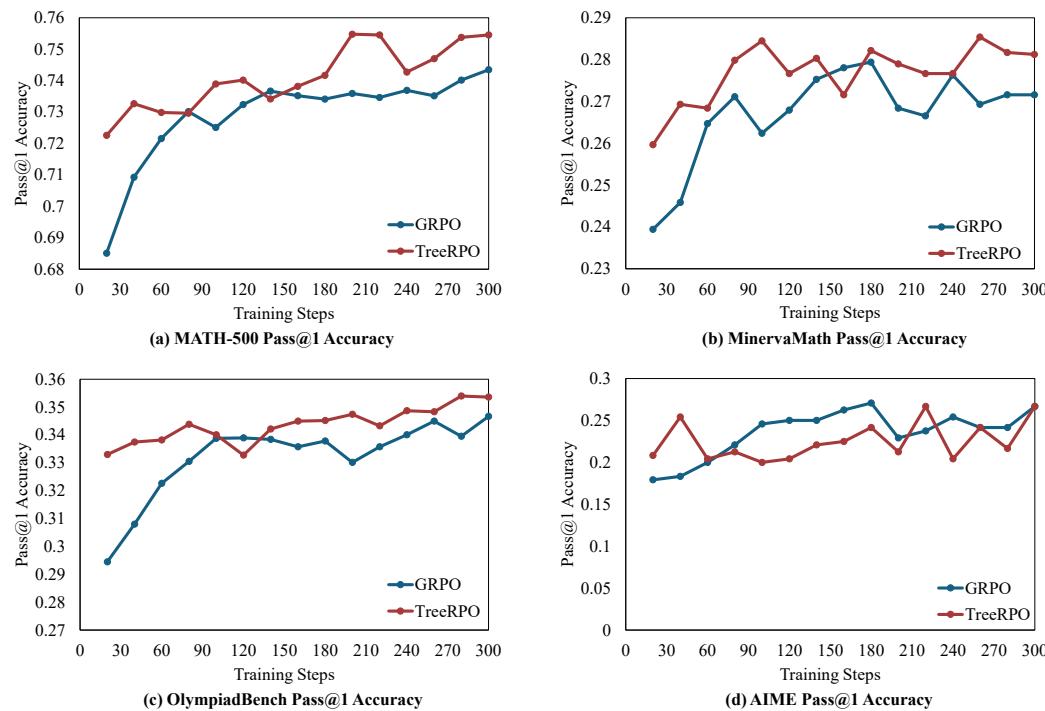
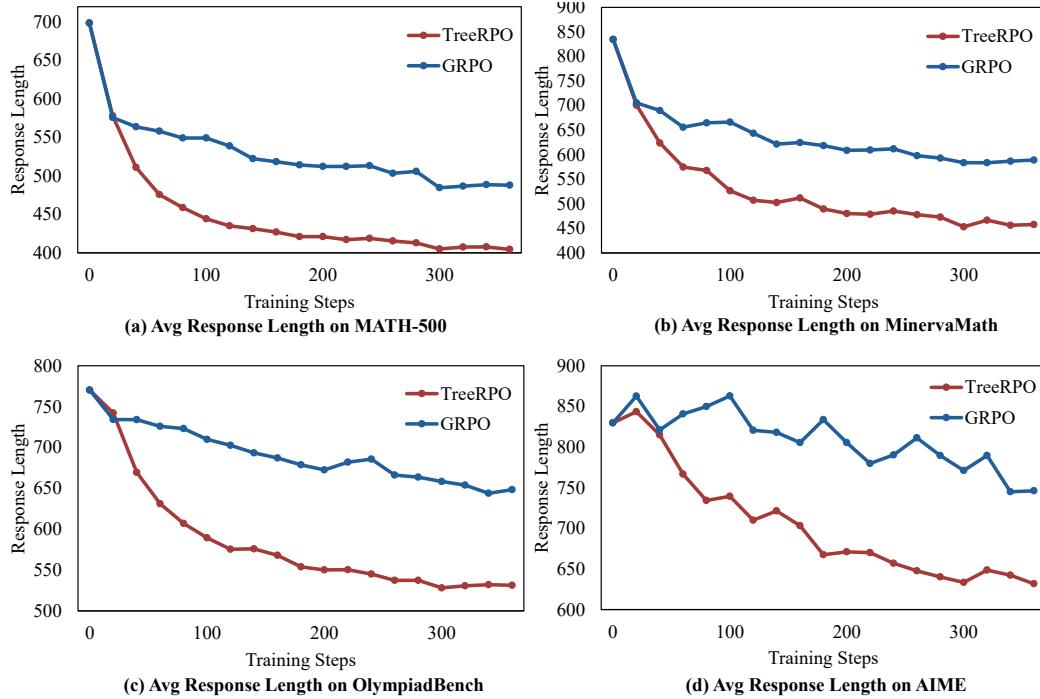


Figure 4: Performance comparison of our TRERPO and GRPO on the four selected benchmarks: Math-500, MinervaMath, OlympiadBench, and AIME. The experiments are conducted with Qwen2.5-Math-1.5b.

378 shown in Figure 3, our TREERPO outperform GRPO on all of the tested benchmarks. We further
 379 show the dynmaic results of Qwen2.5-Math-7B in Figure 4. After training 300 steps for Qwen2.5-
 380 Math-1.5B, our TREERPO outperforms GRPO by 2.7% on MATH-500, 3.5% on MinervaMath,
 381 2.4% on OlympiadBench, and 3.0% on AIME, respectively. As illustrated in Figure 1, TREERPO
 382 outperforms the overall performance of GRPO by **2.9%**. In conclusion, TREERPO has demon-
 383 strated consistent superiority on multiple benchmarks.



409 Figure 5: Response Length comparison of our TREERPO and GRPO on the four selected bench-
 410 marks: Math-500, MinervaMath, OlympiadBench, and AIME. The experiments is conducted with
 411 Qwen2.5-Math-1.5b

415 **TREERPO demonstrates efficiency advantage in token usage.** We conduct TREERPO and
 416 GRPO on the Qwen2.5-Math-1.5b model with the training split of the MATH dataset, and compute
 417 the average response length on four selected benchmarks: Math-500, MinervaMath, Olympiad-
 418 Bench, and AIME. As illustrated in Figure 5, compared to GRPO, our TREERPO achieves a 17.1%
 419 reduction in token usage on MATH, 22.3% on MinervaMath, 18.0% on OlympiadBench, and 15.3%
 420 on AIME. On average, TREERPO demonstrates a **18.1%** decrease in token usage across the four
 421 benchmarks compared to GRPO, showcasing its superior efficiency. TreeRPO not only demonstrates
 422 an advantage in token efficiency on Qwen2.5-Math-1.5B, but also shows an efficiency advantage on
 423 Qwen2.5-Math-7B, with an average token length that is also shorter than the GRPO baseline. We
 424 show the response case of a simple question in Figure 7. It can be seen that in this simple case,
 425 TREERPO’s response is more concise

427 **The performance of TREERPO under different hyperparameters.** In the experiments, we con-
 428 duct experimental analyses using different batch sizes, and the results are shown in Figure 6. For
 429 GRPO and TREERPO, the batch size $bsz = 16/128$ has very little influence on the final perfor-
 430 mance. Our TREERPO significantly outperforms GRPO in both two Settings. This fully demon-
 431 strates that our TreeRPO algorithm significantly outperforms the GRPO-baseline across different
 432 hyperparameters.

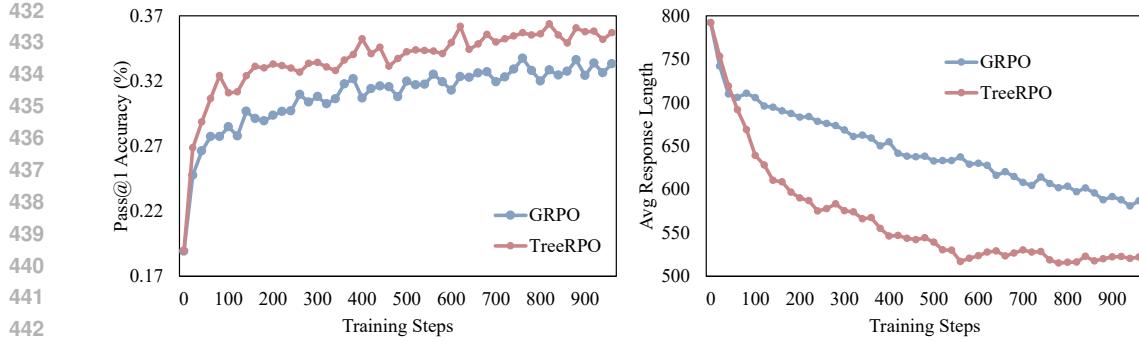


Figure 6: Comparison of TREERPO and GRPO with $bsz = 16$. The pass@1 accuracy and the response length are calculated by taking the average on the four tested benchmarks.

447 **Question:** Solve $\frac{1}{x-5} > 0$. Enter your answer using interval notation.

449 **TreeRPO:**

450 To solve the inequality $\frac{1}{x-5} > 0$, we need to
451 determine when the expression $\frac{1}{x-5}$ is
452 positive. The fraction $\frac{1}{x-5}$ is positive when
453 the denominator $x - 5$ is positive. Therefore,
454 we need to solve the inequality $x - 5 > 0$.
455

456 Adding 5 to both sides of the inequality, we
457 get:

$$x > 5$$

459 So, the solution to the inequality $\frac{1}{x-5} > 0$ is
460 $(5, +\infty)$.

461 The final answer in interval notation is:
462 $\boxed{(5, +\infty)}$.

449 **GRPO:**

450 To solve the inequality $\frac{1}{x-5} > 0$, we need to
451 determine the values of x for which the
452 expression $\frac{1}{x-5}$ is positive. The expression
453 $\frac{1}{x-5}$ is positive when the denominator $x - 5$ is
454 positive. Therefore, we need to solve the
455 inequality $x - 5 > 0$.

456 Let's solve this step-by-step:

457 1. Start with the inequality:

$$x - 5 > 0$$

458 2. Add 5 to both sides of the inequality:

$$x > 5$$

459 So, the solution to the inequality $\frac{1}{x-5} > 0$ is
460 $x > 5$. In interval notation, this is written as
461 $(5, +\infty)$.

462 The final answer is: $\boxed{(5, +\infty)}$.

468 Figure 7: Responses of TREERPO and GRPO of a simple question.

469

5 CONCLUSION

472 In this paper, we propose TREERPO, which conducts tree sampling to construct step-level groups
473 based on vanilla GRPO. TREERPO obtains the reward of the current step by estimating the reward
474 of the subsequent sampling paths of the current step. This is a method that can obtain dense re-
475 ward signals without the need for process reward models (PRMs). The experimental results show
476 that TREERPO demonstrates both effectiveness and efficiency. In the future, we will continuously
477 improve the algorithm based on the current version and expand the scale of LLM training.

478

6 REPRODUCIBILITY STATEMENT

481 To ensure the reproducibility of our research, we have meticulously assembled a comprehensive
482 reproducibility package as part of our supplementary materials. This package is designed to enable
483 the seamless replication of all experiments detailed in our paper. It encompasses anonymized source
484 code that implements the proposed model and training procedures. Additionally, we have included
485 precise configuration files and scripts that specify all hyperparameters and the training commands
486 necessary to reproduce our results.

486 REFERENCES
487488 Zhen Bi, Ningyu Zhang, Yinuo Jiang, Shumin Deng, Guozhou Zheng, and Huajun Chen. When do
489 program-of-thoughts work for reasoning? *arXiv preprint arXiv:2308.15452*, 2023.490 Jiaqi Chen, Bang Zhang, Ruotian Ma, Peisong Wang, Xiaodan Liang, Zhaopeng Tu, Xiaolong Li,
491 and Kwan-Yee K. Wong. Spc: Evolving self-play critic via adversarial games for llm reasoning,
492 2025. URL <https://arxiv.org/abs/2504.19162>.493
494 Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser,
495 Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to
496 solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.497
498 Google DeepMind. Gemini 2.0 flash thinking, 2024. URL <https://deepmind.google/technologies/gemini/flash-thinking/>.500 Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu,
501 Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms
502 via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.503
504 Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han,
505 Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. OlympiadBench:
506 A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific
507 problems. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual
508 Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3828–3850, Bangkok, Thailand, August 2024. Association for Computational Linguistics.
509 doi: 10.18653/v1/2024.acl-long.211. URL <https://aclanthology.org/2024.acl-long.211>.510
511 Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song,
512 and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *NeurIPS*,
513 2021.514
515 Jingcheng Hu, Yinmin Zhang, Qi Han, Dixin Jiang, Xiangyu Zhang, and Heung-Yeung Shum.
516 Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base
517 model, 2025. URL <https://arxiv.org/abs/2503.24290>.518
519 Dong Huang, Guangtao Zeng, Jianbo Dai, Meng Luo, Han Weng, Yuhao Qing, Heming Cui, Zhi-
520 jiang Guo, and Jie M Zhang. Effi-code: Unleashing code efficiency in language models. *arXiv
521 preprint arXiv:2410.10209*, 2024.522
523 Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E.
524 Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model
525 serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating
526 Systems Principles*, 2023.527
528 Xin Lai, Zhuotao Tian, Yukang Chen, Senqiao Yang, Xiangru Peng, and Jiaya Jia. Step-dpo: Step-
529 wise preference optimization for long-chain reasoning of llms, 2024. URL <https://arxiv.org/abs/2406.18629>.530
531 Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ra-
532 masesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam
533 Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with lan-
534 guage models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Ad-
535 vances in Neural Information Processing Systems*, volume 35, pp. 3843–3857. Curran Associates,
536 Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/18abbeef8cfe9203fdf9053c9c4fe191-Paper-Conference.pdf.537
538 Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan
539 Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let’s verify step by step. *arXiv preprint
arXiv:2305.20050*, 2023.

540 Jianqiao Lu, Zhiyang Dou, Hongru Wang, Zeyu Cao, Jianbo Dai, Yunlong Feng, and Zhiqiang
 541 Guo. Autopsv: Automated process-supervised verifier. In Amir Globersons, Lester Mackey,
 542 Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.),
 543 *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Infor-*
 544 *mation Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15,*
 545 *2024, 2024a.* URL http://papers.nips.cc/paper_files/paper/2024/hash/9246aa822579d9b29a140ecdac36ad60-Abstract-Conference.html.

546 Zimu Lu, Aojun Zhou, Houxing Ren, Ke Wang, Weikang Shi, Junting Pan, Mingjie Zhan, and
 547 Hongsheng Li. MathGenie: Generating synthetic data with question back-translation for
 548 enhancing mathematical reasoning of LLMs. In Lun-Wei Ku, Andre Martins, and Vivek Sriku-
 549 mar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Lin-*
 550 *guistics (Volume 1: Long Papers)*, pp. 2732–2747, Bangkok, Thailand, August 2024b. Asso-
 551 ciation for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.151. URL <https://aclanthology.org/2024.acl-long.151/>.

552 Michael Luo, Sijun Tan, Roy Huang, Ameen Patel, Alpay Ariyak, Qingyang Wu, Xiaoxiang Shi,
 553 Rachel Xin, Colin Cai, Maurice Weber, Ce Zhang, Li Erran Li, Raluca Ada Popa, and Ion Stoica.
 554 Deepcoder: A fully open-source 14b coder at o3-mini level, 2025a. Notion Blog.

555 Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y. Tang, Manan Roongta, Colin Cai,
 556 Jeffrey Luo, Li Erran Li, Raluca Ada Popa, and Ion Stoica. Deepscaler: Surpassing o1-preview
 557 with a 1.5b model by scaling rl, 2025b. Notion Blog.

558 OpenAI. Learning to reason with llms, 2024. URL <https://openai.com/index/learning-to-reason-with-llms/>.

559 Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong
 560 Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to fol-
 561 low instructions with human feedback. *Advances in neural information processing systems*, 35:
 562 27730–27744, 2022.

563 Qwen. Qwq-32b: Embracing the power of reinforcement learning, 2024. URL <https://qwenlm.github.io/blog/qwq-32b/>.

564 Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea
 565 Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances*
 566 *in Neural Information Processing Systems*, 36:53728–53741, 2023.

567 John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy
 568 optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

569 Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang,
 570 Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathe-
 571 matical reasoning in open language models, 2024. URL <https://arxiv.org/abs/2402.03300>.

572 Wei Shen, Guanlin Liu, Zheng Wu, Ruofei Zhu, Qingping Yang, Chao Xin, Yu Yue, and Lin Yan.
 573 Exploring data scaling trends and effects in reinforcement learning from human feedback. *arXiv*
 574 *preprint arXiv:2503.22230*, 2025.

575 Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng,
 576 Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint*
 577 *arXiv: 2409.19256*, 2024.

578 Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun
 579 Xiao, Chenzhuang Du, Chonghua Liao, et al. Kimi k1. 5: Scaling reinforcement learning with
 580 llms. *arXiv preprint arXiv:2501.12599*, 2025.

581 Yuxuan Tong, Xiwen Zhang, Rui Wang, Ruidong Wu, and Junxian He. DART-math: Difficulty-
 582 aware rejection tuning for mathematical problem-solving. In *The Thirty-eighth Annual Confer-*
 583 *ence on Neural Information Processing Systems*, 2024. URL <https://openreview.net/forum?id=zLU21oQjD5>.

594 Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhi-
 595 fang Sui. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations. In
 596 Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meet-
 597 ing of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9426–9439,
 598 Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/
 599 2024.acl-long.510. URL <https://aclanthology.org/2024.acl-long.510/>.

600 Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen,
 601 Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen
 602 Yu, Gao Huang, and Junyang Lin. Beyond the 80/20 rule: High-entropy minority tokens drive
 603 effective reinforcement learning for llm reasoning, 2025. URL <https://arxiv.org/abs/2506.01939>.

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

607 XAI. Grok 3 beta — the age of reasoning agents, 2024. URL <https://x.ai/news/grok-3>.

608 Kun Xiang, Heng Li, Terry Jingchen Zhang, Yinya Huang, Zirong Liu, Peixin Qu, Jixi He, Jiaqi
 609 Chen, Yu-Jie Yuan, Jianhua Han, Hang Xu, Hanhui Li, Mrinmaya Sachan, and Xiaodan Liang.
 610 Seephys: Does seeing help thinking? – benchmarking vision-based physics reasoning, 2025.
 611 URL <https://arxiv.org/abs/2505.19099>.

612 An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu,
 613 Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu,
 614 Xingzhang Ren, and Zhenru Zhang. Qwen2.5-math technical report: Toward mathematical ex-
 615 pert model via self-improvement, 2024a. URL <https://arxiv.org/abs/2409.12122>.

616 Zhicheng Yang, Jinghui Qin, Jiaqi Chen, Liang Lin, and Xiaodan Liang. LogicSolver: To-
 617 wards interpretable math word problem solving with logical prompt-enhanced learning. In Yoav
 618 Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Findings of the Association for Compu-
 619 tational Linguistics: EMNLP 2022*, pp. 1–13, Abu Dhabi, United Arab Emirates, December
 620 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-emnlp.1. URL
 621 <https://aclanthology.org/2022.findings-emnlp.1/>.

622 Zhicheng Yang, Yinya Huang, Jing Xiong, Liang Feng, Xiaodan Liang, Yiwei Wang, and Jing Tang.
 623 AlignedCoT: Prompting large language models via native-speaking demonstrations. In Yaser
 624 Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Com-
 625 putational Linguistics: EMNLP 2024*, pp. 2857–2896, Miami, Florida, USA, November 2024b.
 626 Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.163. URL
 627 <https://aclanthology.org/2024.findings-emnlp.163/>.

628 Zhicheng Yang, Yiwei Wang, Yinya Huang, Zhijiang Guo, Wei Shi, Xiongwei Han, Liang Feng,
 629 Linqi Song, Xiaodan Liang, and Jing Tang. Optibench meets resocratic: Measure and improve
 630 LLMs for optimization modeling. In *The Thirteenth International Conference on Learning Rep-
 631 resentations*, 2025. URL <https://openreview.net/forum?id=fsDZwS49uY>.

632 Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik
 633 Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In
 634 A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in
 635 Neural Information Processing Systems*, volume 36, pp. 11809–11822. Curran Associates, Inc.,
 636 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/271db9922b8d1f4dd7aaef84ed5ac703-Paper-Conference.pdf.

637 Fei Yu, Anningzhe Gao, and Benyou Wang. Outcome-supervised verifiers for planning in mathe-
 638 matical reasoning. *arXiv preprint arXiv:2311.09724*, 2023.

639 Longhui Yu, Weisen Jiang, Han Shi, Jincheng YU, Zhengying Liu, Yu Zhang, James Kwok, Zhen-
 640 guo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions
 641 for large language models. In *The Twelfth International Conference on Learning Representations*,
 642 2024. URL <https://openreview.net/forum?id=N8N0hgNDrt>.

648 Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai,
 649 Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guang-
 650 ming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu,
 651 Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao
 652 Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingx-
 653 uan Wang. Dapo: An open-source llm reinforcement learning system at scale, 2025. URL
 654 <https://arxiv.org/abs/2503.14476>.

655
 656 Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Keming Lu, Chuanqi Tan, Chang Zhou,
 657 and Jingren Zhou. Scaling relationship on learning mathematical reasoning with large language
 658 models, 2023.

659
 660 Yu Yue, Yufeng Yuan, Qiying Yu, Xiaochen Zuo, Ruofei Zhu, Wenyuan Xu, Jiaze Chen, Chengyi
 661 Wang, TianTian Fan, Zhengyin Du, Xiangpeng Wei, Xiangyu Yu, Gaohong Liu, Juncai Liu,
 662 Lingjun Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Chi Zhang, Mofan Zhang, Wang Zhang, Hang
 663 Zhu, Ru Zhang, Xin Liu, Mingxuan Wang, Yonghui Wu, and Lin Yan. Vapo: Efficient and
 664 reliable reinforcement learning for advanced reasoning tasks, 2025. URL <https://arxiv.org/abs/2504.05118>.

665
 666 Weiha Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. Simplerl-
 667 zoo: Investigating and taming zero reinforcement learning for open base models in the wild, 2025.
 668 URL <https://arxiv.org/abs/2503.18892>.

669
 670 Zhongshen Zeng, Yinhong Liu, Yingjia Wan, Jingyao Li, Pengguang Chen, Jianbo Dai, Yuxuan
 671 Yao, Rongwu Xu, Zehan Qi, Wanru Zhao, Linling Shen, Jianqiao Lu, Haochen Tan, Yukang
 672 Chen, Hao Zhang, Zhan Shi, Bailin Wang, Zhijiang Guo, and Jiaya Jia. Mr-ben: A meta-
 673 reasoning benchmark for evaluating system-2 thinking in llms. In Amir Globersons, Lester
 674 Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang
 675 (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural
 676 Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 -
 677 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/d81cb1f4dc6e13aeb45553f80b3d6837-Abstract-Conference.html.

678
 679 Yifan Zhang, Jingqin Yang, Yang Yuan, and Andrew Chi-Chih Yao. Cumulative reasoning with
 680 large language models. *arXiv preprint arXiv:2308.04371*, 2023.

683 APPENDIX

685 A THE USE OF LLMs

687
 688 In the preparation of this paper, large language models (LLMs), specifically DeepSeek-V3.1 and
 689 Gemini 2.5, were used solely for the purpose of polishing the writing. The LLM was employed
 690 after the core intellectual content—including the central ideas, theoretical formulations, algorithm
 691 designs, experimental setups, and result analyses—had been fully developed by the authors. The
 692 model’s assistance was limited to rephrasing sentences for improved clarity, fluency, and conciseness.
 693 All prompts provided to the LLM contained only the authors’ original text and instructions for
 694 grammatical or stylistic improvement.

695 B FUTURE WORK AND LIMITATIONS

697
 698 **Remove Redundant Steps.** Yuan et al. (2023) uses Rejection Sampling to collect correct reason-
 699 ing paths for training LLMs. They find that the sampled redundant responses degrade the per-
 700 formance of LLMs. We consider that this phenomenon may also exist in RL. In vanilla GRPO, each
 701 response is treated equally, so responses with high similarity are repeatedly trained, which may cause
 performance disturbances. We believe that eliminating redundant rollouts can enhance performance
 while improving training efficiency through pruning.

702 **Precise Step Segmentation.** The step division of generated sequences in this article is implemented based on a specific token length. Give priority to exploring more precise step division methods.

703

704

705 • One solution to be implemented is to add the **step special token** and train the language model to segment different steps by itself.

706 • Sampling at the tokens where branch paths are more likely to be generated (Wang et al., 2025).

707

708

709 We believe that more precise step cutting will provide more accurate fine-grained reward signals and 710 further enhance the model’s performance.

711 **Scaling on Larger Model Sizes.** Due to the limitations of time and GPU resources, our experiment 712 can only report the 1.5b model for the time being. The experimental results of larger-sized 713 models, such as 7b and 32b, will be updated in the future.

714 **Engineering Efficiency Optimization of Tree Sampling.** Tree sampling is time-consuming, and 715 the tree sampling strategy implemented in this paper is not optimized from the perspective of the 716 KV cache. We believe that the engineering optimization of tree sampling will significantly improve 717 the efficiency of the training procedure.

718

719

720

721

722

723

724

725

726

727

728

729

730

731

732

733

734

735

736

737

738

739

740

741

742

743

744

745

746

747

748

749

750

751

752

753

754

755