

Demystifying Long Chain-of-Thought Reasoning in LLMs

Edward Yeo^{*1} Yuxuan Tong^{*2} Morry Niu¹ Graham Neubig³ Xiang Yue^{*3}

Abstract

Scaling inference compute enhances reasoning in large language models (LLMs), with long chains-of-thought (CoTs) enabling strategies like backtracking and error correction. Reinforcement learning (RL) has emerged as a crucial method for developing these capabilities, yet the conditions under which long CoTs emerge remain unclear, and RL training requires careful design choices. In this study, we systematically investigate the mechanics of long CoT reasoning, identifying the key factors that enable models to generate long CoT trajectories. Through extensive supervised fine-tuning (SFT) and RL experiments, we present four main findings: (1) While SFT is not strictly necessary, it simplifies training and improves efficiency; (2) Reasoning capabilities tend to emerge with increased training compute, but their development is not guaranteed, making reward shaping crucial for stabilizing CoT length growth; (3) Scaling verifiable reward signals is critical for RL. We find that leveraging noisy, web-extracted solutions with filtering mechanisms shows strong potential, particularly for out-of-distribution (OOD) tasks such as STEM reasoning; and (4) Core abilities like error correction are inherently

present in base models, but incentivizing these skills effectively for complex tasks via RL demands significant compute, and measuring their emergence requires a nuanced approach. These insights provide practical guidance for optimizing training strategies to enhance long CoT reasoning in LLMs. Our code is available at: <https://github.com/eddycmu/demystify-long-cot>.

1. Introduction

Large language models (LLMs) (Brown et al., 2020; Touvron et al., 2023; Anthropic, 2023; OpenAI, 2023) have demonstrated remarkable reasoning abilities in domains like mathematics (Cobbe et al., 2021) and programming (Chen et al., 2021). A key technique for enabling reasoning abilities in LLMs is chain-of-thought (CoT) prompting (Wei et al., 2022), which guides models to generate intermediate reasoning steps before arriving at a final answer.

Despite these advancements, LLMs still struggle with highly complex reasoning tasks, such as mathematical competitions (Hendrycks et al., 2021), PhD-level scientific QA (Rein et al., 2024), and software engineering (Jimenez et al., 2024), even with CoT. Recently, OpenAI’s o1 models (OpenAI, 2024) have demonstrated significant breakthroughs in these tasks. A key distinguishing feature of these models is their ability to scale up inference compute with long CoTs, which include strategies such as recognizing and correcting mistakes, breaking down difficult steps, and iterating on alternative approaches, leading to substantially longer and more

^{*}Project Lead. ¹IN.AI ²Tsinghua University. Work started when interning at CMU. ³Carnegie Mellon University. Correspondence to: Xiang Yue <xyue2@andrew.cmu.edu>.

Demystifying Long Chain-of-Thought Reasoning in LLMs

Edward Yeo^{*1} Yuxuan Tong^{*2} Morry Niu¹ Graham Neubig³ Xiang Yue^{*3}

Abstract

Scaling inference compute enhances reasoning in large language models (LLMs), with long chains-of-thought (CoTs) enabling strategies like backtracking and error correction. Reinforcement learning (RL) has emerged as a crucial method for developing these capabilities, yet the conditions under which long CoTs emerge remain unclear, and RL training requires careful design choices. In this study, we systematically investigate the mechanics of long CoT reasoning, identifying the key factors that enable models to generate long CoT trajectories. Through extensive supervised fine-tuning (SFT) and RL experiments, we present four main findings: (1) While SFT is not strictly necessary, it simplifies training and improves efficiency; (2) Reasoning capabilities tend to emerge with increased training compute, but their development is not guaranteed, making reward shaping crucial for stabilizing CoT length growth; (3) Scaling verifiable reward signals is critical for RL. We find that leveraging noisy, web-extracted solutions with filtering mechanisms shows strong potential, particularly for out-of-distribution (OOD) tasks such as STEM reasoning; and (4) Core abilities like error correction are inherently present in base models, but incentivizing these skills effectively for complex tasks via RL demands significant compute, and measuring their emergence requires a nuanced approach. These insights provide practical guidance for optimizing training strategies to enhance long CoT reasoning in LLMs. Our code is available at: <https://github.com/eddycmu/demystify-long-cot>.

present in base models, but incentivizing these skills effectively for complex tasks via RL demands significant compute, and measuring their emergence requires a nuanced approach. These insights provide practical guidance for optimizing training strategies to enhance long CoT reasoning in LLMs. Our code is available at: <https://github.com/eddycmu/demystify-long-cot>.

1. Introduction

Large language models (LLMs) (Brown et al., 2020; Touvron et al., 2023; Anthropic, 2023; OpenAI, 2023) have demonstrated remarkable reasoning abilities in domains like mathematics (Cobbe et al., 2021) and programming (Chen et al., 2021). A key technique for enabling reasoning abilities in LLMs is chain-of-thought (CoT) prompting (Wei et al., 2022), which guides models to generate intermediate reasoning steps before arriving at a final answer. Despite these advancements, LLMs still struggle with highly complex reasoning tasks, such as mathematical competitions (Hendrycks et al., 2021), PhD-level scientific QA (Rein et al., 2024), and software engineering (Jimenez et al., 2024), even with CoT. Recently, OpenAI’s o1 models (OpenAI, 2024) have demonstrated significant breakthroughs in these tasks. A key distinguishing feature of these models is their ability to scale up inference compute with long CoTs, which include strategies such as recognizing and correcting mistakes, breaking down difficult steps, and iterating on alternative approaches, leading to substantially longer and more

^{*}Project Lead. ¹IN.AI ²NTU ³Carnegie Mellon University. Work started when interning at CMU. Correspondence to: Xiang Yue <xyue2@andrew.cmu.edu>.

structured reasoning processes.

Several efforts have attempted to replicate the performance of o1 models by training LLMs to generate long CoTs (Qwen Team, 2024b; DeepSeek-AI, 2025; Kimi Team, 2025; Pan et al., 2025; Zeng et al., 2025). Most of these approaches rely on verifiable rewards, such as accuracy based on ground-truth answers, which helps to avoid reward hacking in reinforcement learning (RL) at scale. However, a comprehensive understanding of how models learn and generate long CoTs remains limited. In this work, we systematically investigate the underlying mechanics of long CoT generation. Specifically, we explore:

1) *Supervised fine-tuning (SFT) for long CoTs* – the most direct way to enable long CoT reasoning. We analyze its scaling behavior and impact on RL, finding that long CoT SFT allows models to reach higher performance and also facilitates easier RL improvements than short CoT.

2) *Challenges in RL-driven CoT scaling* – we observe that RL does not always stably extend CoT length and complexity. To address this, we introduce a cosine length-scaling reward with a repetition penalty, which stabilizes CoT growth while encouraging emergent reasoning behaviors such as branching and backtracking.

3) *Scaling up verifiable signals for long CoT RL* – Verifiable reward signals are essential for stabilizing long CoT RL. However, scaling them up remains challenging due to the limited availability of high-quality, verifiable data. To address this, we explore the use of data containing noisy, web-extracted solutions (Yue et al., 2024b). While these “silver” supervision signals introduce uncertainty, we find that, with an appropriate mixture in SFT and filtration in RL, they show promise, especially in out-of-distribution (OOD) reasoning scenarios such as STEM problem-solving.

4) *Origins of Long CoT Abilities and RL Challenges* – Core skills like branching and error validation are inherently present in base models, but effective RL-driven

incentivization demands careful designs. We examine RL incentives on long CoT generation, trace reasoning patterns in pre-training data, and discuss nuances in measuring their emergence.

2. Problem Formulation

In this section, we define the notation, followed by an overview of SFT and RL methods for eliciting long CoTs.

Research Aim

Our goal is to *demystify long chain-of-thought reasoning* in LLMs. Through systematic analysis and ablations, we extract key insights and offer practical strategies to enhance and stabilize its performance.

2.1. Notation

Let x be a query, and let y be the output sequence. We consider a LLM parameterized by θ , which defines a conditional distribution over output tokens: $(y_t | x; y_{1:t-1})$:

We denote by $\text{CoT}(y) \subseteq y$ the tokens in the generated output that constitute the *chain-of-thought*, which is often a reasoning trace or explanatory sequence. The final answer can be a separate set of tokens or simply the last part of y .

In this work, we use the term *long chain-of-thought (long CoT)* to describe an extended sequence of reasoning tokens that not only exhibits a larger-than-usual token length but also demonstrates more sophisticated behaviors such as:

1) Branching and Backtracking: The model systematically explores multiple paths (branching) and reverts to earlier points if a particular path proves wrong (backtracking).

2) Error Validation and Correction: The model detects inconsistencies or mistakes in its intermediate steps and takes corrective actions to restore coherence and

Research Aim

Our goal is to *demystify long chain-of-thought reasoning* in LLMs. Through systematic analysis and ablations, we extract key insights and offer practical strategies to enhance and stabilize its performance.

2.1. Notation

[illegible]

1) ($\check{Z} \cdot CoT_n$) \cong $SFT - \dots CoT^{\dots}$
 $: i j_2 \frac{1}{2} i \neq \frac{1}{2} j : i j_2 + v i UL : i$

$$\begin{aligned} \mathbb{V} \cap \mathbb{P} \cap \mathbb{Q} &= \mathbb{V} \cap \mathbb{P} \cap \mathbb{Q} \\ \mathbb{V} \cap \mathbb{P} \cap \mathbb{Q} &= \mathbb{V} \cap \mathbb{P} \cap \mathbb{Q} \end{aligned}$$

2) $RLq \dots CoTi U - i \in \frac{1}{2} Q RL V ;$
 $/ 3 s O i U CoT , \cdot | E B' : t \in \frac{1}{2} L Z$
 $i \in \frac{1}{2} i \in z + * & \in \frac{1}{2} i Z \frac{1}{2} Y & \cdot | i$
 $UV \pm i \in \frac{1}{2} \pm / Ei \in \frac{1}{2}^o t \dots L :$
 $i \in z + CoT , z \cdot$

3) i U (Z • CoTRL „ i $\mathbb{F}_{2^2} \times \mathbb{F}_{2^2}$ - i $\mathbb{F}_{2^2} \times \mathbb{F}_{2^2}$ $\pm \mathbb{F}_{2^2} \times \mathbb{F}_{2^2}$ Z BŠ • CoTRL i $\mathbb{F}_{2^2} \times \mathbb{F}_{2^2}$ 6 1 Z $\mathbb{F}_{2^2} \times \mathbb{F}_{2^2}$ i $\mathbb{F}_{2^2} \times \mathbb{F}_{2^2}$ „ i $\mathbb{F}_{2^2} \times \mathbb{F}_{2^2}$ P i U f i $\mathbb{F}_{2^2} \times \mathbb{F}_{2^2}$ w ' : t $\mathbb{F}_{2^2} \times \mathbb{F}_{2^2}$ $\mathbb{F}_{2^2} \times \mathbb{F}_{2^2}$ i $\mathbb{F}_{2^2} \times \mathbb{F}_{2^2}$ + (+ j t $\mathbb{F}_{2^2} \times \mathbb{F}_{2^2}$ $\mathbb{F}_{2^2} \times \mathbb{F}_{2^2}$ $\mathbb{F}_{2^2} \times \mathbb{F}_{2^2}$ $\mathbb{F}_{2^2} \times \mathbb{F}_{2^2}$ pn (Yue e al., 2024b) } 6 i $\mathbb{F}_{2^2} \times \mathbb{F}_{2^2}$ $\mathbb{F}_{2^2} \times \mathbb{F}_{2^2}$ e t n Š' F i $\mathbb{F}_{2^2} \times \mathbb{F}_{2^2}$ i $\mathbb{F}_{2^2} \times \mathbb{F}_{2^2}$ SFT - S ÷ $\mathbb{F}_{2^2} \times \mathbb{F}_{2^2}$ RL - i $\mathbb{F}_{2^2} \times \mathbb{F}_{2^2}$ i $\mathbb{F}_{2^2} \times \mathbb{F}_{2^2}$ OOD ' : o , STEM i $\mathbb{F}_{2^2} \times \mathbb{F}_{2^2}$ $\mathbb{F}_{2^2} \times \mathbb{F}_{2^2}$ $\mathbb{F}_{2^2} \times \mathbb{F}_{2^2}$ $\mathbb{F}_{2^2} \times \mathbb{F}_{2^2}$

[illegible]

2. Problem Formulation

(, , - i ę HŠI & ÷ 6 , i Ğ Z i
• i ę ½ Ğ 0Ts „ SFTCERLi Ğ ½

$$\begin{aligned} & \text{" } i \in V_2 \text{ } \forall y_i : \text{" } i \in V_2 \quad i \in Q * 1 \quad i \in V_2 \\ p_{\text{"}} & \text{" } i \in V_2 \text{ } LLM \quad f_S | t^+ \text{" } i \in V_2 \\ & \text{" } a_i \text{ } i \in V_2 \quad (y_t / x_i, y_{t-1}): \end{aligned}$$

i & GoT(y) y h: " i < 1/2 i 1/2 "
 ° i < B / * " h i < 1/2 1/2 1/2 i < 1/2
 T H i < 1/2 1/2 i < 1/2 — i < 1/2 1/2 1/2 1/2 i
 " i < 1/2

(, i Δ 1/2 i Δ 1/2 • i 1/2 (long CoT) i Δ 1/2
 Δ 1/2 Δ 1/2 Δ 1/2 U „ ° • f i Δ 1/2 i Δ 1/2
 S 8 i Δ 1/2 ° • | i Δ 1/2 i Δ 1/2 Δ 1/2 L: <

1) $\int_{\mathbb{R}^2} \frac{1}{|x|} dx = \int_0^\infty \int_0^{2\pi} \frac{1}{r} r dr d\theta = \int_0^\infty \int_0^{2\pi} 1 dr d\theta = \int_0^\infty 2\pi dr = \infty$

[illegible]

2.2. Supervised Fine-Tuning (SFT)

$$\frac{f(x_i; y_i) g_{i=1}^N}{\sum_{j=1}^N f(x_i; y_j) g_{j=1}^N} \cdot (\text{SFT (Lamb et al., 2016)}) \in \mathbb{R}_{\text{SFT}} =$$

2.3. Reinforcement Learning (RL)

($\bar{\Gamma}$ 1/2 SFT $\bar{\Gamma}$ 1/2 K $\bar{\Gamma}$ 1/2 1/2 1/2 \bullet ($\bar{\Gamma}$:
f` \bullet CoT₁)

$$\begin{aligned} V \pm i p_{1/2} & \rightarrow \tilde{g} V_2^* \rightarrow \tilde{g} V_{2\pm} r_t \rightarrow \tilde{g} V_{2\pm} c \\ n & \rightarrow \tilde{g} V_{2\pm} V_{2\pm} \rightarrow \tilde{g} V_{2\pm} V_{2\pm} \rightarrow \tilde{g} V_{2\pm} V_{2\pm} \rightarrow \tilde{g} V_{2\pm} V_{2\pm} \end{aligned}$$

accuracy.

2.2. Supervised Fine-Tuning (SFT)

A common practice is to initialize the policy via SFT (Lamb et al., 2016) on a dataset $D_{\text{SFT}} = \{x_i, y_i\}_{i=1}^N$, where y_i can be normal or long CoT reasoning tokens.

2.3. Reinforcement Learning (RL)

After optional SFT initialization, we can further optimize the generation of long CoT with reinforcement learning.

Reward Function. We define a scalar reward r_t designed to encourage correct and verifiable reasoning. We only consider the outcome-based reward for the final answer produced, and do not consider process-based reward for the intermediate steps. We denote the term $r_{\text{answer}}(y)$ to capture the correctness of the final solution.

Policy Update. We adopted Proximal Policy Optimization (PPO) (Schulman et al., 2017) as the default policy optimization method in our experiments. We also briefly discuss REINFORCE (Sutton & Barto, 2018) method in subsection 7.3. We adopt a rule-based verifier as the reward function, which compares the predicted answer with the ground truth answer directly. The resulting updates push the policy to generate tokens that yield higher reward.

2.4. Training Setup

We adopt LLaMA-3.1-8B (Meta, 2024) and Qwen2.5-7B-Math (Qwen Team, 2024a) as the base models, which are representative general and math-specialized models respectively. For both SFT and RL, we use the 7,500 training sample prompt set of MATH (Hendrycks et al., 2021) by default, with which verifiable ground truth answers are provided. For SFT when ground truth answers are available, we synthesize responses by rejection sampling (Zelikman et al., 2022; Dong et al., 2023; Yuan et al., 2023; Gulcehre et al., 2023; Singh et al., 2023; Yue et al., 2024a; Tong et al., 2024). Specifically,

we first sample a fixed number N of candidate responses per prompt and then filter by only retaining ones with final answers consistent with the corresponding ground truth answers. We also discuss data like WebInstruct (Yue et al., 2024b) that is more diverse but without gold supervision signals like ground truth answers in §5. We train the models with the OpenRLHF framework (Hu et al., 2024).

2.5. Evaluation Setup

We focus on four representative reasoning benchmarks: MATH-500, AIME 2024, TheoremQA (Chen et al., 2023), and MMLU-Pro-1k (Wang et al., 2024a). Given that our training data is primarily in the mathematical domain, these benchmarks provide a comprehensive framework for both in-domain (MATH-500 test set) and out-of-domain evaluations (AIME 2024, TheoremQA, MMLU-Pro-1k). By default, we generate from the models using a temperature of $t = 0.7$, a top- p value of 0.95, and a maximum output length of 16,384 tokens. Please refer to Appendix E.1 for further details on the evaluation setup.

3. Impact of SFT on Long CoT

In this section, we compare long and short CoT data for SFT and in the context of RL initialization.

3.1. SFT Scaling

To compare long CoT with short CoT, the first step is to equip the model with the corresponding behavior. The most straightforward approach is to fine-tune the base model on CoT data. Since short CoT is common, curating SFT data for it is relatively simple via rejection sampling from existing models. However, how to obtain high-quality long CoT data remains an open question.

Setup. To curate the SFT data, for long CoT, we distill from QwQ-32B-Preview (we discuss other long CoT data construction methods in §3.3). For short CoT, we distill from Qwen2.5-Math-72B-Instruct,

we first sample a fixed number N of candidate responses per prompt and then filter by only retaining ones with final answers consistent with the corresponding ground truth answers. We also discuss data like WebInstruct (Yue et al., 2024b) that is more diverse but without gold supervision signals like ground truth answers in §5. We train the models with the OpenRLHF framework (Hu et al., 2024).

2.4. Training Setup

We adopt LLaMA-3.1-8B (Meta, 2024) and Qwen2.5-7B-Math (Qwen Team, 2024a) as the base models, which are representative general and math-specialized models respectively. For both SFT and RL, we use the 7,500 training sample prompt set of MATH (Hendrycks et al., 2021) by default, with which verifiable ground truth answers are provided. For SFT when ground truth answers are available, we synthesize responses by rejection sampling (Zelikman et al., 2022; Dong et al., 2023; Yuan et al., 2023; Gulcehre et al., 2023; Singh et al., 2023; Yue et al., 2024a; Tong et al., 2024).

Specifically, we first sample a fixed number N of candidate responses per prompt and then filter by only retaining ones with final answers consistent with the corresponding ground truth answers. We also discuss data like WebInstruct (Yue et al., 2024b) that is more diverse but without gold supervision signals like ground truth answers in §5. We train the models with the OpenRLHF framework (Hu et al., 2024).

2.5. Evaluation Setup

We focus on four representative reasoning benchmarks: MATH-500, AIME 2024, TheoremQA (Chen et al., 2023), and MMLU-Pro-1k (Wang et al., 2024a). Given that our training data is primarily in the mathematical domain, these benchmarks provide a comprehensive framework for both in-domain (MATH-500 test set) and out-of-domain evaluations (AIME 2024, TheoremQA, MMLU-Pro-1k). By default, we generate from the models using a temperature of $t = 0.7$, a top- p value of 0.95, and a maximum output length of 16,384 tokens. Please refer to Appendix E.1 for further details on the evaluation setup.

3. Impact of SFT on Long CoT

In this section, we compare long and short CoT data for SFT and in the context of RL initialization.

3.1. SFT Scaling

To compare long CoT with short CoT, the first step is to equip the model with the corresponding behavior. The most straightforward approach is to fine-tune the base model on CoT data. Since short CoT is common, curating SFT data for it is relatively simple via rejection sampling from existing models. However, how to obtain high-quality long CoT data remains an open question.

Setup. To curate the SFT data, for long CoT, we distill from QwQ-32B-Preview (we discuss other long CoT data construction methods in §3.3). For short CoT, we distill from Qwen2.5-Math-72B-Instruct, we first sample a fixed number N of candidate responses per prompt and then filter by only retaining ones with final answers consistent with the corresponding ground truth answers. We also discuss data like WebInstruct (Yue et al., 2024b) that is more diverse but without gold supervision signals like ground truth answers in §5. We train the models with the OpenRLHF framework (Hu et al., 2024).

Specifically, we first sample a fixed number N of candidate responses per prompt and then filter by only retaining ones with final answers consistent with the corresponding ground truth answers. We also discuss data like WebInstruct (Yue et al., 2024b) that is more diverse but without gold supervision signals like ground truth answers in §5. We train the models with the OpenRLHF framework (Hu et al., 2024).

• 3.1 SFT Scaling

We compare long CoT and short CoT data for SFT and in the context of RL initialization.

Figure 1. Scaling curves of SFT and RL on LI ama-3. 1-8B with long CoTs and short CoTs. SFT with long CoTs can scale up to a higher upper limit and has more potential to further improve with RL.

which is a capable short CoT model in math reasoning. Specifically, we perform rejection sampling by first sampling N candidate responses per prompt and then filtering for ones with correct answers. For long CoT, we use $N \in \{32; 64; 128; 192; 256\}$, while for short CoT, we use $N \in \{32; 64; 128; 256\}$, skipping one N for efficiency. In each case, the number of SFT tokens is proportional to N . We use the base model LLaMA-3.1-8B (Meta, 2024). Please refer to Appendix E.3 for more details about the SFT setup.

Result. The dashed lines in Figure 1 illustrate that as we scale up the SFT tokens, long CoT SFT continues to improve model accuracy, whereas short CoT SFT saturates early at a lower accuracy level. For instance, on MATH-500, long CoT SFT achieves over 70% accuracy and has yet to plateau even at 3.5B tokens. In contrast, short CoT SFT converges below 55% accuracy, with an increase in SFT tokens from approximately 0.25B to 1.5B yielding only a marginal absolute improvement of about 3%.

Takeaway 3.1 for SFT Scaling Upper Limit

SFT with long CoT can scale up to a higher performance upper limit than short CoT. (Figure 1)

3.2. SFT Initialization for RL

Since RL is reported to have a higher upper limit than SFT, we compare long CoT and short CoT as different SFT initialization approaches for RL.

Setup. We initialize RL using SFT checkpoints from

§3.1, and train for four epochs, sampling four responses per prompt. Our approach employs PPO (Schulman et al., 2017) with a rule-based verifier from the MATH dataset, using its training split as our RL prompt set. We adopt our cosine length scaling reward with the repetition penalty, which will be detailed in §4. Further details about our RL setup and hyperparameters can be found in Appendix E.4 & E.5.1 respectively.

Result. The gap between solid and dashed lines in Figure 1 shows that models initialized with long CoT SFT can usually be further significantly improved by RL, while models initialized with short CoT SFT see little gains from RL. For example, on MATH-500, RL can improve long CoT SFT models by over 3% absolute, while short CoT SFT models have almost the same accuracies before and after RL.

Takeaway 3.2 for SFT Initialization for RL

SFT with long CoTs makes further RL improvement easier, while short CoTs do not. (Figure 1)

3.3. Sources of Long CoT SFT Data

To curate long CoT data, we compare two approaches: (1) **Construct** long CoT trajectories by prompting short CoT models to generate primitive actions and sequentially combining them; (2) **Distill** long CoT trajectories from existing long CoT models that exhibit emergent long CoT patterns.

Figure 1. (LI ama-3, 1-8B) \bullet (\bullet i EV $\frac{1}{2}$ $\frac{1}{2}$ SFT ER L, i U i $\frac{1}{2}$ $\frac{1}{2}$ (\bullet i $\frac{1}{2}$ SFT i $\frac{1}{2}$ U O i $\frac{1}{2}$ P v i $\frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{2}$ R L $\frac{1}{2}$ $\frac{1}{2}$ 9 i $\frac{1}{2}$ $\frac{1}{2}$

Figure 2. LI ama3. 1-8B CE Owen2. 5-Math-7B ! < (• (i x ½ ± i ½ f - i ½ ½ ½ ½ ... Š † — i ½ , CoT • | i U i ½ MATH-500 i ½ M i ½ , C i ½ i ½ ½ ½ Mi x ½ ½ ½ ½ ½ "Terminated CoTs" , / (‡ : | ... i ½ i ½

3.2. SFT Initialization for RL

[illegible][illegible]

1. \mathbb{P}^n PPO (Schulman et al., 2017) $\mathbb{E} \pi(\theta)$ $\mathbb{E} \pi(\theta)$
 2. \mathbb{P}^n \mathbb{P}^n , $\mathbb{E} \pi(\theta)$ $\mathbb{E} \pi(\theta)$ $\mathbb{E} \pi(\theta)$
 3. \mathbb{P}^n \mathbb{P}^n $\mathbb{E} \pi(\theta)$ $\mathbb{E} \pi(\theta)$ $\mathbb{E} \pi(\theta)$
 4. \mathbb{P}^n \mathbb{P}^n $\mathbb{E} \pi(\theta)$ $\mathbb{E} \pi(\theta)$ $\mathbb{E} \pi(\theta)$
 5. \mathbb{P}^n \mathbb{P}^n $\mathbb{E} \pi(\theta)$ $\mathbb{E} \pi(\theta)$ $\mathbb{E} \pi(\theta)$
 6. \mathbb{P}^n \mathbb{P}^n $\mathbb{E} \pi(\theta)$ $\mathbb{E} \pi(\theta)$ $\mathbb{E} \pi(\theta)$
 7. \mathbb{P}^n \mathbb{P}^n $\mathbb{E} \pi(\theta)$ $\mathbb{E} \pi(\theta)$ $\mathbb{E} \pi(\theta)$
 8. \mathbb{P}^n \mathbb{P}^n $\mathbb{E} \pi(\theta)$ $\mathbb{E} \pi(\theta)$ $\mathbb{E} \pi(\theta)$
 9. \mathbb{P}^n \mathbb{P}^n $\mathbb{E} \pi(\theta)$ $\mathbb{E} \pi(\theta)$ $\mathbb{E} \pi(\theta)$
 10. \mathbb{P}^n \mathbb{P}^n $\mathbb{E} \pi(\theta)$ $\mathbb{E} \pi(\theta)$ $\mathbb{E} \pi(\theta)$

[illegible]

3.3. Sources of Long CoT SFT Data

$$\begin{aligned} &: \dagger \dagger \quad \bullet \cdot \ddot{\text{I}} \in \tfrac{1}{2} \quad \text{CoT} \quad \text{pn} \quad \ddot{\text{I}} \in \tfrac{1}{2} \dagger \$ \ddot{\text{I}} \in \tfrac{1}{2} \\ &\ddot{\text{I}} \in \tfrac{1}{2}(1) \text{ „ } \ddot{\text{I}} \in \tfrac{1}{2} \in \tfrac{1}{2} \text{h} \ddot{\text{I}} \in \tfrac{1}{2} \ddot{\text{I}} \in \tfrac{1}{2} \ddot{\text{I}} \in \tfrac{1}{2} \tfrac{1}{2}! \\ &< \quad \ddot{\text{I}} \in \tfrac{1}{2} \setminus \vee \bullet ! \ddot{\text{I}} \in \tfrac{1}{2} \ddot{\text{I}} \in (2) \ddot{\text{I}} \in \tfrac{1}{2} \bullet \ddot{\text{I}} \in \tfrac{1}{2} \\ &\text{h} \ddot{\text{I}} \in \tfrac{1}{2} \in \tfrac{1}{2} \ddot{\text{I}} \in \tfrac{1}{2} \in \tfrac{1}{2} \bullet \ddot{\text{I}} \in \tfrac{1}{2}! \quad \text{„ } ^{\circ} \quad \bullet \ddot{\text{I}} \in \tfrac{1}{2} \\ &! < - \ddot{\text{I}} \tfrac{1}{2} \end{aligned}$$

i 7/2 : t , i e h e 1/2 h i e 1/2 i e h e 1/2
t * " \ e 1/2 F q D U E.8 s l t
i e 1/2 i e 1/2 " \ clari fy decompose
solution_step reflection answer
i e h e e e 1/2 i j 1/2 ! < < ,
Owen2.5-72B-Instruct e • i e 1/2 \
i e 1/2! < o1-mini -0912 + i e 1/2
" i e x i z 1/2 • i e 1/2 h i e 1/2 i e
QwQ-32-Preview \ : Y ! < (i \$ 1/2 e 1/2
- i j 1/2 1/2 t MATH - i j 1/2 1/2 i e 1/2 "

Figure 2. Both LLaMA-3.1-8B and Qwen2.5-Math-7B models trained under RL with the Classic Reward manifested emergent CoT length scaling past the context window size, resulting in a decline in MATH-500 accuracy. The red points on the charts correspond to the iteration where the accuracy dropped to near zero. “Terminated CoTs” refer to responses that conclude within the context length.

Setup. To construct long CoT trajectories, we developed an Action Prompting framework (Appendix E.8) which defined the following primitive actions: `clarify`, `decompose`, `solution_step`, `reflection`, and `answer`. We employed multi-step prompting with a short CoT model (e.g., Qwen2.5-72B-Instruct) to sequence these actions, while a stronger model, o1-mini-0912, generates reflection steps incorporating self-correction. For distilling long CoT trajectories, we use QwQ-32-Preview as the teacher model. In both approaches, we adopt the MATH training set as the prompt set and apply rejection sampling. To ensure fairness, we use the same base model (LLaMA-3.1-8B), maintain approximately 200k SFT samples, and use the same RL setup as in §3.2.

Result. Table 1 shows that the model distilled from emergent long CoT patterns generalizes better than the constructed pattern, and can be further significantly improved with RL, while the model trained on constructed patterns cannot. Models trained with the emergent long CoT pattern achieve significantly higher accuracies on OOD benchmarks AIME 2024 and MMLU-Pro-1k, improving by 15-50% relatively. Besides, on the OOD benchmark TheoremQA, RL on the long CoT SFT model significantly improves its accuracy by around 20% relative, while the short CoT model’s performance does not change. This is also why we conduct most of our experiments based on distilled long CoT trajectories.

Takeaway 3.3 for Long CoT Cold Start

SFT initialization matters: high-quality, emergent long CoT patterns lead to significantly better generalization and RL gains. (Table 1)

Table 1. Emergent long CoT patterns outperform constructed ones. All the models here are fine-tuned from the base model LLaMA-3.1-8B with the MATH training prompt set.

Training Method	Long CoT SFT Pattern	MATH 500	AIME 2024	Theo. QA	MMLU Pro-1k
SFT	Constructed	48.2	2.9	21.0	18.1
	Emergent	54.1	3.5	21.8	32.0
SFT+RL	Constructed	52.4	2.7	21.0	19.2
	Emergent	59.4	4.0	25.2	34.6

4. Impact of Reward Design on Long CoT

This section examines reward function design, with a focus on its influence on CoT length and model performance.

4.1. CoT Length Stability

Recent studies on long CoT (DeepSeek-AI, 2025; Kimi Team, 2025; Hou et al., 2025) suggest that models naturally improve in reasoning tasks with increased thinking time. Our experiments confirm that models fine-tuned on long CoT distilled from QwQ-32B-Preview tend to extend CoT length under RL training, albeit sometimes unstably. This instability, also noted by Kimi Team (2025); Hou et al. (2025), has been addressed using techniques based on length and repetition penalties to stabi-

(+ 1/2 1/2 1/2 : + n 1/2 s' ... 1/2 1/2 1/2
 „ 1/2 1/2 LLaMA-3.1-8B 1/2 1/2 1/2 200k „
 SFT 7, v • (+ §3.2 - 1/2 1/2 RL 1/2 1/2

1/2 1/2 h 1 > : 1/2 1/2 • 1/2 1/2 ! 1/2 1/2 „ ! <
 1/2 1/2 1/2 1/2 w 1/2 1/2 1/2 1/2 1/2 v 1/2 1/2 1/2 1/2
 RL 1/2 1/2 > W 1/2 1/2 1/2 1/2 1/2 - 1/2 1/2 <

1/2 1/2 (° t • 1/2 1/2 ! - 1/2 1/2 < (OOD
 1/2 1/2 AIME 2024 MMLU-Pro-1k „ 1/2 1/2 > W 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 15-50% d (OOD 1/2 1/2 1/2

TheoremQA RL (• 1/2 1/2 SFT ! < „ 1/2 1/2

† > W 1/2 1/2 1/2 1/2 1/2 1/2 ! < „ 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 : 1/2 1/2 H 1/2 1/2 p z Ei 1/2 1/2 1/2 1/2
 1/2 1/2 „ • 1/2 1/2 h 1/2 1/2 1/2 1/2

• §3.3 • CoT • / ...

SFT 1/2 1/2 1/2 1/2 1/2 1/2 ° t „ • CoT
 ! > W 1/2 1/2 1/2 1/2 1/2 1/2 RL 1/2 1/2 1/2
 1

Table 1. 1/2 • CoT! Z „ 1/2 1/2 1/2 1/2 @
 ! < 1/2 1/2 1/2 1/2 LLaMA-3.1-8B @ e „ v • (+
 † MATH- 1/2 1/2 1/2 1/2

Training Method	Long CoT SFT Pattern	MATH 500	AIME 2024	Theo. QA	MMLU Pro-1k
SFT	Constructed	48.2	2.9	21.0	18.1
	Emergent	54.1	3.5	21.8	32.0
SFT+RL	Constructed	52.4	2.7	21.0	19.2
	Emergent	59.4	4.0	25.2	34.6

4. Impact of Reward Design on Long CoT

, , 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 • | E
 ! < ' 1/2 1/2 1/2 1/2

4.1. CoT Length Stability

1/2 1/2 • 1/2 1/2 CoT „ v (DeepSeek-AI, 2025; Kimi Team, 2025; Hou et al., 2025) h !
 < (Z 1/2 1/2 1/2 v „ 1/2 1/2 1/2 1/2 1/2 1/2
 G 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2
 < (: f ` RL - 1/2 1/2 Z 1/2 1/2 1/2 „ •
 | = j 1/2 1/2 3 s 1/2 1/2 1/2 s' — « Kimi

Figure 3. 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 • •
 | O

Team (2025); Hou et al. (2025) 1/2 O v 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2

lize training.

Setup. We used two different models fine-tuned on long CoT data distilled from QwQ-32B-Preview using the MATH train split, with a context window size of 16K. The models were Llama3.1-8B and Qwen2.5-Math-7B. We used a rule-based verifier along and a simple reward of 1 for correct answers. We shall refer to this as the *Classic Reward*. More details can be found in Appendix E.5.2.

Results. We observed that both models increased their CoT length during training, eventually reaching the context window limit. This led to a decline in training accuracy due to CoTs exceeding the allowable window size. Additionally, different base models exhibited distinct scaling behaviors. The weaker LLaMA-3.1-8B model showed greater fluctuations in CoT length compared to Qwen2.5-Math-7B, as illustrated in Figure 2.

We also found that the rate at which CoTs exceeded the context window size leveled off at a certain threshold below 1 (Figure 2). This suggests that exceeding the limit started to apply significant downward pressure on the CoT length distribution, and highlights the context window size’s role in implicit length penalization. Notably, a trajectory might be penalized even without an explicit exceed-length penalty due to reward or advantage normalization, both of which are standard in RL frameworks.

Takeaway 4.1 for CoT Length Stability

CoT length does not always scale up in a stable fashion. (Figure 2)

4.2. Active Scaling of CoT Length

We found that reward shaping can be used to stabilize emergent length scaling. We designed a reward function to use CoT length as an additional input and to observe a few ordering constraints. Firstly, correct CoTs receive higher rewards than wrong CoTs. Secondly, shorter cor-

Figure 3. The Classic and Cosine Reward functions. The Cosine Reward varies with generation length.

rect CoTs receive higher rewards than longer correct CoTs, which incentivizes the model to use inference compute efficiently. Thirdly, shorter wrong CoTs should receive higher penalties than longer wrong CoTs. This encourages the model to extend its thinking time if it is less likely to get the correct answer.

We found it convenient to use a piecewise cosine function, which is easy to tune and smooth. We refer to this reward function as the *Cosine Reward*, visualized in Figure 3. This is a *sparse* reward, only awarded once at the end of the CoT based on the correctness of the answer. The formula of **CosFn** can be found in equation 1 in the appendix.

$$R(C; L_{\text{gen}}) = \begin{cases} \approx \text{CosFn}(L_{\text{gen}}; L_{\text{max}}; r_0^C; r_L^C); & \text{if } C = 1; \\ \approx \text{CosFn}(L_{\text{gen}}; L_{\text{max}}; r_0^W; r_L^W); & \text{if } C = 0; \\ r_e; & \text{if } L_{\text{gen}} = L_{\text{max}}. \end{cases}$$

Hyperparameters:

$r_0^C=r_0^W$: Reward (correct/wrong) for $L_{\text{gen}} = 0$;

 $r_L^C = r_L^W$: Reward (correct/wrong) for $L_{\text{gen}} = L_{\text{max}}$;

r_e : Exceed length penalty;

Inputs:

C : Correctness (0 or 1);

L_{gen} : Generation length.

Setup. We ran experiments with the Classic Reward and the Cosine Reward. We used the LLaMA3.1-8B fine-tuned on long CoT data distilled from QwQ-32B-Preview using the MATH train split, as our starting point. For more details, see Appendix E.5.3.

4.2. Active Scaling of CoT Length

$\dot{\gamma} \in \mathcal{L} \mathcal{W} \pm Qb \dot{\gamma} \in \mathcal{L} \mathcal{Z} 3 \dot{\gamma} \dot{\gamma} \frac{1}{2} \cdot | |$
 $> \dot{\gamma} \in \mathcal{L} \mathcal{V}_i * V \pm \dot{\gamma} \mathcal{P} \frac{1}{2} \cdot (CoT \cdot | \setminus$
 $: D " e \vee \dot{\gamma} \mathcal{V}_2 \dot{\gamma} \mathcal{V}_2 \frac{1}{2} \cdot | _ - H \text{ cn}$
 $„ CoT \dot{\gamma} \mathcal{V}_2 \dot{\gamma} CoTs - \dot{\gamma} \in \mathcal{L} \mathcal{V} \pm \vee ! f$
 $\dot{\gamma} \in \mathcal{L} \text{cn} CoT \dot{\gamma} \mathcal{V}_2 \cdot „ \text{cn} CoTs - \dot{\gamma} \in \mathcal{L} \mathcal{V} \pm$
 $\dot{\gamma} \in \mathcal{L} ! < \dot{\gamma} \mathcal{H} O \cdot (\dot{\gamma} | - , f \dot{\gamma} \in \mathcal{L} \frac{1}{2}$
 $\dot{\gamma} CoTs \dot{\gamma} \mathcal{V}_2 \cdot „ \dot{\gamma} CoTs - \dot{\gamma} \in \mathcal{L} \mathcal{V}_2 \dot{\gamma} \mathcal{Z} \frac{1}{2} \dot{\gamma}$
 $\pm ! < (* \dot{\gamma} \in \mathcal{L} \mathcal{Q} \text{cn} T H \dot{\gamma} \in \mathcal{L} \mathcal{V}_2 \dot{\gamma} \dot{\gamma} \dot{\gamma}$
 $\dot{\gamma} \in \mathcal{L} \frac{1}{2}$

[illegible]

$$R(C; L_{\text{gen}}) = \begin{cases} \infty & \\ \approx \text{CosFn}(L_{\text{gen}}; L_{\text{max}}; r_0^C; r_L^C); & \text{if } C = 1; \\ \approx \text{CosFn}(L_{\text{gen}}; L_{\text{max}}; r_0^W; r_L^W); & \text{if } C = 0; \\ r_{e_i} & \text{if } L_{\text{gen}} = L_{\text{max}}. \end{cases}$$

Hyperparameters:

$r_0^C=r_0^W$: Reward (correct/wrong) for $L_{\text{gen}} = 0$;

 $r_L^C = r_L^W$: Reward (correct/wrong) for $L_{\text{gen}} = L_{\text{max}}$;

r_e : Exceed length penalty;

Inputs:

C : Correctness (0 or 1);

L_{gen} : Generation length.

i 2 1/2 i 2 1/4 + i 2 1/4 ± 1/2 Y & V ± i 2 1/2
 L + 2 1/2 ± i 2 1/4 + (MATH- i 2 1/2 1/2
 i 2 1/2 32B-Preview, • „ • i 2 1/2 n ®
 „ Li ama3. 1-8B\ : w 1/2 i 2 1/4 i 2 1/4 1/2
 UE.5.3

ĩ ge_{1/2} ĩ ĩ ~~1/2~~_{1/2}Y&V±>W3š†!<(RL
 „•|)>L: ĩ ĩ ~~1/2~~₃š†-ĩ ĩ ~~1/2~~_{1/2}v ĩ ~~1/2~~_{1/2}
 †RLH‡ ĩ ~~4~~_{1/2} ĩ ĩ ~~1/2~~_{1/2}Q1/2<(8ĩ ĩ ~~1/2~~
 „'ĩ ĩ ~~Q~~_{1/2}1/2ĩ ~~5~~_{1/2}

Figure 4. LI ama3. 1-8B • (Y & V ± Cosine Reward) i 2 1/2
1/2 • | t b - i 2 1/2 i 2 1/2 (a) - i 2 1/2 i 2 1/2 (b) i 2 1/2
1 • | i 2 1/2 i 2 1/2 i 2 1/2 8 i 2 1/2 i 2 1/2 i 2 1/2
1/4 b 1/2, c 1/2 h: - i 2 1/2 i 2 1/2 Mi 1/2 i 2 1/2 i 2 1/2

$\bullet \frac{1}{2} 4.2 \leq \tilde{Z} \text{ CoT} \cdot |_{n; \dots} >$

$V \pm Q b i \in \frac{1}{2} \tilde{Z} (\frac{1}{2} \frac{1}{2} \frac{1}{2} \frac{1}{2})$ $i \in \mathbb{Z}$

$\in S_6 \text{ CoT} \cdot |_{(i)} (A \frac{1}{5})$

4.3. Cosine Reward Hyperparameters

$\gamma \in V_{\pm} \dots i$ $p^{1/2} \in \frac{1}{2} \mathbb{Z}$ $i \in \frac{1}{2} \mathbb{Z}$ $i \in \mathbb{Z}$ CoT_n

[illegible]

i e% dV- , i g! % ½ ½ c n T H, V
± • @CoT• | „ ž ž r₀ < r_L CoT• |
%gž i ½ ½ ½ c n V ± i ½ ½ ½ V ½
± Š N CoT• | Š• i ½ ½ ½ i ½ ½ ½
i ½ ½ ½ L: v - c n E i V ½ „ ½ i ½
š † ! < i T H, n ½ i ½ ½ CoTv i ½

• $\frac{1}{2} 4.3 \leq \tilde{Z} Y \& V \pm \dots \tilde{I} \text{ } \frac{1}{2}$
 $Y \& V \pm \tilde{I} \text{ } \frac{1}{2} \frac{1}{2} \tilde{I} \text{ } \frac{1}{2} \quad , \quad \cdot \quad | \quad) \quad > \quad L$
 $\dots \quad (DU \tilde{I} \text{ } \frac{1}{2}) \frac{1}{2}$

4.4. Context Window Size

i j $\frac{1}{2}$ k $\frac{1}{2}$ l : ! < j $\frac{1}{2}$ + i $\frac{1}{2}$ C"
 z i j $\frac{1}{2}$ @ - i $\frac{1}{2}$, ž ! < i $\frac{1}{2}$ +) (

the correct and wrong rewards determines how confident the model has to be about an answer for it to derive a positive expected value from terminating its CoT with an answer.

Takeaway 4.3 for Cosine Reward Hyperparameters

Cosine Reward can be tuned to incentivize various length scaling behaviors. (Appendix Figure 9)

Figure 4. LLaMA3.1-8B trained with length shaping using the Cosine Reward exhibited more stable (a) training accuracy and (b) response length. This stability led to improved performance on downstream tasks (Figure 5). Red points on the charts indicate iterations where training accuracy dropped to near zero.

Result. We found that the Cosine Reward significantly stabilized the length scaling behavior of the models under RL, thereby also stabilizing the training accuracy and improving RL efficiency (Figure 4). We also observed improvements in model performance on downstream tasks (Figure 5).

Takeaway 4.2 for Active Scaling of CoT Length

Reward shaping can be used to stabilize and control CoT length while improving accuracy. (Figure 4, 5)

4.3. Cosine Reward Hyperparameters

The Cosine Reward hyperparameters can be tuned to shape CoT length in different ways.

Setup. We set up RL experiments with the same model fine-tuned on long CoT distilled from QwQ-32B-Preview, but with different hyperparameters for the Cosine Reward function. We tweaked the correct and wrong rewards $r_0^c; r_L^c; r_0^w; r_L^w$ and observed their impact on the CoT lengths. For more details, see Appendix E.5.4.

Result. We see from Figure 9 in the Appendix that if the reward for a correct answer increases with CoT length ($r_0^c < r_L^c$), the CoT length increases explosively. We also see that the lower the correct reward relative to the wrong reward, the longer the CoT length. We interpret this as a kind of trained risk aversion, where the ratio of

4.4. Context Window Size

We know that longer contexts give a model more room to explore, and with more training samples, the model eventually learns to utilize more of the context window. This raises an interesting question – are more training samples necessary to learn to utilize a larger context window?

Setup. We set up 3 experiments using the same starting model fine-tuned on long CoT data distilled from QwQ-32B-Preview with the MATH train split. We also used the latter as our RL prompt set. Each ablation used the Cosine Reward and repetition penalty with a different context window size (4K, 8K, and 16K). For more details, see Appendix E.5.5.

Result. We found that the model with a context window size of 8K performed better than the model with 4K, as expected. However, we observed performance was better under 8K than 16K. Note that all three experiments used the same number of training samples (Figure 6). We see this as an indication that models need more training compute to learn to fully utilize longer context window sizes, which is consistent with the findings of (Hou et al., 2025).

Takeaway 4.4 for Context Window Size

Models might need more training samples to learn to utilize larger context window sizes. (Figure 6)

Figure 5. $V \pm \sigma$ for LLaMA-3.1-8B (σ is the standard deviation)

Figure 5 shows the performance of LLaMA-3.1-8B across different context window sizes (4K, 8K, 16K) and training samples. The figure includes a table of results and a bar chart showing the performance of the model under different conditions. The table shows the performance of the model under different conditions, including the context window size, the number of training samples, and the performance metric. The bar chart shows the performance of the model under different conditions, including the context window size, the number of training samples, and the performance metric. The figure also includes a table of results and a bar chart showing the performance of the model under different conditions. The table shows the performance of the model under different conditions, including the context window size, the number of training samples, and the performance metric. The bar chart shows the performance of the model under different conditions, including the context window size, the number of training samples, and the performance metric.

• 4.4.5 σ for LLaMA-3.1-8B

! < σ for LLaMA-3.1-8B (σ is the standard deviation)

4.5. Length Reward Hacking

Figure 5 shows the performance of LLaMA-3.1-8B across different context window sizes (4K, 8K, 16K) and training samples. The figure includes a table of results and a bar chart showing the performance of the model under different conditions. The table shows the performance of the model under different conditions, including the context window size, the number of training samples, and the performance metric. The bar chart shows the performance of the model under different conditions, including the context window size, the number of training samples, and the performance metric.

• 4.5.5 σ for LLaMA-3.1-8B

! < σ for LLaMA-3.1-8B (σ is the standard deviation)

4.6. Optimal Discount Factors

Figure 5 shows the performance of LLaMA-3.1-8B across different context window sizes (4K, 8K, 16K) and training samples. The figure includes a table of results and a bar chart showing the performance of the model under different conditions. The table shows the performance of the model under different conditions, including the context window size, the number of training samples, and the performance metric. The bar chart shows the performance of the model under different conditions, including the context window size, the number of training samples, and the performance metric.

Figure 5. Performance of LLaMA-3.1-8B trained with different reward functions on a variety of evaluation benchmarks.

Figure 6. Performance of LLaMA-3.1-8B trained with different context window sizes. All experiments used the same number of training samples.

4.5. Length Reward Hacking

We observed that with enough training compute, the model started to show signs of reward hacking, where it increased the lengths of its CoTs on hard questions using repetition rather than learning to solve them. We also noted a fall in the branching frequency of the model, which we estimated by counting the number of times the pivot keyword “alternatively,” appeared in the CoT (Figure 10).

We mitigated this by implementing a simple N -gram repetition penalty (Algorithm 1). We observed that the penalty was most effectively applied on repeated tokens, rather than as a sparse reward for the entire trajectory. Similarly, we found that discounting the repetition penalty when calculating the return was effective. Specific feedback about where the repetition occurred presumably made it easier for the model to learn not to do it (see more in §4.6).

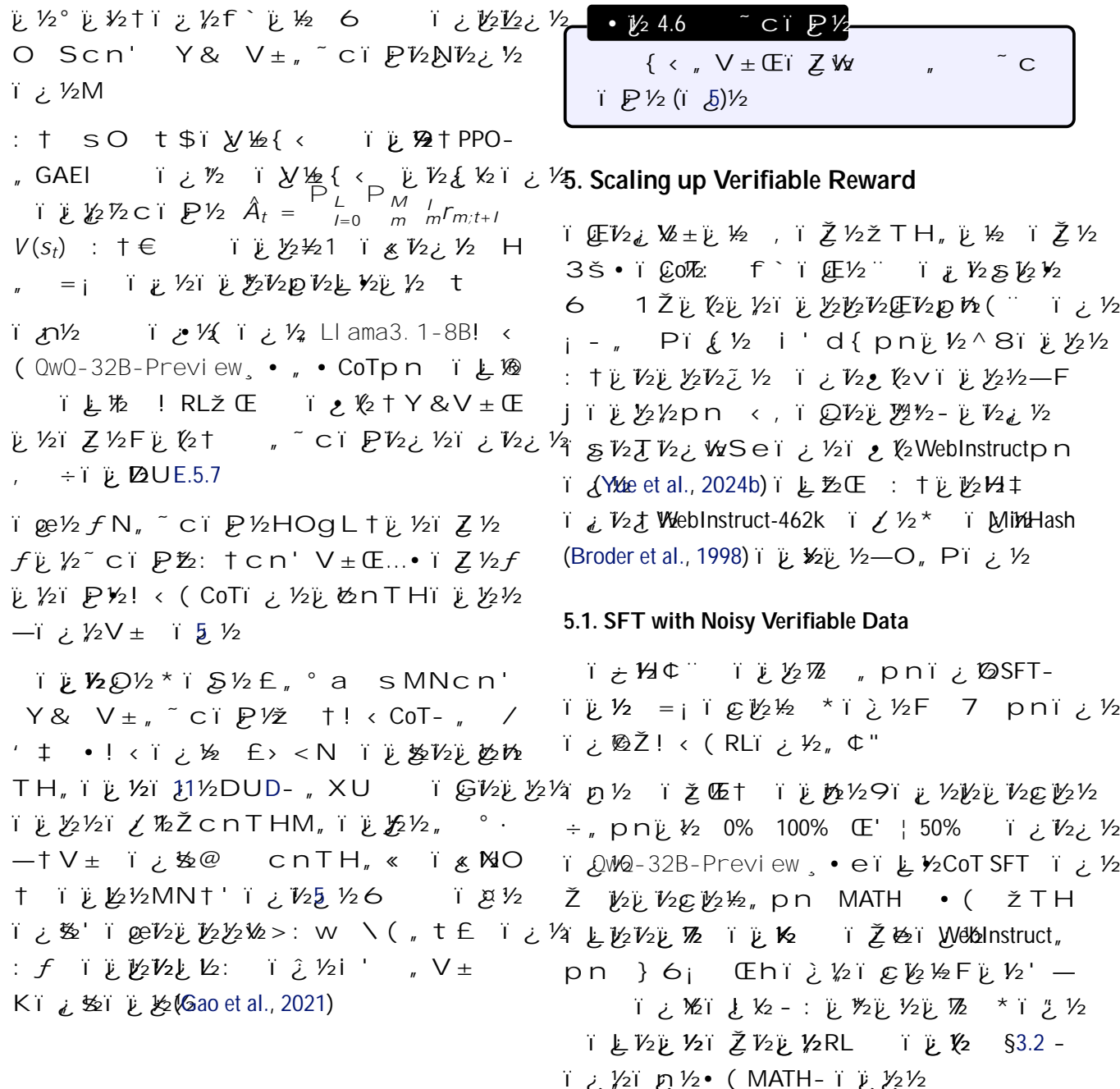
Setup. We used the LLaMA3.1-8B model fine-tuned on long CoT data distilled from QwQ-32B-Preview.

We ran two RL training runs, both using the Cosine Reward, but with and without the repetition penalty. For more details, please refer to Appendix E.5.6.

Result. The repetition penalty resulted in better downstream task performance and also shorter CoTs, meaning there was better utilization of inference compute (Figure 5).

Observation. Our experiments revealed a relationship between the repetition penalty, training accuracy, and the Cosine Reward. When training accuracy was low, the Cosine Reward exerted greater upward pressure on CoT length, leading to increased reward hacking through repetition. This, in turn, required a stronger repetition penalty. Future work could further investigate these interactions and explore dynamic tuning methods for better optimization.

Figure 6. Performance of LLaMA-3.1-8B trained with different context window sizes. All experiments used the same number of training samples.



Takeaway 4.5 for Length Reward Hacking

Length rewards will be hacked with enough compute (Figure 10), but this can be mitigated using a repetition penalty. (Figure 5)

4.6. Optimal Discount Factors

We hypothesized that applying the repetition penalty with temporal locality (i.e., a low discount factor) would be most effective, as it provides a stronger learning signal about the specific offending tokens. However, we also observed performance degradation when the discount factor for the correctness (cosine) reward was too low.

To optimally tune both reward types, we modified the GAE formula in PPO to accommodate multiple reward types, each with its own discount factor: $\hat{A}_t = \sum_{l=0}^L \gamma^l \sum_{m=0}^M \gamma^m r_{m,t+l} - V(s_t)$. For simplicity, we set $\gamma = 1$, which proved effective, though we did not extensively tune this parameter.

Setup. We ran multiple RL experiments with the same Llama3.1-8B model fine-tuned on QwQ-32B-Preview distilled long CoT data. We used the Cosine Reward and repetition penalty but with different combinations of discount factors. For more details, please see Appendix E.5.7.

Result. A lower discount factor effectively enforces the repetition penalty, whereas a higher discount factor enhances the correctness reward and the exceed-length penalty. The higher factor allows the model to be adequately rewarded for selecting a correct answer earlier in the CoT (Figure 5).

We observed a rather interesting phenomenon where decreasing the discount factor of the correctness (cosine) reward increased the branching frequency in the model’s CoT, making the model quickly give up on approaches that did not seem to lead to a correct answer immediately (Figure 11, Extract in Appendix D). We hypothesize that this short-term thinking was due to a relatively small

number of tokens preceding the correct answer receiving rewards, which means stepping stones to the right answer are undervalued. Such behavior degraded performance (Figure 5). However, we think this qualitative result might be of potential interest to the research community, due to its similarity to the relationship between behaviors like delayed gratification and the distribution of rewards given to the biological brain (Gao et al., 2021).

Takeaway 4.6 for Optimal Discount Factors

Different kinds of rewards and penalties have different optimal discount factors. (Figure 5)

5. Scaling up Verifiable Reward

Verifiable reward signals like ones based on ground-truth answers are essential for stabilizing long CoT RL for reasoning tasks. However, it is difficult to scale up such data due to the limited availability of high-quality human-annotated verifiable data for reasoning tasks. As an attempt to counter this, we explore using other data that is more available despite more noise, like reasoning-related QA pairs extracted from web corpora. Specifically, we experiment with the WebInstruct dataset (Yue et al., 2024b). For efficiency, we construct WebInstruct-462k, a deduplicated subset derived via MinHash (Broder et al., 1998).

5.1. SFT with Noisy Verifiable Data

We first explore adding such diverse data to SFT. Intuitively, despite less reliable supervision signals, diverse data might facilitate the model’s exploration during RL.

Setup. We experiment with three setups, varying the proportion of data without gold supervision signals: 0%, 100%, and approximately 50%. We conduct long CoT SFT by distilling from QwQ-32B-Preview. For data with gold supervision signals (MATH), ground truth answers are used for rejection sampling. In contrast, for data from WebInstruct without fully reliable supervision signals but with a much larger scale, we sample one

token preceding the correct answer. We compare SFT (MATH-Pro-1k) and SFT (MATH-Pro-1k + WebInstruct-462k) on MATH, AIME, TheoQA, and MMLU-Pro-1k. The results show that adding WebInstruct data improves performance across all tasks, with the most significant gains in MATH and AIME.

Table 2. Performance comparison of SFT with different data sources. The results are averaged over 5 runs.

Long CoT SFT Data	Training Method	MATH 500	AIME 2024	Theo. QA	MMLU Pro-1k	AVG
100% MATH	SFT	54.1	3.5	21.8	32.0	27.9
	SFT + RL	59.4	4.0	25.2	34.6	30.8
100% WebIT	SFT	41.2	0.8	21.9	41.1	26.3
	SFT + RL	44.6	1.9	22.5	43.3	28.1
50% MATH + 50% WebIT	SFT	53.6	4.4	23.5	41.7	30.8
	SFT + RL	57.3	3.8	25.1	42.0	32.1

Figure 5. Performance comparison of SFT with different data sources. The results are averaged over 5 runs.

5.2. Scaling up RL with Noisy Verifiable Data

We compare RL with different data sources. The results show that adding WebInstruct data improves performance across all tasks, with the most significant gains in MATH and AIME. The performance gains are more pronounced when the proportion of WebInstruct data is higher.

We compare RL with different data sources. The results show that adding WebInstruct data improves performance across all tasks, with the most significant gains in MATH and AIME. The performance gains are more pronounced when the proportion of WebInstruct data is higher.

We compare RL with different data sources. The results show that adding WebInstruct data improves performance across all tasks, with the most significant gains in MATH and AIME. The performance gains are more pronounced when the proportion of WebInstruct data is higher.

Table 3. Performance comparison of RL with different data sources. The results are averaged over 5 runs.

Prompt Set	Verifier Type	MATH 500	AIME 2024	Theo. QA	MMLU Pro-1k
MATH Baseline		59.4	4.0	25.2	34.6
SFT Initialization		46.6	1.0	23.0	28.3
Unfiltered	Rule-Based	45.4	3.3	25.9	35.1
	Model-Based	47.9	3.5	26.2	40.4
Filtered	Rule-Based	48.6	3.3	28.1	41.4
	Model-Based	47.9	3.8	26.9	41.4

We compare RL with different data sources. The results show that adding WebInstruct data improves performance across all tasks, with the most significant gains in MATH and AIME. The performance gains are more pronounced when the proportion of WebInstruct data is higher.

Figure 5. Performance comparison of RL with different data sources. The results are averaged over 5 runs.

response per prompt from the teacher model without filtration. For RL here, we adopt the same setup as in §3.2, using the MATH training set.

Result. Table 2 shows that incorporating silver-supervised data improves average performance. Adding WebInstruct data to long CoT SFT yields a substantial 5–10% absolute accuracy gain on MMLU-Pro-1k over using MATH alone. Furthermore, mixing MATH and WebInstruct data achieves the best average accuracy across benchmarks.

Table 2. Adding data with a silver supervision signal is often beneficial. “WebIT” is the abbreviation of WebInstruct.

Long CoT SFT Data	Training Method	MATH 500	AIME 2024	Theo. QA	MMLU Pro-1k	AVG
100% MATH	SFT	54.1	3.5	21.8	32.0	27.9
	SFT + RL	59.4	4.0	25.2	34.6	30.8
100% WebIT	SFT	41.2	0.8	21.9	41.1	26.3
	SFT + RL	44.6	1.9	22.5	43.3	28.1
50% MATH	SFT	53.6	4.4	23.5	41.7	30.8
+ 50% WebIT	SFT + RL	57.3	3.8	25.1	42.0	32.1

Takeaway 5.1 for SFT with Noisy Verifiable Data

Adding noisy but diverse data to SFT leads balanced performance across different tasks. (Table 2)

5.2. Scaling up RL with Noisy Verifiable Data

We compare two main approaches to obtain rewards from noisy verifiable data: 1) to extract short-form answers and use a rule-based verifier; 2) to use a model-based verifier capable of processing free-form responses.

Here a key factor is whether the QA pair can have a short-form answer. So we also compare whether the dataset is filtered by only retaining samples with short-form answers.

Setup. We implement the model-based verifier by prompting Qwen2.5-Math-7B-Instruct with the raw reference solution. To extract short-form answers, we first prompt Llama-3.1-8B-Instruct to extract from the raw responses and then apply rejection sam-

pling with QwQ-32B-Previe w. Specifically, we generate two responses per prompt from WebInstruct-462k and discard cases where neither response aligns with the extracted reference answers. This process yields approximately 189k responses across 115k unique prompts. Our case studies show that the rejection sampling drops many prompts due to: 1) many WebInstruct prompts lack short-form answers that our rule-based verifier can process effectively, and 2) some prompts are too difficult even for QwQ-32B-Previe w. For SFT we train LLaMA-3.1-8B on the filtered dataset as initialization for reinforcement learning (RL). In the RL stage, we use the full 462k prompt set in the unfiltered setup and the 115k subset in the filtered setup, training with 30k prompts and 4 responses per prompt. Further details about the model-based verifier, the answer extraction and the RL hyperparameters can be found in Appendix & E.5.8 & E.6 & E.7 respectively.

Table 3. Performance of RL with different verifiers and prompt filtering methods. All the models here are fine-tuned from LLaMA-3.1-8B. The “MATH Baseline” is the model trained with SFT and RL on MATH only in Table 2. The other models are trained with SFT by distillation from QwQ-32B-Preview and RL with different setups.

Prompt Set	Verifier Type	MATH 500	AIME 2024	Theo. QA	MLLM Pro-1k
	MATH Baseline	59.4	4.0	25.2	34.6
	SFT Initialization	46.6	1.0	23.0	28.3
Unfiltered	Rule-Based	45.4	3.3	25.9	35.1
	Model-Based	47.9	3.5	26.2	40.4
Filtered	Rule-Based	48.6	3.3	28.1	41.4
	Model-Based	47.9	3.8	26.9	41.4

Result. Table 3 shows that RL with the rule-based verifier on the filtered prompt set with short-form answers achieves the best performance across most benchmarks under the same number of RL samples. This might indicate that rule-based verifier after appropriate filtration can produce the highest-quality reward signals from noisy verifiable data. Moreover, compared to the model trained on human-annotated verified data (MATH), leveraging noisy yet diverse verifiable data still significantly boosts

6. Exploration on RL from the Base Model

[illegible]

6.1. Nuances in Analysis Based on Emergent Behaviors

1. 2. 3. 4. 5. 6. 7. 8. 9. 10. 11. 12. 13. 14. 15. 16. 17. 18. 19. 20. 21. 22. 23. 24. 25. 26. 27. 28. 29. 30. 31. 32. 33. 34. 35. 36. 37. 38. 39. 40. 41. 42. 43. 44. 45. 46. 47. 48. 49. 50. 51. 52. 53. 54. 55. 56. 57. 58. 59. 60. 61. 62. 63. 64. 65. 66. 67. 68. 69. 70. 71. 72. 73. 74. 75. 76. 77. 78. 79. 80. 81. 82. 83. 84. 85. 86. 87. 88. 89. 90. 91. 92. 93. 94. 95. 96. 97. 98. 99. 100. 101. 102. 103. 104. 105. 106. 107. 108. 109. 110. 111. 112. 113. 114. 115. 116. 117. 118. 119. 120. 121. 122. 123. 124. 125. 126. 127. 128. 129. 130. 131. 132. 133. 134. 135. 136. 137. 138. 139. 140. 141. 142. 143. 144. 145. 146. 147. 148. 149. 150. 151. 152. 153. 154. 155. 156. 157. 158. 159. 160. 161. 162. 163. 164. 165. 166. 167. 168. 169. 170. 171. 172. 173. 174. 175. 176. 177. 178. 179. 180. 181. 182. 183. 184. 185. 186. 187. 188. 189. 190. 191. 192. 193. 194. 195. 196. 197. 198. 199. 200. 201. 202. 203. 204. 205. 206. 207. 208. 209. 210. 211. 212. 213. 214. 215. 216. 217. 218. 219. 220. 221. 222. 223. 224. 225. 226. 227. 228. 229. 230. 231. 232. 233. 234. 235. 236. 237. 238. 239. 240. 241. 242. 243. 244. 245. 246. 247. 248. 249. 250. 251. 252. 253. 254. 255. 256. 257. 258. 259. 260. 261. 262. 263. 264. 265. 266. 267. 268. 269. 270. 271. 272. 273. 274. 275. 276. 277. 278. 279. 280. 281. 282. 283. 284. 285. 286. 287. 288. 289. 290. 291. 292. 293. 294. 295. 296. 297. 298. 299. 300. 301. 302. 303. 304. 305. 306. 307. 308. 309. 310. 311. 312. 313. 314. 315. 316. 317. 318. 319. 320. 321. 322. 323. 324. 325. 326. 327. 328. 329. 330. 331. 332. 333. 334. 335. 336. 337. 338. 339. 340. 341. 342. 343. 344. 345. 346. 347. 348. 349. 350. 351. 352. 353. 354. 355. 356. 357. 358. 359. 360. 361. 362. 363. 364. 365. 366. 367. 368. 369. 370. 371. 372. 373. 374. 375. 376. 377. 378. 379. 380. 381. 382. 383. 384. 385. 386. 387. 388. 389. 390. 391. 392. 393. 394. 395. 396. 397. 398. 399. 400. 401. 402. 403. 404. 405. 406. 407. 408. 409. 410. 411. 412. 413. 414. 415. 416. 417. 418. 419. 420. 421. 422. 423. 424. 425. 426. 427. 428. 429. 430. 431. 432. 433. 434. 435. 436. 437. 438. 439. 440. 441. 442. 443. 444. 445. 446. 447. 448. 449. 450. 451. 452. 453. 454. 455. 456. 457. 458. 459. 460. 461. 462. 463. 464. 465. 466. 467. 468. 469. 470. 471. 472. 473. 474. 475. 476. 477. 478. 479. 480. 481. 482. 483. 484. 485. 486. 487. 488. 489. 490. 491. 492. 493. 494. 495. 496. 497. 498. 499. 500. 501. 502. 503. 504. 505. 506. 507. 508. 509. 510. 511. 512. 513. 514. 515. 516. 517. 518. 519. 520. 521. 522. 523. 524. 525. 526. 527. 528. 529. 530. 531. 532. 533. 534. 535. 536. 537. 538. 539. 540. 541. 542. 543. 544. 545. 546. 547. 548. 549. 550. 551. 552. 553. 554. 555. 556. 557. 558. 559. 560. 561. 562. 563. 564. 565. 566. 567. 568. 569. 570. 571. 572. 573. 574. 575. 576. 577. 578. 579. 580. 581. 582. 583. 584. 585. 586. 587. 588. 589. 590. 591. 592. 593. 594. 595. 596. 597. 598. 599. 600. 601. 602. 603. 604. 605. 606. 607. 608. 609. 610. 611. 612. 613. 614. 615. 616. 617. 618. 619. 620. 621. 622. 623. 624. 625. 626. 627. 628. 629. 630. 631. 632. 633. 634. 635. 636. 637. 638. 639. 640. 641. 642. 643. 644. 645. 646. 647. 648. 649. 650. 651. 652. 653. 654. 655. 656. 657. 658. 659. 660. 661. 662. 663. 664. 665. 666. 667. 668. 669. 670. 671. 672. 673. 674. 675. 676. 677. 678. 679. 680. 681. 682. 683. 684. 685. 686. 687. 688. 689. 690. 691. 692. 693. 694. 695. 696. 697. 698. 699. 700. 701. 702. 703. 704. 705. 706. 707. 708. 709. 710. 711. 712. 713. 714. 715. 716. 717. 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. 756. 757. 758. 759. 760. 761. 762. 763. 764. 765. 766. 767. 768. 769. 770. 771. 772. 773. 774. 775. 776. 777. 778. 779. 780. 781. 782. 783. 784. 785. 786. 787. 788. 789. 790. 791. 792. 793. 794. 795. 796. 797. 798. 799. 800. 801. 802. 803. 804. 805. 806. 807. 808. 809. 810. 811. 812. 813. 814. 815. 816. 817. 818. 819. 820. 821. 822. 823. 824. 825. 826. 827. 828. 829. 830. 831. 832. 833. 834. 835. 836. 837. 838. 839. 840.

6.2. Nuances in Analysis Based on Length Scaling

• | i U « α: / | < H Φ " „ i ɛ ʏ ɛ ½ y
 • 6 i ɛ ½ O i ɛ ½ i U i ɛ ½
 @K L c | „ i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½
 „ q i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½
 • — • i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½
 i ɛ ½ i ɛ ½ \$ 6.1 i ɛ ½ d t " i ɛ ½ • |
 ½ i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½
 + ' ' ' python v { : i ɛ ½
 : Qwen2. 5-Math-7B • (i ɛ ½ i ɛ ½ i ɛ ½
 p f i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½
 i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½
 g L i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½
 i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½
 Zeng et al. (2025) i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½
 • " i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½
 6 i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½
 i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½
 ½ d i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½
 ½ h i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½
 i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½
 d i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½
 Φ " i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½
 ½ K L c | M , i ɛ ½ i ɛ ½ i ɛ ½
 Z q i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½
 i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½ i ɛ ½
 6

performance on O.O.D. benchmarks, with absolute gains of up to 2.9% on TheoremQA and 6.8% on MMLU-Pro-1k. In contrast, applying a rule-based verifier to unfiltered data results in the worst performance. This might be caused by its low training accuracy on free-form answers, while the model-based verifier achieves much better performance.

Takeaway 5.2 for RL with Noisy Verifiable Data

To obtain reward signals from noisy verifiable data, the ruled-based verifier after filtering the prompt set for short-form answers works the best. (Table 3)

6. Exploration on RL from the Base Model

DeepSeek-R1 (DeepSeek-AI, 2025) has demonstrated that long chain-of-thought reasoning can emerge by scaling up reinforcement learning compute on a base model. Recent studies (Zeng et al., 2025; Pan et al., 2025) have attempted to replicate this progress by running a relatively small number of RL iterations to observe the emergence of long CoT behavior (e.g., the “aha moment” (DeepSeek-AI, 2025), an emergent realization moment that enables critical functions like self-validation and correction). We also explore the method of RL from the base model in this section.

6.1. Nuances in Analysis Based on Emergent Behaviors

Self-validation behaviors are sometimes flagged as emergent behaviors or “aha-moment” by the model’s exploration, since such patterns are rare in short CoT data. However, we notice that sometimes self-validation behaviors already exist in the base model and reinforcing them through RL requires strict conditions, such as a strong base model.

Setup. We follow the setup from Zeng et al. (2025) to train Qwen2. 5-Math-7B using PPO with a rule-based verifier on approximately 8k MATH level 3-5 questions,

but we use our own rule-based verifier implementation. For inference, we adopt temperature $t = 0$ (greedy decoding), as our preliminary experiments show that $t = 0$ usually significantly outperforms $t > 0$ for models obtained by direct RL from `Owen2.5-Math-7B`. We use the maximum output length of 4096 tokens considering the training context length of 4096 tokens. Note that we use zero-shot prompting for the base model to avoid introducing biases to the output pattern. We select five representative keywords, “wait”, “recheck”, “alternatively”, “retry” and “however” from long CoT cases in previous works (OpenAI, 2024; DeepSeek-AI, 2025; Pan et al., 2025; Zeng et al., 2025), and calculate their frequencies to quantify the extent to which the model does self-validation. Further details about the RL hyperparameters can be found in Appendix E.5.9.

Result. Figure 7 shows that our RL from Qwen2. 5-Math-7B effectively boosts the accuracies, but does not increase the frequency of the “recheck” pattern existing in the output of the base model, nor effectively incentivize other reflection patterns such as “retry” and “alternatively”. This indicates that RL from the base model does not necessarily incentivize reflection patterns, though significantly boosting the performance. Sometimes such behaviors exist in the base model’s output and RL does not substantially enhance them. So we might need to be more careful about recognizing emergent behaviors.

6.2. Nuances in Analysis Based on Length Scaling

The length scaling up is recognized as another important feature of the effective exploration of the model. However, we notice that sometimes length scaling up can be accompanied by the KL divergence decreasing, which raises the possibility that the length is influenced by the KL penalty and is just reverting back to the base model’s longer outputs, rather than reflecting the acquisition of long CoT ability.

Setup. The setup is the same as in §6.1. Besides the

[illegible]

Figure 8. (Owen2.5-Math-7B : f`i 2 1/2 MATH-500 , " i 2 1/2 1/2 | E ‡ , " i 2 1/2 1/2 MATH Lv3-5 - i p n Ve i 2 1/2 1/2 < , KLc |

6.3. Potential Reasons Why Emergent Behavior is Not Observed with Qwen2. 5-Math-7B

[illegible]

6.4. Comparison between RL from the Base Model and RL from Long CoT SFT

[illegible][illegible]

(QwQ-32B-Preview 与 2.5-Math-7B 的对比实验结果如下：

- SFT 模型在 10 个任务上的平均得分是 6.1，而 RL 模型在 10 个任务上的平均得分是 8.5。
- CoT SFT 模型在 10 个任务上的平均得分是 8.5，而 RL 模型在 10 个任务上的平均得分是 8.5。
- CoT SFT 模型在 10 个任务上的平均得分是 8.5，而 RL 模型在 10 个任务上的平均得分是 8.5。

i 0.5 h 4 >: (Qwen2. 5-Math-7B i 0.5
 • CoT SFT! < i 0.5, RL > W i i 0.5 <
 i 0.5, RL v i 0.5 i 0.5 CoT SFT, «
 wSei 0.5 (i 0.5 V ±, • CoT SFT, RL
 s Gi 0.5 i 0.5 < i 0.5, RL i 0.5 > W, 8.7%
 v i 0.5 i 0.5 2.6% < — i 0.5 / i 0.5
 (i 0.5-32B-Preview, •, • CoT, SFT i 0.5
 i 0.5 : ' , ' i 0.5

output token length, we also calculate the “coding rate”. We classify the model’s output as “coding” if it contains the “`python`”, since Qwen2.5-Math-7B uses both natural language and coding to solve mathematical problems. Note that the “coding” output here is actually a special form of natural language output, where the code in it is not executed, and the code’s output is generated by the model.

Zeng et al. (2025) suggest that the initial drop may be due to the model transitioning from generating long coding outputs to shorter natural language outputs. However, Figure 8 (2) indicates that natural language outputs are actually longer than coding outputs, and the initial drop in length occurs in both types of output. Furthermore, Figure 8 (3) shows that the coding rate subsequently increases again, suggesting that the distinction between coding and natural language may not significantly impact the optimization process.

6.3. Potential Reasons Why Emergent Behavior is Not Observed with Qwen2. 5-Math-7B

Table 4. $\bar{y} \pm s$ Wen2, 5-Math-7B, $\frac{1}{2}$ < " ' $\bar{y} \pm s$ $\frac{1}{2}$ $\frac{1}{2}$ SFT pn / i QwQ-32B-Preview $\bar{y} \pm s$ $\frac{1}{2}$ $\frac{1}{2}$, • —
O"

[illegible]

v! i ½ GPT-4o t i ½ W M
6' Y y • „ i ½ U F.2.1 6 •
(MinHash—½ (Broder, 1997) (OpenWebMath(Paste
et al., 2023)- i ½ " i ½ n i ½
i ComamonCrawl(Rana, 2010)- i ½ e½
8 (Ž „ - i ½ i ½ ½ (" i ½ i ½ - >
(' ½ My v - * (7 K i ½ ½ ½ i ½
• i ½ CoT „ i ½ " i ½ i ½ ½ v
4 • i ½ U F.2.2 i ½ i ½
* E „ G i ½ • i ½ i ½ i ½ i ½ i ½
i i ½ i ½ O " i ½ i ½ OpenWebMath- „
8½ ne •

$$\begin{aligned} & \left(\frac{1}{2}, \frac{1}{2} \right) \otimes \frac{1}{2} = \frac{1}{2} \otimes \frac{1}{2} + \frac{1}{2} \otimes \frac{1}{2} = \frac{1}{2} \otimes \frac{1}{2} + \frac{1}{2} \otimes \frac{1}{2} \\ & = \frac{1}{2} \otimes \frac{1}{2} + \frac{1}{2} \otimes \frac{1}{2} = \frac{1}{2} \otimes \frac{1}{2} + \frac{1}{2} \otimes \frac{1}{2} \end{aligned}$$

$\bar{t} \in \frac{1}{2} \mathbb{Z} < \cdot$ / P 6 subsection 6.1 - $\bar{t} \in \frac{1}{2} \mathbb{Q}$
 $\mu, L: \bar{t} \in \frac{1}{2}; \bullet \bar{t} \in \frac{1}{2}$ Hyung Won Chung (Chung,
 2024) $\bar{t} \in \mathbb{Z} + \{ < \mu, \bar{t} \in \frac{1}{2} \}$ $\alpha: f, \mu, ! < \bar{t}$
 $\bar{t} \in \frac{1}{2} \mathbb{Z} \cup \frac{1}{2} \mathbb{Z} \otimes \bar{t} \in \frac{1}{2}$ / $\bullet V \bar{t} \in \frac{1}{2} \bar{t} \in$
 $\mu, ! \bar{t} \in \frac{1}{2}^* e, \forall \bar{t} \in \frac{1}{2} \mathbb{Z} \mathbb{Z} (\bar{t} \in \frac{1}{2} @ \frac{1}{2}$
 $< \mu: f$

(Γ $\frac{1}{2}$! < $\frac{1}{2}$! $\frac{1}{2}$! $\frac{1}{2}$! UO 32B! $\frac{1}{2}$
 + > W,, Γ $\frac{1}{2}$ @,, GPU p $\frac{1}{2}$ '
 $\frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{2}$! $\frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{2}$ • „ : f` F $\frac{1}{2}$ <
 OpenRLHF (Hu et al., 2024) 8O * ^ Γ
 - Γ $\frac{1}{2}$ Γ $\frac{1}{2}$ } „ Γ $\frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{2}$!
 p„ * o, « X” (...X- d $\frac{1}{2}$ PPO Γ $\frac{1}{2}$
 „ — $\frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{2}$ } K Γ $\frac{1}{2}$ z • O $\frac{1}{2}$ $\frac{1}{2}$
 eP6†H† Γ $\frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{2}$ N y
 + / (• CoT $\frac{1}{2}$ 1 Z CoT • | „ Γ $\frac{1}{2}$ $\frac{1}{2}$
 $\frac{1}{2}$ $\frac{1}{2}$ Γ $\frac{1}{2}$ $\frac{1}{2}$ $\frac{1}{2}$ (Kimi Team, 2025)
 Γ $\frac{1}{2}$: h f` $\frac{1}{2}$ $\frac{1}{2}$, Γ $\frac{1}{2}$ $\frac{1}{2}$ ©K
 $\frac{1}{2}$ $\frac{1}{2}$ • CoT $\frac{1}{2}$ $\frac{1}{2}$

π vs π_{old} : REINFORCE++ (Hu, 2025) \(\rightarrow\) PPO,
PPO > REINFORCE++
 $\nabla_{\theta} J(\theta)$

6.4. Comparison between RL from the Base Model and RL from Long CoT SFT

We compare the performance of RL from the base model and RL from long CoT SFT and find that RL from long CoT SFT generally performs better.

Setup. We compare using the base model Qwen2.5-Math-7B. The results of RL from the base model are from the model trained in §6.1. For RL from long CoT SFT, we adopt a setup similar to §3.2. Specifically, we choose the 7.5k MATH training set as the prompt set, curate the SFT data by rejection sampling with 32 candidate responses per prompt using QwQ-32B-Preview, and perform PPO using our cosine length-scaling reward with repetition penalty and our rule-based verifier, sampling 8 responses per prompt and training for 8 epochs. To adapt Qwen2.5-Math-7B with a pre-training context length of only 4096 tokens to long CoT SFT and RL, we multiply its RoPE (Su et al., 2024) by 10 times. We don’t report the results of RL with classic reward from long CoT SFT since it collapses. For evaluation, we adopt our default temperature sampling setup for RL from long CoT SFT as in §2.5 and greedy decoding setup for RL from the base model as in §6.1 for the best performance. Further details about the distillation, SFT hyperparameters and RL hyperparameters can be found in Appendix E.2 & E.3 & E.5.9, respectively.

Table 4. Performance of different models based on Qwen2.5-Math-7B. The SFT data here is distilled with rejection sampling from QwQ-32B-Preview.

| Setup | MATH
500 | AIME
2024 | Theo.
QA | MMLU
Pro-1k | AVG |
|---------------|-------------|--------------|-------------|----------------|-------------|
| Base (0-shot) | 52.0 | 13.3 | 17.1 | 2.4 | 21.2 |
| (Direct) RL | 77.4 | 23.3 | 43.5 | 19.7 | 41.0 |
| SFT | 84.0 | 24.4 | 42.2 | 38.5 | 47.3 |
| SFT + RL | 85.9 | 26.9 | 45.4 | 40.6 | 49.7 |

Result. Table 4 shows that, on Qwen2.5-Math-7B, RL initialized from the long CoT SFT model significantly outperforms RL from the base model and further

improves upon the long CoT SFT itself. Specifically, RL from long CoT SFT with our cosine reward surpasses RL from the base model by a substantial 8.7% on average and improves over the SFT initialization by 2.6%. Notably, simply applying SFT with long CoT distilled from QwQ-32B-Preview already yields strong performance.

6.5. Long CoT Patterns in Pre-training Data

Based on the results in §6.1, we hypothesize that incentivized behaviors, such as the model revisiting its solutions, may have already been partially learned during pre-training. To examine this, we employed two methods to investigate whether such data are already present on the web.

Firstly, we used a generative search engine Perplexity.ai to identify webpages explicitly containing problem-solving steps that approach problems from multiple angles or perform verification after providing an answer. The query we used and the examples we identified are in Appendix F.1).

Secondly, we used GPT-4o to generate a list of phrases that are characteristic of the “aha moment” (Appendix F.2.1), then used the MinHash algorithm (Broder, 1997) to search through OpenWebMath (Paster et al., 2023), a dataset filtered from the CommonCrawl (Rana, 2010) frequently used in pre-training. We found that there was a significant number of matches in discussion forum threads, where the dialogue between multiple users showed similarity to long CoT, with multiple approaches being discussed along with backtracking and error correction (Appendix F.2.2). This raises the intriguing possibility that long CoT originated from human dialogue, although we should also note that discussion forums are a common source of data in OpenWebMath.

Based on these observations, we hypothesize that RL primarily guides the model to recombine skills it already internalized during pre-training towards new behaviors to

7.4. Scaling up Verification

SimpleRL is a baseline for verification. It uses a simple rule-based verifier to check the correctness of the solution. The verifier is implemented as a function that takes a solution and returns a boolean value indicating whether it is correct. The function is defined as follows:

```
def simple_rl_verifier(solution):
    # Simple rule-based verification
    # ... (omitted code) ...
    return is_correct
```

DeepSeek-R1 is a more advanced verifier. It uses a more complex rule-based verifier that can handle more complex problems. The verifier is implemented as a function that takes a solution and returns a boolean value indicating whether it is correct. The function is defined as follows:

```
def deepseek_r1_verifier(solution):
    # DeepSeek-R1 rule-based verification
    # ... (omitted code) ...
    return is_correct
```

7.5. Latent Capabilities in Base Models

Broder, A. On the resemblance and containment of documents. In *Proceedings. Compression and Complexity of SEQUENCES 1997 (Cat. No. 97TB100171)*, pp. 21–29, 1997. doi: 10.1109/SEQUEN.1997.666900.

Broder, A. Z., Charikar, M., Frieze, A. M., and Mitzenmacher, M. Min-wise independent permutations. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pp. 327–336, 1998.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Chen, M., Tworek, J., Jun, H., Yuan, Q., de Oliveira Pinto, H. P., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., Ray, A., Puri, R., Krueger, G., Petrov, M., Khlaaf, H., Sastry, G., Mishkin, P., Chan, B., Gray, S., Ryder, N., Pavlov, M., Power, A., Kaiser, L., Bavarian, M., Winter, C., Tillett, P., Such, F. P., Cummings, D., Plappert, M., Chantzis, F., Barnes, E., Herbert-Voss, A., Guss, W. H., Nichol, A., Paino, A., Tezak, N., Tang, J., Babuschkin, I., Balaji, S., Jain,

Acknowledgment

Anthropic. Introducing claude, 2023. URL <https://www.anthropic.com/index/introducing-claude>.

Broder, A. On the resemblance and containment of documents. In *Proceedings. Compression and Complexity of SEQUENCES 1997 (Cat. No. 97TB100171)*, pp. 21–29, 1997. doi: 10.1109/SEQUEN.1997.666900.

Broder, A. Z., Charikar, M., Frieze, A. M., and Mitzenmacher, M. Min-wise independent permutations. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pp. 327–336, 1998.

Impact Statement

This work aims to improve the understanding of long chain-of-thought reasoning in LLMs. The findings suggest that long CoT may be learned from pre-training data, which could help in designing more effective training methods. The work also highlights the importance of verification in long CoT reasoning, which could be useful for improving the reliability of LLM outputs.

¹<http://www.incompleteideas.net/InIdeas/BitterLesson.html>

improve performance on complex problem-solving tasks. Given the broad scope of this paper, we leave a more in-depth investigation of this behavior to future work.

7. Discussions and Future Work

In this work, we demystify long CoT reasoning in LLMs. In this section, we outline potential future directions.

7.1. Scaling up Model Size

We believe that model size is the primary factor limiting the emergence of the behavior observed in subsection 6.1. Hyung Won Chung (Chung, 2024) recently shared a similar perspective, suggesting that smaller models may struggle to develop high-level reasoning skills and instead rely on heuristic-based pattern recognition. Future research could investigate RL using a larger base model.

7.2. RL Infrastructure Is Still in Its Infancy

While attempting to scale up the model size, we encountered significant challenges in expanding to 32B, ultimately determining that the required number of GPUs was too large to proceed. We observe that open-source RL frameworks (e.g., OpenRLHF (Hu et al., 2024)) often coordinate multiple systems optimized for different training and inference workloads, leading to multiple copies of model parameters being stored in memory. Additionally, algorithms like PPO alternate between these workloads synchronously and sequentially, further limiting efficiency. These factors contribute to low hardware utilization, an issue that is particularly exacerbated in long CoT scenarios due to the higher variance in CoT length, which leads to stragglers during inference (Kimi Team, 2025). We look forward to advancements in machine learning and systems research that will help overcome these limitations and accelerate progress in long CoT modeling.

7.3. REINFORCE Is More Tricky to Tune than PPO

We also explored REINFORCE++ (Hu, 2025) as a faster alternative to PPO for scaling up data. However, we found it to be significantly more unstable than PPO, leading to lower training accuracies (Figure 13). As this instability may be due to an untuned setup (Appendix E.5.10), we refrain from making general claims about the algorithm. We present this as an observation that may be useful to the community.

7.4. Scaling up Verification

While our findings demonstrate that combining rule-based verifiers with prompt set filtering is highly effective, designing such rules and curating prompt sets across different domains remains labor-intensive. More fundamentally, this approach embeds human-designed heuristics into the RL environment, reflecting how we think rather than allowing for emergent learning. As highlighted in The Bitter Lesson¹, manually encoding human intuition tends to be an inefficient long-term strategy. This raises an intriguing question: how can verification signals be scaled effectively? Is there an equivalent of pretraining in the context of designing RL environments? We look forward to future research on silver supervision signals and the potential for self-supervised approaches in RL verification.

7.5. Latent Capabilities in Base Models

Reasoning is a latent capability in base models that has only recently been unlocked. Our analysis suggests that one possible source of this emergent thinking is human dialogue on Internet discussion forums. This raises a broader question: what other abilities exist, waiting to be elicited from the vast reservoir of human knowledge and experience embedded in pre-training data? We look forward to more detailed analyses tracing model behaviors back to their data origins, which could yield new insights

¹<http://www.incompleteideas.net/Incldeas/BitterLesson.html>

S., Saunders, W., Hesse, C., Carr, A. N., Leike, J., Achiam, J., Misra, V., Morikawa, E., Radford, A., Knight, M., Brundage, M., Murati, M., Mayer, K., Welinder, P., McGrew, B., Amodei, D., McCandlish, S., Sutskever, I., and Zaremba, W. Evaluating large language models trained on code, 2021.

Chen, W., Yin, M., Ku, M., Lu, P., Wan, Y., Ma, X., Xu, J., Wang, X., and Xia, T. TheoremQA: A theorem-driven question answering dataset. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.

Chung, H. W. Don't teach. incentivize. Presentation slides, 2024. URL <https://t.co/2sjhynKxzJ>. Slide 48.

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Dao, T. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations (ICLR)*, 2024.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025.

Dong, H., Xiong, W., Goyal, D., Zhang, Y., Chow, W., Pan, R., Diao, S., Zhang, J., KaShun, S., and Zhang, T. Raft: Reward ranked finetuning for generative foundation model alignment. *Transactions on Machine Learning Research*, 2023.

Feng, X., Wan, Z., Wen, M., McAleer, S. M., Wen, Y., Zhang, W., and Wang, J. Alphazero-like tree-search can guide large language model decoding and training, 2023.

Gao, Z., Wang, H., Lu, C., Lu, T., Froudust-Walsh, S., Chen, M., Wang, X.-J., Hu, J., and Sun, W. The neural

basis of delayed gratification. *Science Advances*, 7(49):eabg6611, 2021. doi: 10.1126/sciadv.abg6611.

Gulcehre, C., Paine, T. L., Srinivasan, S., Konyushkova, K., Weerts, L., Sharma, A., Siddhant, A., Ahern, A., Wang, M., Gu, C., et al. Reinforced self-training (rest) for language modeling. *arXiv preprint arXiv:2308.08998*, 2023.

Hendrycks, D., Burns, C., Kadavath, S., Arora, A., Basart, S., Tang, E., Song, D., and Steinhardt, J. Measuring mathematical problem solving with the math dataset. In Vanschoren, J. and Yeung, S. (eds.), *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021.

Hou, Z., Lv, X., Lu, R., Zhang, J., Li, Y., Yao, Z., Li, J., Tang, J., and Dong, Y. Advancing language model reasoning through reinforcement learning and inference scaling, 2025.

Hu, J. Reinforce++: A simple and efficient approach for aligning large language models. *arXiv preprint arXiv:2501.03262*, 2025.

Hu, J., Wu, X., Zhu, Z., Xianyu, Wang, W., Zhang, D., and Cao, Y. Openrlhf: An easy-to-use, scalable and high-performance rlhf framework, 2024.

Jimenez, C. E., Yang, J., Wettig, A., Yao, S., Pei, K., Press, O., and Narasimhan, K. R. SWE-bench: Can language models resolve real-world github issues? In *The Twelfth International Conference on Learning Representations*, 2024.

Kimi Team. Kimi k1.5: Scaling reinforcement learning with llms, 2025.

Lamb, A. M., ALIAS PARTH GOYAL, A. G., Zhang, Y., Zhang, S., Courville, A. C., and Bengio, Y. Professor forcing: A new algorithm for training recurrent networks. In Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., and Garnett, R. (eds.), *Advances in Neural*

and help uncover hidden capabilities within base models.

Impact Statement

This paper aims to provide insights into scaling inference compute and training strategies to enable long chain-of-thought reasoning in large language models. The broader impacts of this work primarily relate to the potential for enhanced reasoning and problem-solving capabilities across various domains, where models capable of interpretable and verifiable reasoning could drive innovation and improve decision-making. Our findings emphasize the importance of ensuring robust training data preparation, stability, and alignment with verifiable ground truths. We encourage future research to actively develop safeguards that ensure these capabilities are used responsibly. This includes careful design of reward shaping and training protocols to minimize unintended consequences while maximizing societal benefits.

Acknowledgment

The authors would thank Yuanzhi Li for insightful discussions on this topic. The authors would also thank the SimpleRL team, particularly Weihao Zeng and Junxian He, for sharing their training experiences and experimental observations. Additionally, the authors appreciate Wenhui Chen, Xiaoyi Ren, Chao Li, Ziqiao Ma, Jiayi Pan, Xingyao Wang, and Seungone Kim for their valuable comments and discussions during the early or final stages of the project. Finally, the authors would acknowledge the DeepSeek-R1 and Kimi-k1.5 teams for their technical report releases, which inspired several additional experiment designs of this paper. This work was supported in part by a Carnegie Bosch Institute Fellowship to Xiang Yue.

1/2 ± .

Anthropic. Introducing claude, 2023. URL <https://www.anthropic.com/index/>

introducing-claude.

Broder, A. On the resemblance and containment of documents. In *Proceedings. Compression and Complexity of SEQUENCES 1997 (Cat. No.97TB100171)*, pp. 21–29, 1997. doi: 10.1109/SEQUEN.1997.666900.

Broder, A. Z., Charikar, M., Frieze, A. M., and Mitzenmacher, M. Min-wise independent permutations. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pp. 327–336, 1998.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Chen, M., Tworek, J., Jun, H., Yuan, Q., de Oliveira Pinto, H. P., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., Ray, A., Puri, R., Krueger, G., Petrov, M., Khlaaf, H., Sastry, G., Mishkin, P., Chan, B., Gray, S., Ryder, N., Pavlov, M., Power, A., Kaiser, L., Bavarian, M., Winter, C., Tillet, P., Such, F. P., Cummings, D., Plappert, M., Chantzis, F., Barnes, E., Herbert-Voss, A., Guss, W. H., Nichol, A., Paino, A., Tezak, N., Tang, J., Babuschkin, I., Balaji, S., Jain, S., Saunders, W., Hesse, C., Carr, A. N., Leike, J., Achiam, J., Misra, V., Morikawa, E., Radford, A., Knight, M., Brundage, M., Murati, M., Mayer, K., Welinder, P., McGrew, B., Amodei, D., McCandlish, S., Sutskever, I., and Zaremba, W. Evaluating large language models trained on code, 2021.

Chen, W., Yin, M., Ku, M., Lu, P., Wan, Y., Ma, X., Xu, J., Wang, X., and Xia, T. TheoremQA: A theorem-driven question answering dataset. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.

Chung, H. W. Don’t teach. incentivize. Presentation slides, 2024. URL <https://t.co/2sjhynKxzJ>. Slide 48.

Information Processing Systems, volume 29. Curran Associates, Inc., 2016.

Lambert, N., Morrison, J., Pyatkin, V., Huang, S., Ivison, H., Brahman, F., Miranda, L. J. V., Liu, A., Dziri, N., Lyu, S., Gu, Y., Malik, S., Graf, V., Hwang, J. D., Yang, J., Bras, R. L., Tafjord, O., Wilhelm, C., Soldaini, L., Smith, N. A., Wang, Y., Dasigi, P., and Hajishirzi, H. Tulu 3: Pushing frontiers in open language model post-training, 2024.

Lightman, H., Kosaraju, V., Burda, Y., Edwards, H., Baker, B., Lee, T., Leike, J., Schulman, J., Sutskever, I., and Cobbe, K. Let’s verify step by step. In *The Twelfth International Conference on Learning Representations*, 2024.

Longpre, S., Hou, L., Vu, T., Webson, A., Chung, H. W., Tay, Y., Zhou, D., Le, Q. V., Zoph, B., Wei, J., and Roberts, A. The flan collection: Designing data and methods for effective instruction tuning. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J. (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 22631–22648. PMLR, 23–29 Jul 2023.

Loshchilov, I. and Hutter, F. Sgdr: Stochastic gradient descent with warm restarts, 2017.

Meta. Introducing meta llama 3: The most capable openly available llm to date., 2024. URL <https://ai.meta.com/blog/meta-llama-3>.

OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

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

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L.,

Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J., and Lowe, R. Training language models to follow instructions with human feedback, 2022.

Pan, J., Zhang, J., Wang, X., and Yuan, L. Tinyzero, 2025. URL <https://github.com/Jiayi-Pan/TinyZero>. Accessed: 2025-01-24.

Paster, K., Santos, M. D., Azerbayev, Z., and Ba, J. Open-webmath: An open dataset of high-quality mathematical web text, 2023.

Qwen Team. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement, 2024a.

Qwen Team. Qwq: Reflect deeply on the boundaries of the unknown, 2024b. URL <https://qwenl.github.io/blog/qwq-32b-preview/>.

Rajbhandari, S., Rasley, J., Ruwase, O., and He, Y. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–16. IEEE, 2020.

Rana, A. Common crawl – building an open web-scale crawl using hadoop, 2010. URL <https://www.sliedeshare.net/hadoopusergroup/common-crawl-presentation>.

Rasley, J., Rajbhandari, S., Ruwase, O., and He, Y. Deep-speed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 3505–3506, 2020.

Rein, D., Hou, B. L., Stickland, A. C., Petty, J., Pang, R. Y., Dirani, J., Michael, J., and Bowman, S. R. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024.

| Demystifying Long Chain-of-Thought Reasoning in LLMs | |
|--|--|
| Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., et al. Training verifiers to solve math word problems. <i>arXiv preprint arXiv:2110.14168</i> , 2021. | Hou, Z., Lv, X., Lu, R., Zhang, J., Li, Y., Yao, Z., Li, J., Tang, J., and Dong, Y. Advancing language model reasoning through reinforcement learning and inference scaling, 2025. |
| Dao, T. FlashAttention-2: Faster attention with better parallelism and work partitioning. In <i>International Conference on Learning Representations (ICLR)</i> , 2024. | Hu, J. Reinforce++: A simple and efficient approach for aligning large language models. <i>arXiv preprint arXiv:2501.03262</i> , 2025. |
| DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. | Hu, J., Wu, X., Zhu, Z., Xianyu, Wang, W., Zhang, D., and Cao, Y. Openrlhf: An easy-to-use, scalable and high-performance rlhf framework, 2024. |
| Dong, H., Xiong, W., Goyal, D., Zhang, Y., Chow, W., Pan, R., Diao, S., Zhang, J., KaShun, S., and Zhang, T. Raft: Reward ranked finetuning for generative foundation model alignment. <i>Transactions on Machine Learning Research</i> , 2023. | Jimenez, C. E., Yang, J., Wettig, A., Yao, S., Pei, K., Press, O., and Narasimhan, K. R. SWE-bench: Can language models resolve real-world github issues? In <i>The Twelfth International Conference on Learning Representations</i> , 2024. |
| Feng, X., Wan, Z., Wen, M., McAleer, S. M., Wen, Y., Zhang, W., and Wang, J. Alphazero-like tree-search can guide large language model decoding and training, 2023. | Kimi Team. Kimi k1.5: Scaling reinforcement learning with llms, 2025. |
| Gao, Z., Wang, H., Lu, C., Lu, T., Froudust-Walsh, S., Chen, M., Wang, X.-J., Hu, J., and Sun, W. The neural basis of delayed gratification. <i>Science Advances</i> , 7 (49):eabg6611, 2021. doi: 10.1126/sciadv.abg6611. | Lamb, A. M., ALIAS PARTH GOYAL, A. G., Zhang, Y., Zhang, S., Courville, A. C., and Bengio, Y. Professor forcing: A new algorithm for training recurrent networks. In Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., and Garnett, R. (eds.), <i>Advances in Neural Information Processing Systems</i> , volume 29. Curran Associates, Inc., 2016. |
| Gulcehre, C., Paine, T. L., Srinivasan, S., Konyushkova, K., Weerts, L., Sharma, A., Siddhant, A., Ahern, A., Wang, M., Gu, C., et al. Reinforced self-training (rest) for language modeling. <i>arXiv preprint arXiv:2308.08998</i> , 2023. | Lambert, N., Morrison, J., Pyatkin, V., Huang, S., Ivison, H., Brahman, F., Miranda, L. J. V., Liu, A., Dziri, N., Lyu, S., Gu, Y., Malik, S., Graf, V., Hwang, J. D., Yang, J., Bras, R. L., Tafjord, O., Wilhelm, C., Soldaini, L., Smith, N. A., Wang, Y., Dasigi, P., and Hajishirzi, H. Tulu 3: Pushing frontiers in open language model post-training, 2024. |
| Hendrycks, D., Burns, C., Kadavath, S., Arora, A., Basart, S., Tang, E., Song, D., and Steinhardt, J. Measuring mathematical problem solving with the math dataset. In Vanschoren, J. and Yeung, S. (eds.), <i>Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks</i> , volume 1, 2021. | Lightman, H., Kosaraju, V., Burda, Y., Edwards, H., Baker, B., Lee, T., Leike, J., Schulman, J., Sutskever, I., and Cobbe, K. Let’s verify step by step. In <i>The Twelfth International Conference on Learning Representations</i> , 2024. |

| Title Suppressed Due to Excessive Size | |
|---|---|
| Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms, 2017. | Wang, Z., Li, Y., Wu, Y., Luo, L., Hou, L., Yu, H., and Shang, J. Multi-step problem solving through a verifier: An empirical analysis on model-induced process supervision, 2024b. |
| Singh, A., Co-Reyes, J. D., Agarwal, R., Anand, A., Patil, P., Liu, P. J., Harrison, J., Lee, J., Xu, K., Parisi, A., et al. Beyond human data: Scaling self-training for problem-solving with language models. <i>arXiv preprint arXiv:2312.06585</i> , 2023. | Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q. V., and Zhou, D. Chain-of-thought prompting elicits reasoning in large language models. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A. (eds.), <i>Advances in Neural Information Processing Systems</i> , volume 35, pp. 24824–24837. Curran Associates, Inc., 2022. |
| Snell, C., Lee, J., Xu, K., and Kumar, A. Scaling llm test-time compute optimally can be more effective than scaling model parameters, 2024. | Yu, L., Jiang, W., Shi, H., YU, J., Liu, Z., Zhang, Y., Kwok, J., Li, Z., Weller, A., and Liu, W. Metamath: Bootstrap your own mathematical questions for large language models. In <i>The Twelfth International Conference on Learning Representations</i> , 2024. |
| Sutton, R. S. and Barto, A. G. <i>Reinforcement Learning: An Introduction</i> . A Bradford Book, Cambridge, MA, USA, 2018. ISBN 0262039249. | Yuan, Z., Yuan, H., Li, C., Dong, G., Tan, C., and Zhou, C. Scaling relationship on learning mathematical reasoning with large language models. <i>arXiv preprint arXiv:2308.01825</i> , 2023. |
| Tong, Y., Zhang, X., Wang, R., Wu, R., and He, J. DART-math: Difficulty-aware rejection tuning for mathematical problem-solving. In <i>The Thirty-eighth Annual Conference on Neural Information Processing Systems</i> , 2024. | Yue, X., Qu, X., Zhang, G., Fu, Y., Huang, W., Sun, H., Su, Y., and Chen, W. MAMmoTH: Building math generalist models through hybrid instruction tuning. In <i>The Twelfth International Conference on Learning Representations</i> , 2024a. |
| Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., and Lample, G. Llama: Open and efficient foundation language models, 2023. | Yue, X., Zheng, T., Zhang, G., and Chen, W. Mammoth2: Scaling instructions from the web. <i>NeurIPS 2024</i> , 2024b. |
| Wang, Y., Ma, X., Zhang, G., Ni, Y., Chandra, A., Guo, S., Ren, W., Arulraj, A., He, X., Jiang, Z., Li, T., Ku, M., Wang, K., Zhuang, A., Fan, R., Yue, X., and Chen, W. MMLU-pro: A more robust and challenging multi-task language understanding benchmark. In <i>The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track</i> , 2024a. | Zelikman, E., Wu, Y., Mu, J., and Goodman, N. Star: Bootstrapping reasoning with reasoning. <i>Advances in Neural Information Processing Systems</i> , 35:15476–15488, 2022. |
| | Zeng, W., Huang, Y., Liu, W., He, K., Liu, Q., Ma, Z., and He, J. 7b model and 8k examples: Emerging reasoning with reinforcement learning is both effective and efficient. https://hkust-nlp.netlify.com/sitemap/simplerl-reason , 2025. Notion Blog. |

| Demystifying Long Chain-of-Thought Reasoning in LLMs | |
|---|---|
| Longpre, S., Hou, L., Vu, T., Webson, A., Chung, H. W., Tay, Y., Zhou, D., Le, Q. V., Zoph, B., Wei, J., and Roberts, A. The flan collection: Designing data and methods for effective instruction tuning. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J. (eds.), <i>Proceedings of the 40th International Conference on Machine Learning</i> , volume 202 of <i>Proceedings of Machine Learning Research</i> , pp. 22631–22648. PMLR, 23–29 Jul 2023. | Qwen Team. Qwq: Reflect deeply on the boundaries of the unknown, 2024b. URL https://qwenl.m.gi.thub.i o/bl og/qwq-32b-previ ew/ . |
| Loshchilov, I. and Hutter, F. Sgdr: Stochastic gradient descent with warm restarts, 2017. | Rajbhandari, S., Rasley, J., Ruwase, O., and He, Y. Zero: Memory optimizations toward training trillion parameter models. In <i>SC20: International Conference for High Performance Computing, Networking, Storage and Analysis</i> , pp. 1–16. IEEE, 2020. |
| Meta. Introducing meta llama 3: The most capable openly available llm to date., 2024. URL https://ai .meta. com/bl og/meta-l l ama-3 . | Rana, A. Common crawl – building an open web-scale crawl using hadoop, 2010. URL https://www. sl i deshare. net/hadoopusergroup/ common-crawl presentati on . |
| OpenAI. Gpt-4 technical report. <i>arXiv preprint arXiv:2303.08774</i> , 2023. | Rasley, J., Rajbhandari, S., Ruwase, O., and He, Y. Deep-speed: System optimizations enable training deep learning models with over 100 billion parameters. In <i>Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining</i> , pp. 3505–3506, 2020. |
| OpenAI. Learning to reason with llms, 2024. URL https://openai . com/i ndex/ l earni ng-to-reason-wi th-l l ms/ . | Rein, D., Hou, B. L., Stickland, A. C., Petty, J., Pang, R. Y., Dirani, J., Michael, J., and Bowman, S. R. GPQA: A graduate-level google-proof q&a benchmark. In <i>First Conference on Language Modeling</i> , 2024. |
| Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J., and Lowe, R. Training language models to follow instructions with human feedback, 2022. | Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms, 2017. |
| Pan, J., Zhang, J., Wang, X., and Yuan, L. Tinyzero, 2025. URL https://gi thub. com/ Ji ayi -Pan/Ti nyZero . Accessed: 2025-01-24. | Singh, A., Co-Reyes, J. D., Agarwal, R., Anand, A., Patil, P., Liu, P. J., Harrison, J., Lee, J., Xu, K., Parisi, A., et al. Beyond human data: Scaling self-training for problem-solving with language models. <i>arXiv preprint arXiv:2312.06585</i> , 2023. |
| Paster, K., Santos, M. D., Azerbayev, Z., and Ba, J. Open-webmath: An open dataset of high-quality mathematical web text, 2023. | Snell, C., Lee, J., Xu, K., and Kumar, A. Scaling llm test-time compute optimally can be more effective than scaling model parameters, 2024. |
| Qwen Team. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement, 2024a. | Su, J., Ahmed, M., Lu, Y., Pan, S., Bo, W., and Liu, Y. Roformer: Enhanced transformer with rotary position embedding. <i>Neurocomputing</i> , 568:127063, 2024. |

| Title Suppressed Due to Excessive Size |
|---|
| Zheng, L., Yin, L., Xie, Z., Sun, C., Huang, J., Yu, C. H., Cao, S., Kozyrakis, C., Stoica, I., Gonzalez, J. E., Barrett, C., and Sheng, Y. SGLang: Efficient execution of structured language model programs. In <i>The Thirty-eighth Annual Conference on Neural Information Processing Systems</i> , 2024. |

Sutton, R. S. and Barto, A. G. *Reinforcement Learning: An Introduction*. A Bradford Book, Cambridge, MA, USA, 2018. ISBN 0262039249.

Tong, Y., Zhang, X., Wang, R., Wu, R., and He, J. DART-math: Difficulty-aware rejection tuning for mathematical problem-solving. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., and Lample, G. Llama: Open and efficient foundation language models, 2023.

Wang, Y., Ma, X., Zhang, G., Ni, Y., Chandra, A., Guo, S., Ren, W., Arulraj, A., He, X., Jiang, Z., Li, T., Ku, M., Wang, K., Zhuang, A., Fan, R., Yue, X., and Chen, W. MMLU-pro: A more robust and challenging multi-task language understanding benchmark. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024a.

Wang, Z., Li, Y., Wu, Y., Luo, L., Hou, L., Yu, H., and Shang, J. Multi-step problem solving through a verifier: An empirical analysis on model-induced process supervision, 2024b.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, b., Xia, F., Chi, E., Le, Q. V., and Zhou, D. Chain-of-thought prompting elicits reasoning in large language models. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A. (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 24824–24837. Curran Associates, Inc., 2022.

Yu, L., Jiang, W., Shi, H., YU, J., Liu, Z., Zhang, Y., Kwok, J., Li, Z., Weller, A., and Liu, W. Metamath: Bootstrap your own mathematical questions for large language models. In *The Twelfth International Conference on Learning Representations*, 2024.

Yuan, Z., Yuan, H., Li, C., Dong, G., Tan, C., and Zhou, C. Scaling relationship on learning mathematical reasoning with large language models. *arXiv preprint arXiv:2308.01825*, 2023.

Yue, X., Qu, X., Zhang, G., Fu, Y., Huang, W., Sun, H., Su, Y., and Chen, W. MAMmoTH: Building math generalist models through hybrid instruction tuning. In *The Twelfth International Conference on Learning Representations*, 2024a.

Yue, X., Zheng, T., Zhang, G., and Chen, W. Mammoth2: Scaling instructions from the web. *NeurIPS 2024*, 2024b.

Zelikman, E., Wu, Y., Mu, J., and Goodman, N. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35:15476–15488, 2022.

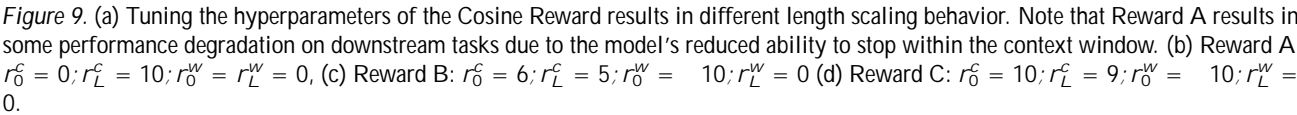
Zeng, W., Huang, Y., Liu, W., He, K., Liu, Q., Ma, Z., and He, J. 7b model and 8k examples: Emerging reasoning with reinforcement learning is both effective and efficient. <https://hkust-nlp.github.io/notebooks/simplerl-reason>, 2025. Notion Blog.

Zheng, L., Yin, L., Xie, Z., Sun, C., Huang, J., Yu, C. H., Cao, S., Kozyrakis, C., Stoica, I., Gonzalez, J. E., Barrett, C., and Sheng, Y. SGLang: Efficient execution of structured language model programs. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

A. Related Work

Besides the above-mentioned works, there are many other studies on the scaling of LLMs. Wei et al. (2022) proposed a scaling law for LLMs, showing that the performance of LLMs scales with the number of parameters, the number of training steps, and the number of data points. This law has been widely used to guide the design of LLMs. Other studies (Lambert et al., 2024; Wei et al., 2022; Longpre et al., 2023; Yu et al., 2024) have also explored the scaling of LLMs from different perspectives. For example, OpenAI (2024) released a report on the scaling of GPT-4, showing that the performance of GPT-4 scales with the number of parameters, the number of training steps, and the number of data points. This report has also been widely used to guide the design of LLMs.

In this paper, we focus on the scaling of LLMs for reasoning. We first review the existing works on the scaling of LLMs for reasoning. Then, we propose a new scaling law for LLMs for reasoning, which shows that the performance of LLMs for reasoning scales with the number of parameters, the number of training steps, and the number of data points. This law is different from the existing scaling laws for LLMs, as it takes into account the reasoning ability of LLMs. We then evaluate the performance of LLMs for reasoning using our scaling law. The results show that our scaling law is more accurate than the existing scaling laws for LLMs.



| Correctness Discount | Repetition Discount | MATH -500 | AIME 2024 | Theo. QA | MMLU -Pro-1k |
|----------------------|---------------------|-------------|------------|-------------|--------------|
| SFT | | 50.4 | 3.5 | 20.6 | 32.4 |
| 1.000 | 1.000 | 55.7 | 5.0 | 25.7 | 34.5 |
| | 0.999 | 58.0 | 4.6 | 26.0 | 36.5 |
| | 0.99 | 57.8 | 3.8 | 24.5 | 33.3 |
| 0.999 | 0.999 | 53.5 | 2.1 | 19.5 | 30.7 |
| | 0.99 | 55.2 | 1.7 | 18.5 | 32.0 |
| 0.99 | 0.99 | 47.9 | 0.2 | 15.6 | 25.5 |

Figure 12. Training response length of models trained with Cosine Reward with and without repetition penalty. We see that repetition penalty reduced the length.

Figure 13. Reinforce with classic reward shows signs of training instability.

Table 5. Performance of model trained with different discount factors for the correctness (cosine) reward and repetition penalty. We see that different reward types have different optimal values.

| Correctness Discount | Repetition Discount | MATH -500 | AIME 2024 | Theo. QA | MMLU -Pro-1k |
|----------------------|---------------------|-----------|-----------|----------|--------------|
| | SFT | 50.4 | 3.5 | 20.6 | 32.4 |
| 1.000 | 1.000 | 55.7 | 5.0 | 25.7 | 34.5 |
| | 0.999 | 58.0 | 4.6 | 26.0 | 36.5 |
| | 0.99 | 57.8 | 3.8 | 24.5 | 33.3 |
| 0.999 | 0.999 | 53.5 | 2.1 | 19.5 | 30.7 |
| | 0.99 | 55.2 | 1.7 | 18.5 | 32.0 |
| 0.99 | 0.99 | 47.9 | 0.2 | 15.6 | 25.5 |

C. Algorithms and Formulas

C.1. Cosine Reward Formula

CosFn(t; T; min; max) = min + 1/2 * (max - min) * (1 + cos(t/T)) (1)

l = 1/2 * (max - min) * (1 + cos(t/T)) (Loshchilov & Hutter, 2017)

C.2. N-gram Repetition Penalty

Algorithm 1 N-gram Repetition Penalty

1: **Input:**

2: s : sequence of tokens

3: l : sequence length

4: N : n-gram size

5: P : penalty value

6: m : maximum sequence length

7: **Output:** r ∈ ℝ^m

8: seq ← s[1 : l] *f*Extract subsequence of length l

9: ngrams ← ∅ *f*Set of observed n-grams

10: r ← 0 ∈ ℝ^m *f*Initialize reward vector

11: **for** j = 1 to seq.length - N + 1 **do**

12: ng ← (seq[j], seq[j + 1], ..., seq[j + N - 1]) *f*Current n-gram

13: **if** ng ∈ ngrams **then**

14: **for** t = j to j + N - 1 **do**

15: r[t] ← P *f*Apply penalty

16: **end for**

17: **end if**

18: ngrams ← ngrams ∪ {ng}

19: **end for**

20: **Output:** r

C. Algorithms and Formulas

C.1. Cosine Reward Formula

$$\text{CosFn}(t; T; \text{min}; \text{max}) = \text{min} + \frac{1}{2}(\text{max} - \text{min})(1 + \cos(\frac{t}{T})) \tag{1}$$

The formula above is commonly used as the learning rate schedule during gradient descent optimization. It was introduced by (Loshchilov & Hutter, 2017).

C.2. N-gram Repetition Penalty

```
Algorithm 1 N-gram Repetition Penalty
1: Input:
2:    $s$  : sequence of tokens
3:    $l$  : sequence length
4:    $N$  : n-gram size
5:    $P$  : penalty value
6:    $m$  : maximum sequence length
7: Output:  $r \in \mathbb{R}^m$ 
8:  $seq \leftarrow s[1 : l]$   $\triangleright$  Extract subsequence of length  $l$ 
9:  $ngrams \leftarrow \emptyset$  ;  $\triangleright$  Set of observed n-grams
10:  $r \leftarrow \mathbf{0} \in \mathbb{R}^m$   $\triangleright$  Initialize reward vector
11: for  $j \leftarrow 1$  to  $j \leftarrow seq.length - N + 1$  do
12:    $ng \leftarrow (seq[j]; seq[j + 1]; \dots; seq[j + N - 1])$   $\triangleright$  Current n-gram
13:   if  $ng \notin ngrams$  then
14:     for  $t \leftarrow j$  to  $j + N - 1$  do
15:        $r[t] \leftarrow -P$   $\triangleright$  Apply penalty
16:     end for
17:   end if
18:    $ngrams \leftarrow ngrams \cup \{ng\}$ 
19: end for
20: Output:  $r$ 
```

D. Extracts

Short-Term Thinking ($\epsilon = 0.99$)

But in this case, we saw that the fixed points don't satisfy the equation

$$p(p(x)) = x:$$

Wait, maybe the solutions to $p(p(x)) = x$ are not the fixed points of p , but something else.

Alternatively, perhaps I can consider that $p(p(x)) = x$ implies that p is a bijection, but since $p(x)$ is quadratic, it's not necessarily invertible over all real numbers.

Wait, but the problem says that $p(p(x)) = x$ has four real solutions, so perhaps there are four values of x that satisfy this equation.

Alternatively, perhaps I need to find the roots of

$$p(p(x)) - x = 0;$$

and solve for x . But that seems complicated. Maybe there's a better way.

D. Extracts

Short-Term Thinking ($p_c = 0.99$)

But in this case, we saw that the fixed points don't satisfy the equation

$$p(p(x)) = x:$$

Wait, maybe the solutions to $p(p(x)) = x$ are not the fixed points of p , but something else.

Alternatively, perhaps I can consider that $p(p(x)) = x$ implies that p is a bijection, but since $p(x)$ is quadratic, it's not necessarily invertible over all real numbers.

Wait, but the problem says that $p(p(x)) = x$ has four real solutions, so perhaps there are four values of x that satisfy this equation.

Alternatively, perhaps I need to find the roots of

$$p(p(x)) - x = 0;$$

and solve for x . But that seems complicated. Maybe there's a better way.

E. Experimental Setup

E.1. Evaluation Setup

- Table 1: Performance of MATH-500, AIME 2024, TheoremQA, and MMLU-Pro-1k across different models and configurations.
- MATH-500** (Hendrycks et al., 2021): Performance on the MATH-500 dataset. The table shows scores for various models, including GPT-4, GPT-3.5, and others, with scores ranging from 12,500 to 12,500.
 - AIME 2024**: Performance on the AIME 2024 dataset. The table shows scores for various models, including GPT-4, GPT-3.5, and others, with scores ranging from 30 to 30.
 - TheoremQA** (Chen et al., 2023): Performance on the TheoremQA dataset. The table shows scores for various models, including GPT-4, GPT-3.5, and others, with scores ranging from 5 to 5.
 - MMLU-Pro-1k** (Wang et al., 2024a): Performance on the MMLU-Pro-1k dataset. The table shows scores for various models, including GPT-4, GPT-3.5, and others, with scores ranging from 12,000 to 12,000.

Figure 14. MMLU-Pro K₁ vs. MMLU-Pro-1k P₁ vs. P₂ vs. P₃ vs. P₄ vs. P₅ vs. P₆ vs. P₇ vs. P₈ vs. P₉ vs. P₁₀ vs. P₁₁ vs. P₁₂ vs. P₁₃ vs. P₁₄ vs. P₁₅ vs. P₁₆ vs. P₁₇ vs. P₁₈ vs. P₁₉ vs. P₂₀ vs. P₂₁ vs. P₂₂ vs. P₂₃ vs. P₂₄ vs. P₂₅ vs. P₂₆ vs. P₂₇ vs. P₂₈ vs. P₂₉ vs. P₃₀ vs. P₃₁ vs. P₃₂ vs. P₃₃ vs. P₃₄ vs. P₃₅ vs. P₃₆ vs. P₃₇ vs. P₃₈ vs. P₃₉ vs. P₄₀ vs. P₄₁ vs. P₄₂ vs. P₄₃ vs. P₄₄ vs. P₄₅ vs. P₄₆ vs. P₄₇ vs. P₄₈ vs. P₄₉ vs. P₅₀ vs. P₅₁ vs. P₅₂ vs. P₅₃ vs. P₅₄ vs. P₅₅ vs. P₅₆ vs. P₅₇ vs. P₅₈ vs. P₅₉ vs. P₆₀ vs. P₆₁ vs. P₆₂ vs. P₆₃ vs. P₆₄ vs. P₆₅ vs. P₆₆ vs. P₆₇ vs. P₆₈ vs. P₆₉ vs. P₇₀ vs. P₇₁ vs. P₇₂ vs. P₇₃ vs. P₇₄ vs. P₇₅ vs. P₇₆ vs. P₇₇ vs. P₇₈ vs. P₇₉ vs. P₈₀ vs. P₈₁ vs. P₈₂ vs. P₈₃ vs. P₈₄ vs. P₈₅ vs. P₈₆ vs. P₈₇ vs. P₈₈ vs. P₈₉ vs. P₉₀ vs. P₉₁ vs. P₉₂ vs. P₉₃ vs. P₉₄ vs. P₉₅ vs. P₉₆ vs. P₉₇ vs. P₉₈ vs. P₉₉ vs. P₁₀₀ vs. P₁₀₁ vs. P₁₀₂ vs. P₁₀₃ vs. P₁₀₄ vs. P₁₀₅ vs. P₁₀₆ vs. P₁₀₇ vs. P₁₀₈ vs. P₁₀₉ vs. P₁₁₀ vs. P₁₁₁ vs. P₁₁₂ vs. P₁₁₃ vs. P₁₁₄ vs. P₁₁₅ vs. P₁₁₆ vs. P₁₁₇ vs. P₁₁₈ vs. P₁₁₉ vs. P₁₂₀ vs. P₁₂₁ vs. P₁₂₂ vs. P₁₂₃ vs. P₁₂₄ vs. P₁₂₅ vs. P₁₂₆ vs. P₁₂₇ vs. P₁₂₈ vs. P₁₂₉ vs. P₁₃₀ vs. P₁₃₁ vs. P₁₃₂ vs. P₁₃₃ vs. P₁₃₄ vs. P₁₃₅ vs. P₁₃₆ vs. P₁₃₇ vs. P₁₃₈ vs. P₁₃₉ vs. P₁₄₀ vs. P₁₄₁ vs. P₁₄₂ vs. P₁₄₃ vs. P₁₄₄ vs. P₁₄₅ vs. P₁₄₆ vs. P₁₄₇ vs. P₁₄₈ vs. P₁₄₉ vs. P₁₅₀ vs. P₁₅₁ vs. P₁₅₂ vs. P₁₅₃ vs. P₁₅₄ vs. P₁₅₅ vs. P₁₅₆ vs. P₁₅₇ vs. P₁₅₈ vs. P₁₅₉ vs. P₁₆₀ vs. P₁₆₁ vs. P₁₆₂ vs. P₁₆₃ vs. P₁₆₄ vs. P₁₆₅ vs. P₁₆₆ vs. P₁₆₇ vs. P₁₆₈ vs. P₁₆₉ vs. P₁₇₀ vs. P₁₇₁ vs. P₁₇₂ vs. P₁₇₃ vs. P₁₇₄ vs. P₁₇₅ vs. P₁₇₆ vs. P₁₇₇ vs. P₁₇₈ vs. P₁₇₉ vs. P₁₈₀ vs. P₁₈₁ vs. P₁₈₂ vs. P₁₈₃ vs. P₁₈₄ vs. P₁₈₅ vs. P₁₈₆ vs. P₁₈₇ vs. P₁₈₈ vs. P₁₈₉ vs. P₁₉₀ vs. P₁₉₁ vs. P₁₉₂ vs. P₁₉₃ vs. P₁₉₄ vs. P₁₉₅ vs. P₁₉₆ vs. P₁₉₇ vs. P₁₉₈ vs. P₁₉₉ vs. P₂₀₀ vs. P₂₀₁ vs. P₂₀₂ vs. P₂₀₃ vs. P₂₀₄ vs. P₂₀₅ vs. P₂₀₆ vs. P₂₀₇ vs. P₂₀₈ vs. P₂₀₉ vs. P₂₁₀ vs. P₂₁₁ vs. P₂₁₂ vs. P₂₁₃ vs. P₂₁₄ vs. P₂₁₅ vs. P₂₁₆ vs. P₂₁₇ vs. P₂₁₈ vs. P₂₁₉ vs. P₂₂₀ vs. P₂₂₁ vs. P₂₂₂ vs. P₂₂₃ vs. P₂₂₄ vs. P₂₂₅ vs. P₂₂₆ vs. P₂₂₇ vs. P₂₂₈ vs. P₂₂₉ vs. P₂₃₀ vs. P₂₃₁ vs. P₂₃₂ vs. P₂₃₃ vs. P₂₃₄ vs. P₂₃₅ vs. P₂₃₆ vs. P₂₃₇ vs. P₂₃₈ vs. P₂₃₉ vs. P₂₄₀ vs. P₂₄₁ vs. P₂₄₂ vs. P₂₄₃ vs. P₂₄₄ vs. P₂₄₅ vs. P₂₄₆ vs. P₂₄₇ vs. P₂₄₈ vs. P₂₄₉ vs. P₂₅₀ vs. P₂₅₁ vs. P₂₅₂ vs. P₂₅₃ vs. P₂₅₄ vs. P₂₅₅ vs. P₂₅₆ vs. P₂₅₇ vs. P₂₅₈ vs. P₂₅₉ vs. P₂₆₀ vs. P₂₆₁ vs. P₂₆₂ vs. P₂₆₃ vs. P₂₆₄ vs. P₂₆₅ vs. P₂₆₆ vs. P₂₆₇ vs. P₂₆₈ vs. P₂₆₉ vs. P₂₇₀ vs. P₂₇₁ vs. P₂₇₂ vs. P₂₇₃ vs. P₂₇₄ vs. P₂₇₅ vs. P₂₇₆ vs. P₂₇₇ vs. P₂₇₈ vs. P₂₇₉ vs. P₂₈₀ vs. P₂₈₁ vs. P₂₈₂ vs. P₂₈₃ vs. P₂₈₄ vs. P₂₈₅ vs. P₂₈₆ vs. P₂₈₇ vs. P₂₈₈ vs. P₂₈₉ vs. P₂₉₀ vs. P₂₉₁ vs. P₂₉₂ vs. P₂₉₃ vs. P₂₉₄ vs. P₂₉₅ vs. P₂₉₆ vs. P₂₉₇ vs. P₂₉₈ vs. P₂₉₉ vs. P₃₀₀ vs. P₃₀₁ vs. P₃₀₂ vs. P₃₀₃ vs. P₃₀₄ vs. P₃₀₅ vs. P₃₀₆ vs. P₃₀₇ vs. P₃₀₈ vs. P₃₀₉ vs. P₃₁₀ vs. P₃₁₁ vs. P₃₁₂ vs. P₃₁₃ vs. P₃₁₄ vs. P₃₁₅ vs. P₃₁₆ vs. P₃₁₇ vs. P₃₁₈ vs. P₃₁₉ vs. P₃₂₀ vs. P₃₂₁ vs. P₃₂₂ vs. P₃₂₃ vs. P₃₂₄ vs. P₃₂₅ vs. P₃₂₆ vs. P₃₂₇ vs. P₃₂₈ vs. P₃₂₉ vs. P₃₃₀ vs. P₃₃₁ vs. P₃₃₂ vs. P₃₃₃ vs. P₃₃₄ vs. P₃₃₅ vs. P₃₃₆ vs. P₃₃₇ vs. P₃₃₈ vs. P₃₃₉ vs. P₃₄₀ vs. P₃₄₁ vs. P₃₄₂ vs. P₃₄₃ vs. P₃₄₄ vs. P₃₄₅ vs. P₃₄₆ vs. P₃₄₇ vs. P₃₄₈ vs. P₃₄₉ vs. P₃₅₀ vs. P₃₅₁ vs. P₃₅₂ vs. P₃₅₃ vs. P₃₅₄ vs. P₃₅₅ vs. P₃₅₆ vs. P₃₅₇ vs. P₃₅₈ vs. P₃₅₉ vs. P₃₆₀ vs. P₃₆₁ vs. P₃₆₂ vs. P₃₆₃ vs. P₃₆₄ vs. P₃₆₅ vs. P₃₆₆ vs. P₃₆₇ vs. P₃₆₈ vs. P₃₆₉ vs. P₃₇₀ vs. P₃₇₁ vs. P₃₇₂ vs. P₃₇₃ vs. P₃₇₄ vs. P₃₇₅ vs. P₃₇₆ vs. P₃₇₇ vs. P₃₇₈ vs. P₃₇₉ vs. P₃₈₀ vs. P₃₈₁ vs. P₃₈₂ vs. P₃₈₃ vs. P₃₈₄ vs. P₃₈₅ vs. P₃₈₆ vs. P₃₈₇ vs. P₃₈₈ vs. P₃₈₉ vs. P₃₉₀ vs. P₃₉₁ vs. P₃₉₂ vs. P₃₉₃ vs. P₃₉₄ vs. P₃₉₅ vs. P₃₉₆ vs. P₃₉₇ vs. P₃₉₈ vs. P₃₉₉ vs. P₄₀₀ vs. P₄₀₁ vs. P₄₀₂ vs. P₄₀₃ vs. P₄₀₄ vs. P₄₀₅ vs. P₄₀₆ vs. P₄₀₇ vs. P₄₀₈ vs. P₄₀₉ vs. P₄₁₀ vs. P₄₁₁ vs. P₄₁₂ vs. P₄₁₃ vs. P₄₁₄ vs. P₄₁₅ vs. P₄₁₆ vs. P₄₁₇ vs. P₄₁₈ vs. P₄₁₉ vs. P₄₂₀ vs. P₄₂₁ vs. P₄₂₂ vs. P₄₂₃ vs. P₄₂₄ vs. P₄₂₅ vs. P₄₂₆ vs. P₄₂₇ vs. P₄₂₈ vs. P₄₂₉ vs. P₄₃₀ vs. P₄₃₁ vs. P₄₃₂ vs. P₄₃₃ vs. P₄₃₄ vs. P₄₃₅ vs. P₄₃₆ vs. P₄₃₇ vs. P₄₃₈ vs. P₄₃₉ vs. P₄₄₀ vs. P₄₄₁ vs. P₄₄₂ vs. P₄₄₃ vs. P₄₄₄ vs. P₄₄₅ vs. P₄₄₆ vs. P₄₄₇ vs. P₄₄₈ vs. P₄₄₉ vs. P₄₅₀ vs. P₄₅₁ vs. P₄₅₂ vs. P₄₅₃ vs. P₄₅₄ vs. P₄₅₅ vs. P₄₅₆ vs. P₄₅₇ vs. P₄₅₈ vs. P₄₅₉ vs. P₄₆₀ vs. P₄₆₁ vs. P₄₆₂ vs. P₄₆₃ vs. P₄₆₄ vs. P₄₆₅ vs. P₄₆₆ vs. P₄₆₇ vs. P₄₆₈ vs. P₄₆₉ vs. P₄₇₀ vs. P₄₇₁ vs. P₄₇₂ vs. P₄₇₃ vs. P₄₇₄ vs. P₄₇₅ vs. P₄₇₆ vs. P₄₇₇ vs. P₄₇₈ vs. P₄₇₉ vs. P₄₈₀ vs. P₄₈₁ vs. P₄₈₂ vs. P₄₈₃ vs. P₄₈₄ vs. P₄₈₅ vs. P₄₈₆ vs. P₄₈₇ vs. P₄₈₈ vs. P₄₈₉ vs. P₄₉₀ vs. P₄₉₁ vs. P₄₉₂ vs. P₄₉₃ vs. P₄₉₄ vs. P₄₉₅ vs. P₄₉₆ vs. P₄₉₇ vs. P₄₉₈ vs. P₄₉₉ vs. P₅₀₀ vs. P₅₀₁ vs. P₅₀₂ vs. P₅₀₃ vs. P₅₀₄ vs. P₅₀₅ vs. P₅₀₆ vs. P₅₀₇ vs. P₅₀₈ vs. P₅₀₉ vs. P₅₁₀ vs. P₅₁₁ vs. P₅₁₂ vs. P₅₁₃ vs. P₅₁₄ vs. P₅₁₅ vs. P₅₁₆ vs. P₅₁₇ vs. P₅₁₈ vs. P₅₁₉ vs. P₅₂₀ vs. P₅₂₁ vs. P₅₂₂ vs. P₅₂₃ vs. P₅₂₄ vs. P₅₂₅ vs. P₅₂₆ vs. P₅₂₇ vs. P₅₂₈ vs. P₅₂₉ vs. P₅₃₀ vs. P₅₃₁ vs. P₅₃₂ vs. P₅₃₃ vs. P₅₃₄ vs. P₅₃₅ vs. P₅₃₆ vs. P₅₃₇ vs. P₅₃₈ vs. P₅₃₉ vs. P₅₄₀ vs. P₅₄₁ vs. P₅₄₂ vs. P₅₄₃ vs. P₅₄₄ vs. P₅₄₅ vs. P₅₄₆ vs. P₅₄₇ vs. P₅₄₈ vs. P₅₄₉ vs. P₅₅₀ vs. P₅₅₁ vs. P₅₅₂ vs. P₅₅₃ vs. P₅₅₄ vs. P₅₅₅ vs. P₅₅₆ vs. P₅₅₇ vs. P₅₅₈ vs. P₅₅₉ vs. P₅₆₀ vs. P₅₆₁ vs. P₅₆₂ vs. P₅₆₃ vs. P₅₆₄ vs. P₅₆₅ vs. P₅₆₆ vs. P₅₆₇ vs. P₅₆₈ vs. P₅₆₉ vs. P₅₇₀ vs. P₅₇₁ vs. P₅₇₂ vs. P₅₇₃ vs. P₅₇₄ vs. P₅₇₅ vs. P₅₇₆ vs. P₅₇₇ vs. P₅₇₈ vs. P₅₇₉ vs. P₅₈₀ vs. P₅₈₁ vs. P₅₈₂ vs. P₅₈₃ vs. P₅₈₄ vs. P₅₈₅ vs. P₅₈₆ vs. P₅₈₇ vs. P₅₈₈ vs. P₅₈₉ vs. P₅₉₀ vs. P₅₉₁ vs. P₅₉₂ vs. P₅₉₃ vs. P₅₉₄ vs. P₅₉₅ vs. P₅₉₆ vs. P₅₉₇ vs. P₅₉₈ vs. P₅₉₉ vs. P₆₀₀ vs. P₆₀₁ vs. P₆₀₂ vs. P₆₀₃ vs. P₆₀₄ vs. P₆₀₅ vs. P₆₀₆ vs. P₆₀₇ vs. P₆₀₈ vs. P₆₀₉ vs. P₆₁₀ vs. P₆₁₁ vs. P₆₁₂ vs. P₆₁₃ vs. P₆₁₄ vs. P₆₁₅ vs. P₆₁₆ vs. P₆₁₇ vs. P₆₁₈ vs. P₆₁₉ vs. P₆₂₀ vs. P₆₂₁ vs. P₆₂₂ vs. P₆₂₃ vs. P₆₂₄ vs. P₆₂₅ vs. P₆₂₆ vs. P₆₂₇ vs. P₆₂₈ vs. P₆₂₉ vs. P₆₃₀ vs. P₆₃₁ vs. P₆₃₂ vs. P₆₃₃ vs. P₆₃₄ vs. P₆₃₅ vs. P₆₃₆ vs. P₆₃₇ vs. P₆₃₈ vs. P₆₃₉ vs. P₆₄₀ vs. P₆₄₁ vs. P₆₄₂ vs. P₆₄₃ vs. P₆₄₄ vs. P₆₄₅ vs. P₆₄₆ vs. P₆₄₇ vs. P₆₄₈ vs. P₆₄₉ vs. P₆₅₀ vs. P₆₅₁ vs. P₆₅₂ vs. P₆₅₃ vs. P₆₅₄ vs. P₆₅₅ vs. P₆₅₆ vs. P₆₅₇ vs. P₆₅₈ vs. P₆₅₉ vs. P₆₆₀ vs. P₆₆₁ vs. P₆₆₂ vs. P₆₆₃ vs. P₆₆₄ vs. P₆₆₅ vs. P₆₆₆ vs. P₆₆₇ vs. P₆₆₈ vs. P₆₆₉ vs. P₆₇₀ vs. P₆₇₁ vs. P₆₇₂ vs. P₆₇₃ vs. P₆₇₄ vs. P₆₇₅ vs. P₆₇₆ vs. P₆₇₇ vs. P₆₇₈ vs. P₆₇₉ vs. P₆₈₀ vs. P₆₈₁ vs. P₆₈₂ vs. P₆₈₃ vs. P₆₈₄ vs. P₆₈₅ vs. P₆₈₆ vs. P₆₈₇ vs. P₆₈₈ vs. P₆₈₉ vs. P₆₉₀ vs. P₆₉₁ vs. P₆₉₂ vs. P₆₉₃ vs. P₆₉₄ vs. P₆₉₅ vs. P₆₉₆ vs. P₆₉₇ vs. P₆₉₈ vs. P₆₉₉ vs. P₇₀₀ vs. P₇₀₁ vs. P₇₀₂ vs. P₇₀₃ vs. P₇₀₄ vs. P₇₀₅ vs. P₇₀₆ vs. P₇₀₇ vs. P₇₀₈ vs. P₇₀₉ vs. P₇₁₀ vs. P₇₁₁ vs. P₇₁₂ vs. P₇₁₃ vs. P₇₁₄ vs. P₇₁₅ vs. P₇₁₆ vs. P₇₁₇ vs. P₇₁₈ vs. P₇₁₉ vs. P₇₂₀ vs. P₇₂₁ vs. P₇₂₂ vs. P₇₂₃ vs. P₇₂₄ vs. P₇₂₅ vs. P₇₂₆ vs. P₇₂₇ vs. P₇₂₈ vs. P₇₂₉ vs. P₇₃₀ vs. P₇₃₁ vs. P₇₃₂ vs. P₇₃₃ vs. P₇₃₄ vs. P₇₃₅ vs. P₇₃₆ vs. P₇₃₇ vs. P₇₃₈ vs. P₇₃₉ vs. P₇₄₀ vs. P₇₄₁ vs. P₇₄₂ vs. P₇₄₃ vs. P₇₄₄ vs. P₇₄₅ vs. P₇₄₆ vs. P₇₄₇ vs. P₇₄₈ vs. P₇₄₉ vs. P₇₅₀ vs. P₇₅₁ vs. P₇₅₂ vs. P₇₅₃ vs. P₇₅₄ vs. P₇₅₅ vs. P₇₅₆ vs. P₇₅₇ vs. P₇₅₈ vs. P₇₅₉ vs. P₇₆₀ vs. P₇₆₁ vs. P₇₆₂ vs. P₇₆₃ vs. P₇₆₄ vs. P₇₆₅ vs. P₇₆₆ vs. P₇₆₇ vs. P₇₆₈ vs. P₇₆₉ vs. P₇₇₀ vs. P₇₇₁ vs. P₇₇₂ vs. P₇₇₃ vs. P₇₇₄ vs. P₇₇₅ vs. P₇₇₆ vs. P₇₇₇ vs. P₇₇₈ vs. P₇₇₉ vs. P₇₈₀ vs. P₇₈₁ vs. P₇₈₂ vs. P₇₈₃ vs. P₇₈₄ vs. P₇₈₅ vs. P₇₈₆ vs. P₇₈₇ vs. P₇₈₈ vs. P₇₈₉ vs. P₇₉₀ vs. P₇₉₁ vs. P₇₉₂ vs. P₇₉₃ vs. P₇₉₄ vs. P₇₉₅ vs. P₇₉₆ vs. P₇₉₇ vs. P₇₉₈ vs. P₇₉₉ vs. P₈₀₀ vs. P₈₀₁ vs. P₈₀₂ vs. P₈₀₃ vs. P₈₀₄ vs. P₈₀₅ vs. P₈₀₆ vs. P₈₀₇ vs. P₈₀₈ vs. P₈₀₉ vs. P₈₁₀ vs. P₈₁₁ vs. P₈₁₂ vs. P₈₁₃ vs. P₈₁₄ vs. P₈₁₅ vs. P₈₁₆ vs. P₈₁₇ vs. P₈₁₈ vs. P₈₁₉ vs. P₈₂₀ vs. P₈₂₁ vs. P₈₂₂ vs. P₈₂₃ vs. P₈₂₄ vs. P₈₂₅ vs. P₈₂₆ vs. P₈₂₇ vs. P₈₂₈ vs. P₈₂₉ vs. P₈₃₀ vs. P₈₃₁ vs. P₈₃₂ vs. P₈₃₃ vs. P₈₃₄ vs. P₈₃₅ vs. P₈₃₆ vs. P₈₃₇ vs. P₈₃₈ vs. P₈₃₉ vs. P₈₄₀ vs. P₈₄₁ vs. P₈₄₂ vs. P₈₄₃ vs. P₈₄₄ vs. P₈₄₅ vs. P₈₄₆ vs. P₈₄₇ vs. P₈₄₈ vs. P₈₄₉ vs. P₈₅₀ vs. P₈₅₁ vs. P₈₅₂ vs. P₈₅₃ vs. P₈₅₄ vs. P₈₅₅ vs. P₈₅₆ vs. P₈₅₇ vs. P₈₅₈ vs. P₈₅₉ vs. P₈₆₀ vs. P₈₆₁ vs. P₈₆₂ vs. P₈₆₃ vs. P₈₆₄ vs. P₈₆₅ vs. P₈₆₆ vs. P₈₆₇ vs. P₈₆₈ vs. P₈₆₉ vs. P₈₇₀ vs. P₈₇₁ vs. P₈₇₂ vs. P₈₇₃ vs. P₈₇₄ vs. P₈₇₅ vs. P₈₇₆ vs. P₈₇₇ vs. P₈₇₈ vs. P₈₇₉ vs. P₈₈₀ vs. P₈₈₁ vs. P₈₈₂ vs. P₈₈₃ vs. P₈₈₄ vs. P₈₈₅ vs. P₈₈₆ vs. P₈₈₇ vs. P₈₈₈ vs. P₈₈₉ vs. P₈₉₀ vs. P₈₉₁ vs. P₈₉₂ vs. P₈₉₃ vs. P₈₉₄ vs. P₈₉₅ vs. P₈₉₆ vs. P₈₉₇ vs. P₈₉₈ vs. P₈₉₉ vs. P₉₀₀ vs. P₉₀₁ vs. P₉₀₂ vs. P₉₀₃ vs. P₉₀₄ vs. P₉₀₅ vs. P₉₀₆ vs. P₉₀₇ vs. P₉₀₈ vs. P₉₀₉ vs. P₉₁₀ vs. P₉₁₁ vs. P₉₁₂ vs. P₉₁₃ vs. P₉₁₄ vs. P₉₁₅ vs. P₉₁₆ vs. P₉₁₇ vs. P₉₁₈ vs. P₉₁₉ vs. P₉₂₀ vs. P₉₂₁ vs. P₉₂₂ vs. P₉₂₃ vs. P₉₂₄ vs. P₉₂₅ vs. P₉₂₆ vs. P₉₂₇ vs. P₉₂₈ vs. P₉₂₉ vs. P₉₃₀ vs. P₉₃₁ vs. P₉₃₂ vs. P₉₃₃ vs. P₉₃₄ vs. P₉₃₅ vs. P₉₃₆ vs. P₉₃₇ vs. P₉₃₈ vs. P₉₃₉ vs. P₉₄₀ vs. P₉₄₁ vs. P₉₄₂ vs. P₉₄₃ vs. P₉₄₄ vs. P₉₄₅ vs. P₉₄₆ vs. P₉₄₇ vs. P₉₄₈ vs. P₉₄₉ vs. P₉₅₀ vs. P₉₅₁ vs. P₉₅₂ vs. P₉₅₃ vs. P₉₅₄ vs. P₉₅₅ vs. P₉₅₆ vs. P₉₅₇ vs. P₉₅₈ vs. P₉₅₉ vs. P₉₆₀ vs. P₉₆₁ vs. P₉₆₂ vs. P₉₆₃ vs. P₉₆₄ vs. P₉₆₅ vs

Statistical Metrics We calculate the average accuracy with at least 4 random seeds. To tame the variance caused by the small size of AIME 2024, we sample 16 responses per prompt.

Implementation We adopt the vLLM library to accelerate the inference and SymEval², an elaborate answer grader capable of processing complex mathematical objects like matrices and functions, keeping consistent with the sampling and reward implementation in our RL setup. Note that a few RL experiments are carried out with an earlier version of the grader, causing nuanced performance differences.

E.2. Details about Distillation

To distill long CoT trajectories from QwQ-32B-Preview, we adopt the temperature $t = 1.0$, the top- p value of 0.95 and the maximum output length of 8192 tokens. Our preliminary experiments show that 8192 tokens show almost the same accuracy with QwQ-32B-Preview on MATH-500 as 16384 tokens, while costing significantly less time.

To distill short CoT trajectories from Qwen2.5-Math-72B-Instruct, we adopt the temperature $t = 0.7$, the top- p value of 0.95 and the maximum output length of 4096 tokens, since Qwen2.5-Math-72B-Instruct has a context limit of 4096 tokens and our preliminary experiments observe a non-negligible ratio of nonsense output when using $t = 1.0$.

Note the data is distilled with SGLang (Zheng et al., 2024) with an early version of our code.

When applying rejection sampling, we adopt the SymEval verifier as the grader.

E.3. Details about SFT Setup

We use OpenRLHF (Hu et al., 2024) for our SFT experiments. By default, we adopt the SFT hyperparameters in Table 6.

For efficiency, we utilize Flash Attention 2 (Dao, 2024) and ZeRO (Rajbhandari et al., 2020) based on the DeepSpeed library (Rasley et al., 2020). We uniformly set the micro batch size as 1 since we don’t observe acceleration when increasing it.

| Table 6. SFT Hyperparameters | | | |
|------------------------------|----------------|------|--------|
| Batch Size | Context Length | LR | Epochs |
| 256 | 128K | 5e-6 | 2 |

E.4. Details about RL Setup

We use OpenRLHF (Hu et al., 2024) for our RL experiments. When describing hyperparameters, we adopt the same naming conventions as OpenRLHF.

²<https://github.com/tongyx361/symeval>

| Table 7. Hyperparameters | | | | | | | | | |
|--------------------------|---------------|-------|----------|---------|-----|--------|----------------------------|-----------------------------|------|
| Base Model | Rewards | GAE | Episodes | Samples | BS | Epochs | Context Length | LR | KL |
| Llama3.1-8B | Cosine: | | | | | | | | |
| | $r_0^c = +2$ | | | | | | | | |
| | $r_L^c = +1$ | | | | | | | | |
| | $r_0^w = 10$ | $= 1$ | 4 | 4 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6 | 0.01 |
| | $r_L^w = 0$ | $= 1$ | | | | | | | |
| | $r_e = 10$ | | | | | | | | |
| | Rep. Penalty: | | | | | | | | |
| | $P = 0.05$ | | | | | | | | |
| | $N = 40$ | | | | | | | | |

E.5.2. DETAILS OF SECTION 4.1 (CoT LENGTH STABILITY)

SFT prompt QwQ-32B-Preview - $\tilde{t} = 1.0$, $\tilde{p} = 0.95$ CoT prompt (MATH - $\tilde{t} = 0.7$, $\tilde{p} = 0.95$)

| Table 8. Hyperparameters | | | | | | | | | |
|--------------------------|-------------|----------------|----------|---------|-----|--------|----------------------------|-------------------------------|------|
| Base Model | Rewards | GAE | Episodes | Samples | BS | Epochs | Context Length | LR | KL |
| Llama3.1-8B | Correct: +1 | $= 1$
$= 1$ | 8 | 8 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 4.5e-6 | 0.01 |
| Qwen2.5-Math-7B | Correct: +1 | $= 1$
$= 1$ | 8 | 8 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 4.5e-6 | 0.01 |

E.5. Experiment Hyperparameters

Note that the BS column below refers to both `rollout_batch_size` (the number of prompts used in a sampling-training iteration) and `train_batch_size` (the number of samples used in a training update) because we adopt the same number for these two hyperparameters in most of our RL setups. Also, the `Samples` column refers to the number of samples per prompt.

E.5.1. DETAILS OF SECTION 3.2 (SFT INITIALIZATION FOR RL)

SFT Data: CoT data distilled from QwQ-32B-Preview or Qwen2.5-Math-72B-Instruct with the MATH train split with different number of candidate responses per prompt.

Table 7. Hyperparameters

| Base Model | Rewards | GAE | Episodes | Samples | BS | Epochs | Context Length | LR | KL |
|-------------|--|----------------|----------|---------|-----|--------|----------------------------|-----------------------------|------|
| Llama3.1-8B | Cosine:
$r_0^C = +2$
$r_L^C = +1$
$r_0^W = 10$
$r_L^W = 0$
$r_e = 10$ | $= 1$
$= 1$ | 4 | 4 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6 | 0.01 |
| | Rep. Penalty:
$P = 0.05$
$N = 40$ | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |

E.5.2. DETAILS OF SECTION 4.1 (CoT LENGTH STABILITY)

SFT Data: Long CoT data distilled from QwQ-32B-Preview with the MATH train split.

Table 8. Hyperparameters

| Base Model | Rewards | GAE | Episodes | Samples | BS | Epochs | Context Length | LR | KL |
|-----------------|-------------|----------------|----------|---------|-----|--------|----------------------------|-------------------------------|------|
| Llama3.1-8B | Correct: +1 | $= 1$
$= 1$ | 8 | 8 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 4.5e-6 | 0.01 |
| Qwen2.5-Math-7B | Correct: +1 | $= 1$
$= 1$ | 8 | 8 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 4.5e-6 | 0.01 |

E.5.3. DETAILS OF SECTION 4.2 (ACTIVE SCALING OF CoT LENGTH)

SFT Data: QwQ-32B-Preview with $\gamma = 1/2$ • CoT prompt • (MATH - $\gamma = 1/2$)

Table 9. Hyperparameters

| Base Model | Rewards | GAE | Episodes | Samples | BS | Epochs | Context Length | LR | KL |
|-------------|--|--------------------------------|----------|---------|-----|--------|----------------------------|-------------------------------|------|
| Llama3.1-8B | Correct: +1 | $= 1$
$= 1$ | 8 | 8 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 4.5e-6 | 0.01 |
| Llama3.1-8B | Cosine:
$r_0^C = +2$
$r_L^C = +1$
$r_0^W = 10$
$r_L^W = 0$
$r_e = 10$ | $= 1$
$= 1$ | 8 | 8 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 4.5e-6 | 0.01 |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| Llama3.1-8B | Correct: +1 | $= 1$
$= 1$ | 8 | 16 | 512 | 2 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6 | 0.01 |
| Llama3.1-8B | Cosine:
$r_0^C = +2$
$r_L^C = +1$
$r_0^W = 10$
$r_L^W = 0$
$r_e = 10$ | $= 1$
$= 1$ | 8 | 16 | 512 | 2 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6 | 0.01 |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| Llama3.1-8B | Cosine:
$r_0^C = +2$
$r_L^C = +1$
$r_0^W = 10$
$r_L^W = 0$
$r_e = 10$ | $= 1$
$c = 1$
$p = 0.99$ | 8 | 16 | 512 | 2 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6 | 0.01 |
| | Rep. Penalty:
$P = 0.05$
$N = 40$ | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |

Demystifying Long Chain-of-Thought Reasoning in LLMs

E.5.3. DETAILS OF SECTION 4.2 (ACTIVE SCALING OF CoT LENGTH)

SFT Data: Long CoT data distilled from QwQ-32B-Preview with the MATH train split.

Table 9. Hyperparameters

| Base Model | Rewards | GAE | Episodes | Samples | BS | Epochs | Context Length | LR | KL |
|-------------|---|--|----------|---------|-----|--------|----------------------------|-------------------------------|------|
| Llama3.1-8B | Correct: +1 | $\gamma = 1$
$\beta = 1$ | 8 | 8 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 4.5e-6 | 0.01 |
| Llama3.1-8B | Cosine:
$r_0^C = +2$
$r_L^C = +1$
$r_0^W = 10$
$r_L^W = 0$
$r_e = 10$ | $\gamma = 1$
$\beta = 1$ | 8 | 8 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 4.5e-6 | 0.01 |
| Llama3.1-8B | Correct: +1 | $\gamma = 1$
$\beta = 1$ | 8 | 16 | 512 | 2 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6 | 0.01 |
| Llama3.1-8B | Cosine:
$r_0^C = +2$
$r_L^C = +1$
$r_0^W = 10$
$r_L^W = 0$
$r_e = 10$ | $\gamma = 1$
$\beta = 1$ | 8 | 16 | 512 | 2 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6 | 0.01 |
| Llama3.1-8B | Cosine:
$r_0^C = +2$
$r_L^C = +1$
$r_0^W = 10$
$r_L^W = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $\gamma = 1$
$\beta = 1$
$\rho = 0.99$ | 8 | 16 | 512 | 2 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6 | 0.01 |

Title Suppressed Due to Excessive Size

E.5.4. DETAILS OF SECTION 4.3 (COSINE REWARD HYPERPARAMETERS)

SFT prompt: QwQ-32B-Preview - $\frac{1}{2}$ CoT prompt • (MATH - $\frac{1}{2}$)

Table 10. Hyperparameters

| Base Model | Rewards | GAE | Episodes | Samples | BS | Epochs | Context Length | LR | KL |
|-------------|--|--|----------|---------|-----|--------|----------------------------|-----------------------------|------|
| Llama3.1-8B | Cosine:
$r_0^C = 0$
$r_L^C = +10$
$r_0^W = 0$
$r_L^W = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $\gamma = 1$
$\beta_c = 1$
$\rho = 0.99$ | 4 | 4 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6 | 0.01 |
| Llama3.1-8B | Cosine:
$r_0^C = +6$
$r_L^C = +5$
$r_0^W = 10$
$r_L^W = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $\gamma = 1$
$\beta_c = 1$
$\rho = 0.99$ | 4 | 4 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6 | 0.01 |
| Llama3.1-8B | Cosine:
$r_0^C = +10$
$r_L^C = +9$
$r_0^W = 10$
$r_L^W = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $\gamma = 1$
$\beta_c = 1$
$\rho = 0.99$ | 4 | 4 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6 | 0.01 |

E.5.4. DETAILS OF SECTION 4.3 (COSINE REWARD HYPERPARAMETERS)

SFT Data: Long CoT data distilled from QwQ-32B-Preview with the MATH train split.

Table 10. Hyperparameters

| Base Model | Rewards | GAE | Episodes | Samples | BS | Epochs | Context Length | LR | KL |
|-------------|--|--|----------|---------|-----|--------|----------------------------|-----------------------------|------|
| Llama3.1-8B | Cosine:
$r_0^c = 0$
$r_L^c = +10$
$r_0^w = 0$
$r_L^w = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $\gamma = 1$
$c = 1$
$\rho = 0.99$ | 4 | 4 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6 | 0.01 |
| | Cosine:
$r_0^c = +6$
$r_L^c = +5$
$r_0^w = 10$
$r_L^w = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $\gamma = 1$
$c = 1$
$\rho = 0.99$ | 4 | 4 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6 | 0.01 |
| | Cosine:
$r_0^c = +10$
$r_L^c = +9$
$r_0^w = 10$
$r_L^w = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $\gamma = 1$
$c = 1$
$\rho = 0.99$ | 4 | 4 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6 | 0.01 |

E.5.5. DETAILS OF SECTION 4.4 (CONTEXT WINDOW SIZE)

SFT prompt length = QwQ-32B-Preview - $\frac{1}{2}$ CoT prompt • (MATH - $\frac{1}{2}$)

Table 11. Hyperparameters

| Base Model | Rewards | GAE | Episodes | Samples | BS | Epochs | Context Length | LR | KL |
|-------------|---|--|----------|---------|-----|--------|----------------------------|-----------------------------|------|
| Llama3.1-8B | Cosine:
$r_0^c = +2$
$r_L^c = +1$
$r_0^w = 10$
$r_L^w = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $\gamma = 1$
$c = 1$
$\rho = 0.99$ | 8 | 8 | 512 | 1 | Prompt: 2048
Gen: 2048 | Actor: 5e-7
Critic: 9e-6 | 0.01 |
| | Cosine:
$r_0^c = +2$
$r_L^c = +1$
$r_0^w = 10$
$r_L^w = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $\gamma = 1$
$c = 1$
$\rho = 0.99$ | 8 | 8 | 512 | 1 | Prompt: 2048
Gen: 6144 | Actor: 5e-7
Critic: 9e-6 | 0.01 |
| | Cosine:
$r_0^c = +2$
$r_L^c = +1$
$r_0^w = 10$
$r_L^w = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $\gamma = 1$
$c = 1$
$\rho = 0.99$ | 8 | 8 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6 | 0.01 |

E.5.6. DETAILS OF SECTION 4.5 (LENGTH REWARD HACKING)

SFT prompt length = QwQ-32B-Preview - $\frac{1}{2}$ • CoT prompt • (MATH - $\frac{1}{2}$)

E.5.5. DETAILS OF SECTION 4.4 (CONTEXT WINDOW SIZE)

SFT Data: Long CoT data distilled from QwQ-32B-Preview with the MATH train split.

Table 11. Hyperparameters

| Base Model | Rewards | GAE | Episodes | Samples | BS | Epochs | Context Length | LR | KL |
|-------------|---|-----------------------------------|----------|---------|-----|--------|----------------------------|-----------------------------|------|
| Llama3.1-8B | Cosine:
$r_0^c = +2$
$r_L^c = +1$
$r_0^w = 10$
$r_L^w = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $= 1$
$c = 1$
$\rho = 0.99$ | 8 | 8 | 512 | 1 | Prompt: 2048
Gen: 2048 | Actor: 5e-7
Critic: 9e-6 | 0.01 |
| | Cosine:
$r_0^c = +2$
$r_L^c = +1$
$r_0^w = 10$
$r_L^w = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $= 1$
$c = 1$
$\rho = 0.99$ | 8 | 8 | 512 | 1 | Prompt: 2048
Gen: 6144 | Actor: 5e-7
Critic: 9e-6 | 0.01 |
| | Cosine:
$r_0^c = +2$
$r_L^c = +1$
$r_0^w = 10$
$r_L^w = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $= 1$
$c = 1$
$\rho = 0.99$ | 8 | 8 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6 | 0.01 |

E.5.6. DETAILS OF SECTION 4.5 (LENGTH REWARD HACKING)

SFT Data: Long CoT data distilled from QwQ-32B-Preview with the MATH train split.

Table 12. Hyperparameters

| Base Model | Rewards | GAE | Episodes | Samples | BS | Epochs | Context Length | LR | KL |
|-------------|---|-----------------------------------|----------|---------|-----|--------|----------------------------|-----------------------------|------|
| Llama3.1-8B | Cosine:
$r_0^c = +2$
$r_L^c = +1$
$r_0^w = 10$
$r_L^w = 0$
$r_e = 10$ | $= 1$
$= 1$ | 8 | 16 | 512 | 2 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6 | 0.01 |
| | Cosine:
$r_0^c = +2$
$r_L^c = +1$
$r_0^w = 10$
$r_L^w = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $= 1$
$c = 1$
$\rho = 0.99$ | 8 | 16 | 512 | 2 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6 | 0.01 |
| | Cosine:
$r_0^c = +2$
$r_L^c = +1$
$r_0^w = 10$
$r_L^w = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $= 1$
$c = 1$
$\rho = 0.99$ | 8 | 16 | 512 | 2 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6 | 0.01 |

E.5.7. DETAILS OF SECTION 4.6 (OPTIMAL DISCOUNT FACTORS)

SFT prompt QwQ-32B-Preview - 1/2 CoT prompt • (MATH - 1/2 1/2

E.5.8. DETAILS OF SECTION 5.2 (RL WITH NOISY VERIFIABLE DATA)

SFT prompt in QW-32B-Preview - in QW-Instruct - in LLaMA-3.1-70B-CoT prompt - [in LLaMA-3.1-70B-CoT

Table 13. Hyperparameters

| Base Model | Rewards | GAE | Episodes | Samples | BS | Epochs | Context Length | LR | KL |
|-------------|---|---|----------|---------|-----|--------|----------------------------|-----------------------------|------|
| Llama3.1-8B | Cosine:
$r_0^c = +2$
$r_L^c = +1$
$r_0^w = 10$
$r_L^w = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $\gamma = 1$
$c = 1$
$\rho = 1$ | 4 | 4 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6 | 0.01 |
| | Cosine:
$r_0^c = +2$
$r_L^c = +1$
$r_0^w = 10$
$r_L^w = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $\gamma = 1$
$c = 1$
$\rho = 0.999$ | 4 | 4 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6 | 0.01 |
| | Cosine:
$r_0^c = +2$
$r_L^c = +1$
$r_0^w = 10$
$r_L^w = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $\gamma = 1$
$c = 1$
$\rho = 0.99$ | 4 | 4 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6 | 0.01 |
| | Cosine:
$r_0^c = +2$
$r_L^c = +1$
$r_0^w = 10$
$r_L^w = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $\gamma = 1$
$c = 0.999$
$\rho = 0.999$ | 4 | 4 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6 | 0.01 |
| | Cosine:
$r_0^c = +2$
$r_L^c = +1$
$r_0^w = 10$
$r_L^w = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $\gamma = 1$
$c = 0.999$
$\rho = 0.99$ | 4 | 4 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6 | 0.01 |
| Llama3.1-8B | Cosine:
$r_0^c = +2$
$r_L^c = +1$
$r_0^w = 10$
$r_L^w = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $\gamma = 1$
$c = 0.99$
$\rho = 0.99$ | 4 | 4 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6 | 0.01 |

Table 14. Hyperparameters

| Base Model | RL Prompt Set Verifier | Rewards | GAE | Episodes Instances | Samples | BS | Epochs | Context Length | LR KL |
|-------------|---------------------------------------|---|--|--------------------|---------|-----|--------|----------------------------|---|
| Llama3.1-8B | Unfiltered (30k sampled) Symeval | Cosine:
$r_0^c = +2$
$r_L^c = +1$
$r_0^w = 10$
$r_L^w = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $\gamma = 1$
$c = 1$
$\rho = 0.99$ | 1
30k instances | 4 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6
KL: 0.01 |
| | | Cosine:
$r_0^c = +2$
$r_L^c = +1$
$r_0^w = 10$
$r_L^w = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $\gamma = 1$
$c = 1$
$\rho = 0.99$ | 1
30k instances | 4 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6
KL: 0.01 |
| | | Cosine:
$r_0^c = +2$
$r_L^c = +1$
$r_0^w = 10$
$r_L^w = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $\gamma = 1$
$c = 1$
$\rho = 0.99$ | 1
30k instances | 4 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6
KL: 0.01 |
| Llama3.1-8B | Filtered (30k sampled) Symeval | Cosine:
$r_0^c = +2$
$r_L^c = +1$
$r_0^w = 10$
$r_L^w = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $\gamma = 1$
$c = 1$
$\rho = 0.99$ | 1
30k instances | 4 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6
KL: 0.01 |
| Llama3.1-8B | Filtered (30k sampled) LLM-as-a-judge | Cosine:
$r_0^c = +2$
$r_L^c = +1$
$r_0^w = 10$
$r_L^w = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $\gamma = 1$
$c = 1$
$\rho = 0.99$ | 1
30k instances | 4 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6
KL: 0.01 |

SFT Data: 115k filtered from 462k instances of long CoT data distilled from QwQ-32B-Preview with WebInstruct

| Base Model | RL Prompt Set Verifier | Rewards | GAE | Episodes Instances | Samples | BS | Epochs | Context Length | LR KL |
|-------------|---|---|--------------------------------|--------------------|---------|-----|--------|----------------------------|---|
| Llama3.1-8B | Unfiltered (30k sampled) Symeval | Cosine:
$r_0^C = +2$
$r_L^C = +1$
$r_0^W = 10$
$r_L^W = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $= 1$
$c = 1$
$p = 0.99$ | 1
30k instances | 4 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6
KL: 0.01 |
| Llama3.1-8B | Unfiltered (30k sampled) LLM-as-a-judge | Cosine:
$r_0^C = +2$
$r_L^C = +1$
$r_0^W = 10$
$r_L^W = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $= 1$
$c = 1$
$p = 0.99$ | 1
30k instances | 4 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6
KL: 0.01 |
| Llama3.1-8B | Filtered (30k sampled) Symeval | Cosine:
$r_0^C = +2$
$r_L^C = +1$
$r_0^W = 10$
$r_L^W = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $= 1$
$c = 1$
$p = 0.99$ | 1
30k instances | 4 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6
KL: 0.01 |
| Llama3.1-8B | Filtered (30k sampled) LLM-as-a-judge | Cosine:
$r_0^C = +2$
$r_L^C = +1$
$r_0^W = 10$
$r_L^W = 0$
$r_e = 10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | $= 1$
$c = 1$
$p = 0.99$ | 1
30k instances | 4 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 9e-6
KL: 0.01 |

| Base Model | Rewards | GAE | Episodes | Samples | BS | Epochs | Context Length | LR | KL |
|-----------------|---|---------------|----------|---------|----------------------|--------|----------------------------|-------------------------------|------|
| Qwen2.5-Math-7B | Correct: +1
Wrong: -0.5
No Answer: -1 | = 0.95
= 1 | 20 | 8 | 1024
(Train: 128) | 1 | Prompt: 1024
Gen: 3072 | Actor: 5e-7
Critic: 9e-6 | 0.01 |
| Qwen2.5-Math-7B | Correct: +1 | = 1
= 1 | 8 | 8 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 4.5e-6 | 0.01 |
| Qwen2.5-Math-7B | Cosine:
$r_0^c = +2$
$r_L^c = +1$
$r_0^w = -10$
$r_L^w = 0$
$r_e = -10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | = 1
= 1 | 8 | 8 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 4.5e-6 | 0.01 |

SFT pn $\dot{\mathcal{U}}_{\mathcal{Q}} \mathcal{M}$ -32B-Preview - $\dot{\mathcal{U}}_{\mathcal{Q}} \frac{1}{2}$ • CoT pn • (MATH - $\dot{\mathcal{U}}_{\mathcal{Q}} \frac{1}{2} \frac{1}{2}$

| Base Model | Rewards | Gamma | Episodes | Samples | BS | Epochs | Context Length | LR | KL | Clip |
|-------------|-------------|-------|----------------------|---------|-----|--------|----------------------------|------|------|------|
| Llama3.1-8B | Correct: +1 | = 1 | 8
(stopped early) | 8 | 512 | 1 | Prompt: 2048
Gen: 14336 | 5e-7 | 0.01 | 0.1 |

i 2 2† Qwen2.5-7B-Instruct\ : i 2 2< „ Ei b 2 2f « 2 2† i 2 2 HCE• CoT„ i 2 2
* - † • CoTi 2 2 ÷ i 2 2 2 2 2 i 2 2† i 2 2 2 2 2

E.5.9. DETAILS OF SECTION 6 (EXPLORATION ON RL FROM THE BASE MODEL)

Table 15. Hyperparameters

| Base Model | Rewards | GAE | Episodes | Samples | BS | Epochs | Context Length | LR | KL |
|-----------------|---|---------------|----------|---------|----------------------|--------|----------------------------|-------------------------------|------|
| Qwen2.5-Math-7B | Correct: +1
Wrong: 0.5
No Answer: -1 | = 0.95
= 1 | 20 | 8 | 1024
(Train: 128) | 1 | Prompt: 1024
Gen: 3072 | Actor: 5e-7
Critic: 9e-6 | 0.01 |
| Qwen2.5-Math-7B | Correct: +1 | = 1
= 1 | 8 | 8 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 4.5e-6 | 0.01 |
| Qwen2.5-Math-7B | Cosine:
$r_0^c = +2$
$r_L^c = +1$
$r_0^w = -10$
$r_L^w = 0$
$r_e = -10$
Rep. Penalty:
$P = 0.05$
$N = 40$ | = 1
= 1 | 8 | 8 | 512 | 1 | Prompt: 2048
Gen: 14336 | Actor: 5e-7
Critic: 4.5e-6 | 0.01 |

E.5.10. DETAILS OF SECTION 7.3 (REINFORCE IS MORE TRICKY TO TUNE THAN PPO)

SFT Data: Long CoT data distilled from QwQ-32B-Preview with the MATH train split.

Table 16. Hyperparameters

| Base Model | Rewards | Gamma | Episodes | Samples | BS | Epochs | Context Length | LR | KL | Clip |
|-------------|-------------|-------|----------------------|---------|-----|--------|----------------------------|------|------|------|
| Llama3.1-8B | Correct: +1 | = 1 | 8
(stopped early) | 8 | 512 | 1 | Prompt: 2048
Gen: 14336 | 5e-7 | 0.01 | 0.1 |

E.6. Implementation of the Model-Based Verifier

We used Qwen2.5-7B-Instruct as our model-based verifier. It was provided with both the reference answer and the suffix of the long CoT. We truncated the long CoT to avoid confusing the verifier. We used the following prompt.

Prompt Template for Model-Based Verifier

Given the following last 20 lines of the LLM response to a math question and the reference solution to that question, evaluate if the LLM response is correct based only on the LLM’s final answer.

LLM response (last 20 lines):
{out}

Reference solution:
{ref}

Explain your thought process step-by-step before responding with ‘Judgement: <correct/wrong/not_found>’

E.7. Implementation of Short-Form Answer Extraction

We used Llama-3.1-8B-Instruct as our model-based verifier. It was provided with both the reference answer and the suffix of the long CoT. We truncated the long CoT to avoid confusing the verifier. We used the following prompt.

Prompt Template for Short-Form Answer Extraction

Problem: {Problem}

Solution: {Solution}

Based on the Problem and the Solution, extract a short final answer that is easy to check. Provide the short final answer in the format of "The final answer is $\\boxed{\\dots}$ "

\$\$

- If the answer is a mathematical object, write it in LaTeX, e.g., "The final answer is $\\boxed{\\frac{1}{2}}$ "

\$\$

- If the answer is a boolean, write it as "True" or "False", e.g., "The final answer is $\\boxed{\\text{True}}$ "

\$\$

- If the Problem can’t be answered in a short form, respond with "" like "The final answer is $\\boxed{}$ "

\$\$

We used Llama-3.1-8B-Instruct as our model-based verifier. It was provided with both the reference answer and the suffix of the long CoT. We truncated the long CoT to avoid confusing the verifier. We used the following prompt.

Prompt Template for Model-Based Verifier

Given the following last 20 lines of the LLM response to a math question and the reference solution to that question, evaluate if the LLM response is correct based only on the LLM's final answer.

LLM response (last 20 lines):

...
{out}

Reference solution:

{ref}

Explain your thought process step-by-step before responding with 'Judgement: <correct/wrong/not_found>'

E.7. Implementation of Short-Form Answer Extraction

We use the Llama-3.1-8B-Instruct model to extract short-form answer from QA pairs in WebInstruct, with the following prompt template:

Prompt Template for Short-Form Answer Extraction

Problem: {Problem}

Solution: {Solution}

Based on the Problem and the Solution, extract a short final answer that is easy to check.

Provide the short final answer in the format of "The final answer is \$\$

\boxed{...}

\$\$"

- If the answer is a mathematical object, write it in LaTeX, e.g., "The final answer is \$\$

\boxed{\frac{1}{2}}

\$\$"

- If the answer is a boolean, write it as "True" or "False", e.g., "The final answer is \$\$

\boxed{True}

\$\$"

- If the Problem can't be answered in a short form, respond with "" like "The final answer is \$\$

\boxed{}

\$\$"

For generation parameters, we use temperature $t = 0$ (greedy decoding) and set the maximum output length as 512

$$\tilde{f} \in \mathcal{U}(\tilde{\mathcal{O}}) \text{ s.t. } \tilde{f} \text{ is } \tilde{\mathcal{O}}\text{-maximal. } f \in \mathcal{F} \text{ is } \mathcal{O}\text{-maximal.}$$

E.8. Action Prompting Framework

Let \mathcal{A} be a set of actions. For each action $a \in \mathcal{A}$, let \mathcal{O}_a be the set of objects that a can act on. Let \mathcal{F} be a set of functions. Let \mathcal{O} be a set of objects. Let \mathcal{I} be a set of instructions. Let \mathcal{M} be a set of models. Let \mathcal{P} be a set of problems. Let \mathcal{S} be a set of solutions. Let \mathcal{V} be a set of verifiers. Let \mathcal{W} be a set of wrappers. Let \mathcal{X} be a set of extractors. Let \mathcal{Y} be a set of yielders. Let \mathcal{Z} be a set of zippers. Let \mathcal{A} be a set of actions. Let \mathcal{O}_a be the set of objects that a can act on. Let \mathcal{F} be a set of functions. Let \mathcal{O} be a set of objects. Let \mathcal{I} be a set of instructions. Let \mathcal{M} be a set of models. Let \mathcal{P} be a set of problems. Let \mathcal{S} be a set of solutions. Let \mathcal{V} be a set of verifiers. Let \mathcal{W} be a set of wrappers. Let \mathcal{X} be a set of extractors. Let \mathcal{Y} be a set of yielders. Let \mathcal{Z} be a set of zippers.

• clarify: $\hat{f} \in \mathcal{U}(\hat{\mathcal{O}})$ s.t. \hat{f} is $\hat{\mathcal{O}}$ -maximal. $f \in \mathcal{F}$ is \mathcal{O} -maximal.

• \mathcal{A} : $\mathcal{S} \cap \mathcal{M} \cap \mathcal{O} \cap \mathcal{I} \cap \mathcal{P} \cap \mathcal{S} \cap \mathcal{V} \cap \mathcal{W} \cap \mathcal{X} \cap \mathcal{Y} \cap \mathcal{Z}$

• solution_step: $j \rightarrow \mathcal{O} \cap \mathcal{M} \cap \mathcal{O} \cap \mathcal{I} \cap \mathcal{P} \cap \mathcal{S} \cap \mathcal{V} \cap \mathcal{W} \cap \mathcal{X} \cap \mathcal{Y} \cap \mathcal{Z}$

• reflection: $\tilde{f} \in \mathcal{U}(\tilde{\mathcal{O}})$ s.t. \tilde{f} is $\tilde{\mathcal{O}}$ -maximal. $f \in \mathcal{F}$ is \mathcal{O} -maximal. $\tilde{f} \in \mathcal{U}(\tilde{\mathcal{O}})$ s.t. \tilde{f} is $\tilde{\mathcal{O}}$ -maximal. $f \in \mathcal{F}$ is \mathcal{O} -maximal.

• answer: $\tilde{f} \in \mathcal{U}(\tilde{\mathcal{O}})$ s.t. \tilde{f} is $\tilde{\mathcal{O}}$ -maximal.

E.8.1. CONTROL FLOW

$\mathcal{F} \cap \mathcal{L} \cap \mathcal{M} \cap \mathcal{O} \cap \mathcal{I} \cap \mathcal{P} \cap \mathcal{S} \cap \mathcal{V} \cap \mathcal{W} \cap \mathcal{X} \cap \mathcal{Y} \cap \mathcal{Z}$

tokens.

After generation, we simply extract the short-form answer from within the *nboxed f. . . g.*

E.8. Action Prompting Framework

We studied the publicly released CoTs of o1-preview and identified that its thoughts could be categorized into a few types of actions (listed below). To construct long CoTs, we designed prompts for each of these actions and implemented a multi-step prompting framework to sequence them. The framework ceded control flow of the CoT to the LLM, with the LLM making branching or looping decisions while the framework acted more passively as a state machine reacting to the LLM outputs. The framework took care of the boilerplate around constructing the CoT with an append-only log and managed all of the orchestration.

- `clarify`: Making some observations about the problem in order to identify an approach to solve it.
- `decompose`: Breaking the current problem down into smaller and easier sub-problems to solve.
- `solution_step`: Computing a single step in the solution. In the context of math, this could be doing some arithmetic or symbolic manipulation.
- `reflection`: Evaluating the current approach and partial solution to see if any mistakes were made, any sub-goals were achieved, or if alternative approaches should be considered instead. Note that we used a strong teacher model o1-mini for the `reflection` action as that one was a more difficult prompt to respond to correctly as it requires self-correction.
- `answer`: Responding with a final answer and terminating the CoT.

E.8.1. CONTROL FLOW

Simplified description of the interaction between the framework and LLM.

Algorithm 2 Action Prompting State Machine

```
1: Input: prompt
2: Output: chain_of_thought sequence
3: chain_of_thought ← [prompt] fInitialize singleton chain of thought sequence from promptg
4: state ← “clarify”
5: while True do
6:   if state = “clarify” then
7:     output ← prompt_action.clarify()
8:     (state; thought) ← parse(output)
9:     chain_of_thought.append(thought)
10:  else if state = “decompose” then
11:    output ← prompt_action.decompose()
12:    (state; thought) ← parse(output)
13:    chain_of_thought.append(thought)
14:  else if state = “solution_step” then
15:    output ← prompt_action.solution_step()
16:    (state; thought) ← parse(output)
17:    chain_of_thought.append(thought)
18:  else if state = “reflection” then
19:    output ← prompt_action.reflection()
20:    (state; thought) ← parse(output)
21:    chain_of_thought.append(thought)
22:  else if state = “answer” then
23:    output ← prompt_action.answer()
24:    (state; thought) ← parse(output)
25:    chain_of_thought.append(thought)
26:    return chain_of_thought fTerminate after answer actiong
27:  end if
28: end while
```

| Demystifying Long Chain-of-Thought Reasoning in LLMs |
|--|
| Algorithm 2 Action Prompting State Machine |
| 1: Input: <i>prompt</i>
2: Output: <i>chain_of_thought</i> sequence
3: <i>chain_of_thought</i> [<i>prompt</i>] <i>f</i> Initialize singleton chain of thought sequence from <i>promptg</i>
4: <i>state</i> “clarify”
5: while True do
6: if <i>state</i> = “clarify” then
7: <i>output</i> <i>prompt_action.clarify()</i>
8: (<i>state; thought</i>) <i>parse(output)</i>
9: <i>chain_of_thought.append(thought)</i>
10: else if <i>state</i> = “decompose” then
11: <i>output</i> <i>prompt_action.decompose()</i>
12: (<i>state; thought</i>) <i>parse(output)</i>
13: <i>chain_of_thought.append(thought)</i>
14: else if <i>state</i> = “solution_step” then
15: <i>output</i> <i>prompt_action.solution_step()</i>
16: (<i>state; thought</i>) <i>parse(output)</i>
17: <i>chain_of_thought.append(thought)</i>
18: else if <i>state</i> = “reflection” then
19: <i>output</i> <i>prompt_action.reflection()</i>
20: (<i>state; thought</i>) <i>parse(output)</i>
21: <i>chain_of_thought.append(thought)</i>
22: else if <i>state</i> = “answer” then
23: <i>output</i> <i>prompt_action.answer()</i>
24: (<i>state; thought</i>) <i>parse(output)</i>
25: <i>chain_of_thought.append(thought)</i>
26: return <i>chain_of_thought</i> <i>f</i> Terminate after answer action <i>g</i>
27: end if
28: end while |

| Title Suppressed Due to Excessive Size |
|--|
| E.8.2. ACTION PROMPTING TEMPLATES |
| <div> <div>Action: Clarify</div> <div> <p>You are a very talented mathematics professor.</p> <p>In a few sentences, VERY CONCISELY rephrase the problem to clarify its meaning and explicitly state what needs to be solved. Highlight any assumptions, constraints and potential misinterpretations.</p> <p>Do NOT attempt to solve the problem yet -- you are just clarifying the problem in your mind.</p> <p><problem>
{goal}
</problem></p> <p>Answer in the following format:</p> <p><clarification>
Problem clarification as instructed above
</clarification>
<goal>
Summarize the problem into a single statement describing the goal, e.g. Find the value of the variable w.
</goal></p> </div> </div> |

E.8.2. ACTION PROMPTING TEMPLATES

Action: Clarify

You are a very talented mathematics professor.
In a few sentences, VERY CONCISELY rephrase the problem to clarify its meaning and explicitly state what needs to be solved. Highlight any assumptions, constraints and potential misinterpretations.
Do NOT attempt to solve the problem yet -- you are just clarifying the problem in your mind.

<problem>
{goal}
</problem>

Answer in the following format:

<clarification>
Problem clarification as instructed above
</clarification>
<goal>
Summarize the problem into a single statement describing the goal, e.g. Find the value of the variable w.
</goal>

Action: Decompose

You are a talented mathematics professor.
You already have a partial solution to a problem.
In a single sentence, propose candidates for the next subgoal as the next step of the partial solution that will help you make progress towards the current goal.
Do not repeat any subgoal, we don't want any infinite loops!
Do not suggest using a computer or software tools.

<current goal>
{current_goal}
</current goal>
<parent goal>
{parent_goal}
</parent goal>
<partial solution>
{solution}
</partial solution>

Format your answer as follows:

<thinking>
step-by-step thinking of what the next possible subgoal should be, as well as some other alternatives that might also work
remember, we want to solve the parent goal WITHOUT repeating the subgoals that are already DONE.
do not suggest verification or checking.
{parent_goal}
</thinking>
<sentence>
single sentence describing the subgoal
phrase it as if you were thinking to yourself and are considering this as a hypothesis (don't express too much certainty)
</sentence>
<sentence>
single sentence describing an *ALTERNATIVE* subgoal, without repeating previous ones
start off with "Alternatively, "
</sentence>
<sentence>
single sentence describing an *ALTERNATIVE* subgoal, without repeating previous ones
start off with "Alternatively, "
</sentence>

Action: Decompose

You are a talented mathematics professor.
You already have a partial solution to a problem.
In a single sentence, propose candidates for the next subgoal as the next step of the partial solution that will help you make progress towards the current goal.
Do not repeat any subgoal, we don't want any infinite loops!
Do not suggest using a computer or software tools.

<current goal >
{current_goal}
</current goal >
<parent goal >
{parent_goal}
</parent goal >
<partial solution>
{solution}
</partial solution>

Format your answer as follows:

<thinking>
step-by-step thinking of what the next possible subgoal should be, as well as some other alternatives that might also work
remember, we want to solve the parent goal WITHOUT repeating the subgoals that are already DONE.
do not suggest verification or checking.
{parent_goal}
</thinking>
<sentence>
single sentence describing the subgoal
phrase it as if you were thinking to yourself and are considering this as a hypothesis (don't express too much certainty)
</sentence>
<sentence>
single sentence describing an *ALTERNATIVE* subgoal, without repeating previous ones
start off with "Alternatively,"
</sentence>
<sentence>
single sentence describing an *ALTERNATIVE* subgoal, without repeating previous ones
start off with "Alternatively,"
</sentence>

Action: Solution Step

You are an extremely PEDANTIC mathematics professor who loves to nitpick.
You already have a partial solution to a problem. Your task is to solve *only* the current goal.
You should include symbols and numbers in every sentence if possible.

<current goal >
{current_goal}
</current goal >
<partial solution>
{solution}
</partial solution>

BE VERY CONCISE. Include calculations and equations in your response if possible, and make sure to solve them instead of just describing them.
DO NOT SOLVE THE WHOLE QUESTION, JUST THE CURRENT GOAL: {current_goal}
Do not repeat any calculations that were already in this prior step:
{prior_step}

Action: Solution Step

You are an extremely PEDANTIC mathematics professor who loves to nitpick.
You already have a partial solution to a problem. Your task is to solve *only* the current goal.
You should include symbols and numbers in every sentence if possible.

<current goal >
{current_goal}
</current goal >
<partial solution>
{solution}
</partial solution>

BE VERY CONCISE. Include calculations and equations in your response if possible, and make sure to solve them instead of just describing them.
DO NOT SOLVE THE WHOLE QUESTION, JUST THE CURRENT GOAL: {current_goal}
Do not repeat any calculations that were already in this prior step:
{prior_step}

Action: Reflection

You are a talented mathematics professor.
You already have a partial solution to a math problem.
Verify whether the current subgoal has been achieved.

<current goal >
{current_goal}
</current goal >
{parent_goal}
<partial solution>
{solution}
</partial solution>

Format your answer as follows:

<verification>
Come up with a quick, simple and easy calculation to double check that the solution is correct.
This calculation should not re-compute the solution in the same way, as that would defeat the purpose of double-checking.
Use one of the following strategies:
- An easier, alternative method to arrive at the answer
- Substituting specific values into equations and checking for consistency
- Working backwards from the answer to derive the given inputs and then checking for consistency
Be concise. Do not suggest using a computer.
At the end of your verification, restate the answer from the current solution. Do not calculate it if it hasn't been solved.
Phrase it as if you are reflecting as you solve the problem.
</verification>
<current_goal_achieved>
true or false, depending on whether the solution is correct and the current goal has been achieved: {current_goal}
</current_goal_achieved>
<parent_goal_achieved>
true or false, depending on whether the parent goal has been achieved:
{parent_goal.target}
</parent_goal_achieved>
<new_goal >
If the solution is not correct or the current goal has not been achieved, suggest an alternative current goal here in a single sentence.
Start off with "Alternatively,"
Your goal should be sufficiently different from subgoals that have been solved or that have timed out:
{parent_goal_tree}
</new_goal >

Action: Reflection

You are a talented mathematics professor.
You already have a partial solution to a math problem.
Verify whether the current subgoal has been achieved.

<current_goal >
{current_goal}
</current_goal >
{parent_goal}
<partial_solution>
{solution}
</partial_solution>

Format your answer as follows:

<verification>
Come up with a quick, simple and easy calculation to double check that the solution is correct.
This calculation should not re-compute the solution in the same way, as that would defeat the purpose of double-checking.
Use one of the following strategies:
- An easier, alternative method to arrive at the answer
- Substituting specific values into equations and checking for consistency
- Working backwards from the answer to derive the given inputs and then checking for consistency
Be concise. Do not suggest using a computer.
At the end of your verification, restate the answer from the current solution. Do not calculate it if it hasn't been solved.
Phrase it as if you are reflecting as you solve the problem.
</verification>
<current_goal_achieved>
true or false, depending on whether the solution is correct and the current goal has been achieved: {current_goal}
</current_goal_achieved>
<parent_goal_achieved>
true or false, depending on whether the parent goal has been achieved:
{parent_goal.target}
</parent_goal_achieved>
<new_goal >
If the solution is not correct or the current goal has not been achieved, suggest an alternative current goal here in a single sentence.
Start off with "Alternatively,"
Your goal should be sufficiently different from subgoals that have been solved or that have timed out:
{parent_goal_tree}
</new_goal >

Action: Answer

Extract the final answer, making sure to obey the formatting instructions.
Solution:
{solution}

Formatting instructions:
{format}

Demystifying Long Chain-of-Thought Reasoning in LLMs

Action: Answer

Extract the final answer, making sure to obey the formatting instructions.

Sol uti on:
{sol uti on}

Formatti ng i nstructi ons:
{format}

Title Suppressed Due to Excessive Size

F. Long CoT Patterns in Pre-training Data

F.1. Snapshot of webpages

Explicit verification

$x + 7 = 10$

This problem can be solved by subtracting 7 from each side.

$x + 7 - 7 = 10 - 7$

$x = 3$

Once the problem is solved, the solution can be verified by rewriting the problem with 3 substituted for x .

$3 + 7 = 10$

$10 = 10$

Both sides are equal, verifying that $x = 3$ is a valid solution.

Explicit verification that found an error

$x + 7 = 10$

A student rushing through her homework might mistakenly write $x = 2$ as the solution to this problem. If she takes a moment to rework the equation with her answer, she will realize the answer is incorrect.

$x + 7 = 10$

$2 + 7 = 10$

$9 = 10$

Since $9 \neq 10$, the student knows she needs to go back and find a different solution to the problem.

41

F. Long CoT Patterns in Pre-training Data

F.1. Snapshot of webpages

Source: brilliant.org

The following two examples demonstrate how explicit verification after answering a question can naturally exist on a webpage.

Explicit verification

$x + 7 = 10$
This problem can be solved by subtracting 7 from each side.
 $x + 7 - 7 = 10 - 7$
 $x = 3$
Once the problem is solved, the solution can be verified by rewriting the problem with 3 substituted for x .
 $3 + 7 = 10$
 $10 = 10$
Both sides are equal, verifying that $x = 3$ is a valid solution.

Explicit verification that found an error

$x + 7 = 10$ A student rushing through her homework might mistakenly write $x = 2$ as the solution to this problem. If she takes a moment to rework the equation with her answer, she will realize the answer is incorrect.
 $x + 7 = 10$
 $2 + 7 = 10$
 $9 = 10$
Since $9 \neq 10$, the student knows she needs to go back and find a different solution to the problem.

• : kidswholovemath.substack.com

Attempt the question from different perspective

The Double Check Game
Regardless of the scenario, we can play the double check game!
The game is simple: we try to solve the problem in as many different ways as possible.
Elementary School Example
Math problem is: $78 - 57 = ?$
To play the game, we try to solve the problem in as many different ways as possible.
The first solution:
 $? = 78 - 57$
Break apart the 57:
 $? = 78 - 50 - 7$
 $? = 28 - 7$
 $? = 21$
A second solution:
 $? = 78 - 57$
Subtract an easier number from 78:
 $? = 78 - 60 + 3$
 $? = 18 + 3$
 $? = 21$
A third solution:
 $? = 78 - 57$
Subtract 57 from an easier number:
 $? = 80 - 57 - 2$
 $? = 23 - 2$
 $? = 21$
...

Source: kidswholovemath.substack.com

Attempt the question from different perspective

The Double Check Game

Regardless of the scenario, we can play the double check game!
The game is simple: we try to solve the problem in as many different ways as possible.

Elementary School Example

Math problem is: $78 - 57 = ?$
To play the game, we try to solve the problem in as many different ways as possible.

The first solution:

$? = 78 - 57$
Break apart the 57:
 $? = 78 - 50 - 7$
 $? = 28 - 7$
 $? = 21$

A second solution:

$? = 78 - 57$
Subtract an easier number from 78:
 $? = 78 - 60 + 3$
 $? = 18 + 3$
 $? = 21$

A third solution:

$? = 78 - 57$
Subtract 57 from an easier number:
 $? = 80 - 57 - 2$
 $? = 23 - 2$
 $? = 21$
...

F.2. OpenWebMath

F.2.1. QUERIES

“I used GPT-4o to test the CoT - 8 1/2 x < 4 1/2 1/2 < 1 1/2 . I 1/2 (OpenWebMath- 1/2 w • i 1/2 h i 1/2 „ E ^ ' „ ‡ c

“Aha” Phrases

"Let's think step by step."
"Let's go through this one step at a time."
"Breaking it down step by step..."
"Thinking about it logically, first..."
"Step 1: Let's figure out the starting point."
"If we follow the steps carefully, we get..."
"To solve this, let's analyze it piece by piece."
"Going through this systematically, we have..."
"Okay, let's solve this gradually."
"Does that make sense?"
"Is this correct?"
"Wait, does that check out?"
"Am I missing something?"
"Hmm& does that work?"
"Let me verify that."
"That makes sense, right?"
"Hold on, is this right?"
"Let's double-check this."
"Wait, actually..."
"Oh, hold on..."
"Wait a second..."
"Actually, let me rethink that."
"Hmm, let me go back for a moment."
"I might need to check this again."
"Let's pause and reassess."
"Let's check by doing the reverse."
"Let's verify by working backward."
"Can we check this by reversing the process?"
"To confirm, let's undo the steps."
"A good way to verify is by reversing it."
"If we undo the operations, do we get the same result?"
...

Source: [physicsforums.com](https://www.physicsforums.com)

The discussion below took place on a physics forum. The user Songoku is asking for help with homework and another user BvU is trying to assist without revealing the solution directly. We see the usual pivot keywords indicating self-reflection, expression of uncertainty and formulation of hypotheses.

Discussion on a physics forum

Cylinder in 3 D
1. Dec 13, 2017

songoku
1. The problem statement, all variables and given/known data
Let r be a positive constant. Consider the cylinder $x^2 + y^2 \leq r^2$, and let C be the part of the cylinder that satisfies $0 \leq z \leq y$.
(1) Consider the cross section of C by the plane $x = t$ ($-r \leq t \leq r$), and express its area in terms of r, t .
(2) Calculate the volume of C , and express it in terms of r .

...

5. Dec 13, 2017
BvU
Simple case: $x = 0$. So $-1 \leq y \leq 1$. In the yz plane $0 \leq z \leq y$ is a triangle. What about y ?

6. Dec 13, 2017
songoku
I think I am missing something here because I feel I can't really grasp the hint given.
Let me start from the basic again:
1. Let the x - axis horizontal, y - axis vertical and z - axis in / out of page. I imagine there is circle on xy plane with radius r then it extends out of page (I take out of page as z +) to form 3 D cylinder. **Is this correct?**
2. Plane $x = t$ is like the shape of a piece of paper hold vertically with the face of paper facing x - axis (I mean x - axis is the normal of the plane). **Is this correct?**
Thanks

7. Dec 14, 2017
BvU
Yes

8. Dec 14, 2017
songoku

"Consider the cross section of C by plane $x = t$ " means plane $x = t$ cuts the cylinder ?
And the intersection will be rectangle?
...

Demystifying Long Chain-of-Thought Reasoning in LLMs

Discussion on a physics forum

Cylinder in 3 D
1. Dec 13, 2017

songoku
1. The problem statement, all variables and given/known data
Let r be a positive constant. Consider the cylinder $x^2 + y^2 \leq r^2$, and let C be the part of the cylinder that satisfies
 $0 \leq z \leq y$.
(1) Consider the cross section of C by the plane $x = t$ ($-r \leq t \leq r$), and express its area in terms of r, t .
(2) Calculate the volume of C , and express it in terms of r .

...

5. Dec 13, 2017
BvU
Simple case: $x = 0$. So $-1 \leq y \leq 1$. In the yz plane $0 \leq z \leq y$ is a triangle. What about y ?

6. Dec 13, 2017
songoku
I think I am missing something here because I feel I can't really grasp the hint given.
Let me start from the basic again:
1. Let the x - axis horizontal, y - axis vertical and z - axis in / out of page. I imagine there is circle on xy plane with radius r then it extends out of page (I take out of page as z +) to form 3 D cylinder. **Is this correct?**
2. Plane $x = t$ is like the shape of a piece of paper hold vertically with the face of paper facing x - axis (I mean x - axis is the normal of the plane). **Is this correct?**
Thanks

7. Dec 14, 2017
BvU
Yes

8. Dec 14, 2017
songoku

"Consider the cross section of C by plane $x = t$ " means plane $x = t$ cuts the cylinder ?
And the intersection will be rectangle?
...

Title Suppressed Due to Excessive Size

Discussion on Stack Exchange

probability that we stop flipping after exactly ten flips in a biased coin flipping?

...

I thought that let us fix of getting a third head at last that is at 10th flip, so that we would stop there, and the remaining - getting two heads can be accommodated in the 9 trials. so there are $\binom{9}{2}$ choose 2 ways of getting two heads so the probability that we stop flipping after exactly ten flips is $\binom{9}{2} p^3 (1-p)^7$. **Is this correct?**

EDIT - Now the probability of getting exactly 3 heads? I got it to be $\binom{10}{3} p^3 (1-p)^7$. Should we get the same as the previous one? any reason why they should/should not be same?

48 I think you switched $P(H);P(T)$ |but the approach is good. | { lulu Oct 1 '18 at 16:13 • |oh i see now! | thanks! { BAYMAX Oct 1 '18 at 16:13 • @lulu please see the edit { BAYMAX Oct 1 '18 at 16:30 • Your probability for exactly 3 heads is right as well. It should be obvious why the results have to be different. In the first case the outcome of the last flip is fix and in the second case the outcome of the last flip is not fix. { calculus Oct 1 '18 at 16:31...

Source: [StackExchange](#)

The user Baymax is asking for help on a probability problem and we see dialogue with another user Lulu. We see that the quick back-and-forth between them is similar to the kind of nimble branching behavior in long CoT where multiple solutions are quickly assessed and considered. We also see an expression of realization which can be easily re-cast as self-verification in a long CoT.

Discussion on Stack Exchange

probability that we stop flipping after exactly ten flips in a biased coin flipping?

...

I thought that let us fix of getting a third head at last that is at 10th flip, so that we would stop there, and the remaining - getting two heads can be accommodated in the 9 trials. so there are $\binom{9}{2}$ choose 2 ways of getting two heads so the probability that we stop flipping after exactly ten flips is $\binom{9}{2} \frac{1}{4}^3 \frac{3}{4}$. **Is this correct?**

EDIT - Now the probability of getting exactly 3 heads? I got it to be $\binom{10}{3} \frac{1}{4}^3 \frac{3}{4}$. Should we get the same as the previous one? any reason why they should/should not be same?

48 I think you switched $P(H);P(T)$ **but the approach is good.** { Lulu Oct 1 '18 at 16:13• **oh i see now!** | thanks! { BAYMAX Oct 1 '18 at 16:13• @lulu please see the edit { BAYMAX Oct 1 '18 at 16:30• Your probability for exactly 3 heads is right as well. It should be obvious why the results have to be different. In the first case the outcome of the last flip is fix and in the second case the outcome of the last flip is not fix. { calculus Oct 1 '18 at 16:31...

[StackExchange](#)

User88 $\frac{1}{2} \times \frac{1}{2} \times \frac{1}{2} \times \frac{1}{2} \times \frac{1}{2} \times \frac{1}{2} \times \frac{1}{2} \times \frac{1}{2} \times \frac{1}{2} \times \frac{1}{2} = \frac{1}{2^{10}}$

Discussion on Stack Exchange

Choosing units for drug testing

Here's a third puzzle that I found in a book, slightly paraphrased because I don't entirely remember the format of the original.

...

How can he arrange the dosage amounts so that he ends up using all 25 test packages, and the total units of dosage used in the tests are as low as possible?

The book had the answer, but one, it didn't explain how the answer was arrived at, and two, I don't remember what the answer was and no longer have that book with me.

48 **|Am I missing something|**, or is the goal just to find 25 coprime numbers from 25 to 50? { Aza May 20 '14 at 4:33• They don't have to be coprime. There just can't be any two where one is a factor of the other. And the range is from 1 to 50, not 25 to 50. { Joe Z. May 20 '14 at 4:34• **|Wouldn't a single test|** of 1 unit technically satisfy the requirement? Or **|am I missing something? Ah, I guess you have to|** perform exactly 25 tests. { arshajii May 20 '14 at 14:28• **|Yea.** | Wouldn't 1 win? { awesomepi May 20 '14 at 19:24• You have to use all 25 tests. { Joe Z. May 20 '14 at 19:31By logically starting from 26-50 and trying to shrink them one by one you can easily show:
8;12;14;17;18;19;20;21;22;23;25;26;27;29;30;31;33;35;37;39;41;43;45;47;49Which equals 711...

Source: [StackExchange](#)

User88 interacts with multiple other users. Observe that they are helping to clarify each others’ doubts, which is reminiscent of self-correction in long CoT trajectories.

Discussion on Stack Exchange

Choosing units for drug testing

Here’s a third puzzle that I found in a book, slightly paraphrased because I don’t entirely remember the format of the original.

...

How can he arrange the dosage amounts so that he ends up using all 25 test packages, and the total units of dosage used in the tests are as low as possible?

The book had the answer, but one, it didn’t explain how the answer was arrived at, and two, I don’t remember what the answer was and no longer have that book with me.

48 |Am I missing something|, or is the goal just to find 25 coprime numbers from 25 to 50? { Aza May 20 ’14 at 4:33• They don’t have to be coprime. There just can’t be any two where one is a factor of the other. And the range is from 1 to 50, not 25 to 50. { Joe Z. May 20 ’14 at 4:34• |Wouldn’t a single test| of 1 unit technically satisfy the requirement? Or |am I missing something? Ah, I guess you have to| perform exactly 25 tests. { arshajii May 20 ’14 at 14:28• |Yea. | Wouldn’t 1 win? { awesomepi May 20 ’14 at 19:24• You have to use all 25 tests. { Joe Z. May 20 ’14 at 19:31By logically starting from 26-50 and trying to shrink them one by one you can easily show:
8;12;14;17;18;19;20;21;22;23;25;26;27;29;30;31;33;35;37;39;41;43;45;47;49Which equals 711...