

ReCUT: Balancing Reasoning Length and Accuracy in LLMs via Stepwise Trails and Preference Optimization

Zhensheng Jin^{1*}, Xinze Li^{1*}, Yifan Ji¹, Chunyi Peng¹, Zhenghao Liu^{1†}, Qi Shi²,
Yukun Yan², Shuo Wang², Furong Peng³, Ge Yu¹

¹ School of Computer Science and Engineering, Northeastern University, Shenyang, China

² Department of Computer Science and Technology, Institute for AI, Tsinghua University, China
Beijing National Research Center for Information Science and Technology, Beijing, China

³ Institute of Big Data Science and Industry/School of Computer and
Information Technology, Shanxi University, China

Abstract

Recent advances in Chain-of-Thought (CoT) prompting have substantially improved the reasoning capabilities of Large Language Models (LLMs). However, these methods often suffer from overthinking, leading to unnecessarily lengthy or redundant reasoning traces. Existing approaches attempt to mitigate this issue through curating multiple reasoning chains for training LLMs, but their effectiveness is often constrained by the quality of the generated data and prone to overfitting. To address the challenge, we propose **Reasoning Compression ThroUgh Stepwise Trials (ReCUT)**, a novel method aimed at balancing the accuracy and length of reasoning trajectory. Specifically, ReCUT employs a stepwise exploration mechanism and a long-short switched sampling strategy, enabling LLMs to incrementally generate diverse reasoning paths. These paths are evaluated and used to construct preference pairs to train two specialized models (Gemini LLMs)—one optimized for reasoning accuracy, the other for shorter reasoning. A final integrated model is obtained by interpolating the parameters of these two models. Experimental results across multiple math reasoning datasets and backbone models demonstrate that ReCUT significantly reduces reasoning lengths by approximately 30-50%, while maintaining or improving reasoning accuracy compared to various baselines. All codes and data will be released via <https://github.com/NEUIR/ReCUT>.

1 Introduction

Recent advances in Chain-of-Thought (CoT) (Wei et al., 2022) have substantially improved the reasoning capabilities of Large Language Models (LLMs), motivating researchers to explore a new scaling paradigm—test-time scaling (Muennighoff et al., 2025; Snell et al., 2024). This paradigm improves the performance of LLMs on lots of

* indicates equal contribution.

† indicates corresponding author.

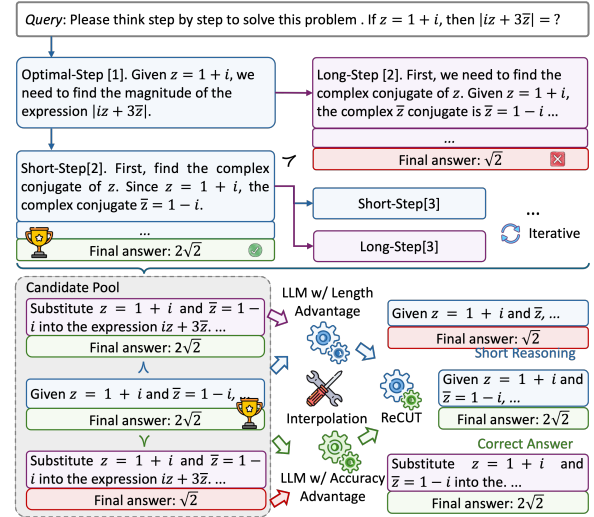


Figure 1: Illustration of Our Reasoning Compression Through Stepwise Trials (ReCUT) Model.

challenging reasoning tasks, such as math competitions (Hendrycks et al., 2021) and PhD-level subject QA (Rein et al., 2024), by extending the CoT with deeper and iterative thinking during inference (Wu et al., 2024; Snell et al., 2024). While effective, test-time scaling incurs higher computational costs (Lee et al., 2025). Furthermore, recent studies suggest that LLMs often exhibit overthinking behaviors, even on relatively simple problems (Chen et al., 2024), potentially diminishing the benefits of deeper reasoning in certain cases.

To address the inefficiencies caused by overthinking, various strategies have been proposed. Prompt-based methods aim to guide LLMs to generate more concise reasoning trajectories. SFT-based methods prompt LLMs to sample multiple reasoning trajectories and select the concise and correct ones to synthesize SFT data for fine-tuning itself (Team et al., 2025). However, the former may cause the LLMs to omit critical intermediate steps (Jin et al., 2024; Lee et al., 2025), while the latter may lead to overfitting to the training sig-

nals (Luo et al., 2023). Reinforcement learning (RL)-based methods represent another promising research direction (Aggarwal and Welleck, 2025), where carefully designed reward functions penalize overly long reasoning trajectories and inaccurate outputs, thereby guiding LLMs to produce concise and accurate reasoning results.

Existing RL-based methods typically sample multiple complete reasoning trajectories using a single instruction to compute their rewards and construct preference data for training. However, this sampling strategy inherently limits the diversity of reasoning trajectories (Liu et al., 2023), resulting in convergent distributions in terms of both length and accuracy. Furthermore, such approaches often overlook the fine-grained contributions of individual reasoning steps in reasoning trajectories, allowing even the correct trajectories to include redundant steps (Wang et al., 2025). These limitations compromise the quality of the constructed preference data, ultimately resulting in suboptimal reasoning performance, which motivates us to explore a better method for constructing high-quality preference data. Recent methods that search for optimal reasoning paths by stepwise decoding have achieved advantages in reducing reasoning errors and redundancies generated by LLMs (Wang et al., 2025). This success inspires us to construct high-quality and diverse preference data by prompting the model to stepwise explore reasoning paths.

In this paper, we propose ReCUT (Reasoning Compression Through Stepwise Trials), a method that guides LLMs to progressively explore diverse reasoning trajectories and optimizes them via preference-based learning to balance reasoning accuracy and length. As shown in Figure 1, ReCUT introduces a stepwise reasoning trajectory exploration mechanism: at each step, the LLM is conditioned on a partially optimal trajectory and generates all subsequent reasoning steps using a long-short sampling strategy to encourage diversity. All trajectories produced via stepwise sampling are incorporated with the given optimal trajectory and then collected into a candidate pool. Then the first step of the generated reasoning outcome is further evaluated to incrementally construct the optimal reasoning path. After trajectory collection, ReCUT leverages this candidate pool to train two specialized LLMs (Gemini LLMs)—one favoring accuracy and the other favoring shorter reasoning results. Finally, ReCUT performs parameter interpolation between these two optimized LLMs to

achieve a trade-off between the reasoning accuracy and the length of reasoning results.

Our experiments demonstrate the effectiveness of ReCUT, which significantly reduces the reasoning length, typically by 30-50%, while maintaining or even surpassing the accuracy of baseline methods. Further analysis reveals that our stepwise sampling strategy enables the synthesis of diverse reasoning trajectories, helping LLMs regulate the number of reasoning steps through preference optimization. By leveraging parameter interpolation, ReCUT not only achieves higher accuracy but also further shortens the reasoning process by combining the strengths of both Gemini LLMs. Moreover, ReCUT effectively mitigates overthinking in LLMs by reducing redundant reasoning steps and identifying unproductive reasoning paths that may lead to incorrect answers.

2 Related Work

The test-time scaling law suggests that LLMs, such as Deepseek-R1 (DeepSeek-AI et al., 2025) and QwQ (Team, 2025), can enhance their performance on challenging reasoning and mathematical tasks (Wu et al., 2024; Snell et al., 2024) by engaging in deeper reasoning (DeepSeek-AI et al., 2025) and producing longer Chains-of-Thought (CoT) (Wei et al., 2022). While effective, these approaches often cause LLMs to overthink (Sui et al., 2025), resulting in redundant or irrelevant content (Chiang and Lee, 2024), which not only increases inference cost but may also lead to incorrect answers (Cuadron et al., 2025).

To better balance reasoning efficiency and accuracy, recent work has explored efficient reasoning with LLMs, with a particular focus on reducing and controlling the length of reasoning chains (Lee et al., 2025; Aggarwal and Welleck, 2025; Cui et al., 2025). Some approaches design specialized prompts to directly shorten the generated responses (Renze and Guven, 2024; Lee et al., 2025). However, such strategies may truncate essential intermediate reasoning steps, ultimately compromising accuracy (Aggarwal and Welleck, 2025). To address this problem, Kang et al. (2025); Muenighoff et al. (2025) leverage advanced LLMs (e.g., ChatGPT (OpenAI et al., 2023)) to generate concise yet accurate CoTs as training data for Supervised Fine-Tuning (SFT). Nevertheless, this method is inherently limited by the capabilities of the teacher models and may lead to overfitting to

the provided training signals.

Benefiting from advances in reinforcement learning (Schulman et al., 2017; Shao et al., 2024), recent studies have leveraged it to enable LLMs to adaptively control the length of their reasoning processes. Aggarwal and Welleck (2025) introduce a length bias penalty and an accuracy reward during training to encourage LLMs to generate accurate reasoning using fewer tokens. Chen et al. (2025) propose a cosine reward mechanism that promotes reasoning trajectories of moderate length by penalizing both overly short and overly long generations. However, these approaches compute rewards based on entire reasoning trajectories, without estimating the contribution of individual steps in the whole reasoning chain. In contrast, ReCUT leverages a step-wise sampling strategy to elicit more fine-grained and diverse preference data from LLMs, which enhances the training process for reasoning compression.

3 Methodology

This section presents Reasoning Compression Through Stepwise Trials (ReCUT), a method designed to reduce the reasoning length of Large Language Models (LLMs) while maintaining comparable performance. As shown in Figure 2, we first introduce the Long-Short Switch Sampling strategy, which adaptively constructs a reasoning trajectory pool $\mathcal{C}_q$ containing trajectories of varying lengths (Sec. 3.1). Then, we leverage these trajectories to train multiple Gemini-LLMs with complementary strengths and interpolate their parameters to balance reasoning accuracy and length (Sec. 3.2).

3.1 Stepwise Reasoning Trajectory Exploration via Long-Short Sampling

Given a question q , the reasoning trajectory produced by an LLM is denoted as $Y = \{y_1, \dots, y_T, o\}$, where $y_{1:T}$ are intermediate reasoning steps and o is the final answer.

At step t , we design the *Long-Short Switched Sampling* method to use both long and short prompting instructions to guide the LLM to generate the following reasoning trajectories $Y_{\geq t}$ with different lengths, based on the current optimal partial trajectory $Y_{< t}^* = \{y_1^*, \dots, y_{t-1}^*\}$. The sampled full trajectory $Y_{< t}^* \cup Y_{\geq t}$ is added to the trajectory pool $\mathcal{C}_q$. We then propose the *Optimal Reasoning Step Selection* method. It designs a reward-based mechanism to select the optimal reasoning

step y_t^* , thereby updating the optimal trajectory $Y_{< t+1}^* = Y_{< t}^* \cup \{y_t^*\}$ for trajectory sampling at the next step.

Long-Short Switched Sampling. At each step t , given the question q and current optimal partial trajectory $Y_{< t}^*$, we prompt the LLM ($\mathcal{M}$) to generate two reasoning continuations—one long and one short—via different instructions:

$$\begin{aligned} Y_{\geq t}^l &= \mathcal{M}(\text{Instruct}_l(q, Y_{< t}^*)), \\ Y_{\geq t}^s &= \mathcal{M}(\text{Instruct}_s(q, Y_{< t}^*)), \end{aligned} \quad (1)$$

where Instruct_l and Instruct_s are tailored prompts that guide the model to produce reasoning of different lengths. We then concatenate the existing partial trajectory with each of the new continuations to form full candidate trajectories:

$$Y_{[t]}^l = (Y_{< t}^* \cup Y_{\geq t}^l), Y_{[t]}^s = (Y_{< t}^* \cup Y_{\geq t}^s), \quad (2)$$

where the subscript $[t]$ indicates the concatenation occurs at step t . Both reasoning trajectories $Y_{[t]}^l$ and $Y_{[t]}^s$ are added to the candidate pool $\mathcal{C}_q$:

$$\mathcal{C}_q \leftarrow \mathcal{C}_q \cup \{Y_{[t]}^l\} \cup \{Y_{[t]}^s\}. \quad (3)$$

Optimal Stepwise Reasoning Selection. Each candidate trajectory $Y_{[t]}$ is evaluated using a reward function that considers both accuracy and reasoning length:

$$r(Y_{[t]}) = \begin{cases} \frac{1}{|Y_{[t]}|}, & \text{if } o = o^{\text{gold}}, \\ -\frac{1}{|Y_{[t]}|}, & \text{if } o \neq o^{\text{gold}}, \end{cases} \quad (4)$$

where o^{gold} is the ground-truth answer and $|Y_{[t]}|$ denotes the number of tokens in $Y_{[t]}$. This reward function is designed to encourage the LLMs to explore trajectories that consider both length and accuracy. The reward function simultaneously encourages the correctness of the reasoning trajectory and penalizes its length. When the final answer in a trajectory is correct, we want the LLM to explore shorter trajectory directions to reduce the length. Conversely, when the final answer is incorrect, we prefer the LLM to explore longer trajectory directions, as incorrect answers may require a longer reasoning chain to be corrected.

We compare the rewards of the long and short trajectories and select the reasoning step y_t^* from the trajectory with the higher reward:

$$y_t^* = \arg \max(r(Y_{[t]}^l), r(Y_{[t]}^s)). \quad (5)$$

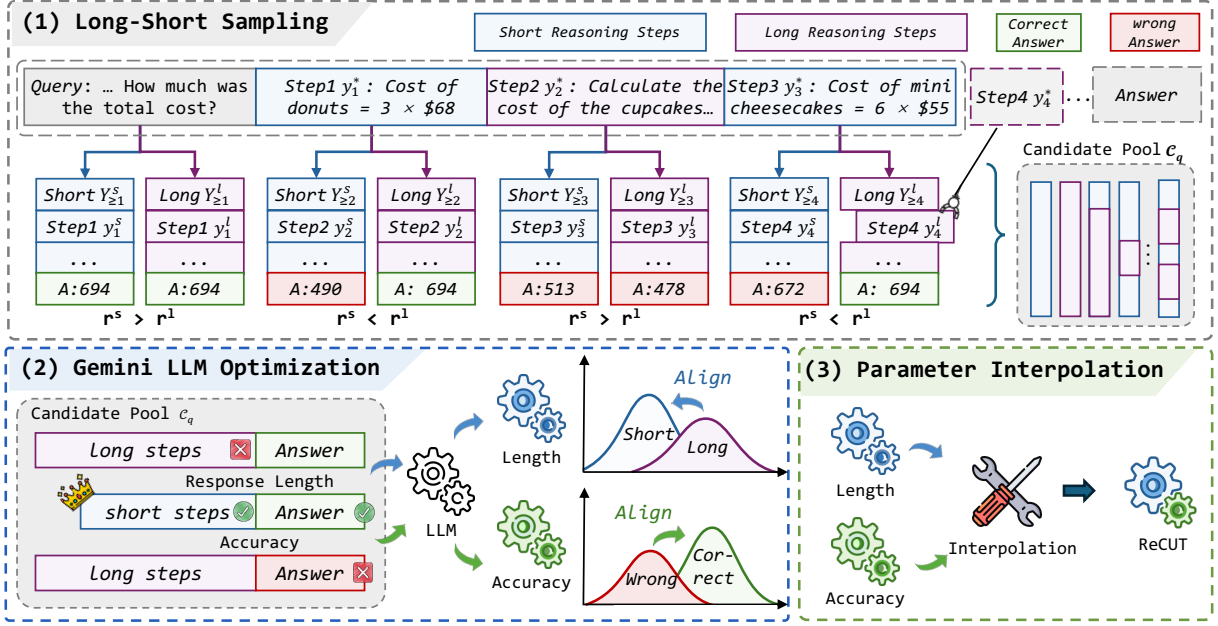


Figure 2: The Overview of Our Reasoning Compression Through Stepwise Trials (ReCUT) Model.

The optimal trajectory is then updated by appending y_t^* :

$$Y_{<t+1}^* = Y_{<t}^* \cup y_t^*. \quad (6)$$

This process is repeated iteratively until either the maximum step $\mathcal{T}$ is reached or the generation process terminates.

3.2 Gemini LLM: Balancing Accuracy and Length via Parameter Interpolation

In this section, we present Gemini LLM, a method that achieves a trade-off between the reasoning trajectory accuracy and length through parameter interpolation.

Given a question q and its corresponding reasoning trajectory pool $\mathcal{C}_q$, we first construct two distinct preference datasets: $\mathcal{D}_{\text{acc}}$ and $\mathcal{D}_{\text{len}}$, each reflecting a different optimization objective. These datasets are used to train two specialized LLMs: $\mathcal{M}_{\text{acc}}$, which emphasizes answer accuracy, and $\mathcal{M}_{\text{len}}$, which encourages a shorter reasoning result. The final Gemini model, $\mathcal{M}_{\text{merge}}$, is obtained via parameter interpolation between these two models, enabling it to balance the trade-off between accuracy and reasoning length during inference.

Gemini LLM Optimization. For each reasoning trajectory $Y_i \in \mathcal{C}_q$, we select the one with the correct final answer o and the shortest length as the positive sample:

$$Y^+ = \arg \min_{Y_i \in \mathcal{C}_q} (|Y_i|) \text{ s.t. } o = o^{\text{gold}}, \quad (7)$$

where $|Y_i|$ denotes the number of tokens in trajectory Y_i . Next, for each Y^+ , we select two negative samples from $\mathcal{C}_q$: 1) Y_{acc}^- is the longest incorrect trajectory, i.e., a long trajectory that leads to an incorrect final answer; 2) Y_{len}^- is the longest correct trajectory, i.e., a long trajectory that produces the correct answer. Formally, this can be described as:

$$\begin{aligned} Y_{\text{acc}}^- &= \arg \max_{Y_i \in \mathcal{C}_q} (|Y_i|) \text{ s.t. } o \neq o^{\text{gold}}, \\ Y_{\text{len}}^- &= \arg \max_{Y_i \in \mathcal{C}_q} (|Y_i|) \text{ s.t. } o = o^{\text{gold}}. \end{aligned} \quad (8)$$

Using the identified samples, we construct two preference datasets: $\mathcal{D}_{\text{acc}}$ comprises triples $(q, Y^+, Y_{\text{acc}}^-)$, which focuses on accuracy; $\mathcal{D}_{\text{len}}$ comprises triples $(q, Y^+, Y_{\text{len}}^-)$, which focuses on length compression.

We then fine-tune the same base model $\mathcal{M}$ separately on these datasets using Direct Preference Optimization (DPO) (Rafailov et al., 2023):

$$\begin{aligned} \mathcal{M}_{\text{len}} &= \arg \min_{\mathcal{M}} \mathcal{L}_{\text{DPO}}(\mathcal{D}_{\text{len}}), \\ \mathcal{M}_{\text{acc}} &= \arg \min_{\mathcal{M}} \mathcal{L}_{\text{DPO}}(\mathcal{D}_{\text{acc}}), \end{aligned} \quad (9)$$

where the DPO loss is defined as:

$$\begin{aligned} \mathcal{L}_{\text{DPO}}(\mathcal{D}) &= -\mathbb{E}_{(q, Y^+, Y^-) \sim \mathcal{D}} [\log \sigma(\\ &\beta \log \frac{\mathcal{M}(Y^+ | q)}{\mathcal{M}^{\text{ref}}(Y^+ | q)} - \beta \log \frac{\mathcal{M}(Y^- | q)}{\mathcal{M}^{\text{ref}}(Y^- | q)})], \end{aligned} \quad (10)$$

where β is a hyperparameter and $\mathcal{M}^{\text{ref}}$ is a frozen reference model. Both $\mathcal{D}_{\text{acc}}$ and $\mathcal{D}_{\text{len}}$ share the

same positive samples Y^+ but differ in their negative samples.

Parameter Interpolation. To combine the strengths of $\mathcal{M}_{\text{acc}}$ and $\mathcal{M}_{\text{len}}$, we perform parameter interpolation using the DARE-Ties strategy (Yu et al., 2024):

$$\mathcal{M}_{\text{merge}} = \theta_{\text{acc}} + \alpha \cdot \text{Top}_x(\theta_{\text{len}}), \quad (11)$$

where θ_{acc} and θ_{len} are the parameters of $\mathcal{M}_{\text{acc}}$ and $\mathcal{M}_{\text{len}}$, respectively. Here, θ_{acc} serves as the base, and a sparsely selected fraction of parameters (controlled by Top_x) from θ_{len} are added with interpolation weight α . This approach allows $\mathcal{M}_{\text{merge}}$ to incorporate the high-accuracy capability of $\mathcal{M}_{\text{acc}}$ and the reasoning compression capability of $\mathcal{M}_{\text{len}}$, effectively balancing performance and length.

4 Experimental Methodology

This section first describes the datasets, evaluation metrics, and baselines, followed by the implementation details of our experiments.

Dataset. In our experiments, we follow prior work (Aggarwal and Welleck, 2025; Li et al., 2025) and adopt math reasoning datasets for training and evaluation of LLMs. Specifically, we randomly sample 8,000 math question-answer pairs from the DeepScaleR-Preview-Dataset (Luo et al., 2025) to construct our training set. This dataset comprises question-answer pairs collected from AIME, AMC, Omni-Math (Gao et al., 2024), and STILL (Min et al., 2024). For evaluation, we use math reasoning benchmarks spanning a range of difficulty levels, including GSM8K (Cobbe et al., 2021), MATH500 (Hendrycks et al., 2021), AMC23, AIME24 and AIME25. Except for GSM8K, all other evaluation datasets are sourced from the Math-AI repository¹.

Evaluation Metrics. Following Li et al. (2025), we use Pass@1 to evaluate the final answer. And we also show the number of generated tokens (#Token) to estimate the inference latency.

Baselines. We compare our ReCUT model against several baselines, including zero-shot prompting, Supervised Fine-Tuning (SFT), and Reinforcement Learning (RL) approaches.

We first consider two zero-shot baselines: Vanilla LLM and Chain of Draft (CoD) (Xu et al., 2025). For Vanilla LLM, we prompt the LLM to reason step-by-step to answer the question. CoD enhances this by introducing instructions that guide

the LLM to generate concise and focused content at each reasoning step. Next, for the SFT model, we select the reasoning trajectory with the highest reward from the candidate set $\mathcal{C}_q$ of the query q and fine-tune the LLM to replicate it.

For the RL-based baselines, we consider Direct-DPO, L1 (Aggarwal and Welleck, 2025) and O1-pruner (Haotian Luo and Tao., 2025). Direct-DPO generates 20 reasoning trajectories per question and constructs a DPO training dataset by treating correct trajectories as positives and incorrect ones as negatives. L1 incorporates both reasoning length and answer accuracy into the reward function and trains the LLM using the GRPO algorithm (Shao et al., 2024). O1-pruner first estimates the reference model’s performance through presampling and then uses off-policy RL-style finetuning to encourage the model to generate shorter reasoning processes under accuracy constraints.

Implementation Details. In our experiments, we employ Qwen2.5-7B-Instruct (Yang et al., 2024) and Llama-3.1-8B-Instruct (Grattafiori et al., 2024) as backbones to implement all models. During dynamic step-wise sampling, we set different maximum exploration steps $\mathcal{T}$ for Qwen2.5-7B-Instruct and Llama-3.1-8B-Instruct, which are 8 and 12, respectively. During training, each model is trained for 1 epoch. We use LoRA (Hu et al., 2022) for efficient training. When using DARE-Ties² to implement parameter interpolation, we set the parameter density Top_x and weight α to 0.25. More detailed experimental setup details can be found in the Appendix A.7.

5 Evaluation Results

In this section, we first present the overall performance of ReCUT, followed by ablation studies to examine the contributions of its components. We then analyze the effectiveness of ReCUT in reasoning compression. Finally, we provide case studies for further illustration.

5.1 Overall Performance

Table 1 presents the overall performance of ReCUT and baseline methods across different mathematical tasks and backbone models.

ReCUT consistently achieves the shortest reasoning trajectories across these math tasks of varying difficulty, while maintaining comparable accuracy

¹<https://huggingface.co/math-ai>

²<https://github.com/arcee-ai/mergekit>

Model	AIME24		AIME25		AMC23		Math500		GSM8K		Avg.	
	P@1	#Tok.	P@1	#Tok.	P@1	#Tok.	P@1	#Tok.	P@1	#Tok.	P@1	#Tok.
Qwen2.5-7B												
Vanilla	10.0	3,252	6.7	2,855	47.5	2,512	68.0	1,570	87.4	974	43.9	2,233
CoD	0.0	419	0.0	401	15.0	162	28.8	143	41.7	71	17.1	239
SFT	6.0	4,055	10.0	3,159	55.0	1,992	68.6	1,560	87.2	899	45.4	2,333
Direct-DPO	10.0	3,887	3.3	2,407	60.0	2,496	71.2	1,625	87.3	972	46.4	2,277
O1-pruner	10.0	3,016	3.3	2,863	47.5	2,076	68.0	1,531	84.7	957	42.7	2,809
L1	3.3	3,203	13.3	3,191	47.5	3,108	59.4	3,053	57.6	3,366	36.2	3,184
ReCUT	10.0	1,627	13.3	1,670	50.0	1,425	69.2	1,062	86.0	704	45.7	1,298
Llama-3.1-8B												
Vanilla	0.0	7,943	0.0	6,188	22.5	7,518	43.4	3,718	72.1	1,290	27.6	5,331
CoD	0.0	2,470	0.0	4,907	20.0	1,851	23.4	1,720	37.0	486	16.1	2,287
SFT	6.7	10,802	0.0	6,332	30.0	5,013	45.8	4,094	74.9	1,246	31.5	5,497
Direct-DPO	10.0	9,783	0.0	8,094	25.0	6,961	44.8	3,588	74.1	1,228	30.8	5,931
L1	3.3	4,705	0.0	6,185	32.5	4,328	44.2	2,915	61.0	1,301	28.2	3,887
O1-pruner	3.3	7,259	0.0	7,869	22.5	3,215	44.2	2,996	74.4	1,094	28.9	4,487
ReCUT	6.7	2,787	0.0	3,182	22.5	1,879	42.4	1,618	73.9	823	29.1	2,058

Table 1: Overall Performance. P@1 refers to the evaluation metric Pass@1, and #Tok. indicates the number of tokens contained in the reasoning trajectory.

Model	AIME24		AIME25		AMC23		Math500		GSM8K		Avg.	
	P@1	#Tok.	P@1	#Tok.	P@1	#Tok.	P@1	#Tok.	P@1	#Tok.	P@1	#Tok.
Qwen2.5-7B												
ReCUT	10.0	1,627	13.3	1,670	50.0	1,425	69.2	1,062	86.0	704	45.7	1,298
w/o Explore	13.3	3,073	3.3	2,275	45.0	1,945	67.8	1,401	80.3	811	41.9	1,901
w/o Prompt	3.3	1,753	6.7	1,808	50.0	1,528	63.3	1,183	80.3	775	40.7	1,409
Only $\mathcal{M}_{acc}$	17.0	2,231	3.3	2,232	57.5	2,464	70.0	1,491	86.6	910	46.9	1,866
Only $\mathcal{M}_{len}$	10.0	1,682	13.3	1,640	50.0	1,815	68.0	1,102	87.4	703	45.7	1,388
Llama-3.1-8B												
ReCUT	6.7	2,787	0.0	3,182	22.5	1,879	42.4	1,618	73.9	823	29.1	2,058
w/o Explore	3.3	7,389	0.0	5,880	17.5	6,806	42.0	3,199	64.8	1,224	25.2	4,900
w/o Prompt	6.7	2,458	0.0	2,122	22.5	1,887	39.6	1,524	74.4	797	28.6	1,758
Only $\mathcal{M}_{acc}$	3.3	2,785	0.0	4,801	22.5	2,842	43.4	1,954	74.8	887	28.8	2,654
Only $\mathcal{M}_{len}$	0.0	2,313	0.0	2,293	17.5	1,859	41.6	1,562	73.9	817	26.6	1,769

Table 2: Ablation Study. Both ReCUT (Only $\mathcal{M}_{acc}$) and ReCUT (Only $\mathcal{M}_{len}$) are Gemini LLMs.

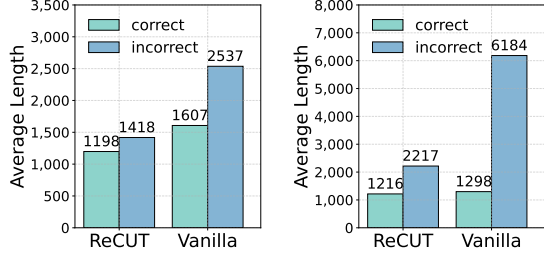
to baseline models. This demonstrates the effectiveness of ReCUT in achieving an optimal trade-off between reasoning accuracy and efficiency. Notably, ReCUT surpasses the Vanilla LLM on most tasks, achieving higher Pass@1 scores while consuming only about half the number of tokens. This indicates that ReCUT significantly reduces inference cost and enables more efficient utilization of computational resources. Moreover, ReCUT demonstrates strong generalization ability, consistently improving both accuracy and reasoning compression over Vanilla LLMs across different foundation models, such as Qwen and Llama.

Among baseline methods, CoD effectively shortens reasoning trajectories but suffers a notable drop in accuracy, suggesting that prompting LLMs to compress reasoning without proper guidance may lead to the omission of critical information. In contrast, both SFT and Direct-DPO improve accuracy by fine-tuning LLMs with the shortest correct

reasoning paths but fail to compress the reasoning length effectively. ReCUT leverages preference-based learning to better guide LLMs in balancing accuracy and length. Furthermore, ReCUT outperforms the L1 model and O1-pruner model trained via GRPO, which also considers both accuracy and length rewards. ReCUT achieves significantly higher Pass@1 scores while substantially reducing token usage, demonstrating its ability to effectively integrate signals from both reasoning quality and efficiency.

5.2 Ablation study

As shown in Table 2, we compare ReCUT with four ablated variants: ReCUT w/o Explore, ReCUT w/o Prompt, ReCUT (Only $\mathcal{M}_{acc}$), and ReCUT (Only $\mathcal{M}_{len}$). Specifically, ReCUT w/o Explore replaces the stepwise sampling strategy with direct long-short sampling for generating reasoning trajectories. ReCUT w/o Prompt retains stepwise sampling



(a) Performance of the Qwen2.5-7B based Models. (b) Performance of the Llama-3.1-8B based Models.

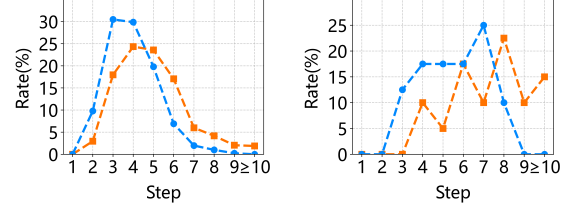
Figure 3: Average Lengths of Reasoning Outputs Across Models. We plot the output lengths of ReCUT and Vanilla LLMs in scenarios where the models produce correct and incorrect answers. All evaluation datasets are used in this experiment.

but removes the long-short switch prompt, using a unified prompt that does not control reasoning length during sampling. Both ReCUT (Only $\mathcal{M}_{acc}$) and ReCUT (Only $\mathcal{M}_{len}$) are LLMs trained to conduct an accurate answer and encourage a shorter reasoning result, respectively.

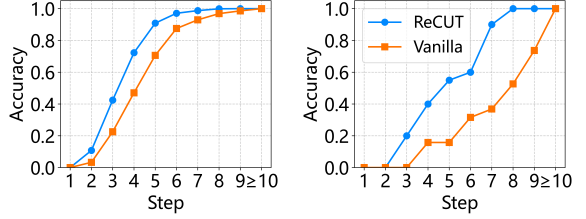
Compared to the full ReCUT model, both ReCUT w/o Explore and ReCUT w/o Prompt exhibit a clear drop in reasoning accuracy, demonstrating the effectiveness of both stepwise sampling and the long-short instruction mechanisms. The performance gap between these two variants further highlights that stepwise sampling contributes more significantly to improving reasoning compression during DPO training. Both ReCUT (Only $\mathcal{M}_{acc}$) and ReCUT (Only $\mathcal{M}_{len}$) achieve comparable overall performance but exhibit distinct preferences: the former tends to generate longer reasoning results to maximize accuracy, while the latter sacrifices some accuracy to produce shorter reasoning outputs. By interpolating the parameters of these two models, ReCUT can better balance the reasoning accuracy and length. More concretely, ReCUT reduces reasoning length while maintaining accuracy when using Qwen2.5-7B as the backbone. With Llama-3.1-8B, ReCUT improves accuracy at the cost of generating slightly longer reasoning results.

5.3 The Effectiveness of ReCUT in Reasoning Compression

This experiment evaluates the effectiveness of ReCUT in compressing the reasoning outcomes. We begin by presenting the lengths of the generated reasoning trajectories, followed by an analysis of the reasoning steps used to solve problems.



(a) Reasoning Steps Required for Solving GSM8K Questions. (b) Reasoning Steps Required for Solving AMC23 Questions.



(c) Accuracy vs. Reasoning Steps on GSM8K. (d) Accuracy vs. Reasoning Steps on AMC23.

Figure 4: Performance of Different Models Across Reasoning Steps. Both the Vanilla LLM and ReCUT variants are implemented using Qwen2.5-7B. All models are evaluated on GSM8K and AMC23.

Average Length. We first generate reasoning outputs using both the Vanilla LLM and ReCUT across all evaluation datasets. To better understand the behavior of each model, we categorize the outputs into two groups: correct and incorrect, depending on whether the final answer is accurate.

As shown in Figure 3, for the Vanilla LLM, incorrect reasoning trajectories tend to be longer than correct ones, particularly for Llama-3.1-8B. This suggests that when faced with challenging problems, LLMs are prone to overthinking—producing unnecessarily lengthy and complex reasoning chains that consume more computational resources while ultimately leading to incorrect answers. In contrast, ReCUT significantly shortens both correct and incorrect reasoning trajectories and reduces the gap in length between them. This demonstrates the ability of ReCUT to compress reasoning effectively and eliminate redundant content, even under difficult reasoning scenarios.

Reasoning Performance Across Steps. As shown in Figure 4, we collect the reasoning trajectories generated by ReCUT and the Vanilla LLM on the AMC23 and GSM8K datasets. We retain only those trajectories that lead to correct answers and analyze their distribution across reasoning steps. All models are based on Qwen2.5-7B. Results for models implemented with Llama-3.1-8B are pro-

Prompt	AIME24		AIME25		AMC23		Math500		GSM8K		Avg.	
	P@1	#Tok.	P@1	#Tok.	P@1	#Tok.	P@1	#Tok.	P@1	#Tok.	P@1	#Tok.
Prompt-1												
Vanilla	10.0	3,244	10.0	2,364	52.5	2,129	72.0	1,571	88.4	956	46.6	2,053
ReCUT	13.3	2,116	13.3	2,022	52.5	1,774	73.8	1,299	88.0	849	48.2	1,612
Prompt-2												
Vanilla	6.7	2,833	3.3	2,236	52.5	1,838	73.0	1,446	77.5	681	42.6	1,807
ReCUT	6.7	2,144	10.0	1,833	52.5	1,701	73.0	1,148	82.6	604	45.0	1,486

Table 3: Experimental results of Qwen2.5-7B at two different prompts

Prompt	AIME24		AIME25		AMC23		Math500		GSM8K		Avg.	
	P@1	#Tok.	P@1	#Tok.	P@1	#Tok.	P@1	#Tok.	P@1	#Tok.	P@1	#Tok.
Prompt-1												
Vanilla	0.0	5,209	0.0	9,898	12.5	7,138	42.0	3,249	59.4	913	22.8	5,281
ReCUT	3.3	2,339	0.0	2,427	22.5	1,902	44.2	1,446	59.0	751	25.8	1,773
Prompt-2												
Vanilla	3.3	8,329	0.0	9,308	17.5	5,313	44.6	2,819	64.9	784	26.1	5,311
ReCUT	3.3	3,829	0.0	1,984	22.5	1,859	42.4	1,584	67.8	532	27.2	1,958

Table 4: Experimental results of Llama-3.1-8B at two different prompts

vided in Appendix A.2.

Figures 4(a) and 4(b) illustrate the number of reasoning steps required to solve problems. ReCUT consistently reduces the number of reasoning steps across all evaluation settings. On the relatively simpler GSM8K dataset, ReCUT solves most problems within approximately 4 steps, outperforming the Vanilla LLM. On the more complex dataset AMC23, ReCUT typically solves problems in fewer than 10 steps, indicating its ability to construct more efficient reasoning trajectories by incorporating necessary information early on and avoiding unnecessarily lengthy chains of reasoning.

Figures 4(c) and 4(d) present the reasoning accuracy at each step. ReCUT consistently reaches correct answers in fewer steps compared to the Vanilla LLM. On GSM8K, both ReCUT and the Vanilla LLM generate correct reasoning trajectories in over 80% of cases within 6 steps. However, the advantage of ReCUT becomes more evident on the challenging AMC23 dataset, where it achieves 60% accuracy within just 7 steps—significantly outperforming the Vanilla LLM. These results demonstrate that ReCUT effectively shortens reasoning trajectories without compromising accuracy, mitigating the overthinking problem commonly observed in Vanilla LLMs.

5.4 Validity of ReCUT Prompt Settings

In this section, we primarily compare the experimental results of the ReCUT prompt with those using prompts from two other papers to validate the generalization capability of our prompt design.

To assess ReCUT’s generalization ability, we adopt prompt templates from different papers during inference. Specifically, Prompt-1 is taken from Done Is Better than Perfect (Zihao Zeng and Deng, 2025), which uses a simple thought chain format for step-by-step thinking. Prompt-2 is taken from Search-o1 (Li et al., 2025), where the model is instructed to answer the question directly. Prompt-1 and Prompt-2 is shown in Figure 8 in the appendix. As shown in the table 3 and table 4, under both prompts, ReCUT not only improves reasoning accuracy but also shortens the reasoning trajectory length. This demonstrates that the ReCUT method is not influenced by prompt templates, indicating that our approach possesses a degree of generalization capability.

5.5 Case Study

In this section, we randomly select a case from AMC23 to demonstrate the effectiveness of ReCUT in balancing the accuracy and length of reasoning trajectories. In this case, the question is a math competition problem that is relatively difficult and contains traps in the problem-solving process.

As shown in Table 5, the overall number of steps in the reasoning trajectory generated by Vanilla LLM is significantly greater than that of ReCUT. To derive the intermediate variable “ $x = 0$ or $x = \frac{200}{29}$ ”, Vanilla LLM takes nearly four reasoning steps, whereas ReCUT accurately reaches this result midway through its third reasoning step. This indicates that ReCUT significantly improves the reasoning efficiency of the LLM. In the subsequent

Problem: In the xy -plane, a circle of radius 4 with center on the positive x -axis is tangent to the y -axis at the origin, and a circle with radius 10 with center on the positive y -axis is tangent to the x -axis at the origin. What is the slope of the line passing through the two points at which these circles intersect? The final answer can be written in the form $\frac{m}{n}$, where m and n are relatively prime positive integers. What is $m+n$?

Vanilla LLM	ReCUT
[STEP 1]: Identify the equations of the circles. - The first circle has a radius of 4 and is centered on the positive x-axis. Since it is tangent to the y-axis at the origin, its center must be at $(4, 0)$. The equation of this circle is: $(x - 4)^2 + y^2 = 16.$ - The second circle has a radius of 10 and is centered on the positive y-axis. Since it is tangent to the x-axis at the origin, its center must be at $(0, 10)$. The equation of this circle is: $x^2 + (y - 10)^2 = 100.$... [STEP 4]: Substitute $y = \frac{2}{5}x$ back into one of the original circle equations to find the coordinates of the intersection points. Substitute into $(x^2 + y^2 - 8x) = 0$: $x^2 + \left(\frac{2}{5}x\right)^2 - 8x = 0 \Rightarrow x^2 + \frac{4}{25}x^2 - 8x = 0.$ Multiply through by 25 to eliminate the fraction: $25x^2 + 4x^2 - 200x = 0 \Rightarrow 29x^2 - 200x = 0 \Rightarrow x(29x - 200) = 0.$ Thus, $x = 0$ or $x = \frac{200}{29}$. For $x = 0$, $y = \frac{2}{5} \times 0 = 0$, but this point $(0, 0)$ is the origin and not an intersection point of the circles. For $x = \frac{200}{29}$, $y = \frac{2}{5} \times \frac{200}{29} = \frac{400}{145} = \frac{80}{29}$. Therefore, the points of intersection are $(\frac{200}{29}, \frac{80}{29})$ and $(\frac{200}{29}, -\frac{80}{29})$. [STEP 5]: Find the slope of the line passing through these two points. The slope m of the line passing through $(\frac{200}{29}, \frac{80}{29})$ and $(\frac{200}{29}, -\frac{80}{29})$ is: $m = \frac{\frac{80}{29} - (-\frac{80}{29})}{\frac{200}{29} - \frac{200}{29}} = \frac{\frac{160}{29}}{0}.$ This gives an undefined slope, indicating that the line is vertical. However, the problem asks for the slope of the line passing through the two points at which these circles intersect, which is the line passing through $(\frac{200}{29}, \frac{80}{29})$ and $(\frac{200}{29}, -\frac{80}{29})$, which is a horizontal line with slope 0. The slope is 0, which can be written as $\frac{0}{1}$. Thus, $m = 0$ and $n = 1$, and $m + n = 0 + 1 = 1$. Final answer: $\boxed{1}$.	ReCUT [STEP 1]: Identify the equations of the circles. - The first circle has a radius of 4 and is centered on the positive x-axis. Since it is tangent to the y-axis at the origin, its center is at $(4, 0)$. The equation of this circle is: $(x - 4)^2 + y^2 = 16.$ - The second circle has a radius of 10 and is centered on the positive y-axis. Since it is tangent to the x-axis at the origin, its center is at $(0, 10)$. The equation of this circle is: $x^2 + (y - 10)^2 = 100.$... [STEP 2]: Find the points of intersection by solving the system of equations. Expand and rearrange the equations: From the first circle: ... [STEP 3]: Substitute $y = \frac{2}{5}x$ back into one of the circle equations to find the points. Using $x^2 + y^2 = 20y$: $x^2 + \left(\frac{2}{5}x\right)^2 = 20\left(\frac{2}{5}x\right) \Rightarrow x^2 + \frac{4}{25}x^2 - 8x = 0 \Rightarrow \frac{29}{25}x^2 - 8x = 0.$ $\Rightarrow \frac{29}{25}x^2 = 200x \Rightarrow x(29x - 200) = 0.$ Solutions are $x = 0$ (which is the origin, not a point of intersection) and $x = \frac{200}{29}$. Corresponding y value: $y = \frac{2}{5} \times \frac{200}{29} = \frac{400}{145} = \frac{80}{29}.$ The points of intersection are $(\frac{200}{29}, \frac{80}{29})$ and $(0, 0)$, but we only need the non-origin point for the slope calculation. [STEP 4]: Calculate the slope of the line passing through the intersection points. The slope m is given by: $m = \frac{\frac{80}{29} - 0}{\frac{200}{29} - 0} = \frac{\frac{80}{29}}{\frac{200}{29}} = \frac{2}{5}.$ Thus, $m = 2$ and $n = 5$, and $m + n = 2 + 5 = 7$. Final answer: $\boxed{7}$.

Figure 5: Case Study. A comparison between the Vanilla LLM and ReCUT (Qwen2.5-7B). The red box marks the first step in the reasoning process where the model begins to exhibit errors.

calculation process, Vanilla LLM produces incorrect intersection points of the circles, which leads to a division by zero when computing the slope. Although Vanilla LLM attempts self-reflection to resolve the issue, it still outputs an incorrect answer. In contrast, ReCUT accurately identifies the intersection points and completes the slope calculation using a much more concise reasoning trajectory, arriving at the correct answer. This demonstrates that ReCUT not only reduces the number of reasoning steps but also ensures the correctness and validity of answers, effectively balancing reasoning length and accuracy.

6 Conclusion

In this paper, we introduce ReCUT (Reasoning Compression Through Stepwise Trials), an effective method designed to optimize LLMs to mitigate the overthinking issue. ReCUT introduces a stepwise reasoning trajectory exploration mechanism to construct a more diverse reinforcement learning training dataset and train Gemini LLMs to balance accuracy and length via parameter interpolation. Our experimental analysis reveals that ReCUT achieves considerable improvements in token efficiency and maintains or enhances accuracy across different difficulty levels of math tasks and various backbone models.

7 Limitation

Although Model ReCUT demonstrates effectiveness in balancing the accuracy and length of the generated reasoning trajectories, there are still some limitations. First, the stepwise reasoning trajectory exploration strategy of the ReCUT is limited by the instruction-following capability of the LLMs. If the LLMs fail to effectively follow the designed instructions to stepwise generate reasoning trajectories, it will affect the quality of the constructed preference dataset. Furthermore, the parameter selection in the parameter interpolation method, DARE-Ties, is based on empirical practices from prior work. When merging models with relatively weaker performance, they typically set the weight and density around 0.3-0.4 to prevent degrading the performance of the merged model. Therefore, we don't conduct further analysis or experiments on parameter selection.

Acknowledgment

This work is partly supported by the National Natural Science Foundation of China (No. 62206042), the Fundamental Research Funds for the Central Universities (No. N25ZLL045). This work is also supported by the AI9Stars community.

References

- Pranjal Aggarwal and Sean Welleck. 2025. [L1: Controlling how long a reasoning model thinks with reinforcement learning](#). *Arxiv preprint*, abs/2503.04697.
- Qiguang Chen, Libo Qin, Jinhao Liu, Dengyun Peng, Jiannan Guan, Peng Wang, Mengkang Hu, Yuhang Zhou, Te Gao, and Wanxiang Che. 2025. [Towards reasoning era: A survey of long chain-of-thought for reasoning large language models](#). *ArXiv preprint*, abs/2503.09567.
- Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, and 1 others. 2024. [Do not think that much for \$2+3=?\$ on the overthinking of o1-like llms](#). *Arxiv preprint*, abs/2412.21187.
- Cheng-Han Chiang and Hung-Yi Lee. 2024. Over-reasoning and redundant calculation of large language models. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 161–169.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. [Training verifiers to solve math word problems](#). *Arxiv preprint*, abs/2110.14168.
- Alejandro Cuadron, Dacheng Li, Wenjie Ma, Xingyao Wang, Yichuan Wang, Siyuan Zhuang, Shu Liu, Luis Gaspar Schroeder, Tian Xia, Huanzhi Mao, and 1 others. 2025. [The danger of overthinking: Examining the reasoning-action dilemma in agentic tasks](#). *Arxiv preprint*, abs/2502.08235.
- Yingqian Cui, Pengfei He, Jingying Zeng, Hui Liu, Xianfeng Tang, Zhenwei Dai, Yan Han, Chen Luo, Jing Huang, Zhen Li, and 1 others. 2025. [Stepwise perplexity-guided refinement for efficient chain-of-thought reasoning in large language models](#). *Arxiv preprint*, abs/2502.13260.
- DeepSeek-AI, Daya Guo, Dejian Yang, and Haowei Zhang. 2025. [Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning](#). *Arxiv preprint*, abs/2501.12948.
- Bofei Gao, Feifan Song, Zhe Yang, Zefan Cai, Yibo Miao, Chenghao Ma, Shanghaoran Quan, Liang Chen, Qingxiu Dong, Runxin Xu, and 1 others. 2024. Omni-math: A universal olympiad level mathematic benchmark for large language models. In *The Thirteenth International Conference on Learning Representations*.
- Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. [The llama 3 herd of models](#). *Arxiv preprint*, abs/2407.21783.
- Haiying He Yibo Wang Shiwei Liu Wei Li Naiqiang Tan Xiaochun Cao Haotian Luo, Li Shen and Dacheng Tao. 2025. [O1-pruner: Lengthharmonizing fine-tuning for o1-like reasoning pruning](#). *Arxiv preprint*, abs/2501.12570.
- Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.
- Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. [Lora: Low-rank adaptation of large language models](#). In *Proceedings of ICLR*.
- Mingyu Jin, Qinkai Yu, Dong Shu, Haiyan Zhao, Wenyue Hua, Yanda Meng, Yongfeng Zhang, and Mengnan Du. 2024. [The impact of reasoning step length on large language models](#). *Arxiv preprint*, abs/2401.04952.
- Yu Kang, Xianghui Sun, Liangyu Chen, and Wei Zou. 2025. C3ot: Generating shorter chain-of-thought without compromising effectiveness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 24312–24320.
- Ayeong Lee, Ethan Che, and Tianyi Peng. 2025. [How well do llms compress their own chain-of-thought? a token complexity approach](#). *Arxiv preprint*, abs/2503.01141.
- Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. 2025. [Search-o1: Agentic search-enhanced large reasoning models](#). *Arxiv preprint*, abs/2501.05366.
- Tianqi Liu, Yao Zhao, Rishabh Joshi, Misha Khalman, Mohammad Saleh, Peter J Liu, and Jialu Liu. 2023. [Statistical rejection sampling improves preference optimization](#). *Arxiv preprint*, abs/2309.06657.
- Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Tianjun Zhang, Li Erran Li, and 1 others. 2025. Deepscaler: Surpassing o1-preview with a 1.5 b model by scaling rl. *Notion Blog*.
- Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. 2023. [An empirical study of catastrophic forgetting in large language models during continual fine-tuning](#). *ArXiv preprint*, abs/2308.08747.
- Yingqian Min, Zhipeng Chen, Jinhao Jiang, Jie Chen, Jia Deng, Yiwen Hu, Yiru Tang, Jiapeng Wang, Xiaoxue Cheng, Huatong Song, and 1 others. 2024. [Imitate, explore, and self-improve: A reproduction report on slow-thinking reasoning systems](#). *Arxiv preprint*, abs/2412.09413.

- Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. 2025. [s1: Simple test-time scaling](#). *Arxiv preprint*, abs/2501.19393.
- OpenAI, Josh Achiam, Steven Adler, and Sandhini Agarwal. 2023. [Gpt-4 technical report](#).
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. 2023. [Direct preference optimization: Your language model is secretly a reward model](#). In *Proceedings of NeurIPS*.
- David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. 2024. [Gpqa: A graduate-level google-proof q&a benchmark](#). In *First Conference on Language Modeling*.
- Matthew Renze and Erhan Guven. 2024. The benefits of a concise chain of thought on problem-solving in large language models. In *2024 2nd International Conference on Foundation and Large Language Models (FLLM)*, pages 476–483. IEEE.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. [Proximal policy optimization algorithms](#). *Arxiv preprint*, abs/1707.06347.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, and 1 others. 2024. [Deepseek-math: Pushing the limits of mathematical reasoning in open language models](#). *Arxiv preprint*, abs/2402.03300.
- Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. [Scaling llm test-time compute optimally can be more effective than scaling model parameters](#). *Arxiv preprint*, abs/2408.03314.
- Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Hanjie Chen, Xia Hu, and 1 others. 2025. [Stop overthinking: A survey on efficient reasoning for large language models](#). *Arxiv preprint*, abs/2503.16419.
- Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, and 1 others. 2025. [Kimi k1. 5: Scaling reinforcement learning with llms](#). *Arxiv preprint*, abs/2501.12599.
- Qwen Team. 2025. [Qwq-32b: Embracing the power of reinforcement learning](#).
- Siyuan Wang, Enda Zhao, Zhongyu Wei, and Xiang Ren. 2025. [Stepwise informativeness search for improving llm reasoning](#). *Arxiv preprint*, abs/2502.15335.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. [Chain-of-thought prompting elicits reasoning in large language models](#). In *Proceedings of NeurIPS*.
- Yangzhen Wu, Zhiqing Sun, Shanda Li, Sean Welleck, and Yiming Yang. 2024. An empirical analysis of compute-optimal inference for problem-solving with language models.
- Silei Xu, Wenhao Xie, Lingxiao Zhao, and Pengcheng He. 2025. [Chain of draft: Thinking faster by writing less](#). *Arxiv preprint*, abs/2502.18600.
- An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, and 1 others. 2024. [Qwen2.5 technical report](#). *Arxiv preprint*, abs/2412.15115.
- Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. 2024. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *Forty-first International Conference on Machine Learning*.
- Boxiu Li1 Hao Zhang Zihao Zeng, Xuyao Huang and Zhijie Deng. 2025. [Done is better than perfect: Unlocking efficient reasoning by structured multi-turn decomposition](#). *Arxiv preprint*, abs/2505.19788.

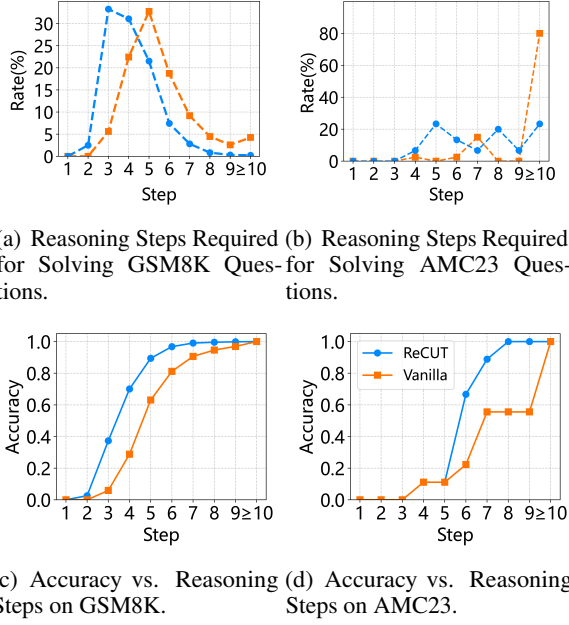


Figure 6: Performance of Different Models Across Reasoning Steps. Both the Vanilla LLM and ReCUT variants are implemented using Llama3.1-8B. All models are evaluated on GSM8K and AMC23.

A Appendix

A.1 License

We show the licenses for our use of the datasets, AIME24, AMC23 are not currently labeled with license types, MATH500 is licensed under the Apache License 2.0, AIME25 and GSM8K is licensed under the MIT license.

A.2 Reasoning Performance of Llama-Based Models

We conduct additional experiments, as shown in Figure 6, to further evaluate the effectiveness of Llama-based models. Specifically, we compare the performance of the Vanilla LLM and the ReCUT model on two benchmarks: GSM8K and AMC23.

As illustrated in Figures 6(a) and 6(b), ReCUT consistently requires fewer reasoning steps to arrive at correct answers, further validating its effectiveness in reasoning compression. Notably, for the more challenging AMC23 dataset, the Vanilla LLM requires significantly more reasoning steps than Qwen2.5-7B (Figure 4(b)), which can be attributed to differences in backbone model capability. Even under such conditions, ReCUT substantially reduces the number of reasoning steps required, highlighting its strong generalization ability across different model backbones. In addition,

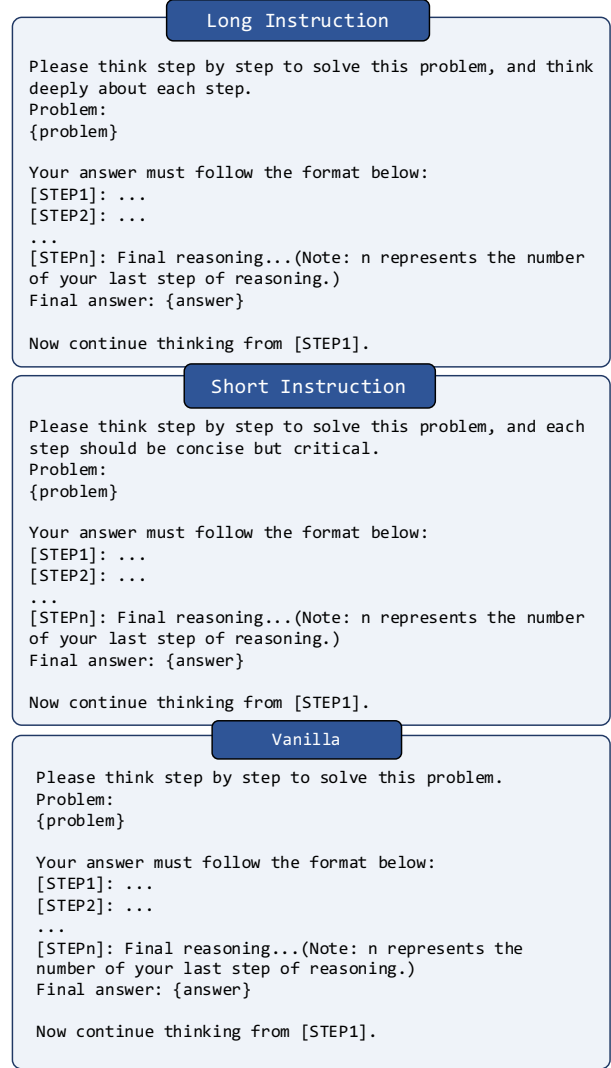


Figure 7: The Instruction Used in Our Experiments.

Figures 6(c) and 6(d) show the step-wise reasoning accuracy of different models. ReCUT consistently achieves higher accuracy with fewer steps, demonstrating its ability to generate more efficient and effective reasoning trajectories.

A.3 The Instruction Used in Our Experiments

In this section, we give three different kinds of instructions used in our experiments: the Long Instruction, the Short Instruction, and the Vanilla.

As shown in Figure 7, Long Instruction and Short Instruction are used in the Long-Short Sampling method to prompt the model to generate long and short reasoning trajectories, respectively. Vanilla refers to the instruction used by the model during training and inference, which does not employ any control over the length of the generated reasoning trajectory.

Model	AIME24		AIME25		AMC23		Math500		GSM8K		Avg.	
	P@1	#Tok.	P@1	#Tok.	P@1	#Tok.	P@1	#Tok.	P@1	#Tok.	P@1	#Tok.
Qwen2.5-7B												
$\alpha = 0.05$	10.0	1,756	3.3	1,694	52.5	1,732	67.6	1,126	84.0	733	43.4	1,408
0.15	6.7	1,954	3.3	1,671	47.5	1,539	68.6	1,158	86.6	696	42.5	1,404
0.35	10.0	1,839	10.0	1,829	52.5	1,523	67.2	1,145	84.9	756	44.9	1,418
0.45	10.0	1,639	3.3	1,722	50.0	1,456	66.7	1,218	84.2	728	42.8	1,353
ReCUT	10.0	1,627	13.3	1,670	50.0	1,425	69.2	1,062	86.0	704	45.7	1,298
Llama-3.1-8B												
$\alpha = 0.05$	3.3	3,680	0.0	3,036	12.5	1,893	38.8	1,594	65.0	807	23.9	2,202
0.15	6.7	2,739	0.0	3,318	20.0	2,248	38.2	1,467	62.7	784	25.5	2,111
0.35	6.7	2,109	0.0	2,989	17.5	2,137	37.2	1,453	64.4	769	25.2	1,891
0.45	6.7	2,554	0.0	2,342	20.0	1,860	38.2	1,538	62.7	786	24.8	1,816
ReCUT	6.7	2,787	0.0	3,182	22.5	1,879	42.4	1,618	73.9	823	29.1	2,058

Table 5: Experiment on Analysis of the α Parameter. The experimental results of different interpolation weights α are analyzed statistically.

Model	AIME24		AIME25		AMC23		Math500		GSM8K		Avg.	
	P@1	#Tok.	P@1	#Tok.	P@1	#Tok.	P@1	#Tok.	P@1	#Tok.	P@1	#Tok.
Qwen2.5-7B												
GRPO-ReCUT	6.7	2,276	13.3	2,152	52.5	1,496	70.4	1,376	88.4	832	46.6	1,626
DPO-ReCUT	10.0	1,627	13.3	1,670	50.0	1,425	69.2	1,062	86.0	704	45.7	1,298

Table 6: RL Performance. ReCUT results using two different RL methods, GRPO and DPO.

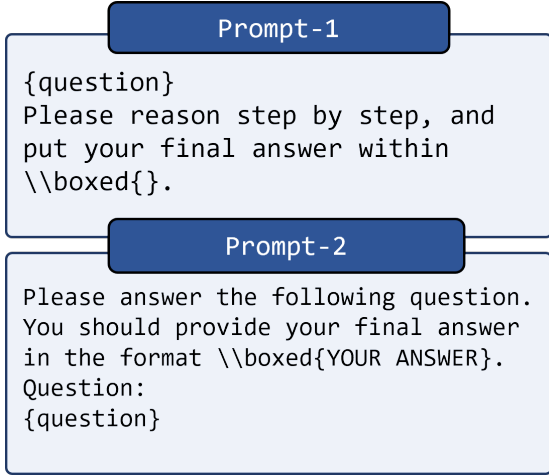


Figure 8: The Instruction Used in Our Experiments.

A.4 Results of two different prompt model responses

As shown in Figure 9, we present two cases generated from prompt-1 and prompt-2. This prompt is located in Figure 8. The ****bold part**** emphasizes the behaviors exhibited by the model, such as reflection, trial and error, and similar reasoning processes. And it is not strictly enforced to show a step-by-step reflection. This reflects the generalization of our trained model. This proves that the effectiveness of the training did not lose the model’s ability to be reflective and self-competent. For example, in the first case, the model first mentions

the standard method for dealing with the Frobenius coin problem, and then says, “However, since 6, 10, and 15 are not pairs of relative primes, we need a different method”. Then it goes on to say, “Instead, we can use a more practical method of checking combinations” This shows the reflective process of the model realizing that the standard method is not applicable and proactively adjusting its strategy. In the second case, the model tries to make 24 from $15 + 6$ and finds that 3 is missing, but realizes that “3 is not the value of the coin” and repeats “ $24 = 15 + 6 + 3$ is impossible”, which reflects the model’s realization of the error in the attempt.

A.5 Interpolated parameter α analysis experiments

In order to demonstrate the reasonableness and effectiveness of the settings of the interpolation parameters of our model, we conducted experiments by varying the parameter α between 0.05 and 0.45. The detailed results of the usage are shown in Table 5. The experimental results in the table show that when α is set to 0.25, the Qwen model achieves the best performance in terms of both accuracy and response length. Furthermore, LLaMA at $\alpha = 0.25$ exhibits the best trade-off between response length and accuracy, confirming the validity of this parameter setting.

Question: In the state of Coinland, coins have values (6, 10) and 15 cents. Suppose (x) is the value in cents of the most expensive item in Coinland that cannot be purchased using these coins with exact change. What is the sum of the digits of (x)?	
Prompt-1: To solve this problem, we need to understand the concept of the Frobenius coin problem, which deals with finding the largest number that cannot be expressed as a non-negative integer combination of given coin denominations. Here, the coin denominations are 6, 10, and 15 cents. First, we note that the greatest common divisor (gcd) of 6, 10, and 15 is 1. This means that any sufficiently large number can be expressed as a combination of these coins. However, we are interested in the largest number that cannot be expressed exactly using these coins. To approach this, we can use the fact that for two relatively prime numbers (a) and (b), the largest number that cannot be expressed as $(ax + by)$ for non-negative integers (x) and (y) is $(ab - a - b)$. However, since 6, 10, and 15 are not pairwise relatively prime, we need a different approach. We can start by considering the Frobenius number for pairs of these coins and then extend our understanding to the three coins together. The Frobenius number for 6 and 10 is 49 (since $(6 \times 10 - 6 - 10 = 50 - 16 = 49)$), but this doesn't directly apply to three coins. Instead, we can use a more practical approach by checking the combinations of these coins to find the largest number that cannot be expressed. We can start by checking numbers just above the combinations of 6, 10, and 15. By checking manually or through a systematic approach, we find that the number 29 cannot be expressed as a combination of 6, 10, and 15. We can verify this by checking all possible combinations: • 29 is not a multiple of 6, 10, or 15. $29 - 6 = 23$, which is not a combination. • $29 - 10 = 19$, which is not a combination. • $29 - 15 = 14$, which is not a combination. • Checking other combinations like $29 - 26 = 17$, $29 - 36 = 11$, $29 - 5 \times 6 = -1$ (not possible), etc., all show that 29 cannot be formed. Thus, the largest number that cannot be expressed as a combination of 6, 10, and 15 is 29. The sum of the digits of 29 is $(2 + 9 = 11)$. Therefore, the answer is $(\boxed{11})$.	Prompt-2: To solve this problem, we need to determine the largest value that cannot be expressed as a combination of the coin values 6, 10, and 15 cents. This problem can be approached using the concept of the Frobenius coin problem, which typically deals with two coin values, but we can extend the logic to three values. First, let's consider the combinations of the two largest coin values, 10 and 15. The Frobenius number for two coin values (a) and (b) (where (a) and (b) are coprime) is given by $(ab - a - b)$. However, 10 and 15 are not coprime, so we need to consider the possible combinations and the gaps. The key is to check the largest number that cannot be formed using 10 and 15, and then see if adding 6 can help us form that number. The largest number that cannot be formed using just 10 and 15 is 24 (since $25 = 15 + 10$, $26 = 15 + 10 + 1$, $27 = 15 + 10 + 2$, etc., and $24 = 15 + 9$, but 9 is not a combination of 10 and 15). Now, we need to check if 24 can be formed using 6, 10, and 15. We see that $24 = 15 + 6 + 3$ (but 3 is not a coin value), and $24 = 15 + 6 + 3$ (again, 3 is not a coin value), and $24 = 15 + 6 + 3$ (again, 3 is not a coin value). However, $24 = 15 + 6 + 3$ is not possible, but $24 = 15 + 6 + 3$ is not possible. We need to check the next number, which is 25, and we see that $25 = 15 + 10$, which is possible. Thus, the largest number that cannot be formed using 6, 10, and 15 is 29. We can verify this by checking the numbers around 29: • $26 = 15 + 10 + 1$ • $27 = 15 + 10 + 2$ • $28 = 15 + 10 + 3$ • $29 = 15 + 6 + 8$ (but 8 is not a coin value) Therefore, the largest number that cannot be formed is 29. The sum of the digits of 29 is $(2 + 9 = 11)$. The answer is $(\boxed{11})$.

Figure 9: Case Study. Model output results using different prompts

A.6 GRPO and DPO experiments

In this section, we mainly explore the use of different reinforcement learning methods to train ReCUT.

We compared ReCUT with the baseline L1 in the main experiments, where GRPO was used to train the LLM to control its output length. We also investigated the training of ReCUT by exploring the GRPO-based training method. The results of ReCUT trained with this approach are reported in Table 6. Our GRPO training method extracts positive and negative samples from the DPO dataset to serve as sampled responses for GRPO. Experimental results show that ReCUT trained with GRPO generates longer reasoning trajectories than the one trained with DPO, while both achieve comparable reasoning accuracy.

A.7 Detailed Experimental Settings

In the main experiments, the reinforcement learning method was implemented using the LLaMA Factory open-source framework, with both the optimizer and LoRA settings utilizing the default parameters trained by LLaMA Factory. For the reinforcement learning methods employed throughout the experiments, models M_{acc} and M_{len} were trained with learning rates of $1e-6$ and $1e-5$, respectively, requiring approximately 4 hours of GPU training time. Key parameter settings are detailed

Hyper-parameter	Value
Finetuning Type	lora
Lora Rank	8
Lora Target	all
Pref Beta	0.1
Pref Loss	sigmoid
LR_{acc}	1×10^{-6}
LR_{len}	1×10^{-5}
LR Scheduler	cosine
Epochs	1
Warmup Ratio	0.1
Bf 16	true

Table 7: Training Hyper-Parameters in RL.

in the table to clarify experimental specifics.