Exploration-Exploitation Reshaping towards Efficient Reasoning for Large Language Models

Yufeng Shi[1*], Weilin Luo[2*], Yuxiang Zhang[2], Zongmeng Zhang[1], Haoyang Liu[1], Yubing Wang[2], Bin Wang[2], Wengang Zhou[1†], Houqiang Li[1†]
[1]University of Science and Technology of China, [2]Noah's Ark Lab, Huawei Technologies
{shiyufeng,zhangzm,dgyoung}@mail.ustc.edu.cn, {zhwg,lihq}@ustc.edu.cn,
{luoweilin3,zhangyuxiang43,wangyubin24,wangbin158}@huawei.com

Abstract

While excelling at solving complex problems, Large Reasoning Models (LRMs) are still constrained by the overthinking issue. Most current studies rely on reward shaping in Reinforcement Learning (RL) to shorten the Chain-of-Thought (CoT) of LRMs, remaining sample-inefficient and non-robust due to the absence of prioritized exploitation and guided exploration. To address these issues, we propose a novel policy optimization framework with **S**elf-**I**mitation and self-**G**uidance **M**ech**A**nisms (SIGMA), which reshapes the exploration and exploitation through two core components: (i) **self-imitation exploitation**, which enables the prioritized exploitation of high-value prompts and rollouts by introducing a self-imitation loss and a dynamic sampling strategy based on compression rate; (ii) **self-guidance exploration**, which provides a preference-aware exploration guidance through diverse and pluggable self-rewriting strategies. Experiments across various datasets indicate that our method achieves superior reasoning efficiency without compromising, and even facilitating, the overall accuracy. Furthermore, ablation studies show that the proposed mechanisms can provide flexible control interfaces for the tradeoff between the reasoning accuracy and efficiency of LRMs.

1 Introduction

Large Reasoning Models (LRMs) achieve test-time scaling through Reinforcement Learning with Verifiable Rewards (RLVR) (Shao et al., 2024), which involves unfolding a chain-of-thought (CoT) reasoning process before delivering the final response. This method is considered to enables LRM to exhibit human-like System-2 slow thinking (Evans, 2003; Li et al., 2025), generating detailed intermediate steps, conducting self-reflection, and even exploring alternative solutions during reasoning. Thus, their ability to solve complex problems such as mathematics (Shao et al., 2024) and commonsense reasoning (Yu et al., 2025a) is significantly enhanced. Models such as Deepseek-R1 (Guo et al., 2025), OpenAI o1 (Jaech et al., 2024), Gemini 2.5 (Comanici et al., 2025), and QwQ (Team, 2025) have demonstrated this capability in practice.

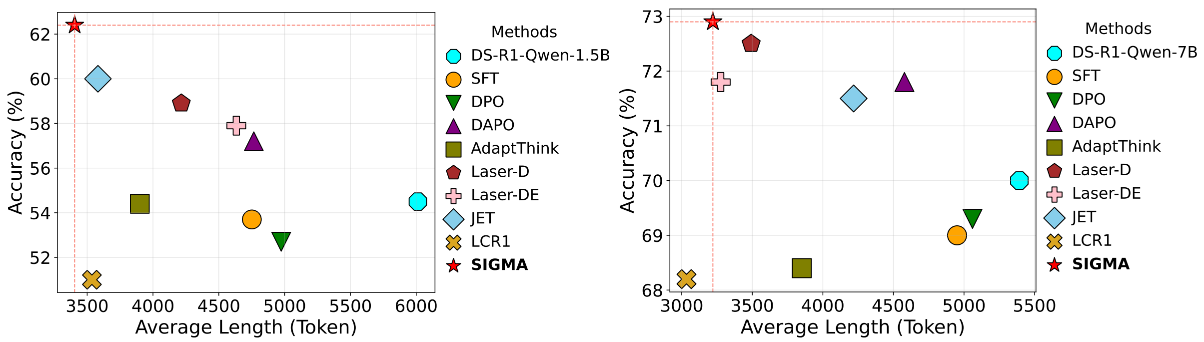
Despite these advances, recent studies have revealed that LRMs typically produce excessively verbose reasoning processes, which is known as the issue of overthinking (Sui et al., 2025). This leads to diminishing returns of test-time scaling and wastes computational resources (Hou et al., 2025). Moreover, additional reasoning steps, such as unnecessary reflection, might even introduce cumulative errors, resulting in incorrect model response (Cuadron et al., 2025). To mitigate this issue, a promising and flexible solution is the RL-based approaches (Team et al., 2025), which jointly optimize accuracy and output conciseness to minimize CoT redundancy with minimal degradation of reasoning performance.

Current researches primarily focus on explicit reward shaping with length penalties in RLVR (Arora and Zanette, 2025; Yi et al., 2025; Liu et al., 2025b; Luo et al., 2025a; Aggarwal and Welleck, 2025; Han et al., 2025a). However, these methods merely apply negative feedback to randomly sampled lengthy responses without explicit guidance to refining a concise reasoning path, resulting in the classical RL issue of inefficient exploitation-exploration (Sutton et al., 1998). To mitigate this issue, inspired by the concept of self-Imitation learning (Oh et al., 2018) in classical RL algorithms, we propose a novel framework with **S**elf-**i**mitation and self-**G**uidance **M**ech**A**nism (SIGMA) that directly reshapes the exploration and exploitation of token-efficient RL.

In terms of exploitation, we strengthen policy optimization by improving utilization mechanism of

*Equal contribution. This work was done when Yufeng Shi was an intern at Noah's Ark Lab, Huawei Technologies.
†Corresponding authors.

Findings of the Association for Computational Linguistics: ACL 2026, pages 30392–30407
July 2-7, 2026 ©2026 Association for Computational Linguistics



(a) Deepseek-R1-Distill-Qwen-1.5B (b) Deepseek-R1-Distill-Qwen-7B

Figure 1: Comparison of our method with baseline methods in terms of average accuracy and average token length across six datasets. Our method achieves higher accuracy with reduced token usage (1.5B and 7B models reduce token consumption by 43.4% and 40.3%, respectively, compared to the base models).

training samples and model rollouts. We propose an efficiency-weighted self-imitation learning strategy that differs from standard correctness-based self-imitation. This strategy drives the policy optimization to focus on samples with concise reasoning pattern by introducing a Supervised Fine-Tuning (SFT) loss according to the compression rate of model rollouts (see Equation 4). In addition, instead of randomly sampling data from training set like RLVR, we also assign a priority to each sample, which decides the sampling probability of the data. These priorities will be updated based on the compression rate of the query at each training iteration, which makes the data sampling a dynamic process. As for exploration, we consider leveraging the model's intrinsic knowledge for preference-aware exploration, reducing low-contribution data produced by undirected model exploration. Specifically, we design an explicit guidance mechanism that prompts the model to regenerate or truncate overly long responses via various self-rewriting strategies, which also mitigates the issue of off-policy to some degree. The proposed self-guidance exploration module can also incorporate a variety of other exploration strategies, offering flexibility for future extensions.

Through explicitly reshaping of exploration and exploitation, evaluations across multiple benchmarks demonstrate that our method significantly enhances the reasoning efficiency. As shown in Figure 1, our method achieves better trade-off between model accuracy and token efficiency.

Our contributions are summarized as follows:

- We propose a novel framework, SIGMA, which introduces various reshaping strategies for exploration and exploitation in RL, providing a combination of regulatory mechanisms for achieving controllable and flexible policy optimization that can satisfy diverse demands.
- We explore several strategies to improve the token efficiency of LRMs without sacrificing model accuracy: (i) enhancing experience exploitation through dynamic sampling and self-imitation learning; (ii) improving data generation quality via self-guidance, directional exploration. In addition, the reshaped exploration mechanism can be easily integrated with various exploration strategies, offering flexibility for future extensions.
- We empirically analyze each component of SIGMA to reveal its impact on the reasoning patterns of LRMs. Experimental results across benchmarks of varying difficulty demonstrate a superior performance on the balance between accuracy and token efficiency compared to existing reward-driven methods.

2 Related Work

2.1 RL with Verifiable Rewards

Reinforcement learning with verifiable rewards (RLVR) significantly enhances the complex reasoning capabilities of large language models (LLMs) by leveraging verifiable signals to provide precise and stable reward shaping. Among these, GRPO (Shao et al., 2024) stands as one of the most influential algorithms, introducing group-normalized policy optimization as an effective alternative to PPO (Schulman et al., 2017). Building upon this foundation, subsequent research has introduced numerous algorithmic improvements (Liu et al., 2025c; Yu et al., 2025b; Zheng et al., 2025;

Chu et al., 2025). For instance, Dr.GRPO (Liu et al., 2025c) mitigates bias by removing variance normalization. DAPO (Yu et al., 2025b) incorporates token-level losses and relaxes policy update constraints by increasing clipping thresholds. Our method builds upon GRPO to jointly optimize model accuracy and token efficiency.

2.2 Efficient Reasoning of LRMs

To mitigate overthinking, numerous studies have proposed methods to improve reasoning efficiency from various perspectives. Prompt-based methods use different prompts, such as instructions with "Be concise" (Renze and Guven, 2024) or token budgets (Han et al., 2025b), to generate concise CoT with less unnecessary reasoning steps. Output-based methods intervene during the model's decoding phase, employing techniques like early-exit (Yang et al., 2025) to accelerate the inference process. SFT-based methods, such as C3oT (Kang et al., 2025), CoT-Value (Ma et al., 2025b), and Token-Skip (Xia et al., 2025), focus on creating training datasets with varying reasoning lengths, emphasizing shorter reasoning, so that models fine-tuned on these datasets produce more concise reasoning. RL-based methods, building upon RLVR, jointly optimize the reasoning efficiency and accuracy of LRMs, and our work falls into this category. Most studies in this category have concentrated on length-based reward shaping (Arora and Zanette, 2025; Yi et al., 2025; Liu et al., 2025b; Luo et al., 2025a; Shen et al., 2025). For example, The LASER series (Liu et al., 2025b) proposes a unified length-based reward shaping framework and further introduces adaptive and difficulty-aware length-based step-wise reward shaping to promote a better trade-off between accuracy and brevity. Our method diverges completely from these methods by reshaping the token-efficient RL problem as a classic exploration-exploitation trade-off in RL, with explicit designs for both aspects to enhance data utilization and model update efficiency.

2.3 RL with Supervised Fine-Tuning

To overcome knowledge limitations of pure RLVR, recent studies have leveraged the high efficiency of supervised fine-tuning (SFT) in extracting knowledge from offline resources (Lv et al., 2025) and explored hybrid methods that combine RL with SFT on external expert data (Ma et al., 2025a; Yan et al., 2025; Liu et al., 2025a; Dong et al., 2025). For instance, ReLIFT (Ma et al., 2025a) alternates between RL and SFT on challenging problems, while LUFFY (Yan et al., 2025) introduces off-policy guidance via mixed-policy optimization with regularized importance sampling to selectively imitate high-quality external responses. Our method shares a similar philosophy, where the self-imitation exploitation module also takes advantage of the efficiency of SFT learning. However, unlike these methods, we do not rely on external knowledge but instead leverage the model's own generated data to enhance data utilization and learning efficiency.

3 Methodology

In this section, we present our method in details. First, we introduce the preliminary (Section 3.1). Then, we present the framework overview of our method (Section 3.2). Finally, we detail the two core modules in our method, i.e., the exploration and exploitation module (Section 3.3 and 3.4).

3.1 Preliminary

Given a prompt q sampled from the dataset $\mathcal{D}$, a large inference model π_θ parameterized by θ, termed the policy model, autoregressively generates a response o of length l according to the probability $\pi_\theta(o|q) = \prod_{i=1}^{l} \pi_\theta(o_i|q, o_{<i})$. Each response o receives a verifiable reward signal $R(q, o)$, valued as 1 if o contains the correct answer and 0 otherwise. The standard optimization objective aims to maximize the expected reward:

$$\max_\theta \mathbb{E}_{q\sim\text{Uniform}(\mathcal{D}), o\sim\pi_\theta(\cdot|q)} \left[R(q, o)\right]. \tag{1}$$

In practice, we often use the training objective from GRPO (Shao et al., 2024) to replace the optimization objective in Equation 1. Specifically, let the initial policy model be π_{ref}, the policiy model before and after the update be $\pi_{\theta_{old}}$ and π_θ respectively. Given an input $q \sim \text{Uniform}(\mathcal{D})$, sample a group of outputs $\{o_1, o_2, \cdots, o_n\} \sim \pi_{\theta_{old}}$, then optimize the π_θ by minimizing the following objective:

$$\begin{aligned}\mathcal{L}_{GRPO}(\theta) = -\frac{1}{n}\sum_{i=1}^{n}\frac{1}{|o_i|}\sum_{t=1}^{|o_i|} \\ \left(\text{CLIP}\left(r_{i,t}(\theta), \widehat{A}_i\right) - \beta\mathbb{D}_{KL}[\pi_\theta \| \pi_{ref}]\right),\end{aligned} \tag{2}$$

where $|o_i|$ denotes the length of the response o_i, $r_{i,t}(\theta) = \pi_\theta(o_{i,t}|q, o_{i,<t})/\pi_{\theta_{old}}(o_{i,t}|q, o_{i,<t})$ is the importance sampling term (Sutton et al., 1999), $\text{CLIP}(r, A, \epsilon) = \min\left(rA, \text{clip}(r, 1-\epsilon, 1+\epsilon)A\right)$ is the clipped surrogate objective (Schulman et al.,

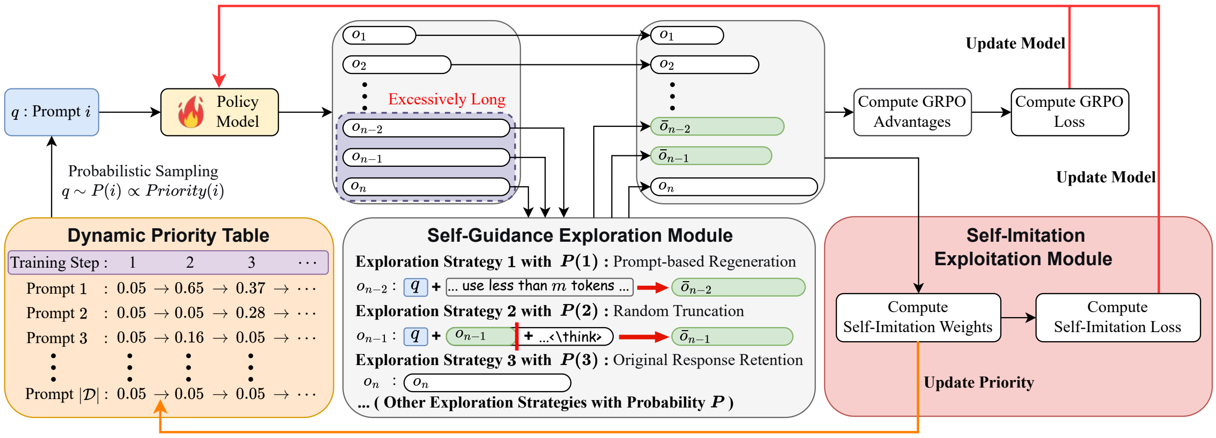


Figure 2: Framework of our proposed method, SIGMA. During RL training, the self-imitation exploitation module, which consists of probabilistic dynamic sampling and self-imitation learning, is employed for both data sampling and model weight updating, while the self-guidance exploration module is employed for policy model rollout.

2017), $\widehat{A}_i = \Big(R(q, o_i) - \text{mean}(\{R(q, o_j\}_{j=1}^{G}\})\Big) / \text{std}(\{R(q, o_j\}_{j=1}^{G}\})$ is the advantage.

3.2 Framework Overview

For the exploitation module, we propose a self-imitation exploitation module incorporating probabilistic dynamic sampling and self-imitation learning. For the exploration module, we design a self-guidance exploration module. In this section, we introduce the overall framework of our method following the data flow, as shown in Figure 2.

First, we design a dynamic priority table that stores the each prompt's priority. A higher priority indicates that the prompt has higher compressibility, i.e., the model is more likely to generate a shorter CoT. Unlike previous methods that sample prompts uniformly from the dataset, we perform probabilistic sampling according to the proportion of priorities, meaning that prompts with higher priorities are sampled with higher probabilities. The prompt q sampled with probability is then fed into the model to generate a group of responses $\{o\}_{i=1}^{n}$ (assuming they are ordered by increasing length).

For excessively long responses $\{o\}_{i=k+1}^{n}$ [1], we feed them into the self-guidance exploration module. This module leverages the model's own knowledge to regenerate or prune these responses, aiming to explore responses within the length limit or shorter within the model's capabilities, thereby alleviating the low data-utilization efficiency caused by undirected exploration. These modified responses $\{\bar{o}\}_{i=k+1}^{n}$, together with unselected responses $\{o\}_{i=1}^{k}$, form a new group.

In the self-imitation exploitation module, we compute the compression ratio for each response in the new group. For responses with positive compression ratios, we compute the self-imitation objective $\mathcal{L}_{SIL}$ and update the priority of the corresponding prompt in the dynamic priority table based on these compression ratios for the next training step. We optimize the policy model by minimizing the GRPO objective and the self-imitation objective, as shown in Equation 3, where the hyperparameter α controls the degree of exploitation.

$$\mathcal{L}_{total} = \mathcal{L}_{GRPO} + \alpha \mathcal{L}_{SIL}. \tag{3}$$

In conclusion, we briefly summarize the interaction mechanism of the three components we proposed: (1) Priority sampling is used to sample problems with higher compression potential, providing high-quality training problems for subsequent self-guidance exploration and self-imitation learning; (2) Self-guidance exploration generates more concise responses based on the high-quality problems provided by dynamic sampling, thereby offering high-quality training responses for subsequent self-imitation learning; (3) Self-imitation learning utilizes the high-quality training data generated in the previous two steps to update the priority of the training problems and model parameters. This allows for better priority sampling in the next iteration and facilitates better self-guidance exploration using the improved model. These 3 components form a virtuous cycle, iteratively refining the policy model to generate more concise reasoning processes. Next, we will provide a detailed description of the design details of these three components.

[1] In practice, we select responses that reach the maximum length limit without providing an answer.

3.3 Self-Imitation Exploitation Module

The self-imitation utilization module incorporates probabilistic dynamic sampling and self-imitation learning. Given a prompt q, the policy model generates corresponding responses $\{o\}_{i=1}^{n}$. We define the compression ratio for each response as follows:

$$C(o_i) = \begin{cases} \left(\frac{mean^* - |o_i|}{mean^*}\right)_+, & \text{if } o_i \text{ is correct} \\ 0, & \text{if } o_i \text{ is incorrect} \end{cases}, \tag{4}$$

where $(\cdot)_+$ denotes $\max(\cdot, 0)$, and $mean^* = \text{mean}\{|o_i| \mid R(q, o_i) > 0\}$ denotes the average length of all correct responses. Note that when all responses $\{o\}_{i=1}^{n}$ are incorrect, there is no need to compute $mean^*$, as all $C(o_i)$ are 0 in this case. We consider that the higher the value of $C(o_i)$, the greater the utility of the response o_i to the model. Therefore, for such data, we perform self-imitation learning inspired by classical RL (Oh et al., 2018) to enable efficient utilization.

Self-Imitation Learning. We use the compression rate $C(o_i)$ as the self-imitation weight for each response o_i, meaning that the larger $C(o_i)$ is, the greater the model's learning effort for that response. Specifically, the current policy model π_θ is optimized using the following self-imitation loss:

$$\mathcal{L}_{SIL}^{\text{origin}} = -\mathbb{E}_{o_i \sim \pi_\theta(\cdot|q)} \left[C(o_i) \log \left(\pi_\theta(o_i|q) \right) \right]. \tag{5}$$

However, note that o_i in Equation 5 requires sampling from the current policy π_θ, whereas in practice we sample from the old policy model $\pi_{\theta_{old}}$. Therefore, similar to PPO (Schulman et al., 2017), we need to perform importance sampling and probability ratio clipping on Equation 5. Given the prompt q and a group of rollouts $\{o_i\}_{i=1}^{n}$, the adjusted self-imitation loss is given as follows:

$$\mathcal{L}_{SIL} = -\frac{1}{\sum_{i=1}^{n} |o_i|} \sum_{i=1}^{n} \sum_{t=1}^{|o_i|} \text{CLIP} \left(r_{i,t}(\theta), C(o_i) \right). \tag{6}$$

For a more concise and correct response o_i, where $C(o_i) > 0$, the SIL loss is non-zero, and both the GRPO loss and the SIL loss are used to optimize the model (note that the gradient directions of the SIL loss and the GRPO loss are consistent in this case). For an incorrect or excessively long response o_i, where $C(o_i) = 0$, the SIL loss is zero, and only the GRPO Loss is used to optimize the model. Thus, self-imitative learning is employed to imitate correct and more concise responses, without explicitly penalizing longer responses.

Probabilistic Dynamic Sampling. As shown in 6, self-imitation learning contributes gradient to the model parameters only when $C(o_i) > 0$. Therefore, we aim to increase the proportion of data where $C(o_i) > 0$ to enhance the model's update efficiency. We maintain a dynamic priority table where each prompt $q_i \in \mathcal{D}$ has a corresponding priority $Priority(i)$. Initially, all priorities are set to a small constant ϵ_{pri}. During self-imitation learning, the priority of a prompt is updated as:

$$Priority(i) = \max \{C(o_{i,j})\}_{j=1}^{n}, \tag{7}$$

where $o_{i,j}$ denotes the j-th response to prompt q_i. During sampling, prompts are selected from the dataset $\mathcal{D}$ according to the probability distribution shown in Equation 8:

$$P(i) = \frac{Priority(i)}{\sum_{i=1}^{|\mathcal{D}|} Priority(i)}. \tag{8}$$

If all responses $o_{i,j}$ for the problem q_i are incorrect, then each $C(o_{i,j}) = 0$. According to Equation 7, the priority for q_i is $Priority(i) = 0$. In the actual implementation, when updating the priority of q_i, we assign a small constant $\epsilon_{pri} = 0.05$ to the problem q_i as its priority. This dynamic sampling mechanism ensures that data with higher compressibility value is utilized with greater probability, which enhances model update efficiency. For a prompt sampled multiple times, as the model updates, the compression rate of the model-generated responses gradually decreases, thereby lowering its priority. Consequently, other data with greater compressibility value are subsequently chosen for training.

3.4 Self-Guidance Exploration Module

Length-based reward methods typically employ undirected model exploration, which may generate excessive long responses offering limited benefits for optimizing token length. We propose to leverage the model's internal knowledge to guide a more directed exploration, thereby improving the quality of generated data and enhancing the update efficiency of self-imitation learning. Specifically, responses that reach the maximum length without providing an answer are considered to offer limited benefits for token efficiency optimization. Inspired by prompt-based and output-based efficient reasoning strategies, we design two directed exploration strategies using the model's own capabilities, along with a strategy to retain the original response (as we observed in experiments that such negative samples

Methods	GSM8K		MATH500		AIME24		AMC		Olympiad		GPQA-Diamond		AVG	
	ACC	Length	ACC	Length	ACC	Length	ACC	Length	ACC	Length	ACC	Length	ACC	Length
DeepSeek-R1-Distill-Qwen-1.5B														
Base	76.0	468	79.6	3617	28.7	11046	63.3	7644	47.0	7679	32.3	5619	54.5	6012
SFT	81.4	559	78.8	2591	27.7	9139	57.2	5394	42.5	5532	34.9	5288	53.7(-0.8)	4751(-21.0)
DPO	80.2	530	78.6	2652	24.0	9966	59.0	5482	44.0	5929	30.3	5283	52.7(-1.8)	4974(-17.3)
DAPO	80.0	826	85.8	3106	26.7	8583	66.3	5666	46.6	5822	37.9	4591	57.2(+2.7)	4766(-20.7)
AdaptThink	82.0	772	79.6	1905	23.7	7434	58.7	3983	49.8	4706	32.3	4601	54.4(-0.1)	3900(-35.1)
Laser-D	84.9	1073	85.2	2424	30.0	7271	65.8	4355	53.2	4813	34.3	5352	58.9(+4.4)	4215(-29.9)
Laser-DE	84.1	1179	84.2	2798	29.7	7960	65.2	5018	50.5	5265	33.8	5575	57.9(+3.4)	4633(-22.9)
LC-R1	75.0	443	77.6	1851	19.0	7155	56.4	3897	44.0	4193	33.8	3678	51.0(-3.5)	3536(-41.2)
JET	83.8	605	83.0	2072	32.0	6641	66.1	3872	51.6	4121	43.4	4182	60.0(+5.5)	3582(-40.4)
SIGMA	81.0	403	84.6	1661	36.7	6924	77.5	3060	54.8	3979	39.8	4402	**62.4(+7.9)**	**3405(-43.4)**
DeepSeek-R1-Distill-Qwen-7B														
Base	87.0	469	92.0	2918	51.3	9812	78.9	6013	63.1	6782	47.5	6359	70.0	5392
SFT	87.3	438	91.4	2658	48.7	8914	78.6	5836	62.7	6441	45.5	5418	69.0(-1.0)	4951(-8.2)
DPO	86.1	438	90.0	2590	53.0	9552	77.2	5797	60.4	6465	49.0	5519	69.3(-0.7)	5060(-6.2)
DAPO	90.1	583	91.6	2720	53.3	8414	81.7	4903	63.4	5361	51.0	5485	71.8(+1.8)	4578(-15.1)
AdaptThink	88.9	304	87.8	1325	50.7	8131	77.2	3871	61.3	4656	44.4	4820	68.4(-1.6)	3851(-28.6)
Laser-D	91.6	965	92.0	1950	52.7	6361	82.8	3505	64.7	3755	51.0	4417	72.5(+2.5)	3492(-35.2)
Laser-DE	91.5	948	92.4	1942	53.0	5809	82.9	3357	64.6	3713	46.5	3884	71.8(+1.8)	3276(-39.2)
LC-R1	86.0	386	87.6	1313	50.0	6329	76.5	3173	59.1	3575	50.0	3429	68.2(-1.8)	3034(-43.7)
JET	86.1	324	91.2	2091	54.0	7981	81.0	4301	63.9	5083	52.5	5530	71.5(+1.5)	4218(-21.8)
SIGMA	89.6	242	90.2	1418	58.0	6275	88.0	2843	63.9	3928	47.8	4622	**72.9(+2.9)**	**3221(-40.3)**

Table 1: Accuracy (ACC) and response length (Length) across different datasets using various methods. AVG denotes the average metric over all datasets, while blue numbers indicate changes relative to the base model.

positively impact RL performance). For a selected response o_i corresponding to a prompt q, we apply one of three modification strategies:

Strategy 1: Prompt-based Regeneration. This strategy discards response o_i and regenerates a new response $\bar{o}_i$ by inserting a token-budget instruction such as "*use less than m tokens*" into the prompt q.

Strategy 2: Random Truncation. For response o_i, we partition it into sentences $o_i = \{s_j\}_{j=1}^{T}$, where s_j is a sentence. Given maximum and minimum truncation lengths T_{max} and T_{min}, a position $k \sim \text{Uniform}(T - T_{max}, T - T_{min})$ is sampled uniformly. The prompt q, the truncated response $o_i = \{s_j\}_{j=1}^{k}$ and a stop-thinking instruction containing "*</think>*" are then concatenated and fed into the model to generate a new response $\bar{o}_i$.

Strategy 3: Original Response Retention. The strategy keeps the original response unchanged.

The specific prompts used in Strategy 1 and 2 are shown in Appendix A. For simplicity, the three strategies above are applied with equal probability ($P(1) = P(2) = P(3) = 1/3$), meaning each strategy is selected with a probability of one-third for modifying a response. In practice, our designed self-guidance exploration module is flexible and extensible: it can incorporate more diverse exploration strategies and adjust probabilities based on actual needs. Exploration strategies may even be invoked conditionally according to user-defined rules rather than solely by probabilistic selection.

4 Experiment

4.1 Settings

Models. Following the common practice established in prior research on token-efficient RL methods (Zhang et al., 2025; Liu et al., 2025b), we select representative and widely-used models, namely DeepSeek-R1-Distill-Qwen-1.5B and DeepSeek-R1-Distill-Qwen-7B, as backbone models to validate the effectiveness of our method across different model sizes. Experiment on Qwen3-8B is also provided, as described in Section 4.4.

Datasets and Metrics. For the training dataset, we employed the same high-quality mathematical dataset, DeepScaleR-Preview-Dataset(Luo et al., 2025b), as used in prior methods (Zhang et al., 2025; Liu et al., 2025b), which consists of approximately 40k competition-level question-answer pairs. For evaluation, we assess the models on six datasets of varying difficulty levels: GSM8K (Cobbe et al., 2021), MATH500 (Lightman et al., 2023), AIME2024 (MAA Committees), AMC (AI-MO, 2024), Olympiad (He et al., 2024), and GPQA-Diamond (Rein et al., 2024). We conduct 16 experiments across all evaluation datasets. For evaluation metrics, we consider both accuracy and response length. We also report the average accuracy change and average length reduction rate of

Component	GSM8K		MATH500		AIME24		AMC		Olympiad		GPQA-Diamond		AVG	
	ACC	Length	ACC	Length	ACC	Length	ACC	Length	ACC	Length	ACC	Length	ACC	Length
DeepSeek-R1-Distill-Qwen-1.5B														
+ SGE	87.5	1429	86.7	2704	34.6	8326	79.5	4617	54.9	5260	38.7	6472	63.6(+1.2)	4801(+41.0)
w/o SGE	83.8	343	82.6	1241	30.7	5190	74.5	2458	49.7	2577	38.0	2948	59.9(-2.5)	2460(-27.8)
w/o SIL	87.0	1680	86.4	2885	31.3	8553	77.5	4651	54.7	5408	39.5	6405	62.7(+0.3)	4930(+44.8)
w/o Dyn	77.4	244	80.3	1520	38.0	6564	75.5	2889	53.3	3867	39.2	5191	60.6(-1.8)	3379(-0.8)
+ Full	81.0	403	84.6	1661	36.7	6924	77.5	3060	54.8	3979	39.8	4402	62.4	3405
DeepSeek-R1-Distill-Qwen-7B														
+ SGE	93.2	1278	93.3	2508	57.3	7596	90.5	4234	64.0	5352	48.7	6112	74.5(+2.5)	4513(+64.9)
w/o SGE	87.4	96	82.1	557	46.7	3588	78.0	1133	57.1	1713	35.7	1497	64.5(-7.5)	1431(-47.7)
w/o SIL	93.2	1101	93.3	2371	59.3	7502	92.0	3397	66.8	4916	47.9	6182	75.4(+3.4)	4245(+55.1)
w/o Dyn	89.0	181	86.5	850	50.0	5472	84.5	2067	60.1	3192	46.3	3497	69.4(-2.6)	2543(-7.1)
+ Full	90.0	126	86.4	919	57.4	6134	90.0	2174	63.3	3406	44.9	3663	72.0	2737

Table 2: Ablation studies of key components in our method SIGMA.

all methods relative to the base model across all evaluation datasets. Following DeepSeek's model cards, we set the evaluation context length to 16k, temperature to 0.6, and top-p to 0.95 for all models.

Baselines. We compare SIGMA with several methods for efficient reasoning: (1) **SFT** fine-tunes backbones using the shortest correct answer among multiple model-generated responses; (2) **DPO** trains backbones using the shortest and longest correct answers from multiple model-generated responses as preference pairs via direct preference optimization (Rafailov et al., 2023); (3) **DAPO** (Yu et al., 2025b) is a widely used variant of GRPO; (4) **AdaptThink** (Zhang et al., 2025) is an RL algorithm that trains models to automatically switch between thinking and non-thinking modes; (5) **LC-R1** (Cheng et al., 2025) is a RL algorithm introducing a novel collaboration of length reward and a compress penalty. (6) The Laser series (Liu et al., 2025b) trains models with step reward functions based on target length, including **Laser-D**, which adaptively adjusts length rewards according to problem difficulty, and **Laser-DE**, a variant that relaxes length penalties for incorrect responses to encourage further exploration. (7) **JET** (Han et al., 2025a) trains models to actively terminate unnecessary reasoning and use quality-controlled length rewards to better balance conciseness and correctness.

We implement our code based on VeRL(Sheng et al., 2025), with detailed hyperparameters provided in Appendix A.

4.2 Main Results

We conduct a series of experiments across benchmarks of varying difficulty to evaluate the overall performance of the proposed SIGMA framework. We compare our framework to the baseline and eight other different policy optimization methods on two LRMs of distinct scales, i.e., Deepseek-R1-Distill-Qwen2.5-1.5B and Deepseek-R1-Distill-Qwen2.5-7B. For abbreviation, these two LRMs are termed as the 1.5B model and the 7B model, respectively. The comparison results are demonstrated in Table 1.

As presented in Table 1, SIGMA exhibits outstanding performance compared to other policy optimization, improving the average accuracy by 7.9% and 2.9% on both the 1.5B and 7B models and delivering the largest reduction in the average reasoning length (43.4% and 40.3% on the two models). The results illustrate the superiority of SIGMA in acquiring the balance between model accuracy and token efficiency, reflecting the effectiveness of our reshaping strategies for exploration and exploitation mechanism.

4.3 Ablation Study

This section presents detailed evaluations of each module in the reshaped exploration-exploitation mechanism, including self-imitation learning, self-guidance exploration, and dynamic sampling. Then we study the impact of exploration strategies and analyze the hyperparameter α in Equation 3.

4.3.1 Ablations on Key Components

The exploration-exploitation mechanism is critical to RL, as it decides the sample efficiency during the policy optimization and the final model ability. In this part, we test the impact of key components of the reshaped exploration-exploitation mechanism. The relevant results are given in Table 2.

The Table 2 shows that self-imitation exploitation can significantly improve the reasoning efficiency and the average reasoning length decreases from 2737 tokens to 1431 tokens when we remove

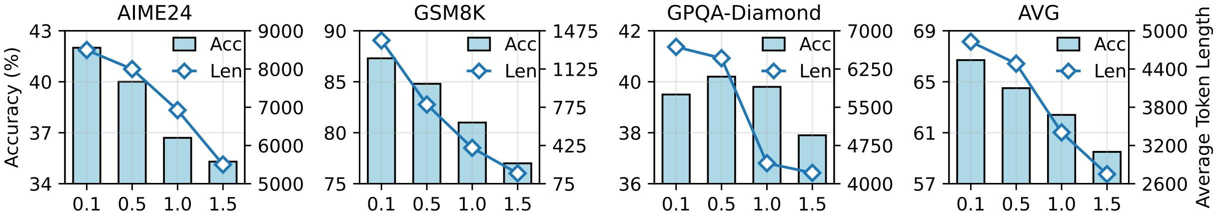


Figure 3: Performance of our method with different α values on DeepSeek-R1-Distill-Qwen-1.5B.

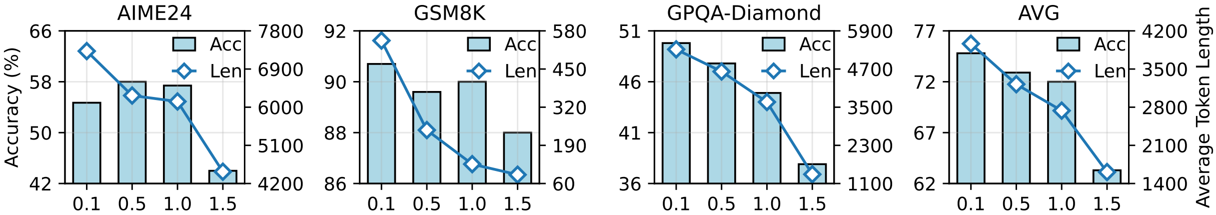


Figure 4: Performance of our method with different α values on DeepSeek-R1-Distill-Qwen-7B.

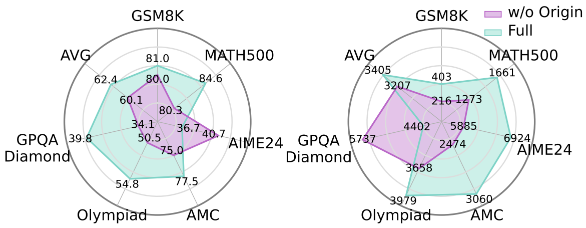


Figure 5: Comparison of accuracy (left) and average token length (right) before and after removing the original response retention on DeepSeek-R1-Distill-Qwen-1.5B.

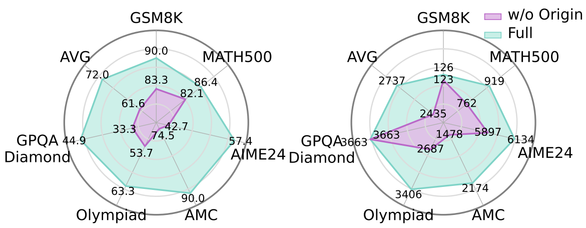


Figure 6: Comparison of accuracy (left) and average token length (right) before and after removing the original response retention on DeepSeek-R1-Distill-Qwen-7B.

the self-guidance exploration module (w/o SGE). However, without our proposed self-guidance strategy, policy optimization fails to explore a high-performance policy distribution. Consequently, the accuracy of 7B model degrades from 72.0% to 64.5%, and the similar trend can also be observed in the results on 1.5B model.

Self-imitation loss provides a specific optimization direction by exploiting the response space more efficiently, which drives the policy model to align with the preference of the selected high-quality response. Without this module (w/o SIL), LLMs still tend to overthink. Although higher accuracy is achieved, this setup induces a 55.1% growth in reasoning costs for 7B and 44.8% for 1.5B, which is hardly tolerable.

Dynamic sampling alters the sampling priority of queries according to their compression ratio. In general, R1-like LLMs are more likely to overthink when addressing complex problems, leading to a higher compression ratio. With the removal of the dynamic sampling module (w/o Dyn), high-difficulty data are no longer preferentially sampled and exploited, resulting in decreased average accuracy—particularly on more challenging benchmarks such as AIME24 and Olympiad. For the 1.5B model on AIME24, SIGMA (36.7%) shows a slight drop in accuracy compared to SIGMA w/o Dyn (38.0%). This may be attributed to the limited capacity of the 1.5B model, which constrains its ability to generate shorter responses for more difficult problems while maintaining correctness. That is, the model's inherent capacity to compress token length without sacrificing accuracy is limited, causing dynamic sampling to favor problems that are easier to compress, such as medium-difficulty problems like Olympiad which the 1.5B model can compress more easily. Thus, this reduces the training demand for more difficult problems.

We also try to remove all our designs on exploitation (+ SGE) , the result is similar to that of w/o self-imitation explitation module. Both LLMs in this configuration retain their "nature" to sacrifice inference efficiency for higher accuracy.

From the aforementioned results, it can be clearly observed that the reshaped exploration and exploitation reflects a tradeoff between reasoning

Methods	GSM8K		MATH500		AIME24		AMC		Olympiad		GPQA-Diamond		AVG	
	ACC	Length	ACC	Length	ACC	Length	ACC	Length	ACC	Length	ACC	Length	ACC	Length
Qwen3-8B														
Base	95.9	2330	95.7	5219	72.0	12088	95.0	7975	73.0	9534	59.7	9577	81.9	7787
SIGMA	95.8	855	95.2	2259	68.0	8080	93.5	4181	72.9	5762	60.3	7024	81.0	4694

Table 3: Accuracy (ACC) and response length (Length) across different datasets using our method on Qwen3-8B. AVG denotes the average metric over all datasets.

accuracy and efficiency. The mechanisms we designed essentially provide flexible and convenient control interfaces for such a tradeoff, allowing the framework to satisfy diverse demands of various application scenarios. Further empirical analysis on the interaction mechanism of the three components can be found in Appendix C.

4.3.2 Impact of Exploration Strategies

In the proposed self-guidance exploration module, various exploration strategies can offer a diversified guide to policy optimization. Despite the different self-guidance strategies, we still need to figure out whether responses sampled from the raw output distribution are necessary to be retained in the response group. In this part, we investigate the efficacy of exploration based on the raw distribution by using prompt-based regeneration and random truncation to modify half of the original responses, respectively. The results are given in Figure 5 and 6. It shows that although the two exploration strategies improve the efficiency of LLM reasoning, reducing the average length by 5.7% on the 1.5B model and by 11.0% on the 7B model, completely dropping the original responses can lead to more severe off-policy issue. Especially for the 7B model, the average accuracy falls from 72.0% to 61.6%.

4.3.3 Parameter Analysis

To further illustrate the influence of the self-imitation loss term and find an optimal hyperparameter setting, we perform a sensitivity analysis of the coefficient before our self-imitation (SI) loss in this part. As shown in Figure 3 and 4, the larger the coefficient of SI loss, the higher the model reasoning efficiency while the lower the accuracy. Interestingly, the decreasing trend of the 7B model's accuracy is not as significant as that of its average output length when the coefficient increases from 0.5 to 1. We attribute this phenomenon to the superior capacity of the 7B model. Due to that, LLMs no longer rely on test-time scaling to boost accuracy. Instead, they are able to correctly answer complex questions through concise reasoning paths. Thus, they are more likely to benefit from larger self-imitation loss term, approaching a Pareto optimality in the trade-off between efficiency and accuracy.

4.4 Experiments on Qwen3-8B

We also conduct experiments with our method on Qwen3-8B, and the experimental results are shown in Table 3. As can be seen from the table, although the base model already achieves strong performance, our method reduces the number of tokens without significantly degrading the base model's performance(the average token count across six datasets decreases from 7787 to 4694, representing a 39.7% reduction in token usage). The slight decrease in accuracy on the Qwen3-8B model is because Qwen3-8B is a more capable model, achieving an accuracy of 80% or even higher on the training dataset (compared to approximately 55% for DeepSeek-R1-Distill-Qwen-7B). As a result, training Qwen3-8B on this dataset does not lead to further improvement in its response accuracy. Even though this training dataset is relatively simple for Qwen3-8B, our method still significantly reduces token length (from 7787 to 4694) without a noticeable drop in accuracy, which demonstrates the effectiveness of our method.

5 Conclusion

We propose a novel framework to explicitly design exploration and exploitation modules, which enables the model trained to balance accuracy and token efficiency. Specifically, we design a self-imitation exploitation module incorporating dynamic sampling mechanism and self-imitation learning, as well as a self-guidance exploration module for directed exploration. Additionally, the exploration module offers flexible extensibility, facilitating the future integration of diverse exploration strategies. Experimental results demonstrate that our method improves inference performance while reducing computational costs.

Limitations

The limitations of our work are as follows: (1) Due to computational resources constraints, we only conduct experiments on 1.5B and 7B models. Applying our method to larger-scale models would be promising. (2) In our experiments, compression rate based priority allocation tends to sample more difficult problems, but it does not distinguish at a finer granularity which specific difficult problems are compressible. Future work may consider explicitly incorporating problem difficulty into the compression rate. (3) Our designed exploitation module exhibits performance degradation when exploitation is excessive (i.e., larger α values). In the future, further exploring how to adaptively control exploitation levels and balance exploitation with exploration holds practical application value. (4) Owing to limited resources and time, the self-guidance exploration module we designed currently incorporates only basic exploration strategies and strategy scheduling. Nevertheless, it provides a framework for integrating various exploration strategies. Investigating more effective exploration strategies and strategy scheduling mechanisms represents a promising future research direction.

References

Pranjal Aggarwal and Sean Welleck. 2025. L1: Controlling how long a reasoning model thinks with reinforcement learning. In *Second Conference on Language Modeling*.

AI-MO. 2024. Amc 2023.

Daman Arora and Andrea Zanette. 2025. Training language models to reason efficiently. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Zhengxiang Cheng, Dongping Chen, Mingyang Fu, and Tianyi Zhou. 2025. Optimizing length compression in large reasoning models. *arXiv preprint arXiv:2506.14755*.

Xiangxiang Chu, Hailang Huang, Xiao Zhang, Fei Wei, and Yong Wang. 2025. Gpg: A simple and strong reinforcement learning baseline for model reasoning. *arXiv preprint arXiv:2504.02546*.

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

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, and 1 others. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.

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 arXiv:2502.08235*.

Yihong Dong, Xue Jiang, Yongding Tao, Huanyu Liu, Kechi Zhang, Lili Mou, Rongyu Cao, Yingwei Ma, Jue Chen, Binhua Li, and 1 others. 2025. Rl-plus: Countering capability boundary collapse of llms in reinforcement learning with hybrid-policy optimization. *arXiv preprint arXiv:2508.00222*.

Jonathan St BT Evans. 2003. In two minds: dual-process accounts of reasoning. *Trends in cognitive sciences*, 7(10):454–459.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, and 1 others. 2025. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638.

Jinyi Han, Ying Huang, Ying Liao, Zishang Jiang, Xikun Lu, Haiquan Zhao, Xinyi Wang, Guanghao Zhou, Sihang Jiang, Jiaqing Liang, and 1 others. 2025a. Your models have thought enough: Training large reasoning models to stop overthinking. *arXiv preprint arXiv:2509.23392*.

Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. 2025b. Token-budget-aware llm reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 24842–24855.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. 2024. OlympiadBench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3828–3850, Bangkok, Thailand. Association for Computational Linguistics.

Bairu Hou, Yang Zhang, Jiabao Ji, Yujian Liu, Kaizhi Qian, Jacob Andreas, and Shiyu Chang. 2025. Thinkprune: Pruning long chain-of-thought of llms via reinforcement learning. *arXiv preprint arXiv:2504.01296*.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, and 1 others. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

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.

Diederik P Kingma. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Zhong-Zhi Li, Duzhen Zhang, Ming-Liang Zhang, Jiaxin Zhang, Zengyan Liu, Yuxuan Yao, Haotian Xu, Junhao Zheng, Pei-Jie Wang, Xiuyi Chen, and 1 others. 2025. From system 1 to system 2: A survey of reasoning large language models. *arXiv preprint arXiv:2502.17419*.

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

Mingyang Liu, Gabriele Farina, and Asuman E. Ozdaglar. 2025a. UFT: Unifying supervised and reinforcement fine-tuning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Wei Liu, Ruochen Zhou, Yiyun Deng, Yuzhen Huang, Junteng Liu, Yuntian Deng, Yizhe Zhang, and Junxian He. 2025b. Learn to reason efficiently with adaptive length-based reward shaping. *arXiv preprint arXiv:2505.15612*.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. 2025c. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*.

Haotian Luo, Li Shen, Haiying He, Yibo Wang, Shiwei Liu, Wei Li, Naiqiang Tan, Xiaochun Cao, and Dacheng Tao. 2025a. O1-pruner: Length-harmonizing fine-tuning for o1-like reasoning pruning. *arXiv preprint arXiv:2501.12570*.

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. 2025b. Deepscaler: Surpassing o1-preview with a 1.5 b model by scaling rl. *Notion Blog*.

Xingtai Lv, Yuxin Zuo, Youbang Sun, Hongyi Liu, Yuntian Wei, Zhekai Chen, Lixuan He, Xuekai Zhu, Kaiyan Zhang, Bingning Wang, and 1 others. 2025. Towards a unified view of large language model post-training. *arXiv preprint arXiv:2509.04419*.

Lu Ma, Hao Liang, Meiyi Qiang, Lexiang Tang, Xiaochen Ma, Zhen Hao Wong, Junbo Niu, Chengyu Shen, Runming He, Yanhao Li, and 1 others. 2025a. Learning what reinforcement learning can't: Interleaved online fine-tuning for hardest questions. *arXiv preprint arXiv:2506.07527*.

Xinyin Ma, Guangnian Wan, Runpeng Yu, Gongfan Fang, and Xinchao Wang. 2025b. CoT-valve: Length-compressible chain-of-thought tuning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6025–6035, Vienna, Austria. Association for Computational Linguistics.

MAA Committees. Aime problems and solutions. `https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions`.

Junhyuk Oh, Yijie Guo, Satinder Singh, and Honglak Lee. 2018. Self-imitation learning. In *International conference on machine learning*, pages 3878–3887. PMLR.

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. *Advances in neural information processing systems*, 36:53728–53741.

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 arXiv:1707.06347*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Yi Shen, Jian Zhang, Jieyun Huang, Shuming Shi, Wenjing Zhang, Jiangze Yan, Ning Wang, Kai Wang, Zhaoxiang Liu, and Shiguo Lian. 2025. DAST: Difficulty-adaptive slow-thinking for large reasoning models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 2322–2331, Suzhou (China). Association for Computational Linguistics.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2025. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, pages 1279–1297.

Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Na Zou, Hanjie Chen, and Xia Hu. 2025. Stop overthinking: A survey on efficient reasoning for large language models. *Transactions on Machine Learning Research*.

Richard S Sutton, Andrew G Barto, and 1 others. 1998. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge.

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. 1999. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12.

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 arXiv:2501.12599*.

Qwen Team. 2025. Qwq-32b: Embracing the power of reinforcement learning.

Heming Xia, Chak Tou Leong, Wenjie Wang, Yongqi Li, and Wenjie Li. 2025. TokenSkip: Controllable chain-of-thought compression in LLMs. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 3351–3363, Suzhou, China. Association for Computational Linguistics.

Jianhao Yan, Yafu Li, Zican Hu, Zhi Wang, Ganqu Cui, Xiaoye Qu, Yu Cheng, and Yue Zhang. 2025. Learning to reason under off-policy guidance. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Chenxu Yang, Qingyi Si, Yongjie Duan, Zheliang Zhu, Chenyu Zhu, Qiaowei Li, Minghui Chen, Zheng Lin, and Weiping Wang. 2025. Dynamic early exit in reasoning models. *arXiv preprint arXiv:2504.15895*.

Jingyang Yi, Justin Wang, and Sida Li. 2025. Shorter-better: Guiding reasoning models to find optimal inference length for efficient reasoning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Longxuan Yu, Delin Chen, Siheng Xiong, Qingyang Wu, Dawei Li, Zhikai Chen, Xiaoze Liu, and Liangming Pan. 2025a. Causaleval: Towards better causal reasoning in language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 12512–12540.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, and 1 others. 2025b. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*.

Jiajie Zhang, Nianyi Lin, Lei Hou, Ling Feng, and Juanzi Li. 2025. AdaptThink: Reasoning models can learn when to think. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 3716–3730, Suzhou, China. Association for Computational Linguistics.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, and 1 others. 2025. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*.

A Experimental Details

Training Details. Our implementation is built upon the VeRL (Sheng et al., 2025) framework. We optimize the policy model using the Adam (Kingma, 2014) optimizer with a learning rate of 1×10^{-6}. During training, we set the train batch size to 128, conduct 8 rollouts per prompt, use a temperature of 0.6, and adopt dynamic batch size with a maximum token length of 10K per GPU. The total training step is set to 300. The hyper-parameter α introduced by our method is selected from $\{0.1, 0.5, 1.0, 1.5\}$. In the main experimental results, we report results with $\alpha = 1.0$ for the DeepSeek-Distill-Qwen-1.5B and $\alpha = 0.5$ for the DeepSeek-Distill-Qwen-7B. Except for experiments analyzing α parameter, all ablation experiments on the All ablation studies on the 1.5B and 7B models, except for the parameter analysis on α, use $\alpha = 1.0$. For experiments on Qwen3-8B, α is set to 1.0. Additionally, we adopt part of the design from DAPO (Yu et al., 2025b), specifically using token-mean mode when computing the GRPO objective, with clipping parameters set to $\epsilon_{low} = 0.2$ and $\epsilon_{high} = 0.28$.

Prompt for Prompt-based Regeneration

Please reason step by step, use less than 6144 tokens, and put your final answer within \\boxed{{}}.

Figure 7: The prompt used in Strategy 1 of the self-guidance exploration module

Prompt for Random Truncation

The number of generated tokens has reached the limit, so stop thinking, start answering and put the final answer within \\boxed{{}}. \n</think>\n\n

Figure 8: The prompt used in Strategy 2 of the self-guidance exploration module

The Prompts Used by the Self-Guidance Exploration Module. The prompt employed for Strat-

α	GSM8K		MATH500		AIME24		AMC		Olympiad		GPQA-Diamond		AVG	
	ACC	Length	ACC	Length	ACC	Length	ACC	Length	ACC	Length	ACC	Length	ACC	Length
DeepSeek-R1-Distill-Qwen-1.5B														
0.1	87.3	1387	88.2	2756	42.0	8497	85.5	4214	57.6	5438	39.5	6684	66.7	4829
0.5	84.8	800	86.9	2324	40.0	7998	79.5	4127	55.7	5192	40.2	6462	64.5	4484
1.0	81.0	403	84.6	1661	36.7	6924	77.5	3060	54.8	3979	39.8	4402	62.4	3405
1.5	77.0	170	80.7	1005	35.3	5502	75.5	2688	50.7	2915	37.9	4213	59.5	2749
DeepSeek-R1-Distill-Qwen-7B														
0.1	90.7	547	93.6	2199	54.7	7323	92.5	3598	67.3	4797	49.8	5318	74.8	3964
0.5	89.6	242	90.2	1418	58.0	6275	88.0	2843	63.9	3928	47.8	4622	72.9	3221
1.0	90.0	126	86.4	919	57.4	6134	90.0	2174	63.3	3406	44.9	3663	72.0	2737
1.5	88.0	90	80.8	622	44.0	4478	73.0	1003	55.8	2075	37.9	1389	63.3	1610

Table 4: Parameter analysis of different α values from Equation 3 on both the DeepSeek-R1-Distill-Qwen-1.5B and DeepSeek-R1-Distill-Qwen-1.5B models. Data with blue backgrounds correspond to the results reported in our main experimental results Table 1.

Strategy	GSM8K		MATH500		AIME24		AMC		Olympiad		GPQA-Diamond		AVG	
	ACC	Length	ACC	Length	ACC	Length	ACC	Length	ACC	Length	ACC	Length	ACC	Length
DeepSeek-R1-Distill-Qwen-1.5B														
full	81.0	403	84.6	1661	36.7	6924	77.5	3060	54.8	3979	39.8	4402	62.4	3405
w/o origin	80.0	216	80.3	1273	40.7	5885	75.0	2474	50.5	3658	34.1	5737	60.1(-2.3)	3207(-5.7)
DeepSeek-R1-Distill-Qwen-7B														
full	90.0	126	86.4	919	57.4	6134	90.0	2174	63.3	3406	44.9	3663	72.0	2737
w/o origin	83.3	123	82.1	762	42.7	5897	74.5	1478	53.7	2687	33.3	3663	61.6(-10.4)	2435(-11.0)

Table 5: Ablation study on exploration strategies for DeepSeek-Distill-Qwen-1.5B and DeepSeek-Distill-Qwen-7B. The blue numbers indicate metric changes relative to the full method employing all three exploration strategies.

egy 1 (Prompt-based regeneration) is shown in Figure 7, and the prompt used for Strategy 2 (Random Truncation) is presented in Figure 8.

B More Detailed Experimental Results

We present here more detailed tabular results of the ablation experiments in Section 4.3. The ablation study on key components is presented in Table 4, and the ablation study on the impact of exploration strategies is shown in Table 5. The results in Table 4 demonstrate the importance of balancing exploration and exploitation. Table 5 suggests that longer original responses positively affect model accuracy, as this enables the RL algorithm to steer away from these incorrect responses during training.

C Analysis of Interaction Mechanism

We tabulate the truncation ratios (when the model's generated responses exceed the maximum token limit, they are truncated) of the responses of the DeepSeek-R1-Distill-Qwen-7B model at different time steps as shown in Table 6. When priority sampling is removed (w/o Dyn), the truncation ratio during training is higher compared to the full method, indicating that high-quality problems positively contribute to the model's ability to generate concise responses. When self-guidance exploration is removed (w/o SGE), the truncation ratio during the early stages of training increases significantly compared to the full algorithm, showing that self-guidance exploration effectively utilizes the model's capabilities to reduce overly long responses. When self-imitation learning is removed (w/o SIL), the truncation ratio remains high during the later stages of training. This is because the model no longer receives learning signals for token length optimization and only updates via the GRPO loss. This highlights the critical role of self-

	Step 0	**Step 100**	**Step 200**	**Step 300**
w/o SGE	0.21	0.08	0.04	0.04
w/o SGE	0.39	0.21	0.06	0.03
w/o SIL	0.24	0.10	0.12	0.09
Full	0.21	0.06	0.02	0.02

Table 6: Truncation rates of DeepSeek-R1-Distill-Qwen-7B at different time steps during training

imitation learning in guiding the model to generate concise responses.

Thus, each component contributes positively to promoting the generation of more concise responses. Furthermore, as shown in Table 6, the full method incorporating all 3 components maintains a consistently low response truncation ratio throughout the training process, enabling the model to more effectively iterate toward generating more concise responses.

D Case Study

As shown in Figure 9, we present a case from the AIME 2024 dataset. DeepSeek-R1-Distill-Qwen-7B consumes a significant amount of tokens (3017 tokens in total) during reasoning to produce the correct answer, often containing unnecessary self-reflections and redundant attempts. In contrast, the model trained with our method uses only 1440 tokens, less than half of the token consumption of the base model, while still arriving at the correct answer. This demonstrates that our method enables models to generate more concise reasoning steps while maintaining accuracy.

To facilitate understanding of the operational boundaries of the self-imitation mechanism, we select an example from the AIME24 dataset in which the DeepSeek-R1-Distill-Qwen-7B model, after SIGAM training with the hyperparameter $\alpha = 1.5$, produces an erroneous response, as illustrated in Figure 10. On an 8×8 grid, moving from the lower left corner to the upper right corner requires 8 right moves (R) and 8 up moves (U). The initial step in the model's reasoning already contains a fundamental error (mistaking the required moves as 7 right and 7 up). Consequently, all subsequent reasoning is based on this incorrect premise. Due to excessive compression, the model's inference lacked the necessary reflection process to correct this fundamental error at the beginning, leading to an incorrect answer. This observation suggests that our current self-imitation mechanism is not yet capable of performing more fine-grained identification of which reflection steps are necessary. This points to a potential direction for further improvement in self-imitation learning.

Question (AIME2024):
Jen enters a lottery by picking 4 distinct numbers from $S = \{1, 2, 3, \cdots, 9, 10\}$. 4 numbers are randomly chosen from S. She wins a prize if at least two of her numbers were 2 of the randomly chosen numbers, and wins the grand prize if all four of her numbers were the randomly chosen numbers. The probability of her winning the grand prize given that she won a prize is $\frac{m}{n}$ where m and n are relatively prime positive integers. Find $m + n$.

Deepseek-R1-Distill-Qwen2.5-7B:
<think> Okay, so Jen is entering this lottery thing by picking 4 distinct numbers from the set S = 1, 2, 3, ..., 10. Then, 4 numbers are randomly chosen from S, and Jen wins a prize if at least two of her numbers match the randomly chosen ones **[...]** But wait, if A is a subset of B, meaning if she wins the grand prize, she definitely wins a prize, then $A \cap B$ is just A **[...]** Wait, let me verify that. 210 - 95 is indeed 115 **[...]** Maybe I made a mistake in computing P(A) or P(B). Let's see. P(A) is the probability that all four numbers match **[...]** Wait, hold on. Let me do that again. $(1/210) \times (42/23) = (42)/(210 \times 23)$. 42 divided by 210 is 1/5, so $(1/5)/23 = 1/(5 \times 23) = 1/115$. Wait, so P(A|B) = 1/115? Hmm, is that correct? Let me think differently. **[...]** Therefore, the answer is 116. Final Answer \boxed{116} **</think>** Jen enters a lottery by picking 4 distinct numbers from the set $S = \{1, 2, 3, \ldots, 9, 10\}$. She wins a prize if at least two of her numbers are among the 4 randomly chosen numbers, and she wins the grand prize if all four of her numbers are chosen. We need to find the probability of her winning the grand prize given that she won a prize, expressed as $\frac{m}{n}$ where m and n are relatively prime positive integers, and then find $m + n$. First, we calculate the total number of possible combinations for choosing 4 numbers from 10, which is $\binom{10}{4} = 210$. The probability of winning the grand prize (event A) is $\frac{1}{210}$. Next, we calculate the probability of winning a prize (event B), which includes winning with exactly 2, 3, or 4 numbers matching. We use the complement rule to find the probability of fewer than two matches (0 or 1 match) and subtract it from 1. - The number of ways to have 0 matches is $\binom{6}{4} = 15$. - The number of ways to have 1 match is $\binom{4}{1} \times \binom{6}{3} = 4 \times 20 = 80$. Thus, the probability of fewer than two matches is $\frac{15+80}{210} = \frac{95}{210}$. Therefore, the probability of at least two matches is $1 - \frac{95}{210} = \frac{115}{210} = \frac{23}{42}$. The probability of winning the grand prize given that she won a prize is: $P(A|B) = \frac{P(A)}{P(B)} = \frac{\frac{1}{210}}{\frac{23}{42}} = \frac{1}{210} \times \frac{42}{23} = \frac{42}{210 \times 23} = \frac{1}{115}$ Thus, the probability is $\frac{1}{115}$, and $m + n = 1 + 115 = 116$. \boxed{116}. **(3017 Tokens)**

SIGMA(Ours):
<think> Okay, so Jen is entering a lottery where she picks 4 distinct numbers from the set S = 1, 2, 3, ..., 9, 10. Then, 4 numbers are randomly chosen from S. Jen wins a prize if at least two of her numbers match the randomly chosen numbers **[...]** But wait, if she wins the grand prize, that means she definitely won a prize, right? Because winning the grand prize is a subset of winning a prize. So, A is a subset of B. Therefore, $P(A \cap B)$ is just P(A) **[...]** They are coprime. So, m + n = 1 + 115 = 116. **Final Answer** \boxed{116} **</think>** Jen enters a lottery by picking 4 distinct numbers from the set $S = \{1, 2, 3, \ldots, 9, 10\}$. The total number of ways to choose 4 numbers from 10 is $\binom{10}{4} = 210$. To find the probability of winning the grand prize (event A), we note that there is only 1 way to choose exactly her 4 numbers. Thus, $P(A) = \frac{1}{210}$. To find the probability of winning a prize (event B), we use the complement rule. The probability of winning a prize is the complement of winning 0 or 1 numbers. - The number of ways to choose 0 matching numbers is $\binom{6}{4} = 15$. - The number of ways to choose 1 matching number is $\binom{4}{1} \times \binom{6}{3} = 4 \times 20 = 80$. Thus, the probability of winning 0 or 1 numbers is $\frac{15+80}{210} = \frac{95}{210}$. Therefore, the probability of winning at least 2 numbers is: $P(B) = 1 - \frac{95}{210} = \frac{115}{210} = \frac{23}{42}$ The conditional probability $P(A|B)$ is: $P(A|B) = \frac{P(A)}{P(B)} = \frac{\frac{1}{210}}{\frac{23}{42}} = \frac{1}{210} \times \frac{42}{23} = \frac{42}{210 \times 23} = \frac{1}{115}$ Thus, the probability is $\frac{1}{115}$, and $m + n = 1 + 115 = 116$. \boxed{116}. **(1440 Tokens)**

Figure 9: Using DeepSeek-R1-Distill-Qwen2.5-7B as the backbone, the responses generated by our SIGMA-trained model and the base model on a problem from AIME 2024. The base model used 3017 tokens to produce the correct answer, while the model trained with our method required only 1440 tokens, effectively reducing redundant tokens.

Question (AIME2024):
Consider the paths of length 16 that follow the lines from the lower left corner to the upper right corner on an 8×8 grid. Find the number of such paths that change direction exactly four times, as in the examples shown below.

SIGMA(Ours) with $\alpha = 1.5$:
<think> I need to count the number of paths on an 8x8 grid from the lower left to the upper right corner that change direction exactly four times. Each path is a sequence of moves, either right or up. Since it's an 8x8 grid, each path must have 15 moves: 7 right and 7 up. Changing direction exactly four times means there are five segments (since each direction change increases the number of segments by one). These segments alternate between horizontal and vertical moves. Since the path starts at the lower left, the first move can be either right or up. Let's consider both cases. Case 1: The path starts with a right move. Then the directions alternate: right, up, right, up, etc. Since there are five segments, the sequence will be right, up, right, up, right. Wait, that's five segments, but starting with right, the directions would be R, U, R, U, R. However, since we need exactly four direction changes, the number of segments is five. But in an 8x8 grid, to go from (0,0) to (7,7), we need 7 right and 7 up moves. Wait, perhaps I should think in terms of runs. Each direction change corresponds to a run switch. So, if there are four direction changes, there are five runs. Since the path starts at the lower left, the first run is either right or up. If it starts with right, the runs are R, U, R, U, R. Similarly, if it starts with up, the runs are U, R, U, R, U. In each case, the number of runs of each direction must add up to 7. For example, if starting with right, the number of right runs is 3 and up runs is 2. Because the sequence is R, U, R, U, R, which is three R runs and two U runs. Similarly, starting with up, it's three U runs and two R runs. So, for each case, we need to distribute 7 right moves into 3 runs and 7 up moves into 2 runs, or vice versa. The number of ways to distribute 7 identical items into 3 distinct runs is C(7-1,3-1) = C(6,2) = 15. Similarly, distributing 7 into 2 runs is C(6,1)=6. Therefore, for each starting direction, the number of paths is 15*6=90. Since there are two starting directions, the total number of paths is 2*90=180. Wait, is that correct? Let me verify. Yes, because if you start with right, you have 3 right runs and 2 up runs. The number of ways to split 7 rights into 3 runs is C(6,2)=15, and 7 ups into 2 runs is C(6,1)=6. So 15*6=90. Similarly for starting with up, it's the same. So total is 180. **</think>** The number of such paths is \boxed{180}.

Figure 10: A response generated by a model trained with SIGMA using DeepSeek-R1-Distill-Qwen2.5-7B as the backbone on a problem from AIME 2024. A high value of α encourages greater reliance on concise responses, which may lead to over-compression and the omission of critical steps. In this example, the model's reasoning was already flawed at the outset, but it skipped the reflection process, thereby failing to correct this fundamental error.