HiPO: Hybrid Policy Optimization for Dynamic Reasoning in LLMs

Anonymous ACL submission

Abstract

Large Language Models (LLMs) increasingly rely on chain-of-thought (CoT) reasoning to improve accuracy on complex tasks. However, always generating lengthy reasoning traces is inefficient, leading to excessive token usage and higher inference costs. This paper introduces the Hybrid Policy Optimization (i.e., HiPO), a framework for adaptive reasoning control that enables LLMs to selectively decide when to engage in detailed reasoning (Think-on) and when to respond directly (Think-off). Specifically, HiPO combines a hybrid data pipeline—providing paired Think-on and Think-off responses—with a hybrid reinforcement learning reward system that balances accuracy and efficiency while avoiding over-reliance on detailed reasoning. Experiments across mathematics and coding benchmarks demonstrate that HiPO can substantially reduce token length while maintaining or improving accuracy. Finally, we hope HiPO can be a principled approach for efficient adaptive reasoning, advancing the deployment of reasoning-oriented LLMs in real-world, resource-sensitive settings.

1 Introduction

Large Language Models (LLMs) have achieved unprecedented success across diverse cognitive tasks, from code generation and mathematical reasoning to scientific problem-solving. A key driver of this progress is the integration of **Chain-of-Thought (CoT)** (Yao et al., 2023; Wei et al., 2023) reasoning—a paradigm where models decompose complex queries into sequential, interpretable steps to derive accurate outputs. These approaches enhance accuracy on challenging problems but also introduce a persistent drawback: **overthinking** (Kumar et al., 2025; Sui et al., 2025; Nayab et al., 2025). Even for trivial queries, models often generate unnecessarily long reasoning chains, leading to inflated token usage, higher latency, and reduced efficiency in interactive applications. This inefficiency creates a fundamental tension between reasoning quality and computational cost, raising the need to adaptively regulate reasoning depth.

Recent work has explored adaptive reasoning control to mitigate overthinking, and can be divided into two categories: (i) training-based adaptive reasoning, where reinforcement learning (RL) (Aggarwal and Welleck, 2025; Arora and Zanette, 2025; Hou et al., 2025; Luo et al., 2025; Shen et al., 2025; Team et al., 2025; Lou et al., 2025) or supervised fine-tuning (SFT) (Munkhdalai et al., 2024; Ma et al., 2025; Chen et al., 2025a; Kang et al., 2025) encourages concise reasoning through length penalties or conciseness rewards; (ii) external control, which constrains reasoning with handcrafted prompts or dynamic instructions (Xu et al., 2025; Renze and Guven, 2024; Chen et al., 2024; Munkhbat et al., 2025). While effective to some extent, these methods suffer from important limitations: coarse supervision signals, monotonic incentives that discourage deeper reasoning on difficult problems, and a lack of principled trade-offs between accuracy, latency, and token efficiency.

To address these challenges, we introduce **HiPO** (**H**ybrid **P**olicy **O**ptimization), a unified framework for adaptive reasoning in LLMs. HiPO is designed to enable models to decide when to "think" (i.e., **Think-on**)and when to skip reasoning (i.e., **Think-off**), thereby striking a balance between correctness and efficiency. Specifically, our approach builds on two key innovations: (1) Hybrid Data Construction Pipeline. As shown in Figure 1, we first collect the training data containing both Think-on and Think-off responses. Each query is automatically categorized based on its difficulty and response correctness. Then, a high-performance model (i.e., DeepSeek-V3 (Liu et al., 2024a)) is used to produce the explicit explanations to justify its reasoning-mode decisions. Finally, for each query, the final response based on the thinking

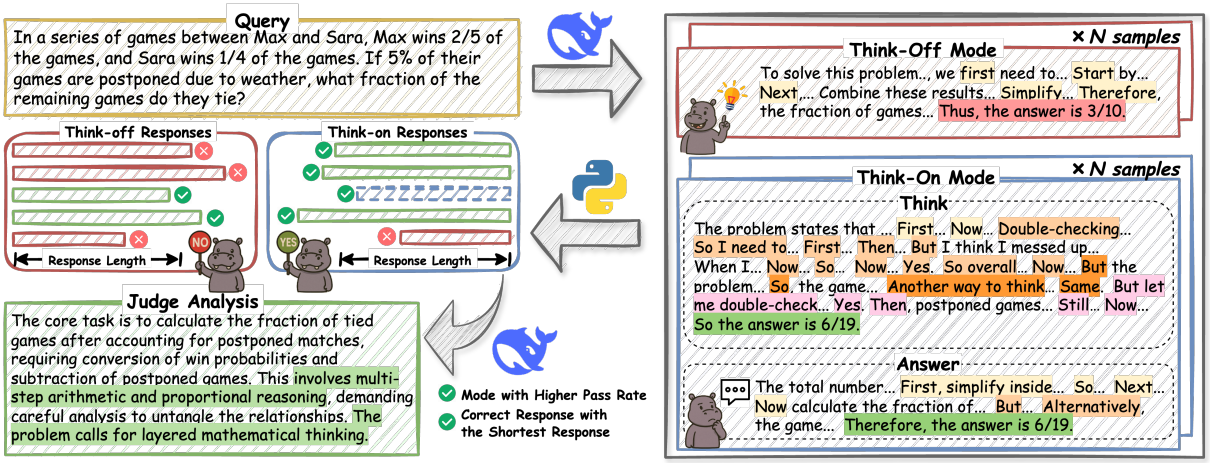


Figure 1: Framework of the hybrid data construction pipeline. For each query, we first produce the think-off and think-on responses and then determine the thinking mode based on predefined criteria. After that, we apply the high-performance model to produce the judge analysis for the corresponding specific mode.

mode and the corresponding explanation construct the hybrid output. (2) Hybrid Reinforcement Learning Reward System. We propose a hybrid reward design that balances Think-on and Think-off decisions. Specifically, a bias adjustment mechanism prevents the model from over-relying on verbose reasoning, while mode-aware advantage functions align reasoning-mode selection with actual performance gains. This ensures stable training and principled control over reasoning depth.

In summary, our contributions are threefold:

- We propose HiPO for adaptive LLM reasoning, which includes the hybrid data construction and hybrid reinforcement learning.
- In the hybrid data construction pipeline, we produce logically rich Think-on and concise Think-off responses with the justification for the thinking mode. Then, for hybrid reinforcement learning, we introduce both the judge analysis and the response reward signal to enable principled control of reasoning depth.
- Experimental results on multiple datasets demonstrate that HiPO can consistently reduce redundant reasoning while improving or maintaining accuracy.

2 Related Works

RL for LLM Reasoning. Recent advances in reinforcement learning (RL) have significantly enhanced LLMs' complex reasoning capabilities, moving beyond supervised fine-tuning (SFT) limitations. State-of-the-art RL algorithms demonstrate superior performance in mathematical reasoning and multi-step problem solving: GRPO (Shao et al., 2024) stabilizes training through intra-group relative reward comparisons; GSPO (Zheng et al., 2025) defines sequence-level importance ratios and applies sequence-level clipping/rewarding/updates to improve efficiency and stabilize MoE training; VAPO (Yue et al., 2025) ensures reward consistency via value-aware optimization; PPO (Schulman et al., 2017) constrains policy updates through clipping mechanisms; and DPO (Rafailov et al., 2024) learns directly from human preferences without explicit reward modeling.

Adaptive Reasoning. Reasoning-oriented LLMs (e.g., CoT (Yao et al., 2023; Wei et al., 2023) and R1-style (DeepSeek-AI et al., 2025) systems) have improved complex problem solving via explicit step-by-step reasoning and self-reflection but also suffer from "overthinking" (Kumar et al., 2025; Sui et al., 2025; Nayab et al., 2025), where simple queries trigger redundant chains that inflate compute, latency, and token usage, hindering interactive deployment. To address this, existing work focuses on: (i) Training-based adaptive reasoning: RL to conditionally trigger CoT, length penalties and conciseness rewards (Aggarwal and Welleck, 2025; Arora and Zanette, 2025; Hou et al., 2025; Luo et al., 2025; Shen et al., 2025; Team et al., 2025; Lou et al., 2025; Zhan et al., 2025), and SFT (Munkhdalai et al., 2024; Ma et al., 2025; Chen et al., 2025a; Kang et al., 2025) to prefer shorter yet correct reasoning; (ii) External control : prompt designs that limit steps or defer CoT (Xu et al., 2025; Renze and Guven, 2024; Chen et al.,

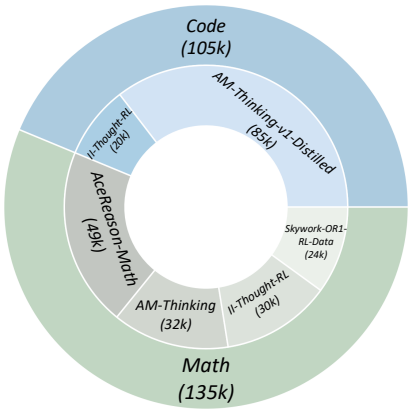


Figure 2: Statistics of Data Sources.

2024; Munkhbat et al., 2025); (iii) Post-hoc Efficiency Optimization: pruning and restructuring chains after generation (Aytes et al., 2025; Xia et al., 2025; Liu et al., 2024b; Sun et al., 2024; Yang et al., 2025). However, existing adaptive reasoning methods lack an explicit, learnable, instance-level mechanism for determining when reasoning is necessary. Their mode-selection signals—whether derived from teacher labels, multi-stage heuristics, or global constraints—do not provide the model with a stable and interpretable decision boundary. This results in limited robustness, reduced interpretability, and unstable gating behavior. To address these challenges, HiPO introduces a framework grounded in endogenous, instance-level judgment and disentangled credit assignment. More discussions are provided in Appendix A.5.

3 Method

Our HiPO framework consists of two important components: (i) a hybrid data construction pipeline that generates training data with both Think-on and Think-off responses; (ii) a hybrid reinforcement learning reward system that combines mode-specific accuracy and global average performance, along with a bias-adjustment mechanism to prevent over-reliance on the Think-on mode.

3.1 Hybrid Data Construction Pipeline

This process begins with a novel data labeling system leveraging state-of-the-art LLMs to assess each query's difficulty and domain characteristics. Queries are then classified into Think-on and Think-off categories based on their intrinsic complexity and the availability of verifiable answers.

3.1.1 Data Source

We construct a corpus for code and mathematics by integrating diverse public sources, as illustrated in Fig. 2, including AM-Thinking-v1-Distilled (Tian et al., 2025), II-Thought-RL (Internet, 2025), AceReason-Math (Chen et al., 2025b), and Skywork-OR1-RL-Data (He et al., 2025).

3.1.2 Data Collection

To effectively enhance the performance of HiPO, we design a structured data construction pipeline aimed at exploring and guiding the model's preference between the Think-on and Think-off reasoning modes. Our training dataset is meticulously curated to be logically rich, cross-domain, and sufficiently challenging.

We adopt a multi-stage data generation process as shown in Figure 1. For each query, the pipeline samples N responses under the Think-on mode and N responses under the Think-off mode using a dedicated reasoning model. All responses are then verified for correctness, and the reasoning mode with the higher pass rate is selected as the preferred mode for that query. Let p_{on} and p_{off} denote the pass rates of the Think-on and Think-off modes, respectively. If the difference in pass rates satisfies $|p_{\text{on}} - p_{\text{off}}| < \delta$, where δ is a predefined threshold, the Think-off mode is selected. This tie-breaking strategy encourages the model to prefer more concise responses when deeper reasoning does not lead to a significant improvement in correctness. For the winning mode, the shortest correct response is retained as the final sample. To expose the model to diverse reasoning scenarios and encourage adaptive behavior, we randomly assign a mode to 1% of the queries, forcing the model to encounter diverse reasoning scenarios. This forces the model to engage with both reasoning styles in varying contexts, which is essential for learning when to switch modes dynamically during inference. Additionally, we incorporate an auxiliary explanation signal to enhance the model's mode alignment capabilities. For each query-response pair, we prompt DeepSeek-V3 (Liu et al., 2024a) to generate a justification explaining why the selected mode is appropriate. This explanation provides a valuable training signal for aligning mode decisions with the underlying reasoning complexity.

3.1.3 Data Format

The training samples follow a unified structure encompassing justification and answer generation. In

Table 1, this design guides the model to decide when reasoning is needed and to generate answers consistent with it. The special tokens are detailed in Table 1, ensuring a clear separation between reasoning and final response for better alignment. Ablation study on mode-explanation supervision is presented in appendix A.7.

3.2 Hybrid RL Reward System

This section details the reinforcement learning process used to teach the model how to effectively balance Think-on and Think-off reasoning modes. The approach is built on a hybrid RL reward system that guides the model's optimization.

3.2.1 Basic Reward Formulation

Consider a group of N sampled responses, for each response $i \in \{1, \ldots, N\}$, we denote its answer correctness by $\mathrm{ACC}_i \in \{0, 1\}$, its format correctness by $\mathrm{FORMAT}_i \in \{0, 1\}$, its basic reward by r_i=$\mathrm{ACC}_i + 0.2 \cdot \mathrm{FORMAT}_i \in \mathbb{R}$, and its reasoning mode by $M_i \in \{\text{on}, \text{off}\}$, where M_i=on indicates the Think-on mode and M_i=off indicates the Think-off mode.

3.2.2 Bias Adjustment Mechanism

A potential risk of the hybrid reward design is that the model may overfit to the more accurate Think-on mode, favoring deep reasoning even when it is unnecessary. This tendency can reduce response efficiency and hinder the flexibility in reasoning behavior. To mitigate this issue, we introduce a bias adjustment mechanism to dynamically regularize the contribution of mode-specific accuracies.

Let $\bar{r}_{\text{on}} = \frac{1}{N_{\text{on}}} \sum_{i:M_i=\text{on}} r_i$ denote the average reward of generated responses under the Think-on mode, and let $\bar{r}_{\text{off}}$ denote the corresponding average for the Think-off mode. We introduce a bias term $b_{\text{off}} = \omega \cdot \bar{r}_{\text{on}}$ to encourage efficient reasoning, where ω controls the bias ratio. The adjustment is applied only when the Think-off mode performs comparably to the Think-on mode, specifically when the performance gap remains within the bias threshold. Let $\delta = \bar{r}_{\text{on}} - \bar{r}_{\text{off}}$ denote the performance gap, the adjusted reward is defined as:

$$\bar{r}'_{\text{off}} = \begin{cases} \bar{r}_{\text{off}} + b_{\text{off}}, & \text{if } 0 \le \delta \le b_{\text{off}}, \\ \bar{r}_{\text{off}}, & \text{otherwise.} \end{cases} \tag{1}$$

This mechanism prevents the model from gaining an unfair advantage by overfitting to the more verbose but more accurate Think-on mode. Moreover, it ensures that the adjusted accuracies remain faithful to the true relative performance between reasoning modes, thereby improving training stability and preserving the intended balance between depth and efficiency.

3.2.3 Supervision RL with HiPO

The final advantage function is formulated as a hybrid signal that integrates both judge analysis and model response. Each response i receives two distinct scalar advantage, including *judge advantage* based on the quality of the mode justification, and *answer advantage* based on correctness and format.

The judge advantage $\mathrm{A}_i^{\text{judge}}$ captures the broader decision-level utility of selecting a particular mode. Let $\bar{r} = \text{mean}(\mathbf{r})$ denote the global mean reward, and define the mode-specific advantages as $\Delta_{\text{on}} = \text{mean}(\mathbf{r}_{\text{on}}) - \bar{r}$ and $\Delta_{\text{off}} = \text{mean}(\mathbf{r}_{\text{off}}) - \bar{r}$. The judge advantage function is then given by:

$$\mathrm{A}_i^{\text{judge}} = \begin{cases} \frac{\Delta_{\text{on}} + \gamma(r_i - \bar{r})}{\text{std}(\mathbf{r})}, & \text{if } M_i = \text{on}, \\ \frac{\Delta_{\text{off}} + \gamma(r_i - \bar{r})}{\text{std}(\mathbf{r})}, & \text{if } M_i = \text{off}. \end{cases} \tag{2}$$

The first term Δ_{M_i} quantifies the global advantage of the chosen mode over the full group average, guiding the model toward selecting modes that yield higher expected rewards. The second term $\gamma \cdot (r_i - \bar{r})$ ensures that the judge's justification is also accountable for the quality of the response under the selected mode, thereby aligning the explanation with actual performance. The normalization factor $\text{std}(\mathbf{r})$ stabilizes the reward signal across groups.

In contrast to the judge advantage function, the advantage A_i^{answer} is computed within the context of the selected reasoning mode. Since the mode M_i has already been determined prior to response generation, it is natural to assess the response quality relative to other responses within the same mode. Let $\bar{r}_{M_i}$ and σ_{M_i} denote the mean and standard deviation of rewards within mode M_i, respectively. The answer advantage is defined as:

$$A_i^{\text{answer}} = \frac{r_i - \bar{r}_{M_i}}{\sigma_{M_i}} \tag{3}$$

This local normalization focuses the learning signal on intra-mode variance, encouraging the model to improve response quality without conflating mode preference.

To assign token-level reward for training with reinforcement learning, we define the final reward for each token t in sample i as follows:

$$\mathrm{A}_{i,t} = \begin{cases} \mathrm{A}_i^{\text{answer}}, & \text{if token } t \in \mathcal{T}^{\text{answer}}, \\ \mathrm{A}_i^{\text{judge}}, & \text{if token } t \in \mathcal{T}^{\text{judge}}. \end{cases} \tag{4}$$

Think-on Mode	Think-off Mode
<judge> {judge_analysis} </judge> <think_on> <think> {thinking_content} </think> <answer> {response} </answer>	<judge> {judge_analysis} </judge> <think_off> <answer> {response} </answer>

Special Token	Description
<judge>	Analyzes input query to determine whether reasoning is required.
<think_on/off>	Specifies whether reasoning should be activated ("on") or skipped ("off").
<think>	Marks the beginning of reasoning in Think-on mode.
<answer>	Marks the beginning of the model's answer.

Table 1: Formatting templates (left) and special tokens with their descriptions (right).

where $\mathcal{T}^{\text{judge}}$ and $\mathcal{T}^{\text{answer}}$ denote the token index sets corresponding to the judge segment and the answer segment, respectively, within each response.

Given a query q, HiPO generates a collection of candidate outputs $\{o_i\}_{i=1}^{G}$ from the old policy $\pi_{\theta_{\text{old}}}$. For each output o_i, let $\mathcal{T}_i$ denote the set of token positions in response i, i.e., $\mathcal{T}_i = \mathcal{T}^{\text{judge}} \cup \mathcal{T}^{\text{answer}}$. We define the per-token probability ratio as $\rho_{i,t} = \frac{\pi_\theta(y_{i,t} \mid h_{i,t})}{\pi_{\theta_{\text{old}}}(y_{i,t} \mid h_{i,t})}$, where $y_{i,t}$ is the t-th generated token in o_i and $h_{i,t}$ is its conditioning context. The policy π_θ is optimized by maximizing the following token-level objective:

$$\begin{aligned}\mathcal{J}(\theta) = \mathbb{E}\Big[q \sim P(Q), \{o_i\}_{i=1}^{G} \sim \pi_{\theta_{\text{old}}}(\cdot \mid q)\Big] \\ \cdot \frac{1}{G}\sum_{i=1}^{G}\frac{1}{|\mathcal{T}_i|}\sum_{t\in\mathcal{T}_i}\Big(\min\big(\rho_{i,t}\,A_{i,t}, \\ \text{clip}(\rho_{i,t},\,1-\epsilon,\,1+\epsilon)\,A_{i,t}\big) \\ -\beta\,\mathbb{D}_{\text{KL}}\big(\pi_\theta(\cdot \mid h_{i,t}) \,\big\|\, \pi_{\text{ref}}(\cdot \mid h_{i,t})\big)\Big).\end{aligned} \tag{5}$$

Here $A_{i,t}$ is the token-level advantage defined in Eq. (4) via segment-wise assignment, and $\mathbb{D}_{KL}$ is the token-level KL between the current policy and the reference policy at context $h_{i,t}$.

3.3 Training Paradigm

Our HiPO framework adopts a two-stage training paradigm, consisting of a **cold-start** stage and a **RL** stage. In the cold-start stage, the model is initialized with high-quality, hybrid training data, including both Think-on and Think-off responses. This stage enables the model to acquire fundamental reasoning and answering capabilities, while establishing an initial balance between analytical reasoning and concise responses. In the RL stage, the model is further optimized using the hybrid reward system, which integrates mode-specific accuracy and global average performance. Together, these two stages ensure that HiPO achieves both strong reasoning ability and token efficiency.

4 Experiments

4.1 Experimental setup

Implementation details. Since Qwen3-8B-Instruct can freely switch between inference modes, we chose it for our experiment. However, when the training data is insufficient, training Qwen3-8B-Instruct can easily lead to a decline in performance on the test set (Details can be found in the Appendix A.2). To address this, we conducted Cold-Start tuning to stabilize its performance with relatively large datasets. For the Cold-Start stage, we use the "AM-Thinking-v1-Distilled", "AceReason-Math", "AM-Thinking", "II-Thought-RL(math)" dataset for training. The parameters are set as: maximum learning rate is 8e-5, minimum learning rate is 8e-6 and batch size is 512. For the RL stage, we use the "II-Thought-RL(code)", "Skywork-OR1-RL-Data" dataset for training. The parameters are set as: batch size = 16, maximum response length = 32k, N = 16, ω = 0.01, and $\gamma = 0.3$. Besides, the δ in the data collection stage is set as 0.2.

Baselines. We designed the following baselines for comparison. (1) **Cold-Start**: We perform Cold-Start on the model using the data construction method in Section 3.1. (2) **Cold-Start (On)**: We apply Cold-Start only using the collected data under the Think-on mode. (3) **Cold-Start (On) + GRPO**: We further train the **Cold-Start (On)** model using GRPO. (4) **Cold-Start + GRPO**: We further train the **Cold-Start** model with GRPO. (5) **HiPO**: We train the model following HiPO. (6) **AdaptThink**: We reproduced the code provided in (Zhang et al., 2025). (7) **AutoThink**: We reproduced the code provided in (Tu et al., 2025).

Evaluation benchmarks. We conducted tests on AIME2024, AIME2025, HumanEval (Chen et al., 2021), LiveCodeBench V6 (Jain et al., 2024), MBPP (Austin et al., 2021), MATH-500 (Light-

man et al., 2023), and GPQA-Diamond (Rein et al., 2023) with AVG@32 evaluation setting.

4.2 Main Results

In Table 2, we observe that training the model solely on Think-on data leads the model to engage in reasoning for problems of any difficulty. We use this baseline as a typical example of "overthinking" for comparison. After applying GRPO to the Cold-Start (on) model, there is a significant improvement in accuracy, with an average accuracy increase of 3.1%. However, this does not reduce the token length and thinking rate of the model. On the contrary, to achieve higher accuracy, the token length on simpler datasets increases significantly. When training the model on a dataset containing both Think-on and Think-off data, the accuracy of the resulting Cold-Start model improves by 4.0% compared to the Cold-Start(on) model, while the token length and thinking rate decrease by 10.8% and 22%, respectively. After applying GRPO to the Cold-Start model, there is no significant change in performance. However, when applying HiPO to train the Cold-Start model, the accuracy improves by 6.2%, while the token length and thinking rate decrease dramatically by 30% and 39%, respectively. Moreover, results show that HiPO outperforms existing methods on both efficiency and accuracy. Stability analysis is presented in the appendix A.6. Analysis of Generalization of HiPO on long-context, tool-use, and general science tasks is presented in the Appendix A.8.

4.3 Ablation Study

Effect of selecting the shortest response In the data construction pipeline, we select the shortest response (Cold-Start (Shortest)) as the final sample. To analyze the effect of this strategy, we additionally propose two variants called (**Cold-Start (Longest)** and **Cold-Start (Random)**) by selecting the longest responses and randomly selecting the responses, respectively. In Figure 3(a), Cold-Start (Shortest) shows an improvement in accuracy compared to both Cold-Start (Longest) and Cold-Start (Random), with a decrease in both the Thinking ratio and Token length. Therefore, we adopt this Cold-Start (Shortest) strategy for Cold-Start.

Effect of design strategies for $\mathrm{A}_i^{\textbf{judge}}$ and $\mathrm{A}_i^{\textbf{answer}}$ In the reinforcement learning stage, first, we utilize the term $\text{mean}(\mathbf{r}_{M_i}) - \text{mean}(\mathbf{r})$ to quantify the **global advantage** of the chosen mode over the full group average. Second, the **local normalization** based on the mode-specific mean and standard deviation is used for $\mathrm{A}_i^{\text{answer}}$. To demonstrate the effect of these strategies, as shown in Table 3, we design two variants (i.e., HiPO (w/o global adv) and HiPO (w/o local norm)). For HiPO (w/o global adv), we directly remove the global advantage for $\mathrm{A}_i^{\text{judge}}$. For HiPO (w/o local norm), we just use the global normalization across the responses in a group. In Table 3, we observe that HiPO achieves significant improvements in performance and efficiency when compared to these two variants.

Effect of different γ values. Figure 3(b) shows that, when the value of γ is set to 0.00, the reward for the judge token lacks information about the current response, resulting in lower model accuracy and higher token length. On the other hand, when γ is set too high, the scales of the two terms (*mean*($\mathbf{r}_{\text{off}}$) - *mean*($\mathbf{r}$)) and $(r_i - \textit{mean}(\mathbf{r}))$ become imbalanced, which leads to a decrease in model accuracy and an increase in token length.

Effect of different rollout numbers Figure 4 shows that, when the rollout number N is set to 16, the model achieves better average performance, shorter token length, and lower think rate. We attribute this to the fact that this configuration provides sufficient data to explore diverse possibilities while avoiding excessive samples with redundant reasoning that dilute the training signal. As a result, the model focuses more on learning from higher-quality samples, leading to a more concise strategy with improved accuracy, reduced token length, and lower think rate.

Effect of different ω values Figure 4 shows that, setting ω to 0.01 provides a balanced trade-off between performance and efficiency. This configuration mitigates the overly conservative behavior seen at 0.0 while avoiding the overly aggressive behavior at higher settings, and achieving the largest efficiency gains with minimal performance loss.

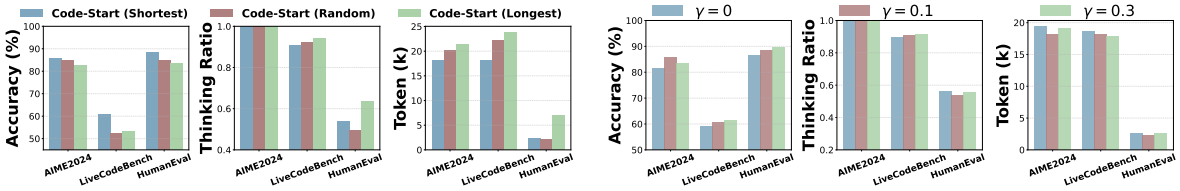
4.4 Further Analysis

Think-on vs. Think-off Dynamics During Training and Inference During the training and evaluation processes, we track how the model's decision-making evolves by monitoring the frequency of reasoning-mode activations and the corresponding output length. Specifically, we logged the frequency of `<think_on>` and `<think_off>` activations at each step. As shown in Figure 5(a), HiPO not only improves final accuracy but also sharpens the model's gating behavior, allowing it to skip unnecessary reasoning. Specifically, the gap be-

Method	AIME2024			AIME2025			LiveCodeBench			HumanEval		
	Acc↑	Length↓	Ratio_T↓	Acc↑	Length↓	Ratio_T↓	Acc↑	Length↓	Ratio_T↓	Acc↑	Length↓	Ratio_T↓
Cold-Start (on)	80.8	21265	1.00	71.7	23791	1.00	56.2	19473	1.00	82.9	2662	1.00
+ GRPO	82.5↑2.1%	21045↓1.0%	1.00−0.0%	76.7↑7%	22695↓4.6%	1.00−0.0%	57.3↑2.0%	19067↓2.1%	1.00−0.0%	95.1↑14.7%	3597↑35.1%	1.00−0.0%
Cold-Start	85.8↑6.2%	18138↓14.7%	1.00−0.0%	76.7↑7.0%	20613↓13.4%	1.00−0.0%	60.8↑8.2%	18158↓6.8%	0.91↓9.0%	88.4↑6.6%	2272↓14.6%	0.54↓46.3%
+ GRPO	86.7↑7.2%	17083↓19.7%	1.00−0.0%	79.17↑10.5%	19869↓16.5%	1.00−0.0%	62.1↑10.6%	18046↓7.3%	0.93↓7.3%	87.8↑5.9%	2220↓16.6%	0.59↓40.8%
AdaptThink	83.3↑3.1%	16598↓21.9%	0.93↓7.0%	74.2↑3.5%	19993↓16.0%	0.84↓16.0%	57.1↑1.6%	16162↓17.0%	0.78↓28.0%	85.4↑3.0%	915↓65.6%	0.16↓84.0%
AutoThink	84.3↑3.5%	17061↓19.8%	0.95↓5.0%	75.0↑4.6%	18784↓21.0%	0.88↓12.0%	57.5↑2.3%	15672↓19.5%	0.80↓20.0%	82.3↓0.7%	1050↓60.6%	0.18↓82.0%
HiPO	87.5↑8.3%	15107↓29.0%	0.98↓1.7%	82.5↑15.1%	17655↓25.8%	0.95↓5.0%	63.0↑12.2%	13558↓30.4%	0.82↓18.5%	90.2↑8.8%	776↓70.9%	0.12↓88.4%

Method	MATH-500			GPQA-Diamond			MBPP			Average		
	Acc↑	Length↓	Ratio_T↓	Acc↑	Length↓	Ratio_T↓	Acc↑	Length↓	Ratio_T↓	Acc↑	Length↓	Ratio_T↓
Cold-Start (on)	92.0	6237	1.00	61.1	10832	1.00	72.0	4411	1.00	73.8	12667	1.00
+ GRPO	93.2↑0.0%	6256↑1.3%	1.00−0.0%	57.6↓5.8%	10633↓1.8%	1.00−0.0%	71.8↓0.3%	5103↑15.7%	1.00−0.0%	76.3↑3.3%	12628↓0.3%	1.00−0.0%
Cold-Start	93.0↑1.1%	5215↓16.4%	0.65↓35.0%	61.6↑0.8%	11172↑3.1%	0.95↓4.5%	71.4↓0.8%	3561↓19.3%	0.42↓58.0%	76.8↑4.1%	11304↓10.8%	0.78↓21.8%
+ GRPO	92.8↑0.9%	5204↓16.6%	0.68↓31.8%	58.6↓4.1%	10581↓2.3%	0.98↓2.0%	72.0−0.0%	4341↓1.6%	0.38↓62.0%	77.0↑4.3%	11049 ↓12.8%	0.79↓20.6%
AdaptThink	92.8↑0.9%	4213↓32.5%	0.55↓45.0%	56.1↓8.2%	10242↓5.4%	0.91↓9.0%	68.0↓5.6%	4165↓5.6%	0.33↓67.0%	73.8−0.0%	10327↓18.5%	0.64↓36.0%
AutoThink	92.8↑0.9%	4261↓31.7%	0.56↓44.0%	58.1↓4.9%	9898↓8.6%	0.89↓11.0%	70.0↓2.8%	4958↓12.4%	0.40↓60.0%	74.3↑0.7%	10240↓19.2%	0.67↓33.0%
HiPO	95.6 ↑3.9%	4090 ↓34.4%	0.54 ↓45.8%	60.1 ↓1.7%	9367 ↓13.5%	0.92↓−8.1%	72.2↑0.3%	1338↓69.7%	0.12↓88.0%	78.4↑6.3%	8842↓30.2%	0.63↓36.5%

Table 2: Based on Qwen3-8B, performance of different methods on multiple benchmarks. **Ratio_T** denotes the ratio of "Think-on" mode over the corresponding benchmark.



(a) (b)

Figure 3: (a) Performance of different response selection strategies. (b) Performance of different γ values.

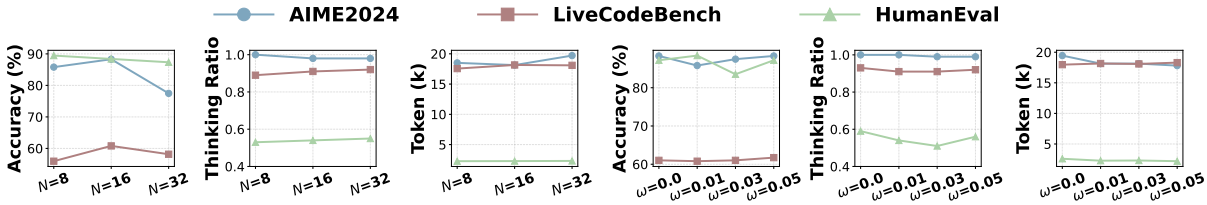


Figure 4: Performance of different rollout numbers and ω values.

tween <think_on> and <think_off> activations decreases from 89.5% at the beginning of training to 53.1% by the end. In Figure 5(b) shows the proportion of Think-on activations across different datasets during inference. Reasoning-intensive tasks, including AIME2024, and LiveCodeBench, consistently demonstrate high Think-on activation rates (>70%) throughout training. Conversely, tasks that require less explicit reasoning, such as HumanEval, exhibit a clear downward trend in Think-on activation as training progresses.

Token Count Dynamics During Training and Inference During RL training, the average token count shows a consistent downward trend in Figure 5(c), which indicates that the model gradually learns to produce more concise responses and highlight the HiPO reward design in encouraging efficient token usage Besides, Figure 5(d) shows the corresponding dynamics in average token counts per generated response during inference, and we also observe consistent token reduction in training.

Generalization on More Models In Table 4, we report the performance of HiPO on Qwen3-1.7B and Qwen3-32B, which shows consistent improvements on both accuracy and efficiency.

Method	AIME2024			LiveCodeBench			HumanEval		
	Acc	Length	Ratio_T	Acc	Length	Ratio_T	Acc	Length	Ratio_T
HiPO	87.50	15107	0.98	63.00	13558	0.82	90.2	776	0.12
HiPO (w/o global adv)	85.83↓1.9%	18064↑19.6%	1.00↑2.0%	56.83↓9.8%	14561↑7.4%	0.86↑4.9%	89.63↓4.9%	1660↑114.9%	0.27↑125.0%
HiPO (w/o local norm)	85.00↓2.9%	18268↑20.9%	1.00↑2.0%	58.37↓7.3%	16029↑18.2%	0.88↑7.3%	89.63↓0.6%	2052↑164.4%	0.32↑166.7%

Table 3: Performance of different design strategies on advantage functions.

Method	AIME24			LiveCodeBench			HumanEval			MBPP		
	Acc	Length	Ratio_T	Acc	Length	Ratio_T	Acc	Length	Ratio_T	Acc	Length	Ratio_T
Qwen3-1.7B												
Cold-Start (On)	63.3	24214	1.00	33.7	25616	1.00	77.4	4172	1.00	54.6	8587	1.00
Cold-Start	65.0↑2.7%	21039↓13.1%	1.00–0.0%	37.4↑11.0%	21364↓16.6%	0.98↓2.0%	81.7↑5.2%	3084↓26.1%	0.39↓61.0%	54.4↓0.4%	6398↓25.5%	0.64↓36.0%
HiPO	68.3↑7.9%	17614↓27.3%	0.98↓6.0%	44.3↑31.4%	19358↓24.4%	0.92↓8.0%	86.0↑11.1%	1973↓52.7%	0.28↓62.0%	62.8↑15.0%	4330↓49.6%	0.47↓53.0%
Qwen3-32B												
Cold-Start (On)	81.7	19551	1.00	65.4	17885	1.00	87.8	4298	1.00	76.2	4753	1.00
Cold-Start	85.0↑4.3%	16542↓15.4%	1.00–0.0%	65.9↑0.8%	14935↓16.5%	0.87↓13.0%	92.1↑4.9%	2785↓35.2%	0.47↓53.0%	78.4↑2.2%	3991↓16.0%	0.51↓49.0%
HiPO	88.3↑8.1%	14873↓23.9%	0.98↓2.0%	68.5↑4.5%	12721↓28.9%	0.82↓18.0%	92.7↑5.6%	824↓80.8%	0.18↓82.0%	84.4↑10.8%	2070↓56.4%	0.24↓76.0%

Table 4: Performance of HiPO on more models.

(a) Training Ratio (b) Testing Ratio

(c) Training Token (d) Testing Token

Figure 5: (a) Think-on and Think-off ratios in training. (b) Think-on ratio of different datasets. (c) Average token usage in RL training. (d) Token usage of different datasets. "LCB" denotes "LiveCodeBench".

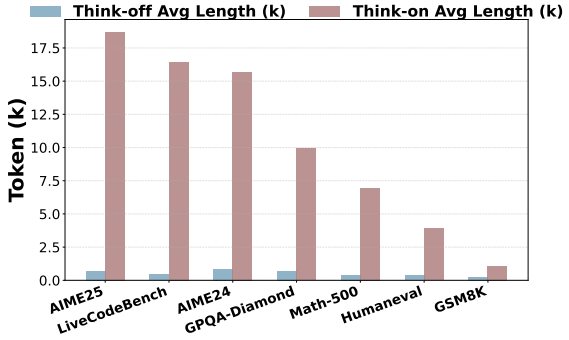


Figure 6: Token length distribution of Think-off vs. Think-on outputs across benchmarks.

Token length distribution of Think-off vs. Think-on As illustrated in the token length distributions across benchmarks such as AIME2025, AIME2024, LiveCodeBench, GPQA-Diamond, Humaneval, GSM8K, and MATH-500 (Figure 6), the "think-off "outputs consistently remain an order of magnitude shorter than their "think-on" counterparts, with average lengths typically in the hundreds versus thousands of tokens. These results further demonstrate the effectiveness of HiPO.

Moreover, we refer readers to see Appendix A.9 and Appendix A.10 for more analysis.

5 Conclusion

In this work, we introduced HiPO, a hybrid framework for adaptive reasoning in LLMs. By combining a hybrid data pipeline with a hybrid reinforcement learning reward system, HiPO enables models to dynamically balance Think-on and Think-off reasoning, mitigating the issue of overthinking while preserving accuracy. Experiments demonstrate that HiPO achieves competitive or superior accuracy with significantly improved token efficiency and reduced reasoning redundancy.

6 Limitations

This study validates HiPO primarily within logic-intensive domains like mathematics and coding. While preliminary generalization is demonstrated in the appendix, extending the framework to subjective or creative tasks—where the boundary between thinking and responding is ambiguous—remains a future direction. This specific focus does not undermine HiPO's core effectiveness in enhancing logical reasoning efficiency.

7 Ethic Statements

7.1 Energy Efficiency and Sustainability

Energy Efficiency and Sustainability. The primary motivation of HiPO is to mitigate the "overthinking" problem in LLMs during complex reasoning, thereby significantly reducing the generation of redundant tokens. By lowering the computational load during the inference phase—achieving a token reduction of approximately 30%—our work contributes to decreasing the carbon footprint and environmental impact associated with the large-scale deployment of reasoning-oriented models.

7.2 Data Sources and Fairness

Data Sources and Fairness. All training data utilized in this study are sourced from publicly available, high-quality scientific datasets, including AM-Thinking, AceReason, and others. We strictly adhered to the respective open-source licensing agreements during the data processing stage. Given that the data primarily focus on the domains of mathematics and programming and do not involve personally identifiable information (PII) or sensitive social attributes, there is no risk of privacy infringement or bias against specific demographic groups.

7.3 Disclosure of LLM Assistance

Disclosure of LLM Assistance. The authors independently conceived and executed all scientific ideas, algorithmic implementations, and experimental data analyses. Large Language Models (LLMs) were employed exclusively as auxiliary tools for language editing and enhancing the clarity of the manuscript. No experimental data, training samples, or reported results were generated using LLMs without rigorous human verification.

References

Pranjal Aggarwal and Sean Welleck. 2025. L1: Controlling how long a reasoning model thinks with reinforcement learning. *Preprint*, arXiv:2503.04697.

Daman Arora and Andrea Zanette. 2025. Training language models to reason efficiently. *Preprint*, arXiv:2502.04463.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. 2021. Program synthesis with large language models. *Preprint*, arXiv:2108.07732.

Simon A. Aytes, Jinheon Baek, and Sung Ju Hwang. 2025. Sketch-of-thought: Efficient llm reasoning with adaptive cognitive-inspired sketching. *Preprint*, arXiv:2503.05179.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, and 39 others. 2021. Evaluating large language models trained on code. *Preprint*, arXiv:2107.03374.

Qiguang Chen, Libo Qin, Jiaqi Wang, Jinxuan Zhou, and Wanxiang Che. 2024. Unlocking the capabilities of thought: A reasoning boundary framework to quantify and optimize chain-of-thought. *Preprint*, arXiv:2410.05695.

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. 2025a. Do not think that much for 2+3=? on the overthinking of o1-like llms. *Preprint*, arXiv:2412.21187.

Yang Chen, Zhuolin Yang, Zihan Liu, Chankyu Lee, Peng Xu, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. 2025b. Acereason-nemotron: Advancing math and code reasoning through reinforcement learning. *arXiv preprint arXiv:2505.16400*.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, and 181 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Jujie He, Jiacai Liu, Chris Yuhao Liu, Rui Yan, Chaojie Wang, Peng Cheng, Xiaoyu Zhang, Fuxiang Zhang, Jiacheng Xu, Wei Shen, Siyuan Li, Liang Zeng, Tianwen Wei, Cheng Cheng, Yang Liu, and Yahui Zhou. 2025. Skywork open reasoner series. Notion Blog.

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. *Preprint*, arXiv:2504.01296.

Intelligent Internet. 2025. Ii-thought : A large-scale, high-quality reasoning dataset.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. 2024. Livecodebench: Holistic and contamination free evaluation of large language models for code. *Preprint*, arXiv:2403.07974.

Yu Kang, Xianghui Sun, Liangyu Chen, and Wei Zou. 2025. C3ot: generating shorter chain-of-thought without compromising effectiveness. In *Proceedings of the Thirty-Ninth AAAI Conference on Artificial*

Intelligence and Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence and Fifteenth Symposium on Educational Advances in Artificial Intelligence, AAAI'25/IAAI'25/EAAI'25. AAAI Press.

Abhinav Kumar, Jaechul Roh, Ali Naseh, Marzena Karpinska, Mohit Iyyer, Amir Houmansadr, and Eugene Bagdasarian. 2025. Overthink: Slowdown attacks on reasoning llms. *Preprint*, arXiv:2502.02542.

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

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024a. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Tengxiao Liu, Qipeng Guo, Xiangkun Hu, Cheng Jiayang, Yue Zhang, Xipeng Qiu, and Zheng Zhang. 2024b. Can language models learn to skip steps? *Preprint*, arXiv:2411.01855.

Chenwei Lou, Zewei Sun, Xinnian Liang, Meng Qu, Wei Shen, Wenqi Wang, Yuntao Li, Qingping Yang, and Shuangzhi Wu. 2025. Adacot: Pareto-optimal adaptive chain-of-thought triggering via reinforcement learning. *Preprint*, arXiv:2505.11896.

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

Xinyin Ma, Guangnian Wan, Runpeng Yu, Gongfan Fang, and Xinchao Wang. 2025. Cot-valve: Length-compressible chain-of-thought tuning. *Preprint*, arXiv:2502.09601.

Tergel Munkhbat, Namgyu Ho, Seo Hyun Kim, Yongjin Yang, Yujin Kim, and Se-Young Yun. 2025. Self-training elicits concise reasoning in large language models. *Preprint*, arXiv:2502.20122.

Tsendsuren Munkhdalai, Manaal Faruqui, and Siddharth Gopal. 2024. Leave no context behind: Efficient infinite context transformers with infini-attention. *arXiv preprint arXiv:2404.07143*.

Sania Nayab, Giulio Rossolini, Marco Simoni, Andrea Saracino, Giorgio Buttazzo, Nicolamaria Manes, and Fabrizio Giacomelli. 2025. Concise thoughts: Impact of output length on llm reasoning and cost. *Preprint*, arXiv:2407.19825.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. *Preprint*, arXiv:2305.18290.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2023. Gpqa: A graduate-level google-proof q&a benchmark. *ArXiv*, abs/2311.12022.

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)*, page 476–483. IEEE.

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

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 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. *Preprint*, arXiv:2503.04472.

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. *Preprint*, arXiv:2503.16419.

Hanshi Sun, Momin Haider, Ruiqi Zhang, Huitao Yang, Jiahao Qiu, Ming Yin, Mengdi Wang, Peter Bartlett, and Andrea Zanette. 2024. Fast best-of-n decoding via speculative rejection. *Preprint*, arXiv:2410.20290.

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, Chuning Tang, Congcong Wang, Dehao Zhang, Enming Yuan, Enzhe Lu, Fengxiang Tang, Flood Sung, Guangda Wei, Guokun Lai, and 77 others. 2025. Kimi k1.5: Scaling reinforcement learning with llms. *Preprint*, arXiv:2501.12599.

Xiaoyu Tian, Yunjie Ji, Haotian Wang, Shuaiting Chen, Sitong Zhao, Yiping Peng, Han Zhao, and Xiangang Li. 2025. Not all correct answers are equal: Why your distillation source matters. *arXiv preprint arXiv:2505.14464*.

Songjun Tu, Jiahao Lin, Qichao Zhang, Xiangyu Tian, Linjing Li, Xiangyuan Lan, and Dongbin Zhao. 2025. Learning when to think: Shaping adaptive reasoning in r1-style models via multi-stage rl. *Preprint*, arXiv:2505.10832.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and

Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models. *Preprint*, arXiv:2201.11903.

Heming Xia, Chak Tou Leong, Wenjie Wang, Yongqi Li, and Wenjie Li. 2025. Tokenskip: Controllable chain-of-thought compression in llms. *Preprint*, arXiv:2502.12067.

Silei Xu, Wenhao Xie, Lingxiao Zhao, and Pengcheng He. 2025. Chain of draft: Thinking faster by writing less. *Preprint*, arXiv:2502.18600.

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

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv: 2305.10601*.

Yu Yue, Yufeng Yuan, Qiying Yu, Xiaochen Zuo, Ruofei Zhu, Wenyuan Xu, Jiaze Chen, Chengyi Wang, TianTian Fan, Zhengyin Du, Xiangpeng Wei, Xiangyu Yu, Gaohong Liu, Juncai Liu, Lingjun Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Chi Zhang, and 8 others. 2025. Vapo: Efficient and reliable reinforcement learning for advanced reasoning tasks. *Preprint*, arXiv:2504.05118.

Zizheng Zhan, Ken Deng, Huaixi Tang, Wen Xiang, Kun Wu, Weihao Li, Wenqiang Zhu, Jingxuan Xu, Lecheng Huang, Zongxian Feng, Shaojie Wang, Shangpeng Yan, Xuxing Chen, Jiaheng Liu, Zhongyuan Peng, Zuchen Gao, Haoyang Huang, Xiaojiang Zhang, Jinghui Wang, and 11 others. 2025. Kat-v1: Kwai-autothink technical report. *Preprint*, arXiv:2507.08297.

Jiajie Zhang, Nianyi Lin, Lei Hou, Ling Feng, and Juanzi Li. 2025. Adaptthink: Reasoning models can learn when to think. *Preprint*, arXiv:2505.13417.

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

A Appendix

A.1 Use of LLMs

Large language models (LLMs) were used exclusively for language-related assistance, including editing, formatting, and improving the clarity of the manuscript; (1) all scientific ideas, experimental designs, implementations, and analyses were independently conceived and carried out by the authors; (2) no LLM-generated content was used as experimental data, training data, or reported results in this work.

A.2 The decline in Qwen3's performance on the test set.

This section reports an empirical performance degradation of Qwen3 on AIME2024, AIME2025, HumanEval, and LiveCodeBench under extended training with AM-DeepSeek-R1-0528-Distilled, AM-Thinking-v1-Distilled, and OpenThoughts3-1.2M; as shown in Figure 7, when the number of training steps reaches 100, the accuracy consistently decreases across all evaluated benchmarks; unless otherwise specified, the batch size is fixed at 512 and all remaining hyperparameters follow the implementation details in the main paper.

A.3 Data Source

Our dataset is constructed from multiple publicly available reasoning corpora spanning both code generation and mathematical problem-solving; as summarized in Table 5, the queries are sourced from AM-Thinking-v1-Distilled [1], II-Thought-RL [2], AceReason-Math [3], and Skywork-OR1-RL-Data [4]; this heterogeneous composition promotes domain diversity and provides a robust foundation for model training and evaluation.

A.4 Prompt Templates

In this section, we provide the prompt templates for the response generation and judge analysis generation.

Response Generation

Please read the following question carefully and provide a clear answer.
—
Query
—

Judge Analysis Generation

You are tasked with analyzing the characteristics of a question to determine why it **requires** complex reasoning.
Your should **not** attempting to answer or infer its solution.
You should analyse user's question to determine the **core task intention**—that is, what the user wants the model to do. (e.g., write and validate code based on a problem description, etc.).
Then briefly outline the basic approach to accomplishing this task (e.g., write SQL code to retrieve imformation, etc.).
Based on the required approach, assess the **reasoning complexity**, and indicate whether it involves multiple steps or deep analysis. Do not solve the question or provide an answer. Focus solely on interpreting the task type, approach, and cognitive demand.
Be concise: your analysis must be no more than two lines and under 500 characters. Use clear, natural, and varied language. End your explanation with a statement indicating that complex reasoning is required (Think-on), but express this conclusion with a natural and diverse phrase, not repeating any single pattern. The meaning must be clear, but the expression can vary.
Please analyze the following question as required above:
—
Model Response
—

A.5 Discussion on limitations of relative methods

Re-evaluation of Adaptive Reasoning Methods. We acknowledge the contributions of prior dynamic inference methods such as AdaCOT, Adapthink, and AutoThink, which introduced mechanisms for dynamic token allocation and early exiting. However,

[1] https://huggingface.co/datasets/a-m-team/AM-Thinking-v1-Distilled
[2] https://huggingface.co/datasets/Intelligent-Internet/II-Thought-RL-v0
[3] https://huggingface.co/datasets/nvidia/AceReason-Math
[4] https://huggingface.co/datasets/Skywork/Skywork-OR1-RL-Data

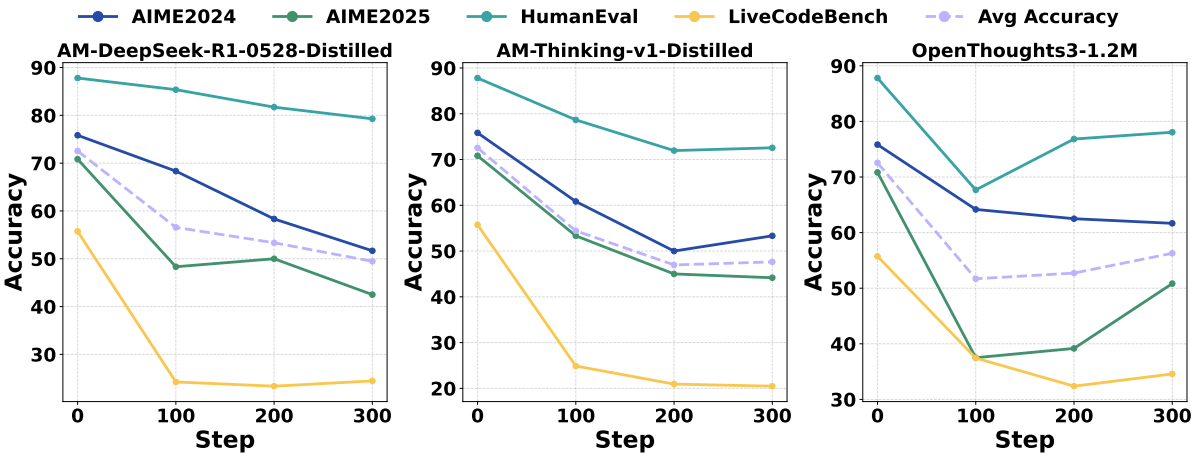


Figure 7: The decline in Qwen3's performance on the AIME2024, AIME2025, HumanEval, LiveCodeBench.

Category	Data Source	# Query
Code	AM-Thinking-v1-Distilled (Tian et al., 2025)	85k
	II-Thought-RL (Internet, 2025)	20k
Math	AceReason-Math (Chen et al., 2025b)	49k
	AM-Thinking-v1-Distilled (Tian et al., 2025)	32k
	II-Thought-RL (Internet, 2025)	30k
	Skywork-OR1-RL-Data (He et al., 2025)	24k

Table 5: Description of data sources.

a critical re-evaluation reveals a shared fundamental limitation: these approaches rely primarily on indirect and post-hoc supervision signals. As detailed in Table 6, these methods typically utilize final task accuracy, generation length, or heuristic confidence scores to guide adaptive behavior. Consequently, they suffer from a lack of causal rationale; rather than learning to identify which intrinsic properties of a problem necessitate complex reasoning, the models optimize for proxies (e.g., "shorter is better") or rely on decoupled classifiers. This results in coarse-grained supervision that fails to distinguish between necessary reasoning steps and redundant computation.

A.6 Stability Analysis

We conduct supplementary experiments to assess the stability of HiPO across random seeds and decoding temperatures; (1) **Experimental setup**: we retrain the model with three random seeds and evaluate on five benchmarks (AIME2024, AIME2025, LiveCodeBench, HumanEval, and MATH-500) under four temperatures (0.2, 0.4, 0.6, 0.8), yielding 12 independent runs; (2) **Performance stability**: across these runs, HiPO achieves an average score of 83.63 with a standard deviation of 0.201, corresponding to a 95% confidence interval of [83.50, 83.76] (α=0.05), whose lower bound (83.50) remains above the best baseline score (81.75), indicating that the observed gains are statistically significant and unlikely to be attributable to random variation; (3) **Generation efficiency**: HiPO produces an average token length of 10402.58 with a standard deviation of 105.12 and a 95% confidence interval of [10335.78, 10469.38], whose upper bound (10469.38) is lower than the minimum baseline length (11365), indicating consistently more concise generations while maintaining superior performance; Table 7 summarizes the results.

As shown in Table 8, the "shortest response" strategy achieves the highest Average Accuracy (83.86) among the compared alternatives while simultaneously attaining the lowest average Repetition Rate (3.08%), alleviating concerns that selecting shorter responses inherently induces template-

Method	Core Mechanism	Supervision Signal	Fundamental Limitation
AdaCOT	Dynamic step adjustment	Final accuracy; Length budget	**Coarse-grained Supervision:** The model learns to penalize length broadly ("long is bad") rather than identifying specific redundant steps.
Adapthink	Multi-round "rethinking"	Final accuracy	**Rigid Adaptation:** Adaptation is confined to a fixed "retry" paradigm that lacks a true assessment of intrinsic problem difficulty.
AutoThink	Classifier-based gating	Classifier accuracy	**Decoupled Learning:** The gating module is trained separately, leading to a misalignment between gating decisions and the model's actual reasoning capabilities.

Table 6: **Analysis of Limitations in Prior Adaptive Reasoning Methods.** Existing approaches primarily rely on indirect supervision signals (e.g., length penalties or final accuracy), failing to establish a causal link between problem difficulty and reasoning expenditure.

Seed	Average Accuracy (%)				Average Token Length			
	0.2	0.4	0.6	0.8	0.2	0.4	0.6	0.8
Seed 1	83.74	83.78	83.86	83.61	10423	10521	10237	10543
Seed 2	83.65	83.33	83.57	83.46	10617	10452	10421	10276
Seed 3	83.95	83.94	83.81	83.83	10340	10356	10344	10301

Table 7: Stability analysis of HiPO across different seeds and temperatures.

like artifacts; furthermore, its Truncation Rate is also among the lowest, suggesting improved completeness; taken together, these results indicate that the proposed filtering strategy preferentially retains more concise and less redundant reasoning trajectories, rather than promoting overfitting to a particular response style.

A.7 Ablation study on mode-explanation supervision

We introduce mode-explanation supervision to provide an explicit training signal for the gating mechanism beyond final-answer supervision, with the goal of improving both interpretability and mode selection quality; (1) this supervision encourages the model to articulate whether a given problem requires "Think-on" reasoning, rather than learning the gating policy solely from task outcomes; (2) to evaluate its effect, we ablate this component and report results on five benchmarks (AIME2024, AIME2025, LiveCodeBench, HumanEval, and MATH-500) in Table 9.

The results indicate that mode-explanation supervision substantially affects both accuracy and efficiency: (1) removing this supervision reduces the average accuracy from 80.23 to 70.67 (a drop of 9.56 points), suggesting that accurate mode selection is important for overall task performance; (2) the Think-on Ratio increases from 0.64 to 0.80 without supervision, implying a reduced ability to discriminate difficulty and a tendency to overactivate the computationally expensive Think-on mode even when it may be unnecessary; (3) correspondingly, the average token length increases from 9,814 to 14,645 (a 49.2% increase), demonstrating a marked loss in generation efficiency, and together these observations support that mode-explanation supervision is a critical component for learning an effective gating policy that balances performance and computational cost.

A.8 Analysis of Generalization of HiPO on long-context, tool-use, and general science tasks

We further evaluate the generalization of HiPO on long-context, tool-use, and general science tasks; (1) on GPQA-Diamond, HiPO achieves 60.1 compared to 59.6 for Qwen3-8B (Thinking) while reducing the average token length to 9,367 and the Think-on Ratio to 0.92, indicating improved ef-

Filtering Strategy	AIME2024			LiveCodeBench			HumanEval			Average		
	Acc	Rep.	Trunc.	Acc	Rep.	Trunc.	Acc	Rep.	Trunc.	Acc	Rep.	Trunc.
Shortest response (Ours)	87.5	2.08	2.29	63.0	3.92	4.14	90.2	3.16	3.20	80.23	3.05	3.21
Longest response	81.7	15.21	16.46	55.8	18.76	19.82	85.6	17.34	18.18	74.37	17.10	18.15
Non-Filtering	84.2	13.54	14.79	58.9	16.89	17.94	87.8	16.01	17.04	76.97	15.48	16.59
Filter overlong samples	80.8	8.54	9.17	56.2	11.23	12.05	84.8	10.94	11.78	73.93	10.24	11.00
GPT-4o score selection	85.0	5.83	6.46	60.3	8.12	9.03	88.5	7.55	8.50	77.93	7.17	8.00

Table 8: Ablation study on different data filtering strategies. **Rep.**: Repetition Rate (%), **Trunc.**: Truncation Rate (%).

Method	AIME2024			LiveCodeBench			HumanEval			Average		
	Acc	Length	Ratio_T	Acc	Length	Ratio_T	Acc	Length	Ratio_T	Acc	Length	Ratio_T
HiPO	87.5	15107	0.98	63.0	13558	0.82	90.2	776	0.12	80.23	9814	0.64
w/o Explanation Supervision	78.3	21956	0.99	51.2	19823	0.94	82.5	2156	0.48	70.67	14645	0.80

Table 9: Ablation study on mode-explanation supervision.

Benchmark	Model	Score	Token Length	Ratio (%)
LongBench (Long-Context)	Qwen3-8B (Thinking)	34.3	7,670	100
	HiPO-8B	**35.5**	**6,496**	**79**
BFCL-V3 (Tool-Augmented)	Qwen3-8B (Thinking)	13.9	6,196	100
	HiPO-8B	**16.4**	**4,789**	**33**
GPQA-Diamond (General Science)	Qwen3-8B (Thinking)	59.6	10,925	100
	HiPO-8B	**60.1**	**9,367**	**92**

Table 10: Generalization of HiPO on long-context, tool-use, and general science tasks.

ficiency without sacrificing general-domain reasoning performance; (2) following this, we extend evaluation to LongBench and BFCL-V3, as summarized in Table 10; (3) across these benchmarks, HiPO matches or exceeds the baseline while reducing token length (e.g., 6,496 vs. 7,670 on LongBench) and substantially lowering Think-on usage on tool-augmented queries (33% on BFCL-V3), suggesting that HiPO can allocate additional reasoning selectively rather than uniformly applying Think-on to all inputs.

A.9 Robustness analysis of HiPO under varying levels of supervision noise

Noisy "mode-explanation" supervision could potentially misguide policy learning and compromise reliability; (1) we use DeepSeek-V3 as the judge model due to its strong instruction-following behavior and coherent rationale generation, and for the constrained task of judging whether complex reasoning is required, we expect its explanations to be sufficiently consistent for supervision; (2) nevertheless, to quantitatively assess sensitivity to potential "judge noise", we simulate label noise by randomly shuffling 5%–30% of the mode-explanation labels, retrain the model under each noise level, and evaluate the resulting performance.

The results in Table 11 support two conclusions: (1) supervision quality matters, as increasing noise produces a monotonic degradation in Average Accuracy accompanied by higher Think-on usage and longer generations, consistent with a less confident gating policy; (2) the method exhibits robustness in that performance degrades gradually rather than exhibiting abrupt failure even under 10%–20% corruption, suggesting that the learned policy remains stable under moderate supervision noise albeit with reduced efficiency; overall, these findings highlight both the importance of clean mode-explanations and the resilience of HiPO to imperfect supervision.

A.10 Effect of scale of the cold-start data.

The SFT cold-start stage is a prerequisite for stable Reinforcement Learning (RL), as it initializes the policy within a semantically sound search space to

Noise Ratio	AIME2024			LiveCodeBench			HumanEval			Average		
	Acc	Length	Ratio_T	Acc	Length	Ratio_T	Acc	Length	Ratio_T	Acc	Length	Ratio_T
0%	87.5	15107	0.98	63.0	13558	0.82	90.2	776	0.12	80.23	9814	0.64
5%	85.8	16823	0.99	60.5	15234	0.86	88.6	1056	0.21	78.30	11038	0.69
10%	83.3	18456	0.99	57.8	16892	0.90	86.2	1423	0.32	75.77	12257	0.74
20%	80.0	19823	0.99	54.2	18045	0.93	83.5	1856	0.41	72.57	13241	0.78
30%	79.4	20567	0.99	52.8	18934	0.95	83.2	2134	0.46	71.80	13878	0.80

Table 11: Robustness analysis of HiPO under varying levels of supervision noise.

Cold-Start Scale	AIME2024			LiveCodeBench			HumanEval			Average		
	Acc	Length	Ratio_T	Acc	Length	Ratio_T	Acc	Length	Ratio_T	Acc	Length	Ratio_T
100	79.6	19856	0.99	56.6	18234	0.96	84.1	1256	0.45	73.43	13115	0.80
1K	81.3	18542	0.99	57.5	17456	0.93	85.4	1089	0.38	74.73	12362	0.77
10K	82.1	17823	0.98	57.9	16223	0.93	85.9	984	0.31	75.30	11677	0.74
100K	86.3	16045	0.98	61.5	14256	0.86	89.0	823	0.18	78.93	10375	0.67
Ours (196K)	87.5	15107	0.98	63.0	13558	0.82	90.2	776	0.12	80.23	9814	0.64

Table 12: Effect of cold-start data scale on HiPO performance.

mitigate the intractability of sparse rewards. Our "dose-response" analysis (Table 12) demonstrates that scaling SFT data from 100 to 100K samples is vital for preventing policy collapse and improving both accuracy and inference efficiency. Furthermore, extensive cold-start training refines the "Think-on" gating mechanism, transitioning the model from erratic computation to precise, task-adaptive mode selection. Ultimately, this stage ensures reliable RL convergence and sophisticated behavioral alignment.