
Think Only When You Need with Large Hybrid-Reasoning Models

Lingjie Jiang^{*†‡} Xun Wu^{*†} Shaohan Huang[†] Qingxiu Dong^{†‡} Zewen Chi[†]
Li Dong[†] Xingxing Zhang[†] Tengchao Lv[†] Lei Cui[†] Furu Wei^{†◇}

[†] Microsoft Research [‡] Peking University

lingjiejiang@stu.pku.edu.cn; {xunwu, shaohanh, fuwei}@microsoft.com

<https://aka.ms/GeneralAI>

Abstract

Recent Large Reasoning Models (LRMs) have shown substantially improved reasoning capabilities over traditional Large Language Models (LLMs) by incorporating extended thinking processes prior to producing final responses. However, excessively lengthy thinking introduces substantial overhead in terms of token consumption and latency, particularly unnecessary for simple queries. In this work, we introduce Large Hybrid-Reasoning Models (LHRMs), the first kind of model capable of adaptively determining whether to perform thinking based on the contextual information of user queries. To achieve this, we propose a two-stage training pipeline comprising Hybrid Fine-Tuning (HFT) as a cold start, followed by online reinforcement learning with the proposed Hybrid Group Policy Optimization (HGPO) to implicitly learn to select the appropriate thinking mode. Furthermore, we introduce a metric called Hybrid Accuracy to quantitatively assess the model’s capability for hybrid thinking. Extensive experimental results show that LHRMs can adaptively perform hybrid thinking on queries of varying difficulty and type. It outperforms existing LRMs and LLMs in reasoning and general capabilities while significantly improving efficiency. Together, our work advocates for a reconsideration of the appropriate use of extended thinking processes, and provides a solid starting point for building hybrid thinking systems.

1 Introduction

Recent remarkable advancements in Large Reasoning Models (LRMs), such as DeepSeek-R1 [15], OpenAI o1/o3 series [38] and others [46, 37], have catalyzed a significant paradigm shift from conventional Large Language Models (LLMs) toward LRMs. Compared to LLMs, LRMs exhibit substantially improved reasoning capabilities and generalization across tasks such as programming, mathematics, and commonsense reasoning [49, 15, 48]. These improvements are primarily attributed to the generation of extended thinking traces, often marked by special tokens such as `<think>`, prior to producing the final responses. Despite the effectiveness of LRMs, existing work primarily focuses on converting LLMs into LRMs to enhance performance across various domains, typically through reinforcement learning algorithms such as GRPO [15] and REINFORCE++ [17], or by fine-tuning on distilled thinking traces [36].

However, broader implications regarding the potential challenges faced by LRMs in real-world applications remain underexplored. A prominent issue is the problem of **overthinking** [? ?], where excessive computational resources are expended on simple queries with negligible performance gains, even for trivial inputs such as a single word (e.g. “Hello”). These findings highlight several pressing limitations of current LRMs and raise a critical question:

* Equal contribution. ◇ Corresponding author.

<p>Question 1: Let a, b, c be real numbers such that $ax^2 + bx + c \leq 1$ for all $0 \leq x \leq 1$. Find the largest possible value of $a + b + c$. Let's think step by step and output the final answer within boxed(). Ground Truth: 17</p> <p>Qwen2.5-7B-Instruct: To find the largest possible value of $a + b + c$ given that $ax^2 + bx + c \leq 1$ for all $0 \leq x \leq 1$, we need to analyze the constraints imposed by this inequality at specific points.</p> <p>...</p> <p>All conditions are satisfied. Now, calculate $a + b + c$: $a + b + c = 2 + -2 + 1 = 2 + 2 + 1 = 5$ Thus, the largest possible value of $a + b + c$ is 5. ✗</p> <p>DeepSeek-R1-Distill-Qwen-7B: <think> Okay, so I have this problem where I need to find the largest possible value of $a + b + c$ given that $ax^2 + bx + c \leq 1$ for all x between 0 and 1. Hmm, let me think about how to approach this. ... </think></p> <p>The maximum value of $a + b + c$ is achieved when the quadratic function touches 1 at $x=0$ and $x=1$, and -1 at $x=1/2$.</p> <p>...</p> <p>For $0 \leq x \leq 1$, $0 \leq (x - \frac{1}{2})^2 \leq \frac{1}{4}$, so $-1 \leq f(x) \leq 1$. Therefore, the largest possible value of $a + b + c$ is 17 ✓</p> <p>LHRMs-7B: <think> Okay, so I need to find the largest possible value of $a + b + c$ given that the quadratic function $ax^2 + bx + c$ is bounded by 1 in absolute value for all x between 0 and 1. Hmm, let's start by understanding the problem. ... </think></p> <p>Given the problem of finding the largest possible value of $a + b + c$ under the constraint that $ax^2 + bx + c \leq 1$ for all $0 \leq x \leq 1$, we start by considering the Chebyshev polynomials and their properties.</p> <p>...</p> <p>After verifying that this polynomial indeed satisfies the given constraint and considering other potential polynomials, we conclude that the maximum value of $a + b + c$ is achieved by the polynomial $8x^2 - 8x + 1$. The largest possible value of $a + b + c$ is 17 ✓</p>
<p>Question 2: Can you help me please?</p> <p>Qwen2.5-7B-Instruct: Of course! I'd be happy to help you. Could you please provide some more information about what you need assistance with? ✓</p> <p>DeepSeek-R1-Distill-Qwen-7B: <think> Okay, the user just said "Can you help me please?" I need to respond in a friendly and helpful way. I should acknowledge their request, show willingness to assist, and ask them to specify what they need help with. Keeping it open-ended will encourage them to provide more details.</p> <p>Hmm, maybe start with a positive affirmation. Something like, "Yes, I can help! What do you need assistance with today?" That sounds good. It's polite and inviting them to share more about their request.</p> <p>I should make sure my response is clear and concise. No need for complicated words. Just a straightforward offer to help. Let me put that together. </think></p> <p>Of course! I'm here to help. What do you need assistance with today? ✗</p> <p>LHRMs-7B: <no_think> Of course! What do you need help with? </no_think> ✓</p>

Figure 1: Example responses from Qwen2.5-7B-Instruct, DeepSeek-R1-Distill-Qwen-7B, and our LHRMs-7B on a reasoning related task (top) and a daily QA task (bottom). While LLMs respond concisely to simple queries, they struggle with complex reasoning. LRMs handle reasoning well via explicit thinking steps but often overuse them on simple tasks, resulting in slower responses and degraded user experience. In contrast, LHRMs adaptively determines when to engage in thinking, preserving strong reasoning ability while enabling faster, more natural interactions in everyday scenarios. Additional examples are provided in Appendix G.

How to build a hybrid thinking system that can achieve an optimal balance between system 2 reasoning and system 1 ability?

To fill these gaps and make LRMs more efficient, intelligent, and better aligned with human usage requirements, we draw inspiration from human cognitive processes, where complex problems often require deliberate thinking while simple ones can be handled intuitively.

Building on this insight, we introduce **Large Hybrid-Reasoning Models (LHRMs)**, a novel class of models that, to the best of our knowledge, are the first to explicitly address the limitations of existing LRMs by dynamically deciding whether to invoke extended thinking based on the semantic and contextual characteristics of user queries. As shown in Figure 1, this adaptive thinking mechanism allows LHRMs to balance computational efficiency with task complexity, better aligning with real-world deployment scenarios and enhancing user experience without compromising performance. To achieve this, we propose a two-stage training pipeline. The first stage, Hybrid Fine-Tuning (HFT), provides an effective cold start by enabling the model to learn to support two distinct thinking modes on the same query without mode collapse. This is followed by Hybrid Group Policy Optimization, a variant of online RL designed to teach the model when to engage in extended thinking while simultaneously generating more helpful and harmless responses. Furthermore, we introduce a metric termed Hybrid Accuracy, which correlates strongly with human expert judgment, to quantitatively evaluate the model’s capability for hybrid thinking.

Extensive experimental results and human studies conducted on Qwen-2.5 series models ranging from 1.5B to 7B parameters across multiple domains (including mathematics, programming, and general tasks) demonstrate that our LHRMs effectively performs hybrid thinking by adapting to queries of varying difficulty and types. Moreover, LHRMs outperforms existing LRMs and LLMs

in both reasoning and general capabilities, while significantly improving efficiency. In general, our main contribution is as follows:

1. We introduce Large Hybrid-Reasoning Models, the first kind of model which can adaptively determine whether to perform thinking based on the contextual information of user queries, mimicking human reasoning patterns and mitigating the over-thinking problem.
2. We develop a two-stage training pipeline comprising Hybrid Fine-Tuning as a cold start and a novel RL method, Hybrid Group Policy Optimization, which enables the model to learn when to engage in thinking while simultaneously generating more helpful and harmless responses.
3. Extensive experiments across diverse model scales and domains demonstrate that LHRMs accurately performs hybrid thinking and significantly improves both efficiency and general capabilities compared to existing LRMs and LLMs.

2 Large Hybrid-Reasoning Models

2.1 Problem Formulation

Specifically, let q be an input query. The LHRMs has access to two thinking modes: *Thinking* and *No-Thinking*, denoted as $\mathcal{M} = \{\top, \bot\}$, respectively. Each mode defines a conditional distribution $\mathcal{P}(a \mid q, m)$ over answers $a \in \mathcal{A}$, i.e.,

$$\mathcal{P}(a \mid q, m), \quad m \in \mathcal{M}. \quad (1)$$

For each query q , the model aims to select the optimal reasoning mode $m^*(q)$ from the candidate set \mathcal{M} , such that a task-specific utility function $\mathcal{U}(q, a)$ is maximized in expectation:

$$m^*(q) = \arg \max_{m \in \mathcal{M}} \mathbb{E}_{a \sim \mathcal{P}(a \mid q, m)} [\mathcal{U}(q, a)]. \quad (2)$$

The overall objective is to learn a policy $\pi : \mathcal{P} \rightarrow \mathcal{M}$ to select a mode $m \in \mathcal{M}$ that maximizes the expected utility $\mathcal{U}(q, A)$ over the query distributions $\Theta = \{(\mathcal{D}_i, \mathcal{U}_i)\}_{i=1}^N$:

$$\max_{\pi} \frac{1}{N} \sum_{i=1}^N \mathbb{E}_{\mathcal{D}_i \sim \Theta, \mathcal{D}_i \Leftrightarrow \mathcal{U}_i} \left[\mathbb{E}_{a \sim \mathcal{P}(a \mid q, \pi(q)), q \sim \mathcal{D}_i} [\mathcal{U}_i(q, a)] \right]. \quad (3)$$

We summarize two key challenges and our corresponding solutions as follows:

C1. How to learn the policy π based on the \mathcal{U}_i . We first propose a two-stage training phase to learn an effective LHRMs by: Hybrid Fine-Tuning as cold start (§ 2.2), followed by online reinforcement learning with Hybrid Group Policy Optimization (§ 2.3) to guide LHRMs to more effectively select between two reasoning modes to generate more appropriate and high-quality responses, respectively.

C2. How to evaluate the hybrid thinking capability of a policy π . We propose an evaluation metric named hybrid accuracy, which is designed to assess the hybrid thinking ability of models in § 2.4.

2.2 Stage I: Hybrid Fine-Tuning

We begin with a hybrid-formatted supervised fine-tuning stage named Hybrid Fine-Tuning (HFT) that integrates both reasoning-intensive (*Thinking*) and direct-answer (*No-Thinking*) data. This approach mitigates the instability often observed in cold-start scenarios [15], and establishes a robust initialization for next stage reinforcement learning.

Data Construction. The hybrid fine-tuning dataset consists of both reasoning-intensive and direct-answer examples. The think-style subset includes high-quality math, code, and science questions sourced from existing datasets [35, 12, 39, 45, 50], with answers generated by Deepseek-R1 [15] and verified for correctness. Each instance is annotated with `<think>` and `</think>` tags to mark intermediate reasoning steps. For the non-think-style subset, we collect simple queries from WildChat-1M [58] using a FastText-based classifier [21] to exclude complex reasoning tasks. The remaining factual and conversational examples are wrapped with `<no_think>` and `</no_think>` tags to indicate that direct answers are sufficient.

After deduplication and the removal of overlaps with evaluation benchmarks, we obtain a final set of 1.7M hybrid-formatted training examples. This curated collection provides a high-quality,

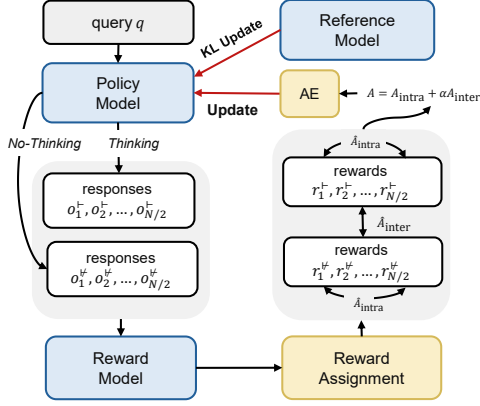


Figure 2: Demonstration of Hybrid Group Policy Optimization. HGPO proceeds by: (1) sampling multiple responses for each query q using both reasoning modes; (2) scoring the responses with the reward model and assigning these rewards based on Eq. 9; and (3) computing the advantage and policy loss, followed by updating the policy model. AE denotes advantage estimator and reward assignment denotes Eq. 9.

Algorithm 1 Hybrid Group Policy Optimization

Input model trained at Stage I $\pi_{\theta_{\text{HFT}}}$; reward models R_ϕ ; queries \mathcal{P} ; hyperparameters ϵ, β, μ

```

1: policy model  $\pi_\theta \leftarrow \pi_{\theta_{\text{HFT}}}$ 
2: for iteration = 1, ..., I do
3:   reference model  $\pi_{\text{ref}} \leftarrow \pi_\theta$ 
4:   for step = 1, ..., M do
5:     Sample a batch  $\mathcal{P}_b$  from  $\mathcal{P}$ 
6:     Update the old policy model  $\pi_{\theta_{\text{old}}} \leftarrow \pi_\theta$ 
7:     Sample  $N$  outputs  $\mathcal{O} \sim \pi_{\theta_{\text{old}}}(\cdot | q, m)$  for
        $q \in \mathcal{P}_b$  and  $m \in \mathcal{M}$ 
8:     Compute rewards  $r(o_i^m)$  for  $o_i^m \in \mathcal{O}$  by
       running  $R_\phi$ 
9:     Assign each  $r(o_i^m)$  to  $r_{\text{inter}}(o_i^m)$  and
        $r_{\text{intra}}(o_i^m)$  using Eq. 9
10:    Compute  $A_i^t$  for the  $t$ -th token of  $o_i^m$ 
       through Eq. 10
11:    for iteration = 1, ...,  $\mu$  do
12:      Update the  $\pi_\theta$  by maximizing the objec-
       tive  $\mathcal{J}_{\text{HGPO}}$  in Eq. 11
13:    end for
14:  end for
15: end for
Output  $\pi_\theta$ 

```

format-consistent foundation for cold-start training, enabling the model to effectively handle both reasoning-heavy and direct-response tasks.

Optimize Objective. HFT trains the model to predict the next token based on prior context. For the constructed dataset $\mathcal{D}_{\text{HFT}} = \{(x^i, y^i)\}_{i=1}^N$, the objective is:

$$\mathcal{L}_{\text{HFT}}(\theta) = -\mathbb{E}_{[(x,y) \sim \mathcal{D}_{\text{HFT}}]} \left[\sum_{t=1}^{|y|} \log \pi_\theta(y_t | x, y_{1:t-1}) \right], \quad (4)$$

2.3 Stage II: Hybrid Group Policy Optimization

After Stage I, LHRMs acquires an initial ability to support two distinct reasoning modes in \mathcal{M} on the same query q without collapsing. However, our ultimate goal is to enable the LHRMs to adaptively select the most appropriate reasoning mode $m^*(q)$ (Eq. 2) for each query q to improve its applicability in real-world scenarios. To this end, we propose an effective RL algorithm named Hybrid Group Policy Optimization (HGPO) to explicitly teach the model how to adaptively select between reasoning modes, i.e., to learn the policy π in Eq. 3, while simultaneously improving the model’s foundational ability (e.g., helpfulness and harmlessness).

HGPO is illustrated in Figure 2 and Algorithm 1. Following prior RL approaches [1, 17, 41], HGPO discards the critic model and instead estimates the values using multiple samples for each prompt q to reduce the computational cost training. Specifically, for each question q in the task prompt set \mathcal{P} , HGPO samples two groups of outputs \mathcal{O} from the old policy $\pi_{\theta_{\text{HFT}}}$, using *Thinking* and *No-Thinking* mode, respectively.

Sampling Strategy. Given a query $q \in \mathcal{P}$, we sample N candidate responses from the old policy $\pi_{\theta_{\text{HFT}}}$ under two distinct reasoning modes $\mathcal{M} = \{\vdash, \not\vdash\}$. Specifically,

$$\{o_i^+\}_{i=1}^{N/2} \sim \pi_{\theta_{\text{HFT}}}(\cdot | q, m = \vdash), \quad \{o_i^-\}_{i=1}^{N/2} \sim \pi_{\theta_{\text{HFT}}}(\cdot | q, m = \not\vdash) \quad (5)$$

We define the complete output candidate set as:

$$\mathcal{O}(q) = \{o_i^+\}_{i=1}^{N/2} \cup \{o_i^-\}_{i=1}^{N/2}. \quad (6)$$

Reward Scoring and Assignment. A reward function R_ϕ ² is applied to each candidate output, yielding two reward sets:

$$\mathcal{R}^\vdash = \{r(o_i^\vdash)\}_{i=1}^{N/2}, \quad \mathcal{R}^\nabla = \{r(o_i^\nabla)\}_{i=1}^{N/2}. \quad (7)$$

Since the reward scores assigned by the reward model may vary significantly across different domains, we apply a rule-based assignment scheme to normalize the existing reward scores. Specifically, we compute two types of binary reward: **inter-group rewards** r_{inter} and **intra-group rewards** r_{intra} to jointly capture both the relative quality across different reasoning modes and the answer quality within each individual reasoning mode. Given the average reward for each mode computed as

$$\bar{\mathcal{R}}^\vdash = \frac{2}{N} \sum_{i=1}^{N/2} r(o_i^\vdash), \quad \bar{\mathcal{R}}^\nabla = \frac{2}{N} \sum_{i=1}^{N/2} r(o_i^\nabla), \quad (8)$$

we have r_{inter} and r_{intra} assigned as:

$$r_{\text{inter}}(o_i^m) = \begin{cases} 1, & \text{if } m = \arg \max_{m' \in \{\vdash, \nabla\}} \{\bar{\mathcal{R}}^\vdash, \bar{\mathcal{R}}^\nabla + \delta\} \\ 0, & \text{otherwise} \end{cases}, \quad r_{\text{intra}}(o_i^m) = \begin{cases} 1, & \text{if } i = \arg \max_{j \in \{1, \dots, N/2\}} r_j^m \\ 0, & \text{otherwise} \end{cases}. \quad (9)$$

Here margin δ is a hyperparameter that governs the trade-off between two reasoning modes.

Advantage Estimation. After getting the final two kinds of rewards for \mathcal{O} , we can adopt advantage estimators like REINFORCE++ [17], GRPO [41], and RLOO [1] to estimate the advantages. Here we use GRPO as the default advantage estimator, i.e.,

$$A_i^t = \underbrace{\left[\frac{r_{\text{intra}}(o_i) - \text{mean}(r_{\text{intra}}(o_j))}{\text{std}(r_{\text{intra}}(o_j))} \right]}_{\text{GRPO for intra-group advantage } A_{\text{intra}}} + \mathbb{1}\{o_i^t \in \Phi\} \cdot \alpha \underbrace{\left[\frac{r_{\text{inter}}(o_i) - \text{mean}(r_{\text{inter}}(o_j))}{\text{std}(r_{\text{inter}}(o_j))} \right]}_{\text{GRPO for inter-group advantage } A_{\text{inter}}}, \quad (10)$$

where A_i^t is the final estimated per-token advantage for each response in \mathcal{O} , $\Phi = \{\text{<think>, <no_think>}\}$ and α is the hyperparameter for controlling the weight of inter-group advantages. We also explore the impact of using different estimators in § 3.3.

Optimize Objective. Following [1, 17, 41], HGPO optimizes the policy model π_θ by maximizing the following objective:

$$\mathcal{J}_{\text{HGPO}}(\theta) = \mathbb{E}_{[q \sim \mathcal{P}, \{o_i^m\}_{i=1}^N \sim \pi_{\theta_{\text{HFT}}}(\mathcal{O}|q), m \in \mathcal{M}]}$$

$$\frac{1}{N} \sum_{i=1}^N \sum_{t=1}^{|o_i|} \left[\min \left(\frac{\pi_\theta(o_i^{m,t}|q, o_i^{m,<t})}{\pi_{\theta_{\text{HFT}}}(o_i^{m,t}|q, o_i^{m,<t})} A_i^t, \text{clip} \left(\frac{\pi_\theta(o_i^{m,t}|q, o_i^{m,<t})}{\pi_{\theta_{\text{HFT}}}(o_i^{m,t}|q, o_i^{m,<t})}, 1 - \epsilon, 1 + \epsilon \right) A_i^t \right) - \beta \mathbb{D}_{\text{KL}}(\pi_\theta || \pi_{\text{ref}}) \right], \quad (11)$$

where

$$\mathbb{D}_{\text{KL}}(\pi_\theta || \pi_{\text{ref}}) = \frac{\pi_{\text{ref}}(o_i^m|q)}{\pi_\theta(o_i^m|q)} - \log \frac{\pi_{\text{ref}}(o_i^m|q)}{\pi_\theta(o_i^m|q)} - 1. \quad (12)$$

Here $\pi_{\theta_{\text{HFT}}}$ denotes the model we get from Stage I in § 2.2 and ϵ and β are hyper-parameters.

2.4 Evaluating Hybrid Thinking Capability

To more comprehensively evaluate the performance of LHRMs, beyond conventional downstream task metrics, we propose a new metric called **Hybrid Accuracy** (\mathcal{H}_{Acc}), which aims to measure the LHRMs' ability to correctly select the appropriate reasoning pattern.

Given the task prompt set $\mathcal{P} = \{p_i\}_{i=1}^K$, LHRMs is first applied to sample N responses under two distinct reasoning modes \vdash and ∇ . A reward model is then employed as a scorer to evaluate and assign scores to both sets of generated responses. The mode with the higher average score is regarded as the ground-truth preferred reasoning mode for each prompt, denoted as m_{gt} . In cases where the average scores of the two modes are equal or the difference between them falls below a predefined margin, the mode yielding the shorter response is selected as m_{gt} . Subsequently, we allow LHRMs to autonomously select a reasoning mode m_p for each prompt. The proportion of prompts for which LHRMs' selected mode matches the ground-truth mode is reported as

$$\mathcal{H}_{\text{acc}} = \frac{1}{K} \sum_{i=1}^K \mathbb{1}[\text{Equal}(m_{\text{gt}}, m_p)] \quad \text{s.t.} \quad m_{\text{gt}}, m_p \in \{\vdash, \nabla\}. \quad (13)$$

²For queries with definitive answers, we use rule-based rewards [41, 15] for a better reward estimation; otherwise, a trained parametric reward model is applied.

3 Experimental Results

3.1 Experimental Setup

Compared Baselines. To validate the effectiveness of our proposed method LHRMs, we conduct a comprehensive comparison against state-of-the-art LLMs and LRMs derived from the same base models. Specifically, we build our LHRMs on both 1.5B and 7B parameter versions of Qwen-2.5-math-base [52], and compare our method with several Qwen-2.5-derived variants, including:

- **LLMs:** we compare our model with Qwen-2.5 Math series [52] and Instruct series [51], which show great capabilities in coding, mathematics and general tasks.
- **LRMs:** Here we use DeepSeek-R1-Distill series [10], which are distilled using the reasoning data generated by DeepSeek-R1[10] and attain strong reasoning ability.
- **Hybrid:** Due to the absence of existing models capable of hybrid thinking, we compare our final model against the baseline we obtained in our Stage I (§ 2.2) (denoted as HFT). Additionally, we construct two further baselines by applying Direct Preference Optimization (DPO) [40] and Rejection Sampling Fine-Tuning (RFT) [55] to the checkpoint $\pi_{\theta_{\text{HFT}}}$ from Stage I, using the same training data as in Stage II (§ 2.3). These baselines are referred to as HFT-DPO and HFT-RFT, respectively. Implementation details of DPO and RFT can be found in Appendix A.

Evaluation Settings. We primarily evaluate model performance based on the following aspects:

- **Reasoning Capabilities.** We evaluate models on a comprehensive set of widely-used reasoning benchmarks, including math related like MATH500 [26], AIME24 [2], AMC23, Olympiad Bench [16], and code related like LiveCodeBench [20], MBPP [4] and MBPP+ [30].
- **General Capabilities.** We assess the models’ general capabilities through open-ended generation tasks using LLMs as judges. Specifically, we adopt AlpacaEval 2.0 [11] and Arena-Hard [25] to assess instruction-following ability and alignment with human preferences.
- **Hybrid Thinking Capabilities.** We compute Hybrid Accuracy (\mathcal{H}_{Acc}) (§ 2.4) on MATH500 to evaluate the model’s ability to correctly select the appropriate reasoning mode.

More details about the evaluation settings can be found in Appendix D.

Training Settings. For stage I, we finally obtained 1.7 M SFT data and details about construction pipeline, sources and statistics can be found in § 2.2 and Appendix C. All models are trained for 3 epochs with the AdamW optimizer, employing a 10% linear warmup followed by a cosine learning rate decay schedule. The maximum learning rate is set to $1\text{e}-4$, with a batch size of 128 and a maximum sequence length of 32k tokens. Training the 7B model in the SFT phase takes approximately 2.5 days on 4 nodes of NVIDIA 8×H100 stations.

For stage II, we construct the training dataset by randomly sampling 76K queries from Deepscaler [32] and Tulu3 [23] (details can be found in Appendix C). We use Llama-3.1-Tulu-3-8B-RM³ as the parametric reward model in Eq. 7. We use VeRL [43] to conduct experiments. By default, we use a constant 1×10^{-6} learning rate together with AdamW optimizer for policy model, and use a batch size of 256 and micro batchsize of 8. The rollout stage collects 256 prompts and samples 4 responses for each prompt. We set $\alpha = 1.0$ and margin = 0.2 for RL training. We set KL coefficient to 0.001 and $\epsilon = 0.5$ in Eq. 11 in all experiments. The RL phase takes 2 days on NVIDIA 4×H100 Stations.

3.2 Main Results

Overall Performance. The overall results of different models on the aforementioned benchmarks are presented in Table 1. We observe that LHRMs consistently outperform all comparable baselines across both the 1.5B and 7B model scales, achieving average improvements of **9.2%** and **7.1%** compared to HFT-DPO, and **18.3%** and **11.5%** compared to HFT-RFT, at the 1.5B and 7B scales, respectively. Specifically, it surpasses the strongest competing baseline, HFT-DPO, by **10.3%** and **13.6%** on the AIME24 benchmark at 1.5B and 7B scales, respectively. On the 7B scale, LHRMs further outperforms HFT-DPO by **50.2%** on Alpaca and **93.4%** on Arena-Hard. Notably, AIME24 and Arena-Hard represent the most challenging benchmarks in the math and general domains, respectively. These results demonstrate the strong reasoning and general capabilities of our LHRMs.

³<https://huggingface.co/allenai/Llama-3.1-Tulu-3-8B-RM>

Table 1: Performance comparison across different tasks. “-” indicates that the model does not support hybrid thinking and is therefore left blank. The last column (Avg.) reports the average performance across all evaluation tasks, excluding the \mathcal{H}_{acc} metric. Bold numbers indicate the best performance. *Type* refers to the model’s reasoning mode, where *Hybrid* denotes models that adaptively select between *Thinking* and *No-Thinking* modes for each query.

Methods	Type	MATH				Code			General		\mathcal{H}_{acc}	Avg.
		MATH500	AIME24	AMC23	Olympiad	LiveCode	MBPP	MBPP+	Alpaca	Arena		
1.5B size model												
Qwen2.5-Math-1.5B	LLMs	42.4	3.3	22.5	16.7	0.4	16.1	14.3	0.1	1.8	-	13.1
Qwen2.5-1.5B-Instruct	LLMs	51.0	3.3	52.8	38.7	2.2	60.1	51.9	8.8	1.1	-	30.0
Qwen2.5-Math-1.5B-Instruct	LLMs	72.0	6.7	60.0	38.1	3.7	26.7	23.8	2.8	4.7	-	26.5
DeepSeek-R1-Distill-Qwen-1.5B	LRMs	83.9	28.9	62.9	43.3	16.8	54.2	46.3	5.6	2.7	-	38.3
HFT-1.5B	Hybrid	87.8	32.7	75.0	48.9	15.7	54.8	47.4	13.1	6.9	41.4	42.5
HFT-RFT-1.5B	Hybrid	82.2	22.0	67.5	44.1	14.2	49.7	42.6	13.6	8.5	48.1	38.3
HFT-DPO-1.5B	Hybrid	86.8	32.6	75.0	48.7	17.2	50.5	42.6	13.3	6.9	45.8	41.5
LHRMs-1.5B	Hybrid	87.8	35.3	75.0	50.4	17.2	61.1	54.0	16.9	10.4	54.4	45.3
7B size model												
Qwen2.5-Math-7B	LLMs	57.0	13.3	22.5	21.8	6.0	31.5	27.3	2.0	7.0	-	20.9
Qwen2.5-7B-Instruct	LLMs	77.0	13.3	52.8	29.1	14.6	79.9	67.5	36.2	25.8	-	44.0
Qwen2.5-Math-7B-Instruct	LLMs	82.4	10.0	62.5	41.6	2.6	40.2	34.7	3.8	10.0	-	32.0
DeepSeek-R1-Distill-Qwen-7B	LRMs	92.8	55.5	91.5	58.1	37.6	74.3	64.3	19.1	17.9	-	56.8
HFT-7B	Hybrid	93.6	56.7	95.0	58.5	34.7	70.6	59.8	23.7	14.0	34.2	56.4
HFT-RFT-7B	Hybrid	87.8	55.3	82.5	55.0	35.8	81.0	68.8	28.1	14.0	49.7	56.6
HFT-DPO-7B	Hybrid	93.8	58.7	92.5	60.6	38.8	80.1	68.3	23.3	13.0	37.1	58.9
LHRMs-7B	Hybrid	93.8	66.7	95.0	61.2	38.8	81.5	69.6	35.0	26.0	71.9	63.1

Furthermore, we find that our LHRMs achieves the best hybrid thinking performance, as measured by \mathcal{H}_{acc} , significantly outperforming all baselines. For example, it exceeds HFT-DPO by **93.8%** and RFT by **44.7%**. This further demonstrates that our HGPO effectively enables the model to learn correct hybrid thinking behaviors, providing a promising and practical pathway for building hybrid thinking systems. Next, we provide a detailed investigation into each of the two training stages.

Effect of HFT at Stage I. By comparing HFT with the Qwen2.5 and Deepseek-R1-Distill series, we observe that HFT significantly enhances both reasoning and general capabilities, while demonstrating robust cold-start performance by maintaining stability during hybrid reasoning without failure or collapse. These results validate the effectiveness of the proposed HFT (§ 2.2) framework.

Effect of HGPO at Stage II. By comparing LHRMs with HFT, we find that HGPO further substantially improves the model’s reasoning and general capabilities, while enabling more effective selection of thinking modes (over **31.4%** and **110.2%** \mathcal{H}_{acc} improvements at 1.5B and 7B size, respectively). These findings demonstrate the effectiveness of HGPO.

Furthermore, when comparing LHRMs with HFT-DPO and HFT-RFT, we observe that LHRMs achieves superior downstream performance on both reasoning and general tasks, along with higher accuracy in reasoning mode selection. This highlights the effectiveness of HGPO.

Notably, we also observe that LHRMs exhibits stronger cross-domain generalization capabilities. Although RL training is conducted solely on math and general-domain data, LHRMs-1.5B achieves substantial improvements on code-related tasks, with gains of **11.1%** on MBPP and **13.9%** on MBPP+. In contrast, both HFT-DPO-1.5B and HFT-RFT-1.5B show performance drops (**7.8%** and **9.3%**, respectively) on the same tasks. This indicates that LHRMs is able to generalize learned hybrid thinking patterns across domains—a property not observed in DPO or RFT.

3.3 Analysis

Effect of Different Advantage Estimators. To investigate the impact of different advantage estimators on HGPO training, we replace GRPO in Eq. 10 with RLOO [1] and REINFORCE++ [17]. Implementation details can be found in Appendix B. As shown in Figure 3 (a), all estimators yield competitive performance, indicating that HGPO is robust to the choice of advantage estimator.

Effect of Margin δ in Eq. 9. We investigate how varying the margin δ influences the model’s bias toward the two reasoning modes. As shown in Figure 3 (b), different values of δ result in distinct

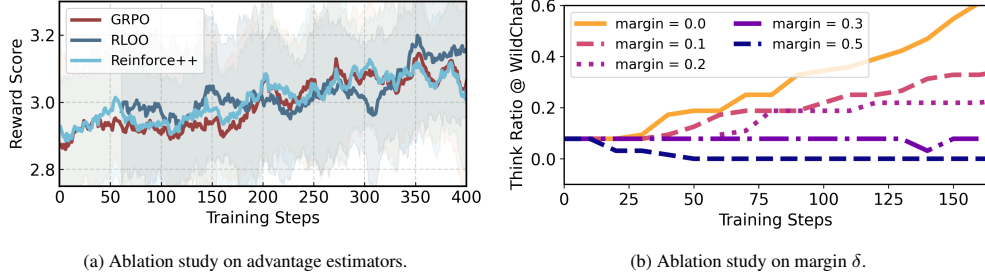


Figure 3: Ablation study on the effects of advantage estimators and margin δ . δ in Eq. 9.

hybrid reasoning behaviors. Specifically, a larger δ encourages the model to favor the *No-Thinking* mode. This suggests that δ can serve as a control knob for tailoring hybrid reasoning behavior to specific application needs. For instance, a higher δ may be preferred when real-time responsiveness is prioritized, whereas a lower δ is more suitable when reasoning quality is the primary concern.

Thinking Ratio Study within Domain. Figure 4 illustrates the distribution of thinking ratio across varying difficulty levels in the MATH500 benchmark for both HFT-7B and LHRMs-7B. We observe that HFT-7B maintains a consistently high thinking ratio across all difficulty levels. In contrast, after applying the Stage II HGPO, LHRMs-7B exhibits a decreasing thinking ratio as the difficulty level decreases while getting a higher overall performance (as shown in Table 1). This trend suggests that our HGPO can effectively enable the model to adaptively perform hybrid thinking based on the input query. Specifically, the model learns to respond to simpler queries using a *No-Thinking* strategy, thereby reducing reasoning cost and inference latency without sacrificing accuracy. For more challenging queries, the model consistently engages in full thinking, achieving higher reasoning accuracy where it matters most.

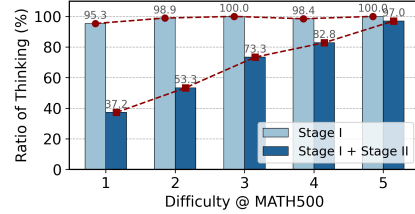


Figure 4: Analysis of thinking ratio of LHRMs within a single domain.

Thinking Ratio Study across Domain. We investigate how HGPO affects model performance on unseen domains not encountered during the RL phase. Specifically, we conduct RL using data from the math and general domains, and then evaluate the resulting models on the code domain. As shown in Figure 5, we observe that the model attained from HFT exhibits a relatively low think ratio in the code domain. However, as RL progresses—despite being conducted on unrelated domains such as math—the think ratio in the code domain gradually increases, accompanied by improved performance. This suggests that our HGPO is capable of transferring hybrid thinking patterns learned in a specific domain (e.g., math) to other domains (e.g., code), demonstrating strong generalization and transferability.

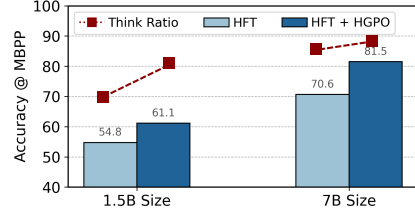


Figure 5: Analysis of thinking ratio of LHRMs across different domains.

Thinking Ratio Study across Model Size. We analyze how model size influences hybrid thinking behavior during the RL phase. In Figure 6, we present the think ratio of the 1.5B and 7B models on the MATH500 benchmark. We observe that, as RL training progresses, the think ratio of the 7B model gradually decreases, while that of the 1.5B model increases. This indicates that hybrid thinking behavior during RL is strongly correlated with model capacity and knowledge storage. Specifically, larger models with stronger knowledge reserves tend to shift towards using *No-Thinking* strategies on solvable problems to reduce reasoning overhead and improve response efficiency. In contrast, smaller models are more inclined to invoke *Thinking* more frequently, as a way to compensate for their limited capabilities through enhanced reasoning.

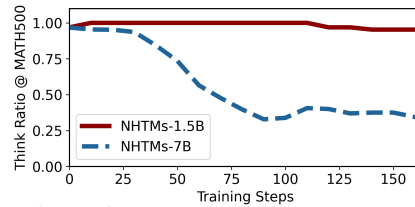


Figure 6: Ablation study on model size.

Question: If $a * b = 9a + 2b - ab + 5$, what is the value of $5 * 1$?

DeepSeek-R1-Distill-Qwen-7B: <think> Okay, let's tackle this problem. So, we have this operation defined as a star b equals 9a plus 2b minus ab plus 5. And we need to find the value of 5 star 1. Hmm, let me make sure I understand the question... (2941 characters) </think> To solve $5 * 1$ using the defined operation $a * b = 9a + 2b - ab + 5$:

1. Substitute $a = 5$ and $b = 1$ into the expression: $5 * 1 = 9(5) + 2(1) - (5)(1) + 5$
2. Calculate each term: $- 9 \times 5 = 45 - 2 \times 1 = 2 - 5 \times 1 = 5$
3. Substitute these values back into the expression: $45 + 2 - 5 + 5$
4. Perform the arithmetic operations step-by-step: $- 45 + 2 = 47 - 47 - 5 = 42 - 42 + 5 = 47$

Thus, the value of $5 * 1$ is 47 ✓

LHRMs-7B: <no_think> To solve the problem, we need to substitute $a = 5$ and $b = 1$ into the given function $a * b = 9a + 2b - ab + 5$. Step-by-step solution:

1. Start with the given function: $a * b = 9a + 2b - ab + 5$
2. Substitute $a = 5$ and $b = 1$ into the function: $5 * 1 = 9(5) + 2(1) - 5(1) + 5$
3. Perform the multiplication operations: $5 * 1 = 45 + 2 - 5 + 5$
4. Simplify the expression by performing the addition and subtraction operations: $5 * 1 = 45 + 2 - 5 + 5 = 47$

Therefore, the final answer is: 47 ✓</no_think>

Figure 7: Example outputs from DeepSeek-R1-Distill-Qwen-7B and our LHRMs-7B model. For easy questions, LHRMs-7B selects the *No-Thinking* mode.

Case Analysis. We present representative model responses in Figure 1 and 7. We observe that for relatively simple queries—such as everyday questions and basic math questions, LHRMs tends to autonomously select the *No-Thinking* mode, achieving fast and accurate responses. In contrast, for more complex problems that require deeper reasoning, LHRMs adaptively switches to the *Thinking* mode to produce more precise solutions. More example responses can be found in Appendix G.

Additional Analysis. To further verify the effectiveness of our method, we conduct additional analyses in Appendix E, including the impact of hybrid training on reasoning ability (E.1), an efficiency analysis (E.2), an ablation comparing HGPO with standard GRPO (E.3), and an annotation consistency validation for Hybrid Accuracy (E.4).

4 Related Works

Test-Time Scaling (TTS). Test-time scaling [12, 15, 37, 19] has been validated as a promising approach to enhance model performance beyond scaling model size [44]. There are two primary approaches to implementing TTS [37]: **Parallel** and **Sequential**. The former is achieved by sampling multiple solutions and select the one by a selector (e.g., an outcome reward model), like Best-of-N [5, 19, 24], and Monte-Carlo Tree Search (MCTS) [28, 56, 61, 8]. The latter aims to achieve TTS by enabling the model to generate longer outputs, i.e., Chain-of-Thoughts (CoT) within a single sampling pass by prompting, finetuning or reinforcement learning. Beyond the field of NLP, TTS has also been shown to effectively improve the test-time performance of trained models in other domains, like image generation, video generations [27] and multi-modality learning [13, 18, 6].

Large Reasoning Models. Recent advances in Large Reasoning Models (LRMs), such as DeepSeek-R1 [10], OpenAI o1/o3 series [38], and others [46, 37, 3, 14], have led to a growing focus on Large Reasoning Models (LRMs). Compared to general LLMs, LRMs extend the capabilities of LLMs by generating long chains of reasoning steps with reflection before outputting the final answers. LRMs are typically developed by applying reinforcement learning such as GRPO [15], REINFORCE++ [17], and related approaches [7, 53], or distilled from stronger models [45, 35, 12, 57, 54].

5 Conclusion

In this work, we focus on building a large language model that effectively balances reasoning capabilities and general-purpose performance. To this end, we propose a novel evaluation metric, \mathcal{H}_{acc} , designed to consistently assess a model’s ability to perform hybrid thinking across diverse tasks. We further introduce a two-stage training pipeline consisting of HFT and HGPO. Experimental results demonstrate that this pipeline significantly improves performance on both reasoning-centric and general downstream tasks. Moreover, it enhances the model’s hybrid thinking capabilities, leading to a better user experience by reducing unnecessary reasoning on simple queries commonly observed in LRMs, and mitigating the insufficient reasoning capabilities in conventional LLMs.

References

- [1] A. Ahmadian, C. Cremer, M. Gallé, M. Fadaee, J. Kreutzer, O. Pietquin, A. Üstün, and S. Hooker. Back to Basics: Revisiting REINFORCE-Style Optimization for Learning from Human Feedback in LLMs. In L. Ku, A. Martins, and V. Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL 2024, Bangkok, Thailand, August 11-16, 2024, pages 12248–12267. Association for Computational Linguistics, 2024.
- [2] AI-MO. Aime 2024, 2024.
- [3] Anthropic. Claude 3.7 sonnet and claude code. <https://www.anthropic.com/news/claude-3-7-sonnet>, 2025.
- [4] J. Austin, A. Odena, M. Nye, M. Bosma, H. Michalewski, D. Dohan, E. Jiang, C. Cai, M. Terry, Q. Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.
- [5] B. Brown, J. Juravsky, R. Ehrlich, R. Clark, Q. V. Le, C. Ré, and A. Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling, 2024.
- [6] L. Chen, L. Li, H. Zhao, Y. Song, and Vinci. R1-v: Reinforcing super generalization ability in vision-language models with less than \$3. <https://github.com/Deep-Agent/R1-V>, 2025. Accessed: 2025-02-02.
- [7] D. Cheng, S. Huang, X. Zhu, B. Dai, W. X. Zhao, Z. Zhang, and F. Wei. Reasoning with exploration: An entropy perspective on reinforcement learning for llms, 2025.
- [8] S. Choi, T. Fang, Z. Wang, and Y. Song. Kcts: Knowledge-constrained tree search decoding with token-level hallucination detection, 2023.
- [9] G. Cui, L. Yuan, N. Ding, G. Yao, W. Zhu, Y. Ni, G. Xie, Z. Liu, and M. Sun. Ultrafeedback: Boosting language models with high-quality feedback, 2023.
- [10] DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025.
- [11] Y. Dubois, B. Galambosi, P. Liang, and T. B. Hashimoto. Length-controlled alpaca-eval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475*, 2024.
- [12] H. Face. Open r1: A fully open reproduction of deepseek-r1, January 2025.
- [13] K. Feng, K. Gong, B. Li, Z. Guo, Y. Wang, T. Peng, B. Wang, and X. Yue. Video-r1: Reinforcing video reasoning in mllms. *arXiv preprint arXiv:2503.21776*, 2025.
- [14] Google. Gemini 2.5 flash. <https://deepmind.google/technologies/gemini/flash/>, 2025.
- [15] D. Guo, D. Yang, H. Zhang, J. Song, R. Zhang, R. Xu, Q. Zhu, S. Ma, P. Wang, X. Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.
- [16] C. He, R. Luo, Y. Bai, S. Hu, Z. L. Thai, J. Shen, J. Hu, X. Han, Y. Huang, Y. Zhang, J. Liu, L. Qi, Z. Liu, and M. Sun. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems, 2024.
- [17] J. Hu. Reinforce++: A simple and efficient approach for aligning large language models. *arXiv preprint arXiv:2501.03262*, 2025.
- [18] W. Huang, B. Jia, Z. Zhai, S. Cao, Z. Ye, F. Zhao, Y. Hu, and S. Lin. Vision-r1: Incentivizing reasoning capability in multimodal large language models. *arXiv preprint arXiv:2503.06749*, 2025.
- [19] R. Irvine, D. Boubert, V. Raina, A. Liusie, Z. Zhu, V. Mudupalli, A. Korshuk, Z. Liu, F. Cremer, V. Assassi, C.-C. Beauchamp, X. Lu, T. Rialan, and W. Beauchamp. Rewarding chatbots for real-world engagement with millions of users, 2023.

- [20] N. Jain, K. Han, A. Gu, W.-D. Li, F. Yan, T. Zhang, S. Wang, A. Solar-Lezama, K. Sen, and I. Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. [arXiv preprint arXiv:2403.07974](#), 2024.
- [21] A. Joulin, E. Grave, P. Bojanowski, M. Douze, H. Jégou, and T. Mikolov. Fasttext.zip: Compressing text classification models. [arXiv preprint arXiv:1612.03651](#), 2016.
- [22] A. Köpf, Y. Kilcher, D. Von Rütte, S. Anagnostidis, Z. R. Tam, K. Stevens, A. Barhoum, D. Nguyen, O. Stanley, R. Nagyfi, et al. Openassistant conversations-democratizing large language model alignment. [Advances in Neural Information Processing Systems](#), 36:47669–47681, 2023.
- [23] N. Lambert, J. Morrison, V. Pyatkin, S. Huang, H. Ivison, F. Brahman, L. J. V. Miranda, A. Liu, N. Dziri, S. Lyu, Y. Gu, S. Malik, V. Graf, J. D. Hwang, J. Yang, R. L. Bras, O. Tafjord, C. Wilhelm, L. Soldaini, N. A. Smith, Y. Wang, P. Dasigi, and H. Hajishirzi. Tulu 3: Pushing frontiers in open language model post-training, 2024.
- [24] N. Levi. A simple model of inference scaling laws, 2024.
- [25] T. Li, W.-L. Chiang, E. Frick, L. Dunlap, T. Wu, B. Zhu, J. E. Gonzalez, and I. Stoica. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. [arXiv preprint arXiv:2406.11939](#), 2024.
- [26] H. Lightman, V. Kosaraju, Y. Burda, H. Edwards, B. Baker, T. Lee, J. Leike, J. Schulman, I. Sutskever, and K. Cobbe. Let’s verify step by step. [arXiv preprint arXiv:2305.20050](#), 2023.
- [27] F. Liu, H. Wang, Y. Cai, K. Zhang, X. Zhan, and Y. Duan. Video-t1: Test-time scaling for video generation. [arXiv preprint arXiv:2503.18942](#), 2025.
- [28] J. Liu, A. Cohen, R. Pasunuru, Y. Choi, H. Hajishirzi, and A. Celikyilmaz. Don’t throw away your value model! generating more preferable text with value-guided monte-carlo tree search decoding, 2024.
- [29] J. Liu, C. S. Xia, Y. Wang, and L. Zhang. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. [Advances in Neural Information Processing Systems](#), 36:21558–21572, 2023.
- [30] J. Liu, S. Xie, J. Wang, Y. Wei, Y. Ding, and L. Zhang. Evaluating language models for efficient code generation. [arXiv preprint arXiv:2408.06450](#), 2024.
- [31] H. Luo, L. Shen, H. He, Y. Wang, S. Liu, W. Li, N. Tan, X. Cao, and D. Tao. O1-pruner: Length-harmonizing fine-tuning for o1-like reasoning pruning, 2025.
- [32] M. Luo, S. Tan, J. Wong, X. Shi, W. Tang, M. Roongta, C. Cai, J. Luo, T. Zhang, E. Li, R. A. Popa, and I. Stoica. Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl, 2025. Notion Blog.
- [33] X. Ma, G. Wan, R. Yu, G. Fang, and X. Wang. Cot-valve: Length-compressible chain-of-thought tuning, 2025.
- [34] S. Mahdavi, M. Li, K. Liu, C. Thrampoulidis, L. Sigal, and R. Liao. Leveraging online olympiad-level math problems for llms training and contamination-resistant evaluation. [arXiv preprint arXiv:2501.14275](#), 2025.
- [35] J. Mattern, S. Jaghouar, M. Basra, J. Straube, M. D. Ferrante, F. Gabriel, J. M. Ong, V. Weisser, and J. Hagemann. Synthetic-1: Two million collaboratively generated reasoning traces from deepseek-r1, 2025.
- [36] I. Moshkov, D. Hanley, I. Sorokin, S. Toshniwal, C. Henkel, B. Schifferer, W. Du, and I. Gitman. Aimo-2 winning solution: Building state-of-the-art mathematical reasoning models with openmathreasoning dataset. [arXiv preprint arXiv:2504.16891](#), 2025.
- [37] N. Muennighoff, Z. Yang, W. Shi, X. L. Li, L. Fei-Fei, H. Hajishirzi, L. Zettlemoyer, P. Liang, E. Candès, and T. Hashimoto. s1: Simple test-time scaling. [arXiv preprint arXiv:2501.19393](#), 2025.

- [38] OpenAI. Openai gpt-4.5 system card. OpenAI Publication, 2025.
- [39] G. Penedo, A. Lozhkov, H. Kydlíček, L. B. Allal, E. Beeching, A. P. Lajarín, Q. Gallouédec, N. Habib, L. Tunstall, and L. von Werra. Codeforces. <https://huggingface.co/datasets/open-r1/codeforces>, 2025.
- [40] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn. Direct preference optimization: Your language model is secretly a reward model. Advances in Neural Information Processing Systems, 36:53728–53741, 2023.
- [41] Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, X. Bi, H. Zhang, M. Zhang, Y. Li, Y. Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. arXiv preprint arXiv:2402.03300, 2024.
- [42] Y. Shen, J. Zhang, J. Huang, S. Shi, W. Zhang, J. Yan, N. Wang, K. Wang, Z. Liu, and S. Lian. Dast: Difficulty-adaptive slow-thinking for large reasoning models, 2025.
- [43] G. Sheng, C. Zhang, Z. Ye, X. Wu, W. Zhang, R. Zhang, Y. Peng, H. Lin, and C. Wu. Hybridflow: A flexible and efficient rlhf framework. arXiv preprint arXiv: 2409.19256, 2024.
- [44] C. Snell, J. Lee, K. Xu, and A. Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. arXiv preprint arXiv:2408.03314, 2024.
- [45] O. Team. Open Thoughts. <https://open-thoughts.ai>, Jan. 2025.
- [46] Q. Team. Qwq: Reflect deeply on the boundaries of the unknown, November 2024.
- [47] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. Advances in neural information processing systems, 35:24824–24837, 2022.
- [48] Z. Xie, L. Chen, W. Mao, J. Xu, L. Kong, et al. Teaching language models to critique via reinforcement learning. arXiv preprint arXiv:2502.03492, 2025.
- [49] F. Xu, Q. Hao, Z. Zong, J. Wang, Y. Zhang, J. Wang, X. Lan, J. Gong, T. Ouyang, F. Meng, et al. Towards large reasoning models: A survey of reinforced reasoning with large language models. arxiv. 10.48550. arXiv preprint arXiv.2501.09686, 2025.
- [50] Z. Xu, Y. Liu, Y. Yin, M. Zhou, and R. Poovendran. Kodcode: A diverse, challenging, and verifiable synthetic dataset for coding, 2025.
- [51] A. Yang, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Li, D. Liu, F. Huang, H. Wei, H. Lin, J. Yang, J. Tu, J. Zhang, J. Yang, J. Yang, J. Zhou, J. Lin, K. Dang, K. Lu, K. Bao, K. Yang, L. Yu, M. Li, M. Xue, P. Zhang, Q. Zhu, R. Men, R. Lin, T. Li, T. Tang, T. Xia, X. Ren, X. Ren, Y. Fan, Y. Su, Y. Zhang, Y. Wan, Y. Liu, Z. Cui, Z. Zhang, and Z. Qiu. Qwen2.5 technical report. arXiv preprint arXiv:2412.15115, 2024.
- [52] A. Yang, B. Zhang, B. Hui, B. Gao, B. Yu, C. Li, D. Liu, J. Tu, J. Zhou, J. Lin, K. Lu, M. Xue, R. Lin, T. Liu, X. Ren, and Z. Zhang. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. arXiv preprint arXiv:2409.12122, 2024.
- [53] C. Yang, Q. Si, Y. Duan, Z. Zhu, C. Zhu, Q. Li, M. Chen, Z. Lin, and W. Wang. Dynamic early exit in reasoning models, 2025.
- [54] Y. Ye, Z. Huang, Y. Xiao, E. Chern, S. Xia, and P. Liu. Limo: Less is more for reasoning, 2025.
- [55] Z. Yuan, H. Yuan, C. Li, G. Dong, K. Lu, C. Tan, C. Zhou, and J. Zhou. Scaling relationship on learning mathematical reasoning with large language models. arXiv preprint arXiv:2308.01825, 2023.
- [56] S. Zhang, Z. Chen, Y. Shen, M. Ding, J. B. Tenenbaum, and C. Gan. Planning with large language models for code generation, 2023.
- [57] H. Zhao, H. Wang, Y. Peng, S. Zhao, X. Tian, S. Chen, Y. Ji, and X. Li. 1.4 million open-source distilled reasoning dataset to empower large language model training, 2025.

- [58] W. Zhao, X. Ren, J. Hessel, C. Cardie, Y. Choi, and Y. Deng. Wildchat: 1m chatGPT interaction logs in the wild. In The Twelfth International Conference on Learning Representations, 2024.
- [59] L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. P. Xing, H. Zhang, J. E. Gonzalez, and I. Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena, 2023.
- [60] Y. Zheng, R. Zhang, J. Zhang, Y. Ye, Z. Luo, Z. Feng, and Y. Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. In Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations), Bangkok, Thailand, 2024. Association for Computational Linguistics.
- [61] A. Zhou, K. Yan, M. Shlapentokh-Rothman, H. Wang, and Y.-X. Wang. Language agent tree search unifies reasoning acting and planning in language models, 2024.

Contents

1	Introduction	1
2	Large Hybrid-Reasoning Models	3
2.1	Problem Formulation	3
2.2	Stage I: Hybrid Fine-Tuning	3
2.3	Stage II: Hybrid Group Policy Optimization	4
2.4	Evaluating Hybrid Thinking Capability	5
3	Experimental Results	6
3.1	Experimental Setup	6
3.2	Main Results	6
3.3	Analysis	7
4	Related Works	9
5	Conclusion	9
A	Implementation Details for DPO and RFT	15
B	Implementation Details for RLOO and Reinforce++	15
C	Dataset Statistics	16
D	Evaluation Settings	16
E	Additional Experiments	17
E.1	Comparison between Hybrid and Non-Hybrid Models	17
E.2	Efficiency and Compression Analysis	17
E.3	HGPO vs. Standard GRPO under Equivalent Training Resources	18
E.4	Annotation Consistency Analysis for Hybrid Accuracy	19
F	Future Work	19
G	Example Outputs	20

A Implementation Details for DPO and RFT

In this section, we detail the construction pipeline of training data for both DPO and RFT used in § 3. To construct datasets for DPO and RFT, we adopt the same set of queries used in HGPO for offline sampling. Specifically, for each query q , we sample two responses using each of the two thinking modes $m \in \{\top, \bot\}$, resulting in a set of four responses per query as

$$\mathcal{O}(q) = \{o_i^\top\}_{i=1}^2 \cup \{o_i^\bot\}_{i=1}^2. \quad (14)$$

Then the reward function R_ϕ ⁴ in Eq. 15 is applied to score these responses as:

$$\mathcal{R}^\top = \{r(o_i^\top)\}_{i=1}^2, \quad \mathcal{R}^\bot = \{r(o_i^\bot)\}_{i=1}^2. \quad (15)$$

Given the average reward for each mode computed as

$$\bar{\mathcal{R}}^\top = \frac{1}{2} \sum_{i=1}^2 r(o_i^\top), \quad \bar{\mathcal{R}}^\bot = \frac{1}{2} \sum_{i=1}^2 r(o_i^\bot), \quad (16)$$

DPO. For DPO, the training dataset which contains win and lose sample is constructed as following:

$$\mathcal{D}_{\text{DPO}} = \left\{ (q, o_w, o_l) \mid o_w = \arg \max_{o \in \{o_i^{m_w}\}_{i=1}^2} (r(o)), \quad o_l = \arg \min_{o \in \{o_i^{m_l}\}_{i=1}^2} (r(o)) \right\}, \quad (17)$$

while

$$m_w = \arg \max_{m \in \{\top, \bot\}} \bar{\mathcal{R}}^m, \quad m_l = \arg \min_{m \in \{\top, \bot\}} \bar{\mathcal{R}}^m. \quad (18)$$

After getting \mathcal{D}_{DPO} , we optimize the model π_θ initialized from $\pi_{\theta_{\text{HFT}}}$ by using training objective:

$$\max_{\pi_\theta} \mathbb{E}_{(x, o_w, o_l) \sim \mathcal{D}_{\text{DPO}}} \left[\log \sigma \left(\beta \log \frac{\pi_\theta(o_w | x)}{\pi_{\theta_{\text{HFT}}}(o_w | x)} - \beta \log \frac{\pi_\theta(o_l | x)}{\pi_{\theta_{\text{HFT}}}(o_l | x)} \right) \right]. \quad (19)$$

RFT. For RFT, the training dataset is constructed as following:

$$\mathcal{D}_{\text{RFT}} = \left\{ (q, o) \mid o = \arg \max_{o \in \mathcal{O}} (\mathcal{R}^\top \cup \mathcal{R}^\bot) \right\}. \quad (20)$$

The training objectives for RFT are presented as:

$$\mathcal{L}_{\text{RFT}}(\theta) = -\mathbb{E}_{[(x, o) \sim \mathcal{D}_{\text{RFT}}]} \left[\sum_{t=1}^{|o|} \log \pi_\theta(o_t | x, o_{1:t-1}) \right], \quad (21)$$

For implementations, we use LLaMA-Factory [60]⁵ as the codebase for both DPO and RFT. To ensure a fair comparison, we maintain the same learning rate, batch size, and total number of training samples as used in Stage II of HGPO.

B Implementation Details for RLOO and Reinforce++

First of all, We compare different advantage estimators including REINFORCE++ [17], GRPO [41], and RLOO [1], toggling the existence of our HGPO. To make different algorithms compatible with the compound of intra-group rewards and inter-group rewards, we accordingly make adaptations similar to Eq. 10. For Reinforce++, we have

$$A_i^t = \underbrace{\sum_{s=t}^{|o_i|} \gamma^{s-t} \cdot r_{\text{intra}}(o_i)}_{\text{REINFORCE++ for intra-group advantage } A_{\text{intra}}} + \underbrace{\mathbb{1}\{o_i^t \in \Phi\} \cdot \alpha [r_{\text{inter}}(o_i)]}_{\text{REINFORCE++ for inter-group advantage } A_{\text{inter}}}, \quad (22)$$

⁴Same with § 2.3, for queries with definitive answers, we use rule-based rewards [41, 15]; otherwise, a trained parametric reward model [23] is applied.

⁵<https://github.com/hiyouga/LLaMA-Factory.git>

Here is a hyperparameter representing the decay factor, which is set to 0.99 in our experiments. For RLOO, we have

$$A_i^t = \underbrace{\left[r_{\text{intra}}(o_i) - \frac{1}{N-1} \sum_{j \neq i} r_{\text{intra}}(o_j) \right]}_{\text{RLOO for intra-group advantage } A_{\text{intra}}} + \mathbb{1}\{o_i^t \in \Phi\} \cdot \alpha \underbrace{\left[r_{\text{inter}}(o_i) - \frac{1}{N-1} \sum_{j \neq i} (r_{\text{inter}}(o_j)) \right]}_{\text{RLOO for inter-group advantage } A_{\text{inter}}}. \quad (23)$$

From § 3.3, we show that **HGPO is a general plug-in for almost any advantage estimator**, which largely extends the use cases of HGPO. We implement both RLOO and Reinforce++ on VeRL [43]⁶.

C Dataset Statistics

Stage I. Figure 8 shows the token length distributions for the *Thinking* and *No-Thinking* datasets we construct in Stage I. The *Thinking* data has an average length of 575 tokens, with the 25th and 75th percentiles at 362 and 672 tokens, respectively, and a maximum length of 9,148 tokens. In contrast, the *No-Thinking* data exhibits a significantly higher average length of 4,897 tokens, with the 25th and 75th percentiles at 1,719 and 6,738 tokens, and a maximum of 23,997 tokens.

We present the scores and statistics of our constructed dataset for HFT in Table 2. The dataset covers a diverse range of domains, primarily including reasoning-intensive tasks such as mathematics and code, as well as general-purpose question answering.

Stage II. We report the details of the training data used in Stage II in Table 3.

Table 2: Data distribution and source of Stage I.

Category	Source	Data Size	Total	Link
General	WildChat-1M [58]	649,569	674,908	https://huggingface.co/datasets/allenai/WildChat-1M
	OSSAT2 [22]	25,339		https://huggingface.co/datasets/OpenAssistant/oasst1
MATH	SYNTHETIC-1 [35]	343,988	631,325	https://huggingface.co/datasets/PrimeIntellect/SYNTHETIC-1
	OpenR1-Math [12]	93,533		https://huggingface.co/datasets/open-r1/OpenR1-Math-220k
	OpenThought [45]	55,566		https://huggingface.co/datasets/open-thoughts/OpenThoughts-114k
	AoPS [34]	137,497		https://huggingface.co/datasets/di-zhang-fdu/AOPS
	AIME	741		https://huggingface.co/datasets/di-zhang-fdu/AIME_1983_2024
Coding	SYNTHETIC-1 [35]	107,543	381,845	https://huggingface.co/datasets/PrimeIntellect/SYNTHETIC-1
	OpenR1-Codeforces [39]	8,926		https://huggingface.co/datasets/open-r1/codeforces
	OpenThought [45]	19,447		https://huggingface.co/datasets/open-thoughts/OpenThoughts-114k
	KodCode [50]	245,929		https://huggingface.co/datasets/KodCode/KodCode-V1
Others	SYNTHETIC-1 [35]	6,508	6,508	https://huggingface.co/datasets/PrimeIntellect/SYNTHETIC-1
Total	–	1,694,586	1,694,586	–

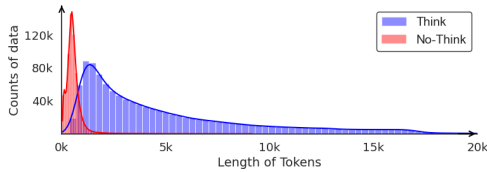


Figure 8: Token length distributions of *Thinking* and *No-Thinking* data in Stage I.

Table 3: Data distribution and source of Stage II. Note that our experiments are conducted on subsets of the two datasets rather than their full versions.

Category	Source	Data Size	Used
General	Tulu 3 [23]	337,000	50,000
	Tulu 3 preference dataset		
MATH	DeepScaler [32]	40,300	40,000
	DeepScaleR-Preview-Dataset		

D Evaluation Settings

In this section, we introduce the details for downstream tasks evaluation in § 3. During inference, we set the temperature to 0.6, the top-p value to 0.95, and the maximum generation length to 35,000 tokens.

⁶<https://github.com/volcengine/verl.git>

- For MATH500, AIME24, GPQA Diamond, and LiveCodeBench, we adopt the OpenR1 [12] evaluation framework. For MBPP and MBPP+, we use the EvalPlus [29, 30] framework with Chain-of-Thought (CoT) [47] prompting. We report pass@1 for all reasoning related benchmarks.
- For AlpacaEval 2.0, we use GPT-4-Turbo as the baseline model, while for Arena-Hard, we adopt evaluation queries from Arena-Hard-v2.0 and use GPT-4o-mini as the baseline. To reduce length-related bias, only the final generated summary is submitted to the judge in think mode.

We provide a summarization and links of evaluation tools we used in experiments in Table 4.

Table 4: Summarization of evaluation tools we used in experiments.

Task	Tool Name	Source
MATH500 [26], AIME24 [2], Livecodebench [20]	open-r1 [12]	https://github.com/huggingface/open-r1
Olympiad Bench, AMC23	math-evaluation-harness	https://github.com/ZubinGou/math-evaluation-harness
MBPP [4, 30], MBPP+ [4, 30]	evalplus [29, 30]	https://github.com/evalplus/evalplus
AlpacaEval [11]	aplaca_eval	https://github.com/tatsu-lab/alpaca_eval
ArenaHard v2.0 [25]	arena-hard-auto	https://github.com/lmarena/arena-hard-auto

E Additional Experiments

E.1 Comparison between Hybrid and Non-Hybrid Models

To further validate the robustness of our hybrid training framework, we perform a controlled comparison between hybrid-trained and non-hybrid models to assess whether hybrid training compromises reasoning capability.

Specifically, we fine-tuned both the 1.5B and 7B variants of *Qwen-2.5-Math* on approximately 1.1M math and code reasoning samples formatted in the *Thinking-style*. These models were trained using identical optimization settings as our hybrid models, allowing for a direct and fair comparison. The two model configurations are defined as follows:

PureThink-SFT models. These models are trained purely on ~ 1.1 M *Thinking-style* math and code data using standard supervised fine-tuning (SFT), without any hybrid formatting or No-Thinking examples.

HFT (Think-Only) models. These models are trained using our Hybrid Fine-Tuning (HFT) pipeline, which jointly incorporates both Thinking and No-Thinking data. During inference, the models are constrained to always operate in the Thinking mode to enable a direct comparison with PureThink-SFT.

As presented in Table 5, both model types achieve nearly identical results across multiple reasoning benchmarks at both the 1.5B and 7B scales. This consistency indicates that hybrid training does not impair reasoning performance. Instead, it provides the additional flexibility of supporting adaptive reasoning modes without compromising capability.

Table 5: Comparison between Hybrid and Non-Hybrid Models on Reasoning-Intensive Benchmarks.

Model (Training Type)	MATH500	AIME24	AMC23	Olympiad	LiveCodeBench	MBPP	MBPP+	Avg.
PureThink-SFT-1.5B	87.6	34.7	72.5	49.9	17.2	63.5	53.2	54.1
HFT-1.5B (Think Only)	87.8	32.7	75.0	50.0	16.8	62.2	52.9	53.9
PureThink-SFT-7B	94.0	56.7	92.5	59.9	38.7	80.7	68.3	70.1
HFT-7B (Think Only)	93.6	56.7	95.0	59.7	38.4	80.3	68.9	70.4

E.2 Efficiency and Compression Analysis

To further analyze the efficiency of the proposed hybrid reasoning approach, we compared the average *thinking token length* and corresponding accuracies of our models against several leading Chain-of-Thought (CoT) compression methods, including **O1-Pruner** [31], **CoT-Valve** [33], and **DAST** [42].

As shown in Table 6, our Large Hybrid-Reasoning Models (LHRMs) achieve a better balance between compression and accuracy. Compared to full-length thinking baselines (HFT-1.5B-Think and HFT-7B-Think), LHRMs significantly reduce reasoning lengths across most tasks while maintaining or improving performance.

Although some prior CoT compression approaches produce shorter outputs (e.g., O1-Pruner, DAST), they often exhibit accuracy degradation. In contrast, our adaptive hybrid framework attains **competitive compression with stronger accuracy**, validating its effectiveness for efficient and reliable inference.

Table 6: Efficiency and compression analysis: benchmark accuracy (outside parentheses) and average thinking token length (inside parentheses) across different CoT compression methods and model variants.

Model	MATH500	AIME24	AMC23	Olympiad	LiveCodeBench	MBPP	MBPP++	AlpacaEval2.0	ArenaHard2.0	Avg.
1.5B Size Models										
HFT-1.5B-Think	87.8 (4379)	32.7 (13431)	75.0 (10181)	50.0 (10480)	16.8 (13628)	62.2 (5090)	52.9 (5090)	15.9 (1927)	9.3 (10616)	44.7 (8314)
HFT-1.5B	87.8 (4330)	32.7 (13431)	75.0 (10181)	48.9 (10480)	15.7 (7460)	54.8 (2803)	47.4 (2803)	13.1 (1101)	6.9 (4287)	42.5 (6320)
O1-Pruner	85.4 (2884)	29.3 (10655)	69.0 (6158)	49.7 (6568)	16.8 (12233)	49.7 (711)	41.8 (711)	13.0 (595)	6.3 (3470)	40.1 (4887)
DAST	84.8 (2428)	26.9 (8429)	64.0 (4637)	46.4 (4968)	16.8 (8238)	55.8 (1412)	46.3 (1413)	14.3 (898)	6.0 (2760)	40.1 (3909)
CoT-Valve	87.0 (3399)	32.0 (11078)	62.5 (8054)	46.1 (7827)	17.1 (9614)	60.1 (4042)	51.9 (4042)	14.1 (1491)	8.1 (6253)	42.1 (6200)
LHRMs-1.5B	87.8 (3722)	35.3 (13491)	75.0 (9065)	50.4 (9490)	17.2 (9342)	61.1 (3103)	54.0 (3103)	16.9 (1250)	10.4 (5289)	45.3 (6428)
7B Size Models										
HFT-7B-Think	93.8 (3658)	56.7 (10778)	95.0 (6456)	59.7 (8376)	38.4 (12046)	80.3 (3251)	68.9 (3251)	30.6 (1731)	23.3 (7442)	60.7 (6332)
HFT-7B	93.6 (3604)	56.7 (10778)	95.0 (6253)	58.5 (7870)	34.7 (7739)	70.6 (1658)	59.8 (1658)	23.7 (779)	14.0 (2216)	56.4 (4728)
O1-Pruner	93.0 (2885)	52.0 (9015)	91.0 (4725)	60.8 (5801)	35.1 (8351)	79.4 (2210)	67.9 (2210)	24.4 (566)	16.0 (2023)	57.7 (4198)
DAST	92.6 (2697)	46.0 (8835)	86.0 (4723)	58.1 (4965)	32.9 (6045)	69.4 (1074)	58.9 (1074)	22.3 (835)	13.8 (1765)	53.3 (3557)
CoT-Valve	93.4 (2964)	50.0 (9692)	93.0 (5122)	58.5 (7151)	37.3 (9970)	80.1 (2987)	67.8 (2987)	26.5 (1553)	23.2 (3445)	58.9 (5097)
LHRMs-7B	93.8 (2616)	66.7 (11031)	95.0 (4976)	61.2 (7540)	38.8 (8432)	81.5 (1906)	69.6 (1906)	35.0 (1086)	26.0 (3416)	63.1 (4768)

These findings highlight the practicality of our adaptive hybrid reasoning framework, which maintains strong accuracy while significantly improving inference efficiency.

E.3 HGPO vs. Standard GRPO under Equivalent Training Resources

To verify that the improvement brought by our **Hybrid Guided Policy Optimization (HGPO)** is not due to additional computational resources, we conducted controlled reinforcement learning (RL) experiments comparing HGPO with standard GRPO under identical setups.

Both baseline models (HFT-1.5B and HFT-7B) were trained using the same RL dataset consisting of 76K queries (sourced from DeepScaleR and Tulu3) and identical hyperparameters. This ensures that any observed differences arise purely from the optimization objective rather than training cost.

As summarized in Table 7, both strategies yield similar performance on reasoning-intensive tasks. However, HGPO-trained models consistently outperform GRPO-trained ones on general-purpose benchmarks, resulting in higher overall averages. We attribute this improvement to HGPO’s **adaptive selection constraint**, which promotes more effective utilization of the “no-think” mode for simple queries and reserves reasoning capacity for complex ones.

In contrast, GRPO lacks this adaptive mechanism and underutilizes the “no-think” mode during rollouts, leading to weaker general-task performance. Moreover, HGPO models achieve significantly higher **hybrid accuracy** (H_{acc}), reflecting stronger ability to select the appropriate reasoning mode per query.

These results confirm that adaptive thinking enhances model effectiveness **without requiring extra computational resources**, establishing HGPO as a key contributor to the efficiency and capability of hybrid reasoning systems.

Table 7: Performance comparison between HGPO and standard GRPO under equivalent training resources. H_{acc} represents hybrid accuracy.

Model	MATH500	AIME24	AMC23	OlympiadBench	LiveCodeBench	MBPP	MBPP+	AlpacaEval2.0	ArenaHard2.0	H_{acc}	Avg.
HFT-1.5B	87.8	32.7	75.0	48.9	15.7	54.8	47.4	13.1	6.9	41.4	42.5
HFT-GRPO-1.5B	89.0	34.7	73.1	50.2	17.9	61.4	52.9	14.3	8.1	47.4	44.6
LHRMs-1.5B	87.8	35.3	75.0	50.4	17.2	61.1	54.0	16.1	10.4	54.4	45.3
HFT-7B	93.6	56.7	95.0	58.5	34.7	70.6	59.8	23.7	14.7	34.2	56.4
HFT-GRPO-7B	95.2	64.0	94.4	61.5	39.6	79.6	67.2	25.8	18.7	41.9	60.7
LHRMs-7B	93.8	66.7	95.0	61.2	38.8	81.5	69.6	35.0	26.7	71.9	63.1

E.4 Annotation Consistency Analysis for Hybrid Accuracy

To ensure the fairness and reliability of our Hybrid Accuracy (\mathcal{H}_{acc}) evaluation, we further computed \mathcal{H}_{acc} using both GPT-4o and human annotations in addition to the Tulu-3-8B-RM reward model. For human evaluations, we randomly sampled 100 queries per model and engaged three annotators, each labeling 800 samples independently.

As shown in Table 8, the relative ranking of models based on Hybrid Accuracy remains consistent across all annotation sources (GPT-4o, human, and Tulu-3-8B-RM). All evaluation methods confirm that LHRMs-7B and LHRMs-1.5B achieve substantial improvements in selecting the correct reasoning mode.

Furthermore, we analyzed the agreement across different annotation sources, summarized in Table 9. Human-to-human agreement averaged around 60%, while human-to-RM agreement was slightly higher (62.5%). Notably, GPT-4o showed the highest consistency, comparable to agreement levels reported in MT-Bench [59] and UltraFeedback [9]. These results validate the robustness and fairness of using Tulu-3-8B-RM for hybrid accuracy evaluation.

Table 8: Hybrid Accuracy (%) of different models evaluated by Tulu-3-8B-RM, GPT-4o, and human annotators.

Model	RM	GPT-4o	Human	Avg.
HFT-1.5B	41.4	41.3	42.0	41.6
HFT-1.5B	48.1	48.2	46.3	47.5
HFT-DPO-1.5B	45.8	43.4	43.3	44.2
LHRMs-1.5B	54.4	52.8	53.7	53.6
HFT-7B	34.2	30.0	39.0	34.4
HFT-RFT-7B	49.7	48.6	47.7	48.7
HFT-DPO-7B	37.1	35.4	39.0	37.2
LHRMs-7B	71.9	71.5	72.0	71.8

Table 9: Agreement rates (%) between annotation sources. GPT-4o shows the highest consistency, confirming reliable alignment with human judgment.

	GPT-4o	RM	H-1	H-2	H-3	Avg.
GPT-4o	-	78.5	69.8	70.1	69.5	69.8
RM	-	-	61.0	64.1	62.3	62.5
H-1	-	-	-	59.6	61.8	60.7
H-2	-	-	-	-	58.6	59.1
H-3	-	-	-	-	-	60.2

F Future Work

While our current work focuses on enabling adaptive hybrid thinking based on query context, there remain several promising directions for future exploration. One potential enhancement is to allow the model to first reason about the context and complexity of the input before deciding whether to engage in extended thinking or direct response generation. This preliminary reasoning step could improve the accuracy of mode selection, leading to better balance between performance and efficiency. We plan to explore this approach in future work to further strengthen the adaptability and generality of our hybrid thinking framework.

G Example Outputs

Question: What is the value of $513^2 - 487^2$? Let's think step by step and output the final answer within boxed(). **Ground Truth:** 26000

DeepSeek-R1-Distill-Qwen-7B: <think> Okay, so I need to calculate 513 squared minus 487 squared. Hmm, let me think about how to approach this. I remember that there's a formula for the difference of squares, which is $a^2 - b^2 = (a - b)(a + b)$. Maybe I can use that here instead of calculating each square separately, which might be time-consuming. ... (3837 characters) </think>
 To solve $513^2 - 487^2$, we use the difference of squares formula $a^2 - b^2 = (a - b)(a + b)$.
 1. Identify $a = 513$ and $b = 487$. 2. Calculate $a - b$:

$$513 - 487 = 26$$

3. Calculate $a + b$:

$$513 + 487 = 1000$$

4. Multiply the results from steps 2 and 3:

$$26 \times 1000 = 26000$$

Verification by direct calculation: $513^2 = 263169$ - $487^2 = 237169$ - Subtracting these values: $263169 - 237169 = 26000$
 Both methods confirm the result.
 Therefore, the final answer is 26000 ✓

LHRMs-7B: <no_think> To solve $513^2 - 487^2$, we can use the difference of squares formula, which states that $a^2 - b^2 = (a + b)(a - b)$. Here, $a = 513$ and $b = 487$.
 Step 1: Calculate $a + b$:

$$513 + 487 = 1000$$

Step 2: Calculate $a - b$:

$$513 - 487 = 26$$

Step 3: Use the difference of squares formula:

$$513^2 - 487^2 = (513 + 487)(513 - 487)$$

$$513^2 - 487^2 = 1000 \times 26$$

Step 4: Perform the multiplication:

$$1000 \times 26 = 26000$$

Therefore, the final answer is: 26000 </no_think> ✓

Figure 9: LHRMs-7B uses the No-Thinking mode to solve easy math problems.

Question: You are given an array of integers `nums` of length `n` and a positive integer `k`. The power of an array is defined as: its maximum element if all of its elements are consecutive and sorted in ascending order, and `-1` otherwise. You need to find the power of all subarrays of `nums` of size `k`. Return an integer array `results` of size `n - k + 1`, where `results[i]` is the power of `nums[i..(i + k - 1)]`.

HFT-7B:

<no_think> To solve this problem, we need to evaluate the power of all subarrays of a given size `k` from the input array `nums`. The power of a subarray is defined as its maximum element if all elements are consecutive and sorted in ascending order; otherwise, it is `-1`.

Here's a step-by-step plan to solve the problem:

1. **Iterate through each subarray of size `k`:** We will slide a window of size `k` through the array `nums` and examine each subarray.
2. **Check if the subarray is sorted and consists of consecutive integers:**
 - First, check if the subarray is sorted in ascending order.
 - Then, check if the elements of the subarray are consecutive integers.
3. **Calculate the power of each subarray:** If the subarray meets the criteria, the power is the maximum element; otherwise, it is `-1`.

Here is the Python code that implements this logic:

```
class Solution:
    def resultsArray(self, nums: List[int], k: int) -> List[int]:
        n = len(nums)
        result = []

        for i in range(n - k + 1):
            subarray = nums[i:i + k]
            sorted_subarray = sorted(subarray)
            max_element = subarray[-1]

            if subarray == sorted_subarray and max_element - subarray[0] == k - 1:
                result.append(max_element)
            else:
                result.append(-1)

        return result
```

</no_think> ✗

LHRMs-7B:

<think> Okay, I need to solve this problem where I have to find the power of all subarrays of size `k` in the given array. The power is defined as the maximum element if all elements are consecutive and sorted in ascending order. Otherwise, it's `-1`. Hmm, let's break down the problem. So for each subarray of length `k`, I have to check two things: whether the elements are consecutive and whether they are sorted in ascending order. ... (2645 characters) </think>

To solve this problem, we need to determine the power of all subarrays of a given size `k` from an array of integers. The power of a subarray is defined as the maximum element if the subarray is both sorted in ascending order and consists of consecutive integers. Otherwise, the power is `-1`.

Approach

1. **Iterate through each possible subarray:** For each starting index `i` from 0 to '`n - k`', extract the subarray of length `k`.
2. **Check if the subarray is sorted:** Verify if each element in the subarray is less than the next element. If not, the power is `-1`.
3. **Check if the subarray consists of consecutive integers:** Ensure each element is exactly one more than the previous element. If not, the power is `-1`.
4. **Determine the power:** If both conditions are met, the power is the last element of the subarray (since it is the maximum in a sorted consecutive sequence). Otherwise, the power is `-1`.

```
class Solution:
    def resultsArray(self, nums: List[int], k: int) -> List[int]:
        result = []
        n = len(nums)
        for i in range(n - k + 1):
            sub = nums[i:i+k]
            # Check if the subarray is sorted in ascending order
            is_sorted = True
            for j in range(len(sub) - 1):
                if sub[j] >= sub[j+1]:
                    is_sorted = False
                    break
            if not is_sorted:
                result.append(-1)
                continue
            # Check if the subarray consists of consecutive integers
            is_consecutive = True
            for j in range(len(sub) - 1):
                if sub[j+1] != sub[j] + 1:
                    is_consecutive = False
                    break
            if is_consecutive:
                result.append(sub[-1])
            else:
                result.append(-1)
        return result
```

✓

Figure 10: LHRMs-7B learned to select the Thinking mode to solve hard code problems.

NeurIPS Paper Checklist

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [Yes]

Justification: Yes, our main contribution is building a hybrid thinking system, which is clearly and accurately described in both the abstract and the introduction.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [Yes] .

Justification: We discuss the limitations of our work in Appendix ??.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: We start from the construction of a hybrid thinking system and propose corresponding training algorithms. Our work does not involve theoretical derivation or validation.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [\[Yes\]](#)

Justification: We have provided detailed reports on our experimental data (see in Appendix C), the implementation details of the baseline methods (see in Appendix A) and our own algorithm (see in § 3), the evaluation tools (see in Appendix D) we used in experiments at both maintext and the appendix.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes] .

Justification: Our training data is constructed by combining multiple publicly available datasets, with all details provided in Appendix C. The training code will be included in the supplementary materials.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We have provided detailed reports on our experimental data (see in Appendix C), the implementation details of the baseline methods (see in Appendix A) and our own algorithm (see in § 3), the evaluation tools (see in Appendix D) we used in experiments at both maintext and the appendix.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: NA

Guidelines:

- The answer NA means that the paper does not include experiments.

- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: we have provided details of computer resources in our experiments in § 3.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [Yes]

Justification: We confirm that our paper align well with the NeurIPS Code of Ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [NA]

Justification: There is no societal impact of the work

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: This paper does not involve such risks, and all data used in our study is publicly available, as detailed in Appendix C.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: For all assets used in our experiments, we have properly credited and respected the license and use explicitly.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset’s creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [\[Yes\]](#) .

Justification: We will organize our training code according to the official template and include it in the supplementary materials.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [\[NA\]](#)

Justification: Our model training, data collection, and evaluation are entirely based on automated metrics and models, without any human involvement.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [\[NA\]](#)

Justification: This paper doesn’t involve crowd-sourcing with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [\[Yes\]](#) .

Justification: Our main contribution is building a LLMs system that can perform hybrid thinking.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.