# Do Thinking Tokens Help or Trap? Towards More Efficient Large Reasoning Model

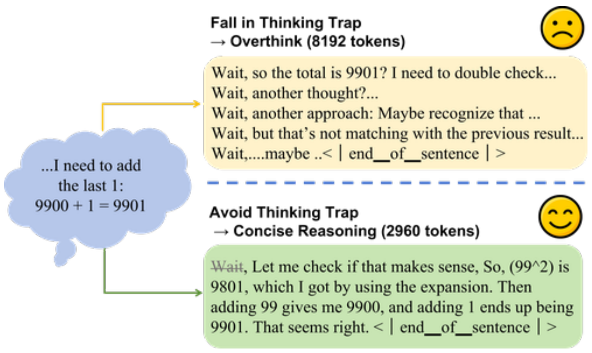
**Bowen Ding**[1,4,*], **Yuhan Chen**[2,*], **Futing Wang**[1,4], **Lingfeng Ming**[3], **and Tao Lin**[4,5,†]
[1] Zhejiang University [2] Boston University [3] ByteDance
[4] School of Engineering, Westlake University
[5] Research Center for Industries of the Future, Westlake University
[4]`{dingbowen, wangfuting, lintao}@westlake.edu.cn`
[2]`erv1n@bu.edu`, [3]`minglingfeng@bytedance.com`

## Abstract

Large Reasoning Models (LRMs) excel at solving complex problems but face an overthinking dilemma. When handling simple tasks, they often produce verbose responses overloaded with thinking tokens (e.g., *wait, however*). These tokens trigger unnecessary high-level reasoning behaviors like reflection and backtracking, reducing efficiency. In this work, our pilot study reveals that these thinking-token-induced behaviors are not essential for effective problem-solving and may even hinder correct reasoning within constrained token budgets. We identify this phenomenon as the thinking trap. To mitigate this issue, we propose **Du**al **P**olicy **P**reference **O**ptimization (DuP-PO), a novel algorithm featuring: (1) A rollout sampling strategy that guarantees balanced exposure to responses with and without thinking tokens; (2) A fine-grained advantage control technique to dynamically regulate the prediction of target tokens; (3) A policy shaping method ensuring stable gradient contributions from thinking tokens. Experimental results on five popular math reasoning benchmarks show that DuP-PO performs well on the popular LRM, which significantly improves their token efficiency during reasoning, while achieving the superior performance of the base model. [1]

## 1 Introduction

Large Reasoning Models (LRMs) excel at generating complex, human-like chains-of-thought (CoTs) for tasks like math, coding, and STEM Q&A (OpenAI, 2024; DeepSeek-AI, 2025; Qwen Team, 2025; Seed et al., 2025). Unlike


Fall in Thinking Trap
→ Overthink (8192 tokens)
...I need to add the last 1: 9900 + 1 = 9901
Avoid Thinking Trap
→ Concise Reasoning (2960 tokens)

Wait, so the total is 9901? I need to double check...
Wait, another thought?...
Wait, another approach: Maybe recognize that ...
Wait, but that's not matching with the previous result...
Wait,....maybe ..< | end__of__sentence | >

~~Wait~~, Let me check if that makes sense, So, (99^2) is 9801, which I got by using the expansion. Then adding 99 gives me 9900, and adding 1 ends up being 9901. That seems right. < | end__of__sentence | >

Figure 1: **The Illustration of Trapped v.s. Efficient Reasoning.** An example from MATH500 where correct inference gets stuck in redundant verification loops, failing to produce a final answer within token limits.

conventional Large Language Models (LLMs), LRMs consistently generate discourse markers such as "*wait, hmm, however*" within their reasoning processes, which we term **thinking tokens** (elaborated in Section 5.3.1). These tokens, in turn, activate advanced cognitive behaviours such as reflection, back-tracking, and thought transitions. This phenomenon is often referred to as an "aha moment" and has traditionally been considered a hallmark of the evolution from a System-1 agent to a System-2 agent (DeepSeek-AI, 2025; Zeng et al., 2025; Muennighoff et al., 2025a).

Recently, a growing number of studies (Sui et al., 2025; Luo et al., 2025a; Feng et al., 2025) indicate that these advanced thought patterns in LRMs can lead to an explosive growth in response length. Even for remarkably simple problems, such as "9900 + 1 =?", LRM responses can become cluttered with extensive and unnecessary reflective thinking patterns, often resulting in outputs exceeding thousands of tokens, as shown in Figure 1. This phenomenon, termed **overthinking**, significantly constrains the practical applicability of LRMs

*Contributed equally.
†Corresponding author.
[1]The project is still under progress and our codes will be released at `https://github.com/Danield21/Dual-Policy-Preference-Optimization`

in real-world scenarios (Chen et al., 2025).

Existing training-based approaches tackle overthinking by collecting variable-length CoT (Xia et al., 2025; Kang et al., 2025; Ma et al., 2025b; Munkhbat et al., 2025) and explicitly penalizing verbose responses (Fu et al., 2024; Yu et al., 2024; Aggarwal and Welleck, 2025). Nevertheless, these methods struggle to achieve an optimal balance between performance gains and token efficiency (Fang et al., 2025; Su and Cardie, 2025), highlighting the need for a deeper understanding of overthinking mechanisms to develop more effective solutions.

Recent work by Muennighoff et al. (2025b) demonstrates that appending thinking tokens (e.g., *wait*) after end-of-thinking delimiters can artificially extend reasoning duration, suggesting that frequent sampling of such tokens may contribute to overthinking. However, the research community has yet to reach consensus on the role that thinking tokens play in the reasoning process, leading us to pose a fundamental question:

*Do Thinking Tokens Help or Trap?*

To answer this question, we conduct systematic experiments analyzing thinking token behaviors in LRMs. Our analysis reveals that thinking tokens may trigger a **thinking trap**, where unproductive reasoning loops that waste computational resources without improving task performance. This challenges the prevailing assumption that more thinking necessarily leads to better reasoning.

Building on the insight, we propose a simple yet effective **r**einforcement **l**earning (RL) algorithm: **Du**al-**P**olicy **P**reference **O**ptimization (DuP-PO), which features three key innovations: (1) **Dual-Policy Sampling** that provides balanced exposure to responses with and without thinking tokens during training; (2) **Token-Level Advantage Scaling** that finely controls the reinforcement of specific tokens based on their utility; and (3) **Policy Shaping** that ensures stable gradient flow for thinking token suppression. This approach enables models to learn when thinking tokens are beneficial versus detrimental, avoiding the thinking trap while maintaining reasoning quality. The main contributions of this work include:

- We identify and characterize the **thinking trap**, where thinking tokens drive unproductive reasoning cycles, providing empirical evidence that challenges their assumed necessity.
- We propose **DuP-PO**, a training-based approach that achieves superior performance-efficiency trade-offs by reinforcing concise correct responses while suppressing problematic thinking token generation.
- We demonstrate DuP-PO's effectiveness across diverse mathematical reasoning benchmarks, achieving 6-20% token reduction with performance improvements through minimal training overhead.

## 2 Related work

### 2.1 Analysis on Thinking Tokens

Since the advent of DeepSeek-R1, thinking tokens, predominantly featuring *wait*, have garnered significant attention. DeepSeek-AI (2025) emphasizes the significance of these tokens, positing that their emergence signals a model's autonomous development of advanced problem-solving strategies. Concurrently, Zhou et al. (2025) and Open-R1-Team (2025) have regarded the model's exhibition of "aha moments" deduced by thinking tokens as an indicator of successfully replicating R1. Furthermore, other works have investigated the origins of these thinking tokens. Liu et al. (2025) suggest that thinking tokens represent pre-existing patterns within base models, which RL methods merely activate.

Moreover, certain studies have delved into the model behaviors and internal mechanisms induced by thinking tokens. Yang et al. (2025d) posit that these thinking tokens can subtly alter the model's perception of problem difficulty, thereby facilitating the resolution of complex problems. Wang et al. (2025a) observe that in Qwen3 series models, tokens like *wait* introduce high uncertainty. Additionally, Qian et al. (2025), by calculating the mutual information between response tokens and the final answer, find mutual information peaks associated with thinking tokens such as *hmm*, *wait*, and *therefore*, suggesting that these tokens possess superior representational capabilities for the answer compared to other tokens.

In contrast, our analysis demonstrates that thinking tokens can induce a trap of redundant cyclic verification, leading to reasoning inefficiencies such as overthinking in the context of mathematical reasoning tasks.

### 2.2 Mitigating Overthinking

To reduce over-thinking in LRMs, prior work explores both training-free and training-based approaches. Training-free methods include prompt-based techniques, which elicit concise reasoning through carefully designed prompts (Ma et al., 2025a), and decoding-based strategies that terminate reasoning based on uncertainty signals or penalize unstable token transitions (Fu et al., 2024; Yang et al., 2025b; Wang et al., 2025c).

Training-based methods promote efficiency through supervised fine-tuning on variable-length CoT data (Xia et al., 2025; Kang et al., 2025; Ma et al., 2025b; Munkhbat et al., 2025; Yu et al., 2024; Cui et al., 2025) or reinforcement learning with length-aware rewards to discourage verbosity (Yeo et al., 2025; Luo et al., 2025b; Team et al., 2025; Aggarwal and Welleck, 2025; Shen et al., 2025; Qu et al., 2025; Hou et al., 2025; Yang et al., 2025c). Hybrid approaches further adapt reasoning depth based on task complexity (Zhang et al., 2025; Tu et al., 2025).

In contrast, our method DuP-PO requires no curated CoT data or predefined reasoning length. It finely regulates thinking token usage, delivering a favorable balance between performance and efficiency.

## 3 Rethinking the Role of Thinking Tokens

In this section, we challenge the prevailing assumption that thinking tokens universally enhance reasoning capabilities in large reasoning models (DeepSeek-AI, 2025; Muennighoff et al., 2025a).

Our empirical analysis of 6,023 test responses from DeepSeek-R1-Distill-Qwen-1.5B (DeepSeek-AI, 2025) reveals a counter-intuitive pattern:

**Observation**

Incorrect responses contain **twice as many thinking tokens** as correct responses.

This finding suggests that thinking token density correlates more strongly with reasoning failures than successes, motivating a fundamental question:

**Question**

Do thinking tokens **genuinely facilitate complex reasoning**? or do they **introduce detrimental overthinking**?

We address this question through two complementary investigations: Section 3.1 examines whether reasoning performance degrades when thinking tokens are removed, while Section 3.2 analyzes specific properties of thinking tokens that may contribute to overthinking behaviors in large reasoning models.

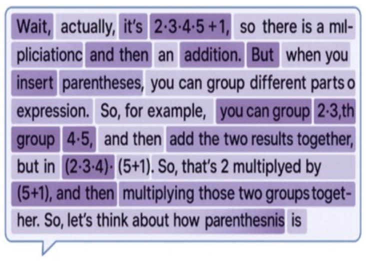
### 3.1 When Fewer Thinking Tokens Maintain a Good Reasoning

To directly test the necessity of thinking tokens, we conduct a controlled experiment where thinking token generation is systematically suppressed through logits penalty (i.e., ThinkTokenPenalty in Section 5.2) on common thinking tokens (*wait, hmm, hold on, alternatively, maybe, however, but, okay*), Then we get a counterintuitive result:

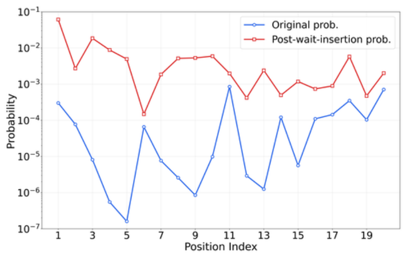
**Observation**

Suppressing thinking token generation produces **minimal degradation** in reasoning performance across difficulty levels.

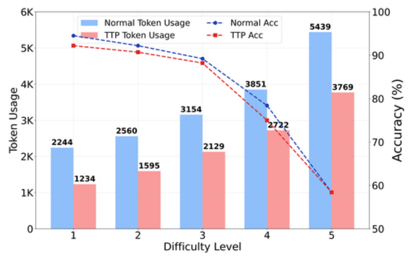
As illustrated in Figure 2c, accuracy remains remarkably stable across MATH500 difficulty levels, with both normal reasoning and ThinkTokenPenalty achieving comparable performance across varying problem difficulties. Simultaneously, the bar chart reveals substantial savings on the reasoning cost, achieving approximately 1,000 token reductions per response across different difficulty levels through disenabling thinking tokens. Furthermore, our error analysis demonstrates that ThinkTokenPenalty significantly reduces reasoning failures attributed to excessive reflection and response truncation, decreasing such failures from 86% to 37%. These empirical findings lead to a critical takeaway:


Wait, actually, it's 2·3·4·5+1, so there is a mil-pliciationc and then an addition. But when you insert parentheses, you can group different parts o expression. So, for example, you can group 2·3,th group 4·5, and then add the two results together, but in (2·3·4)·(5+1). So, that's 2 multiplied by (5+1), and then multiplying those two groups toget-her. So, let's think about how parenthesnis is

(a) The heatmap of token probability.



(b) The illustration of *wait* cascading effect.



(c) Normal reasoning v.s. ThinkTokenPenalty (TTP) on MATH500.

Figure 2: **Analysis of thinking token effects in DeepSeek-R1-Distill-Qwen-1.5B**. (2a) **The heatmap of token probability.** Darker backgrounds indicate higher probabilities, showing systematic over-confidence in thinking tokens like *wait*. (2b) **The illustration of *wait* cascading effect.** Blue circles show the original *wait* probabilities, red squares show 100-fold increases after inserting a single *wait* token, demonstrating auto-regressive amplification across 20 positions. (2c) **Normal reasoning v.s. ThinkTokenPenalty (TTP) on MATH500.** The performance and response token usage across difficulty levels 1-5 is shown, where bars show token usage and dotted lines show accuracy for normal reasoning (blue) and TTP (red). TTP maintains comparable accuracy while reducing token consumption.

**Takeaway**

Thinking tokens are **not necessary** for effective reasoning; their absence often **improves** reasoning efficiency.

We formalize this counterproductive phenomenon as **thinking trap**-reasoning deadlocks caused by excessive self-reflection, as exemplified in Figure 1.

## 3.2 Mechanisms Underlying Thinking Traps

Having established that thinking tokens can impair reasoning performance, we investigate the underlying mechanisms driving this phenomenon. Our analysis identifies two critical properties that enable thinking trap formation: **over-confidence** and **cascading generation**.

**Over-confidence in Thinking Token Prediction.** Analysis of our 6,023 response dataset reveals an average of 106 thinking tokens per response, indicating remarkably high generation frequency. Moreover, Figure 2a demonstrates consistent high-probability patterns for thinking tokens such as *wait* and *but*, evidenced by the prominent highlighting behavior. These observations suggest:

**Hypothesis**

Models assign consistently **high probabilities** to thinking tokens, leading to their **frequent sampling**.

Among these, *wait* emerges as the most prevalent thinking token, constituting 37% of all thinking tokens across responses on average. We therefore focus on *wait* tokens to validate our hypothesis. To test this, we analyze the top 100 responses with the highest *wait* token occurrences, encompassing 492 instances in total. The average model-predicted probability for *wait* at these positions reaches 0.88, confirming systematic over-confidence in *wait* token generation within the LRM.

**Cascading Generation Patterns.** Further analysis of *wait* tokens reveals another concerning pattern: 96% of responses contain multiple (more than one) *wait* tokens. Considering the auto-regressive mechanism of LRM generation, we hypothesize:

**Hypothesis**

Previous thinking token generation **triggers cascading production** of additional thinking tokens.

We validate this through controlled insertion experiments on 100 responses initially containing no *wait* tokens (typically under 3,000 tokens). Inserting a single *wait* token increases subsequent *wait* prediction probabilities by 100-fold over the following 20 positions (Figure 2b), demonstrating strong auto-regressive amplification.

Together, these findings reveal the key mechanisms underlying thinking trap formation:

**Takeaway**

LRMs exhibit **systematic over-confidence** in thinking token utility and **cascading generation behaviors** that **create unproductive reasoning loops**.

## 4 Methodology

Given the insights gained from our analysis on thinking tokens and their detrimental effects on reasoning performance, we propose **Dual-Policy Preference Optimization** (DuP-PO) to mitigate the thinking trap dilemma.

DuP-PO extends GRPO to **precisely control thinking token usage**. Rather than eliminating thinking tokens arbitrarily, DuP-PO learns to strategically regulate their generation, enabling models to engage in productive reasoning while avoiding the thinking trap.

In this section, we first provide an overview of GRPO in Section 4.1, followed by a detailed explanation of the core concepts and implementation of DuP-PO in Section 4.4.

### 4.1 Preliminary

In this section, we review the key components of GRPO that underpin our approach.

### 4.2 GRPO

The GRPO algorithm streamlines PPO (Schulman et al., 2017) by replacing the traditional value network with group-based advantage estimation. The algorithm operates by sampling $G$ response trajectories $\{\boldsymbol{\tau}_i\}_{i=1}^{G}$ from the current policy $\pi_{\theta_{\text{old}}}$ for each query-answer pair $(\boldsymbol{q}, \boldsymbol{a})$ in dataset $\mathcal{D}$. Each trajectory receives a reward score $R_i$ through rule-based evaluation.

The binary reward function exemplifies this approach:

$$R_i = \begin{cases} 1.0 & \text{if } \texttt{is\_equivalent}(\boldsymbol{a}, \boldsymbol{\tau}_i) \\ 0.0 & \text{otherwise} \end{cases} \tag{1}$$

This formulation creates a clear preference hierarchy: trajectories satisfying $\texttt{is\_equivalent}(\boldsymbol{a}, \boldsymbol{\tau}_i)$ are designated as **preferred** and receive positive rewards, while all other trajectories are treated as **unpreferred** with zero rewards.

**Advantage Estimation.** With these reward assignments, GRPO transforms the group-level preferences into token-level training signals. The advantage for any token $\boldsymbol{\tau}_{i,t}$ is computed through group normalization:

$$A_i^t = \frac{R_i - \text{mean}(\{R_i\}_{i=1}^{G})}{\text{std}(\{R_i\}_{i=1}^{G})} \tag{2}$$

This equation indicates that GRPO assigns the identical advantage to all tokens within a trajectory, treating all reasoning content identically regardless of their distinct roles in response quality.

**Policy Optimization Objective.** GRPO aims to maximize the expected advantage of the policy while preventing the current policy from deviating excessively from the reference policy. The optimization objective $\mathcal{J}(\theta)$ incorporates token-level advantages through a clipped policy gradient formulation that balances preference learning with training stability:

$$\mathcal{J}(\theta) = \mathbb{E}_{\mathcal{D}, \pi_{\theta_{\text{old}}}} \left[ \frac{1}{\sum_{i=1}^{G} |\tau_i|} \sum_{i=1}^{G} \sum_{t=1}^{|\tau_i|} \min\left(r_i^t A_i^t, C_i^t A_i^t\right) \right] \tag{3}$$
$$- \beta \cdot \mathbb{D}_{\text{KL}}\left[\pi_\theta || \pi_{\text{ref}}\right]$$

where $r_i^t = \frac{\pi_\theta(\tau_{i,t}|\boldsymbol{q}, \tau_{i,<t})}{\pi_{\theta_{\text{old}}}(\tau_{i,t}|\boldsymbol{q}, \tau_{i,<t})}$ is the importance ratio measuring probability shifts for each token, and $C_i^t = \text{clip}(r_i^t, 1-\epsilon, 1+\epsilon)$ applies clipping to maintain training stability. The clipping trick ensures that when the importance ratio exceeds the trust region bounds $[1-\epsilon, 1+\epsilon]$, the gradient contribution is capped, preventing excessive policy updates that could destabilize training.

### 4.3 Token-Level Policy Gradient

As a policy gradient method, GRPO utilizes the policy gradient to control the token prediction. Building on the optimization objective in Equation (3), the policy gradient for token $\boldsymbol{\tau}_{i,t}$ follows a conditional update rule:

$$\nabla_\theta \mathcal{J}(\boldsymbol{\tau}_{i,t}) = \begin{cases} \nabla_\theta \log \pi_\theta(\boldsymbol{\tau}_{i,t}) A_i^t & \text{if } A_i^t > 0 \\ & \text{and } r_i^t < 1+\epsilon \\ \nabla_\theta \log \pi_\theta(\boldsymbol{\tau}_{i,t}) A_i^t & \text{if } A_i^t < 0 \\ & \text{and } r_i^t > 1-\epsilon \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

which exposes three fundamental properties:

- **Magnitude Scaling:** The advantage magnitude $|A_i^t|$ directly controls gradient strength, allowing proportional reinforcement based on trajectory success. Higher-performing trajectories generate stronger learning signals for all constituent tokens.
- **Directional Control:** The advantage sign determines whether token probabilities increase (positive advantage from preferred trajectories) or decrease (negative advantage from un-preferred ones), providing clear directional guidance for policy updates.
- **Gradient Gating:** The importance ratio $r_i^t$ acts as a gate, zeroing gradients when policy changes exceed clipping bounds. This mechanism prevents unstable updates while preserving meaningful learning signals within the trust region.

These properties form the foundation for DuP-PO's ability to apply distinct training signals to some concerned tokens (i.e., thinking tokens) versus others, enabling precise control over when and how models engage in self-reflection. We leverage these insights to design our algorithm in Section 4.4.

### 4.4 DuP-PO

DuP-PO addresses two key limitations of GRPO: its inability to preferentially reinforce concise preferred trajectories and its failure to effectively suppress unpreferred trajectories that fall into the thinking trap.

The core training objective of DuP-PO extends GRPO's formulation:

$$\mathbb{E}_{\mathcal{D},(\pi_n,\pi_r)}\Big[\frac{1}{\sum_{i=1}^{N+M}|\boldsymbol{\tau}_i|}\sum_{i=1}^{N+M}\sum_{t=1}^{|\boldsymbol{\tau}_i|} \min\left(\hat{r}_i^t\hat{A}_i^t, \operatorname{clip}(\hat{r}_i^t, 1-\epsilon, 1+\epsilon)\hat{A}_i^t\right)\Big] - \beta\cdot\mathbb{D}_{\mathrm{KL}}\left[\pi_\theta||\pi_{\mathrm{ref}}\right]. \tag{5}$$

where $\hat{A}_i^t = m_i^t \cdot A_i^t$ represents scaled advantages and $\hat{r}_i^t$ denotes calibrated importance ratios. It enables three complementary components:

1. **Dual-Policy Sampling** generates both thinking-heavy and thinking-free trajectories during rollout, providing comparative examples that GRPO's single-policy sampling cannot achieve.
2. **Token-level Advantage Scaling** breaks GRPO's uniform advantage constraint by applying differential scaling factors $m_i^t$ to thinking tokens based on the trajectory's characteristic.
3. **Policy Shaping** ensures a consistent gradient flow for thinking token suppression by calibrating importance ratios $\hat{r}_i^t$, overcoming GRPO's clipping limitations that can zero out crucial learning signals.

#### 4.4.1 Dual-Policy Sampling

Our first innovation addresses the single-policy sampling limitation of GRPO through a dual-policy approach during the rollout phase.

**GRPO Sampling Limitation.** GRPO samples all trajectories from a single policy $\pi_{\theta_{\text{old}}}$ (referred to as the normal policy $\pi_n$). For models prone to overthinking, this creates a critical problem: nearly all sampled trajectories contain excessive thinking tokens, providing no examples of the concise responses we want to encourage. Without seeing both thinking-heavy and thinking-free responses for the same query, GRPO cannot learn to prefer concise reasoning over overthinking.

**Rectified Policy Design.** To provide balanced training examples, we introduce a rectified policy ($\pi_r$) that systematically eliminates thinking tokens during generation. This policy operates by setting the logit values of predefined thinking tokens to $-\infty$, effectively zeroing their generation probability. For any token $\boldsymbol{\tau}_{\cdot,t} \in \mathcal{S}_{\text{think}}$ (our predefined set of thinking tokens), the rectified policy is defined as:

$$\pi_r(\boldsymbol{\tau}_{\cdot,t}|\boldsymbol{q},\boldsymbol{\tau}_{\cdot,<t}) = \begin{cases}\delta\approx 0, & \text{if } \boldsymbol{\tau}_{\cdot,t}\in\mathcal{S}_{\text{think}}\\ \pi_n(\boldsymbol{\tau}_{\cdot,t}|\boldsymbol{q},\boldsymbol{\tau}_{\cdot,<t}), & \text{if } \boldsymbol{\tau}_{\cdot,t}\notin\mathcal{S}_{\text{think}}\end{cases}$$

**Balanced Trajectory Generation.** During rollout, DuP-PO samples $N$ trajectories from the rectified policy $\{\boldsymbol{\tau}_i^r\}_{i=1}^N \sim \pi_r$ and $M$ trajectories from the normal policy $\{\boldsymbol{\tau}_i^n\}_{i=1}^M \sim \pi_n$.

This dual-sampling strategy ensures the model observes both response types for each query, enabling it to learn the comparative value of concise versus thinking-heavy approaches. Unlike GRPO's uniform sampling, this provides the contrastive examples necessary for effective thinking token regulation.

#### 4.4.2 Token-Level Advantage Scaling

Our second innovation breaks GRPO's constraint of assigning the identical advantage within a trajectory. DuP-PO applies token-specific advantage scaling based on the token's attribute and the corresponding trajectory preference.

**GRPO Identical Advantage Limitation.** Section 4.2 demonstrates that GRPO assigns identical advantages to all tokens within a trajectory, limiting fine-grained control over token-level optimization. Since advantage magnitude and sign directly determine each token's likelihood adjustment in the updated policy (see Section 4.3), this uniform treatment creates a fundamental constraint: GRPO cannot selectively encourage concise reasoning while suppressing certain overthinking triggers (i.e., thinking tokens) within the same trajectory.

**Advantage Scaling Mechanism.** To overcome this limitation, we introduce a scaling factor $m_i^t$ that modifies the original advantage $A_i^t$ to produce calibrated advantages $\hat{A}_i^t = m_i^t \cdot A_i^t$. The scaling factor is determined by both the advantage sign and trajectory source:

$$m_i^t := \begin{cases} \alpha, & \text{if } A_i^t > 0 \text{ and } \boldsymbol{\tau}_i \sim \pi_r \\ \beta, & \text{if } A_i^t < 0 \text{ and } \boldsymbol{\tau}_i \sim \pi_n \\ & \text{and } \boldsymbol{\tau}_{i,t} \in \mathcal{S}_{\text{think}} \\ 0, & \text{if } A_i^t > 0 \text{ and } \boldsymbol{\tau}_i \sim \pi_n \\ & \text{and } \boldsymbol{\tau}_{i,t} \in \mathcal{S}_{\text{think}} \\ & \text{and } \exists j \text{ s.t. } A_j > 0 \text{ and } \boldsymbol{\tau}_j \sim \pi_r \\ 1, & \text{otherwise} \end{cases} \tag{6}$$

Equation (6) enables four distinct advantage operations, each of which targets a specific scenario:

- **Enhancement ($m_i^t = \alpha > 1$):** Amplifies advantages for preferred trajectories from the rectified policy, strongly reinforcing thinking-free successful responses.
- **Suppression ($m_i^t = \beta > 1$):** Magnifies negative advantages for thinking tokens in unpreferred normal policy trajectories, actively discouraging overthinking behaviors that lead to incorrect responses.
- **Return-to-Zero ($m_i^t = 0$):** Eliminates advantages for thinking tokens in preferred normal policy trajectories when equivalent thinking-free solutions exist, removing redundant learning signals.
- **Identity ($m_i^t = 1$):** Preserves original advantages for all other tokens, maintaining standard GRPO behavior where differential treatment is unnecessary.

This design enables preferential reinforcement of concise preferred trajectories through enhancement while simultaneously suppressing the triggers (i.e., thinking tokens) of overthinking.

### 4.4.3 Policy Shaping on Thinking Tokens

Our third innovation ensures consistent gradient flow for thinking token suppression by addressing GRPO's gradient clipping limitation.

**GRPO Gradient Clipping Limitation.** As established in Section 4.3, GRPO's policy gradient for token $\boldsymbol{\tau}_{i,t}$ is zeroed when the importance ratio $r_i^t$ falls outside the clipping bounds $[1-\epsilon, 1+\epsilon]$. For thinking tokens in unpreferred trajectories (where $A_i^t < 0$), gradients are preserved only when $r_i^t > 1-\epsilon$, which requires:

$$\pi_\theta(\boldsymbol{\tau}_{i,t}|\boldsymbol{q}, \boldsymbol{\tau}_{i,<t}) > (1-\epsilon) \cdot \pi_{\theta_{\text{old}}}(\boldsymbol{\tau}_{i,t}|\boldsymbol{q}, \boldsymbol{\tau}_{i,<t})$$

However, this condition is difficult to satisfy due to thinking tokens' inherently high prediction probabilities. Section 3.2 shows that the thinking token *wait* being around 0.88 in average probability, implying a high suppression threshold. With the standard $\epsilon = 0.2$, thinking tokens are suppressed only when $\pi_\theta(\boldsymbol{\tau}_{i,t}|\boldsymbol{q}, \boldsymbol{\tau}_{i,<t}) > (1-0.2) \times 0.88 = 0.704$. Such high threshold causes critical suppression signals to be clipped away when we need them most.

**Old Policy Calibration.** To ensure consistent gradient flow for thinking token suppression, we calibrate the old policy probability for thinking tokens. Inspired by LUFFY (Yan et al., 2025), we reshape the old policy probability as:

$$\hat{\pi}_{\theta_{\text{old}}}(\boldsymbol{\tau}_{i,t}|\boldsymbol{q}, \boldsymbol{\tau}_{i,<t}) = \frac{\gamma}{1-\epsilon}, \quad \forall \boldsymbol{\tau}_{i,t} \in \mathcal{S}_{\text{think}} \tag{7}$$

where $\gamma$ is a small constant (we use $\gamma = 0.1$). Then, the importance ratio is modified as:

$$\hat{r}_i^t = \frac{\pi_\theta(\boldsymbol{\tau}_{i,t}|q, \boldsymbol{\tau}_{i,<t})}{\hat{\pi}_{\theta_{\text{old}}}(\boldsymbol{\tau}_{i,t}|q, \boldsymbol{\tau}_{i,<t})} = \frac{\pi_\theta(\boldsymbol{\tau}_{i,t}|q, \boldsymbol{\tau}_{i,<t})}{\gamma} \cdot (1-\epsilon) \tag{8}$$

**Guaranteed Gradient Flow.** With this calibration, the clipping condition $\hat{r}_i^t > 1 - \epsilon$ is satisfied whenever $\pi_\theta(\boldsymbol{\tau}_{i,t}|\boldsymbol{q}, \boldsymbol{\tau}_{i,<t}) > \gamma$. Since thinking tokens typically maintain probabilities well above $\gamma = 0.1$ during training, this ensures that suppression gradients for thinking tokens are rarely clipped. Combined with our token-level advantage scaling, this strategy provides reliable and consistent learning signals for thinking token regulation throughout the training process.

## 5 Experiments

In this section, we discuss our experimental setup, detailed implementation and baselines.

### 5.1 Setup

**Model.** The training process is initialized with DeepSeek-R1-Distill-Qwen-1.5B, which is a popular LRM developed by DeepSeek-AI (2025).

**Training Data.** Inspired by recent works (Yang et al., 2025a; Bae et al., 2025), we prioritise the quality over quantity for RL training data, curating 1,000 problems of medium difficulty from the DAPO-MATH-17K dataset (Yu et al., 2025). Specifically, we processed 28,000 random DAPO-MATH-17K samples with the LRM, using 0.6 temperature to generate 4 responses per question with 16K maximum length. The final dataset comprised entries exhibiting an answer correctness rate between 0.25 and 0.5, coupled with an average response length exceeding 8192 tokens. These selected data are of moderate difficulty and feature longer responses, making them prone to the thinking trap. We name the curated dataset as DuPPO-1K.

**Validation Data.** We utilise AIME24 as the validation set, following the setting of DAPO (Yu et al., 2025) and VAPO (Yue et al., 2025). During inference on this validation set, the temperature is set to 0. Adhering to the principle of balancing performance and token usage optimally, we report the evaluation results for the following checkpoints: DeepSeek-R1-Distill-Qwen-1.5 with GRPO at 90-step, DeepSeek-R1-Distill-Qwen-1.5 with DuP-PO at 80-step.

**Benchmarks and Metrics.** To evaluate performance, we selected six popular math reasoning benchmarks: AIME 2024, AIME 2025, AMC (Jia LI et al., 2024), Minerva (Lewkowycz et al., 2022), OlympiadBench (He et al., 2024), and MATH500 (Lightman et al., 2024). Given the comparatively smaller test sets of AIME 2024, AIME 2025, and AMC, we present results using Avg@32. For Minerva, OlympiadBench and MATH500, Pass@1 is used. The temperature is set to 0.6, and the max response length is set to 8,192 for testing. To mitigate systematic errors caused by sampling randomness, all reported results represent the average of three inference runs.

### 5.2 Baselines

We compare DuP-PO against the base model (DeepSeek-R1-Distill-Qwen-1.5B) and two test-time efficiency baselines:

- **NoThink** (Ma et al., 2025a) bypasses internal reasoning by appending "*Okay, I think I have finished thinking.*</think>" after "<think>" in prompts, forcing models to generate concise answers directly.
- **ThinkTokenPenalty** (Wang et al., 2025b) applies the logit penalty to thinking tokens (e.g., *however, alternatively*) during inference. We adopt an aggressive variant that penalizes all thinking tokens to prevent their sampling entirely. This technique aligns with sampling from the rectified policy $\pi_r$ described in Section 4.4.

### 5.3 RL Practice

#### 5.3.1 Implementation Details

**Training and Inference.** We use VeRL (Sheng et al., 2024) to implement DuP-PO, where the training context size, batch size, and the learning rate are 8,192, 128 and $2e{-}6$. The actor policy update batch size is 128. The total rollout number is set as 8, including $N = 4$ responses sampling from the normal policy $\pi_n$, and $M = 4$ responses sampling from the rectified policy $\pi_r$. We remove the KL loss term by setting $\beta = 0$ and set the entropy loss coefficient to 0.01. In terms of the newly added hyperparameters of DuP-PO, we set the *enhancement factor* $\alpha$ and the *suppression factor* $\beta$ as 2 for advantage scaling. For the 1.5B model, we train them within 100 steps. All training experiments are conducted on a single 8xH800

Table 1: **Main results on six mathematical reasoning benchmarks.** **Score** represents the average performance, using Pass@1 for MINERVA, MATH500, and OLYMPIADBENCH (for short OLYMPIAD), and Avg@32 for AMC, AIME24, and AIME25. **Len** denotes the average response length. Higher **Score** values and lower **Len** values indicate better performance. Bold entries highlight the best result in each column.

| Model | MATH500 | | OLYMPIAD | | MINERVA | | AIME24 | | AMC | | AIME25 | | Average | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Score (↑) | Len (↓) | Score (↑) | Len (↓) | Score (↑) | Len (↓) | Score (↑) | Len (↓) | Score (↑) | Len (↓) | Score (↑) | Len (↓) | Score (↑) | Len (↓) | ΔScore (↑) | ΔLen (↓) |
| **DeepSeek-R1-Distill-Qwen-1.5B** | | | | | | | | | | | | | | | | |
| Base model | 79.2 | 3760 | 43.0 | 5992 | 29.3 | 4821 | 21.8 | 7410 | 55.4 | 5812 | 20.4 | 7277 | 43.9 | 6105 | – | – |
| + NoThink | 72.3 | **2022** | 40.5 | **4052** | 24.4 | **1272** | 21.9 | 6157 | 53.8 | 4153 | 18.5 | 6335 | 41.8 | 4502 | −2.1 | −26.3% |
| + ThinkTokenPenalty | 77.7 | 2547 | 43.8 | 4081 | 27.7 | 2392 | 20.1 | **5694** | 57.2 | **3956** | 18.8 | **5053** | 44.0 | **4234** | +0.1 | −30.6% |
| + GRPO | 81.4 | 3345 | 45.0 | 5614 | 30.1 | 4482 | 24.0 | 7169 | 58.7 | 5359 | 23.2 | 6956 | 46.6 | 5724 | +2.7 | −6.2% |
| + DuP-PO | **82.7** | 2830 | **47.4** | 5033 | **30.6** | 3585 | **25.0** | 6795 | **60.9** | 4722 | **21.7** | 6499 | **47.9** | 5162 | +4.0 | −15.4% |

node, and all inference experiments use four NVIDIA RTX 4090 GPUs.

**Thinking Tokens.** For thinking token identification, we categorize common thinking tokens in LRM responses into reflection tokens (*wait, hmm, hold on, okay*) and thought transition tokens (*alternatively, maybe, but, however*). Our analysis in Section 3.2 focuses on the *wait* token as a representative case, while both DuP-PO and ThinkTokenPenalty penalize all identified thinking tokens to maximize token efficiency.

### 5.3.2 Reward Design

We design a rule-based reward function that incorporates both correctness and formatting rewards, similar to the general RL training approach used in DeepSeek-R1 (DeepSeek-AI, 2025). The reward function evaluates two components: answer correctness through Math-Verify parsing and response formatting quality. Specifically, correct answers receive a base reward of 1.0, with an additional 0.1 bonus for well-formatted responses. The reward function is formally defined as:

$$R(\boldsymbol{\tau}, \boldsymbol{a}) = \begin{cases} 1.1, & \text{if } \texttt{is\_equivalent}(\boldsymbol{a}, \boldsymbol{\tau}) \\ & \quad \tau \text{ is well-formatted} \\ 1.0, & \text{if } \texttt{is\_equivalent}(\boldsymbol{a}, \boldsymbol{\tau}), \\ & \quad \tau \text{ is not well-formatted} \\ 0.1, & \text{if } \texttt{not\_equivalent}(\boldsymbol{a}, \boldsymbol{\tau}), \\ & \quad \tau \text{ is well-formatted} \\ 0, & \text{if } \texttt{not\_equivalent}(\boldsymbol{a}, \boldsymbol{\tau}), \\ & \quad \tau \text{ is not well-formatted} \end{cases} \tag{9}$$

where $\texttt{is\_equivalent}(\boldsymbol{a}, \boldsymbol{\tau})$ determines whether the ground truth answer $\boldsymbol{a}$ can be successfully parsed from the response trajectory $\boldsymbol{\tau}$.

## 6 Results and Discussion

We conduct comprehensive experiments across six widely-adopted mathematical reasoning benchmarks to evaluate DuP-PO's effectiveness. Our empirical analysis demonstrates consistent improvements in both performance and token efficiency, as summarized in Table 1.

### 6.1 Comparision with Training-free Baselines

We first establish the performance ceiling of training-free approaches. While the prompt-based NoThink method exhibits the expected performance-efficiency trade-off (26.3% token reduction with a 2.1-point accuracy penalty), **ThinkTokenPenalty** emerges as the superior training-free method, **achieving** 30.6% **token reduction while preserving accuracy** on DeepSeek-R1-Distill-Qwen-1.5B. This finding reveals that thinking tokens in the base model may contribute minimally to reasoning quality, primarily inflating response length. Such result aligns with and extend the key insights from Section 3.1 across a broader range of datasets, providing additional empirical validation that thinking tokens are largely dispensable for reasoning.

However, training-based optimization unlocks substantially greater potential. Compared to ThinkTokenPenalty, DuP-PO outperforms by 3.9 points while maintaining a significant token efficiency, demonstrating superior performance-efficiency trade-offs that training-free methods cannot achieve.

**Takeaway**

Training-based optimization is essential: DuP-PO **substantially outperforms** all training-free approaches on performance.

### 6.2 Comparision with Base Models

Our empirical evaluation on DeepSeek-R1-Distill-Qwen-1.5B reveals that DuP-PO achieves substantial improvements with minimal training overhead. Specifically, the method requires only 80 RL steps to deliver 4.0 points average performance gain alongside 15.4% token reduction.

DuP-PO achieves consistent performance improvements across benchmark complexities with substantial efficiency gains. Simpler benchmarks like MATH500 show pronounced benefits with 3.5-point improvements and 24.7% token savings. On more challenging benchmarks such as AIME24 and AIME25, performance gains remain consistent while token reduction becomes more conservative. AIME24 improves from 21.8 to 25.0 with 8.3% token reduction, and AIME25 from 20.4 to 21.7 with 10.7% fewer tokens. These results indicate DuP-PO's adaptive optimization across problem complexities.

**Takeaway**

DuP-PO consistently **improves both performance and token efficiency** over the base model with **minimal training cost**.

### 6.3 Comparision with GRPO

We conduct a controlled comparison against GRPO to isolate the contributions of our proposed innovations. Standard GRPO training (90 steps) yields 2.7 points average improvement with 6.2% token reduction relative to the base model.

DuP-PO demonstrates clear algorithmic superiority. With fewer training iterations (80 vs 90 steps), our method achieves 1.3 points higher accuracy than GRPO while consuming fewer reasoning tokens (5,162 vs 5,724). This demonstrates superior efficiency with reduced training iterations and lower token consumption during inference.

**Takeaway**

DuP-PO outperforms GRPO in **performance**, **token efficiency**, and **training speed** simultaneously.

## 7 Conclusion

In this paper, we present DuP-PO, a novel reinforcement learning algorithm designed to address the overthinking problem in the 1.5B LRM. Our experimental results demonstrate that frequent use of thinking tokens is not necessarily essential for model performance improvement. Through fine-grained control of thinking tokens, our method achieves superior balance between performance enhancement and token efficiency compared to baseline approaches, requiring only lightweight training on the base LRM across multiple mathematical benchmarks. For future work, we plan to validate the reliability and robustness of our approach by extending experiments to larger model architectures and diverse domain benchmarks.

## References

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

Sanghwan Bae, Jiwoo Hong, Min Young Lee, Hanbyul Kim, JeongYeon Nam, and Donghyun Kwak. 2025. Online difficulty filtering for reasoning oriented reinforcement learning. *Preprint*, arXiv:2504.03380.

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

Yingqian Cui, Pengfei He, Jingying Zeng, Hui Liu, Xianfeng Tang, Zhenwei Dai, Yan Han, Chen Luo, Jing Huang, Zhen Li, and 1 others. 2025. Stepwise perplexity-guided refinement for efficient chain-of-thought reasoning in large language models. *arXiv preprint arXiv:2502.13260*.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *CoRR*, abs/2501.12948.

Gongfan Fang, Xinyin Ma, and Xinchao Wang. 2025. Thinkless: Llm learns when to think. *Preprint*, arXiv:2505.13379.

Sicheng Feng, Gongfan Fang, Xinyin Ma, and Xinchao Wang. 2025. Efficient reasoning models: A survey. *Preprint*, arXiv:2504.10903.

Yichao Fu, Junda Chen, Siqi Zhu, Zheyu Fu, Zhongdongming Dai, Aurick Qiao, and Hao Zhang. 2024. Efficiently serving llm reasoning programs with certaindex. *arXiv preprint arXiv:2412.20993*.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. 2024. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. *Preprint*, arXiv:2402.14008.

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

Jia LI, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang, Kashif Rasul, Longhui Yu, Albert Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. 2024. Numinamath.

Yu Kang, Xianghui Sun, Liangyu Chen, and Wei Zou. 2025. C3ot: Generating shorter chain-of-thought without compromising effectiveness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 24312–24320.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. 2022. Solving quantitative reasoning problems with language models. *Preprint*, arXiv:2206.14858.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let's verify step by step. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Zichen Liu, Changyu Chen, Wenjun Li, Tianyu Pang, Chao Du, and Min Lin. 2025. There may not be aha moment in r1-zero-like training — a pilot study. `https://oatllm.notion.site/oat-zero`. Notion Blog.

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

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

Wenjie Ma, Jingxuan He, Charlie Snell, Tyler Griggs, Sewon Min, and Matei Zaharia. 2025a. Reasoning models can be effective without thinking. *arXiv preprint arXiv:2504.09858*.

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

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel J. Candès, and Tatsunori Hashimoto. 2025a. s1: Simple test-time scaling. *CoRR*, abs/2501.19393.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. 2025b. s1: Simple test-time scaling. *Preprint*, arXiv:2501.19393.

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

Open-R1-Team. 2025. Mini r1 countdown game. Blog post.

OpenAI. 2024. Learning to reason with llms. `https://openai.com/index/learning-to-reason-with-llms/`. Accessed: 2025-05-07.

Chen Qian, Dongrui Liu, Haochen Wen, Zhen Bai, Yong Liu, and Jing Shao. 2025. Demystifying reasoning dynamics with mutual information: Thinking tokens are information peaks in llm reasoning. *Preprint*, arXiv:2506.02867.

Yuxiao Qu, Matthew YR Yang, Amrith Setlur, Lewis Tunstall, Edward Emanuel Beeching, Ruslan Salakhutdinov, and Aviral Kumar. 2025. Optimizing test-time compute via meta reinforcement fine-tuning. *arXiv preprint arXiv:2503.07572*.

Qwen Team. 2025. Qwq-32b-preview. `https://qwenlm.github.io/blog/qwq-32b-preview/`. Accessed: 15 March 2025.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *ArXiv*, abs/1707.06347.

ByteDance Seed, :, Jiaze Chen, Tiantian Fan, Xin Liu, Lingjun Liu, Zhiqi Lin, Mingxuan Wang, Chengyi Wang, Xiangpeng Wei, Wenyuan Xu, Yufeng Yuan, Yu Yue, Lin Yan, Qiying Yu, Xiaochen Zuo, Chi Zhang, Ruofei Zhu, Zhecheng An, and 255 others. 2025. Seed1.5-thinking: Advancing superb reasoning models with reinforcement learning. *Preprint*, arXiv:2504.13914.

Yi Shen, Jian Zhang, Jieyun Huang, Shuming Shi, Wenjing Zhang, Jiangze Yan, Ning Wang, Kai Wang, Zhaoxiang Liu, and Shiguo Lian. 2025. Dast: Difficulty-adaptive slow-thinking for large reasoning models. *arXiv preprint arXiv:2503.04472*.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2024. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*.

Jinyan Su and Claire Cardie. 2025. Thinking fast and right: Balancing accuracy and reasoning length with adaptive rewards. *Preprint*, arXiv:2505.18298.

Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Hanjie Chen, and Xia Ben Hu. 2025. Stop overthinking: A survey on efficient reasoning for large language models. *CoRR*, abs/2503.16419.

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, and 1 others. 2025. Kimi k1.5: Scaling reinforcement learning with llms. *arXiv preprint arXiv:2501.12599*.

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

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. 2025a. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. *Preprint*, arXiv:2506.01939.

Yue Wang, Qiuzhi Liu, Jiahao Xu, Tian Liang, Xingyu Chen, Zhiwei He, Linfeng Song, Dian Yu, Juntao Li, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. 2025b. Thoughts are all over the place: On the underthinking of o1-like llms. *Preprint*, arXiv:2501.18585.

Yue Wang, Qiuzhi Liu, Jiahao Xu, Tian Liang, Xingyu Chen, Zhiwei He, Linfeng Song, Dian Yu, Juntao Li, Zhuosheng Zhang, and 1 others. 2025c. Thoughts are all over the place: On the underthinking of o1-like llms. *arXiv preprint arXiv:2501.18585*.

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

Jianhao Yan, Yafu Li, Zican Hu, Zhi Wang, Ganqu Cui, Xiaoye Qu, Yu Cheng, and Yue Zhang. 2025. Learning to reason under off-policy guidance. *arXiv preprint arXiv:2504.14945*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025a. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

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

Junjie Yang, Ke Lin, and Xing Yu. 2025c. Think when you need: Self-adaptive chain-of-thought learning. *arXiv preprint arXiv:2504.03234*.

Shu Yang, Junchao Wu, Xin Chen, Yunze Xiao, Xinyi Yang, Derek F. Wong, and Di Wang. 2025d. Understanding aha moments: from external observations to internal mechanisms. *Preprint*, arXiv:2504.02956.

Edward Yeo, Yuxuan Tong, Morry Niu, Graham Neubig, and Xiang Yue. 2025. Demystifying long chain-of-thought reasoning in llms. *arXiv preprint arXiv:2502.03373*.

Ping Yu, Jing Xu, Jason Weston, and Ilia Kulikov. 2024. Distilling system 2 into system 1. *arXiv preprint arXiv:2407.06023*.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, and 16 others. 2025. Dapo: An open-source llm reinforcement learning system at scale. *Preprint*, arXiv:2503.14476.

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

Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. 2025. Simplerl-zoo: Investigating and taming zero reinforcement learning for open base models in the wild. *Preprint*, arXiv:2503.18892.

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

Hengguang Zhou, Xirui Li, Ruochen Wang, Minhao Cheng, Tianyi Zhou, and Cho-Jui Hsieh. 2025. R1-zero's "aha moment" in visual reasoning on a 2b non-sft model. *Preprint*, arXiv:2503.05132.

## Contents