RLVR TRAINING OF LLMS DOES NOT IMPROVE THINKING ABILITY FOR GENERAL QA: EVALUATION METHOD AND A SIMPLE SOLUTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning from verifiable rewards (RLVR) stimulates the thinking processes of large language models (LLMs), substantially enhancing their reasoning abilities on verifiable tasks. It is naturally assumed that similar gains should transfer to general question answering (GQA), but this assumption has not been thoroughly validated. To assess whether RLVR effectively improves LLM performance on GQA, we propose a Cross-Generation evaluation framework that measures the quality of intermediate thinking by feeding the generated thinking context into LLMs of varying capabilities. Our evaluation leads to a discouraging finding: the efficacy of the thinking process on GQA tasks is markedly lower than on verifiable tasks, suggesting that explicit training on GQA remains necessary in addition to training on verifiable tasks. We further observe that direct RL training on GQA is less effective than RLVR. Our hypothesis is that, whereas verifiable tasks demand robust logical chains to obtain high rewards, GQA tasks often admit shortcuts to high rewards without cultivating high-quality thinking. To avoid possible shortcuts, we introduce a simple method, Separated Thinking And Response Training (START), which first trains only the thinking process, using rewards defined on the final answer. We show that START improves both the quality of thinking and the final answer across several GQA benchmarks and RL algorithms.

1 INTRODUCTION

Large language models (LLMs) have shown remarkable progress in complex reasoning tasks (Guo et al., 2025), especially when trained or prompted to perform explicit intermediate *thinking* steps. A particular effective paradigm is reinforcement learning from verifiable rewards (RLVR), where models are optimized using outcome-based rewards driven from verifiable answers. By avoiding the need for dense human annotations and relying on automatically verifiable correctness signals, RLVR has become a scalable recipe for eliciting strong thinking behaviors in LLMs. These successes naturally raise an important question: *do the thinking abilities stimulated from RLVR transfer to broader, less easily verifiable tasks,* e.g, general QA (GQA)? A common assumption is that RLVR, by improving the model's general capacity for thinking, should also improve GQA performance, as suggested by rubric-based methods (Gunjal et al., 2026; Kim et al., 2024).

To empirically validate whether these reasoning gains transfer to GQA tasks, we propose a Cross-Generation Evaluation framework designed to isolate the intrinsic quality of the thinking process. This method operates by feeding different thinking traces generated by the source model into different models, treating the thinking traces as part of the prompt. The underlying premise is that high-quality reasoning should be universally beneficial, acting as a scaffold that improves the performance of any model. However, our evaluation reveals a discouraging disparity: while thinking traces learned via RLVR on verifiable tasks significantly boost performance in reasoning contexts, their efficacy drops precipitously when applied to GQA. Specifically, we observe that the marginal performance gains derived from employing a stronger thinking traces are significantly overshadowed by the improvements achieved through utilizing a more capable answering model. This finding challenges the assumption of automatic transfer and suggests that explicit training on GQA tasks remains essential alongside verifiable task training.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

However, our experiments indicate that direct RL is markedly less effective in fostering genuine thinking abilities than verifiable tasks. We identify a critical phenomenon termed *thinking stagnation*: while direct RL yields substantial improvements in the quality of the final answers, the corresponding enhancement in the intermediate thinking process is disproportionately marginal. We hypothesize that this discrepancy arises from the weak coupling between thinking and response quality in general domains. Unlike verifiable tasks, where a correct solution relies strictly on a robust and error-free logical chain, GQA tasks often admit shortcuts to high rewards Turpin et al. (2023); Singhal et al. (2024); Gao et al. (2023); Skalse et al. (2022). In these scenarios, the model can learn to satisfy the reward function by optimizing the final answer only, bypassing the need to cultivate high-quality, structured thinking.

To mitigate the influence of these reward shortcuts, we propose Separated Thinking And Response Training (START), a simple yet effective two-stage training paradigm. The core innovation of START lies in the decoupling of the thinking process from the response generation during the first training stage. Specifically, we first train only the thinking process while keeping the final answer generation fixed, using rewards derived from the resulting complete response. From an reinforcement learning perspective, this approach transforms the answer generation from a high-variance part of the model's exploration space into a stable component of the environment. Through this, we force the RL algorithm to attribute reward solely to the quality of the thinking trace, encouraging the model to cultivate genuine thinking capabilities robustly coupled with answer accuracy. Our empirical results demonstrate that START consistently outperforms joint RL baselines across a variety of RL algorithms and diverse GQA datasets, yielding both higher reward scores and superior answer quality.

The main contributions of this work are summarized as follows: (i) We propose a novel Cross-Generation evaluation framework to isolate the intrinsic value of thinking processes, revealing that unlike in verifiable tasks, the efficacy of the thinking process in GQA is significantly limited. (ii) We identify a thinking stagnation phenomenon where standard RL fails to effectively evolve the model's thinking process. (iii) We introduce START, a two-stage training paradigm that decouples the thinking process from response generation by treating the final answer as a stable part of the environment instead of model exploration.

2 Thinking Stagnation of RL on GQA

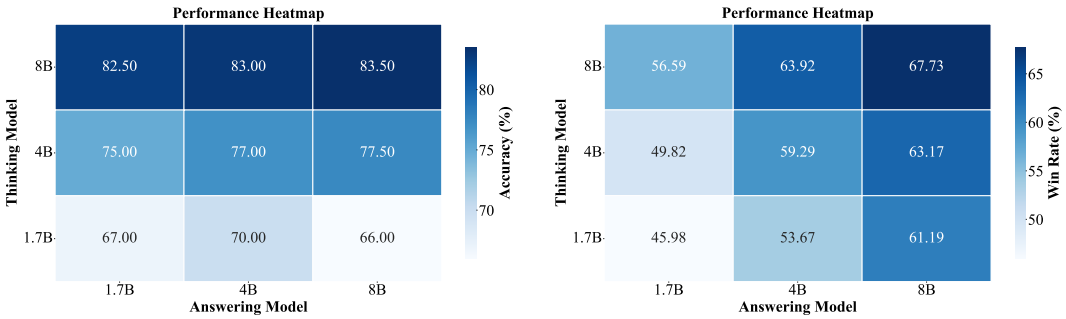
2.1 Uncovering the Limited Efficacy of Thinking in GQA

In this subsection, we investigate whether the significant reasoning enhancements observed in verifiable tasks through RLVR naturally generalize to the broader domain of General Question Answering (GQA).

To empirically disentangle the quality of thinking from the final response, we implement a *Cross-Generation* evaluation framework. Specifically, for a given answer sequence $T_X + A_Y$, T and A represent the thinking trace and the final answer respectively. We first allow model X to generate the thinking trace, then this trace is provided as a fixed prefix (pre-filled context) to model Y, which is tasked to generate the final answer.

To empirically isolate the cognitive utility of thinking traces, we utilize the Qwen3 model series (1.7B, 4B, and 8B) Yang et al. (2025). These models, by virtue of their varying parameter scales, naturally generate thinking traces of distinct quality and depth. For reasoning tasks, we evaluate performance on the MATH dataset Hendrycks et al. (2021) using answer accuracy as the standard metric. For GQA, we employ the AlpacaEval 2.0 benchmark Li et al. (2023), reporting the standard *length-controlled win rate* against *GPT-4-1106-preview* OpenAI et al. (2023) to ensure a robust measure of general answering quality. Detailed hyperparameters and experimental configurations are provided in Appendix A. By implementing the Cross-Generation framework, we construct performance heatmaps that visualize the interaction between thinking and answering efficacy across both domains.

The results from our experiments (Figure 1) reveal a fundamental disparity in how thinking traces translate to final output quality across different domains. In the reasoning tasks (MATH), a high-quality thinking trace serves as the primary determinant of success. Upgrading the thinking trace



(a) Reasoning (MATH) (b) General QA (AlpacaEval 2.0)

Figure 1: **Cross-generation performance heatmaps.** Comparing the influence of thinking vs. answering model capacity. (Left) In reasoning tasks, performance is almost entirely dictated by the quality of the thinking trace. (Right) In general tasks, the answering model capacity remains a dominant factor, and the performance provided by superior thinking is less decisive.

from a 1.7B-level to an 8B-level for a 1.7B answering model results in a massive performance surge, from 67.00% to 82.50%. Crucially, once a high-quality thinking trace is provided, increasing the answering model's capacity yields almost no additional benefit-the accuracy only marginally improves from 82.50% (1.7B model) to 83.50% (8B model). This indicates that in formal reasoning, the thinking process is the clear bottleneck, and its potential is already being effectively exploited. However, the dynamics in the GQA (AlpacaEval) are strikingly different. While a stronger thinking trace does improve the win rate (e.g., from 45.98% to 56.59% for a 1.7B answering model), this improvement is far less decisive. In fact, the gain from upgrading the answering model while keeping the thinking trace fixed is significantly more pronounced: the same 8B thinking trace paired with an 8B answering model jumps to 67.73%, a much larger leap than the gain provided by the thinking trace alone.

These observations suggest that in the general domain, the thinking-to-answering transition is highly inefficient. The latent potential and efficacy of thinking traces in GQA remain largely untapped compared to their impact in reasoning tasks. This underscores the necessity for additional, targeted reinforcement learning training in GQA to bridge this cognitive gap and ensure the model to leverage its internal thinking more effectively.

2.2 The Stagnation of Thinking in Reinforcement Learning on GQA

Building upon the findings in Section 2.1, which revealed the limited efficacy of existing thinking processes in the GQA domain, we conduct further reinforcement learning experiments directly on GQA datasets, aiming to determine if targeted RL signals can force the model to evolve more effective and substantive thinking traces that lead to better task performance.

We continue to employ the Cross-Generation evaluation framework specified in section 2.1. This protocol enables us to assess whether an RL-tuned model has developed a superior thinking engine, or whether the improvements are merely localized to the answering phase. To quantify performance, we report both the reward achieved after training and the Win Rate, defined as the frequency with which a specific configuration's output is preferred over the output of the Base model in a pairwise comparison.

As presented in Table 1, the results on general task datasets (ExpertQA Malaviya et al. (2024), UltraFeedback Cui et al. (2024) and AlpacaEval Li et al. (2023)) reveal a striking phenomenon of thinking-answering decoupling. Across various model scales (Qwen3-1.7B, 4B and 8B) and RL objectives (GRPO and DAPO Yu et al. (2025)), the configuration $T_{\text{post}} + A_{\text{pre}}$ yields nearly negligible gains in Reward and Win Rate. For instance, in Setting A (Qwen3-1.7B on ExpertQA), the Reward only marginally increases from 0.1842 to 0.1849 ($+0.0007$). This suggests that the thinking traces generated by the post-trained model offer no more cognitive utility than those from the base model. Conversely, the $T_{\text{pre}} + A_{\text{post}}$ configuration captures the vast majority of the post-training gains, often matching or even exceeding the performance of the full post-trained model.

Table 1: **Stagnation of thinking evolution in general tasks.** Experimental verification of the thinking-answering decoupling effect. We report Reward and Win Rate (vs. Base) across various model scales, RL algorithms, and datasets. Δ denotes the reward improvement relative to the Base model. Subscripts denote the model state before (*pre*) and after (*post*) RL fine-tuning.

Config.	Reward	Δ R	Win Rate
Setting A: Qwen3-1.7B / GRPO / ExpertQA			
Base	0.1842	—	—
$T_{\text{post}} + A_{\text{pre}}$	0.1849	**+0.0007**	52.87%
$T_{\text{pre}} + A_{\text{post}}$	0.2148	+0.0306	100.0%
Post-trained	0.2182	+0.0340	100.0%
Setting B: Qwen3-4B / GRPO / ExpertQA			
Base	0.1924	—	—
$T_{\text{post}} + A_{\text{pre}}$	0.1921	**-0.0003**	47.77%
$T_{\text{pre}} + A_{\text{post}}$	0.2213	+0.0289	99.36%
Post-trained	0.2224	+0.0300	96.50%
Setting C: Qwen3-8B / GRPO / ExpertQA			
Base	0.1964	—	—
$T_{\text{post}} + A_{\text{pre}}$	0.1958	**-0.0006**	47.13%
$T_{\text{pre}} + A_{\text{post}}$	0.2331	+0.0368	100.00%
Post-trained	0.2355	+0.0391	100.00%

Config.	Reward	Δ R	Win Rate
Setting D: Qwen3-1.7B / DAPO / ExpertQA			
Base	0.1842	—	—
$T_{\text{post}} + A_{\text{pre}}$	0.1868	**+0.0026**	58.92%
$T_{\text{pre}} + A_{\text{post}}$	0.2261	+0.0419	100.0%
Post-trained	0.2352	+0.0510	100.0%
Setting E: Qwen3-1.7B / GRPO / UltraFeedback			
Base	0.1535	—	—
$T_{\text{post}} + A_{\text{pre}}$	0.1539	**+0.0004**	49.25%
$T_{\text{pre}} + A_{\text{post}}$	0.1677	+0.0142	86.50%
Post-trained	0.1658	+0.0123	76.25%
Setting F: Qwen3-1.7B / GRPO / AlpacaEval			
Base	0.1539	—	—
$T_{\text{post}} + A_{\text{pre}}$	0.1539	**+0.0000**	51.53%
$T_{\text{pre}} + A_{\text{post}}$	0.1625	+0.0086	80.82%
Post-trained	0.1642	+0.0103	81.82%

Table 2: **Comparison of thinking evolution across different models and domains.** We report Reward and Win Rate (vs. Base) for ExpertQA, and Accuracy for MATH. Δ denotes the reward improvement relative to the Base model. Subscripts denote the model state before (*pre*) and after (*post*) RL fine-tuning.

Config.	Reward	Δ R	Win Rate
Setting A: Hunyuan-1.8B-Instruct / ExpertQA			
Base	0.1628	—	—
$T_{\text{post}} + A_{\text{pre}}$	0.1740	**+0.0112**	69.11%
$T_{\text{pre}} + A_{\text{post}}$	0.1942	+0.0314	93.31%
Post-trained	0.2111	+0.0483	98.41%
Setting B: R1-Distill-Qwen-1.5B / ExpertQA			
Base	0.1327	—	—
$T_{\text{post}} + A_{\text{pre}}$	0.1478	**+0.0151**	71.97%
$T_{\text{pre}} + A_{\text{post}}$	0.1536	+0.0209	90.13%
Post-trained	0.1664	+0.0337	92.68%

Config.	Accuracy	Δ Acc
Setting C: Qwen3-1.7B / MATH		
Base	55.00%	—
$T_{\text{post}} + A_{\text{pre}}$	62.60%	**+7.60**
$T_{\text{pre}} + A_{\text{post}}$	62.45%	+7.45
Post-trained	64.20%	+9.20

As shown in Table 2, we extend our analysis to models with different architectures, such as Hunyuan-1.8B-Instruct Tencent (2025) and DeepSeek-R1-Distill-Qwen-1.5B Guo et al. (2025). While these models show slightly more "active" thinking traces compared to Qwen3, with the $T_{\text{post}} + A_{\text{pre}}$ configuration yielding relative reward gains of +0.0112 and +0.0151 respectively, these improvements are still significantly overshadowed by the gains in the answering phase. In Setting A, for instance, the $T_{\text{pre}} + A_{\text{post}}$ configuration achieves a +0.0314 reward increase, almost triple what the evolved thinking alone contributes. This indicates that even when some degree of thinking evolution occurs, the RL process continues to prioritize the answering part as the primary vehicle for reward acquisition. In sharp contrast, for reasoning-intensive tasks like Setting C (MATH), the thinking process is no longer stagnant and instead evolves into an engine for reaching the performance ceiling: the $T_{\text{post}} + A_{\text{pre}}$ configuration accounts for the vast majority of the total gain. In this domain, the model is effectively forced to evolve its thinking process to achieve success.

This empirical evidence suggests a clear **Thinking Stagnation** where the RL process primarily optimizes the model's ability to generate high-reward answers from existing thinking traces, rather than evolving the thinking process itself to provide better logical foundations.

3 SEPARATED THINKING AND RESPONSE TRAINING (START)

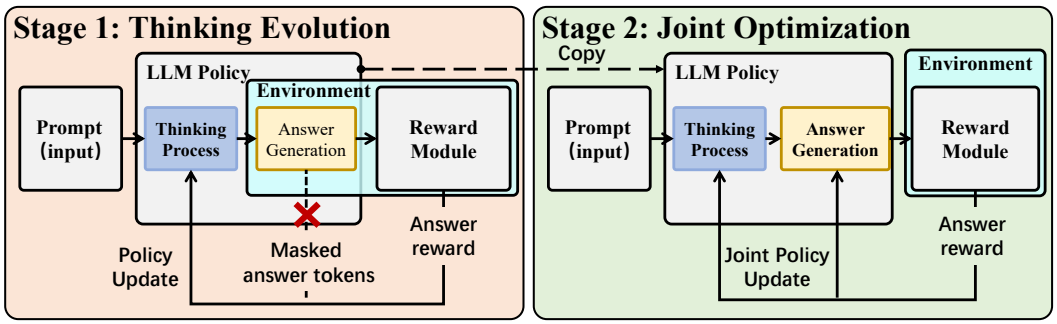


Figure 2: Overall framework of START method.

Based on the Thinking Stagnation uncovered in Section 2, we introduce START (Separated Thinking And Response Training). This framework decouples the training of cognitive thinking from response generation through a two-phase optimization process.

3.1 PRELIMINARIES: RL FOR LLMS AS AN MDP

To formalize the proposed method, we take an reinforcement learning perspective of LLMs and formalize the language generation task as an Markov Decision Process (MDP) Puterman (1994); Ramamurthy et al. (2023), defined by the tuple $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$:

- **State Space $\mathcal{S}$:** A state s_t at step t consists of the initial instruction x and the tokens generated so far, i.e., $s_t = [x, y_1, \ldots, y_{t-1}]$.
- **Action Space $\mathcal{A}$:** The action corresponds to the vocabulary $\mathcal{V}$ of the LLM. At each step t, the model samples a token $y_t \in \mathcal{V}$.
- **Transition Dynamics P:** In language generation, the transition is deterministic where the next state s_{t+1} is formed by appending the action y_t to the current sequence.
- **Reward Function R:** We focus on outcome-based RL where a scalar reward $r(x, s)$ is provided only after the complete sequence $s = [T, A]$ is generated, where T denotes thinking tokens and A denotes answer tokens.
- **Policy π_θ:** The policy $\pi_\theta(y_t|s_t)$ represents the LLM being optimized, which defines the probability distribution over the vocabulary given the current state.

In standard end-to-end RL frameworks, the goal is to maximize the expected reward $\mathbb{E}_{s\sim\pi_\theta}[r(x, s)]$. Under this formulation, the entire sequence $s = [T, A]$ is treated as the policy's actions, meaning both thinking and answering reside within the same **exploration space**.

3.2 CHANGE ANSWERING FROM EXPLORATION TO ENVIRONMENT

In standard reinforcement learning for LLMs, the entire generated sequence $s = [T, A]$ (where T denotes thinking tokens and A denotes answer tokens) is typically treated as the policy π_θ's action. Consequently, both thinking and answering reside within the same exploration space. While in verifiable tasks, a correct solution relies strictly on a robust and error-free logical chain, the weak coupling between thinking and answering in general tasks implies that the model is not strictly required to explore superior thinking processes to achieve higher rewards. Consequently, the reward signal becomes more directly associated with the answer tokens, allowing the model to improve performance while the thinking process remains in a state of stagnation.

To break this shortcut, START proposes a fundamental shift in the optimization perspective: **transforming the answering phase from a part of the policy's exploration space into a fixed component of the environment**. During the initial stage of training, the policy π_θ is only considered responsible for generating the thinking trace T. The subsequent generation of the answer A is no

longer treated as part of model exploration, but is instead redefined as part of the environmental dynamics.

From the perspective of the RL agent, the answering phase becomes a stationary transition process that maps a given thought T to a final outcome. By reclassifying the answering phase as part of the environment, we effectively remove it from the reward optimization, making exploring and evolving superior thinking traces T the only viable path for the policy to achieve higher rewards. This shift ensures that the optimization pressure is purely concentrated on the evolution of internal thinking.

3.3 Phase I: Thinking Evolution via Gradient Masking

To operationalize the conceptual shift described in Section 3.2, we introduce a gradient masking mechanism during the first phase of training. During the backward pass of the RL objective (e.g., GRPO), we apply a gradient mask to the answer part. Specifically, the loss function for Phase I is modified as:

$$\mathcal{L}_{START} = \mathbb{E}\left[\sum_{i \in T \cup A} M_i \cdot \nabla_\theta \log \pi_\theta(s_i | x, s_{<i}) \cdot \hat{A}\right]$$

where M_i is a binary mask defined as:

$$M_i = \begin{cases} 1 & \text{if } s_i \in T \\ 0 & \text{if } s_i \in A \end{cases}$$

Through this masking, the answer A, while still sampled from the model, functions as a fixed response head that maps the generated thought T to a reward. Because the gradients for tokens in A are zeroed out, the model is physically unable to improve its reward by merely adjusting its answering. In this configuration, the only way for the policy to improve the advantage $\hat{A}$ is to evolve thinking traces that are more logically rigorous or informative. This phase ensures that the evolution of internal thinking is the sole driver of performance gains.

3.4 Phase II: Joint Optimization

Once the model has developed a robust thinking engine in Phase I, we transition to Phase II: a standard full-parameter RL fine-tuning. In this stage, the gradient mask is removed ($M_i = 1$ for all $i \in T \cup A$), allowing the model to jointly optimize both thinking and answering. The primary goal of Phase II is to align the answering with the newly acquired thinking capabilities, ensuring that the model can effectively extract and present the logical insights generated in T.

3.5 Integrated with Standard RL Algorithms

START is a simple but effective method, which could be naturally integrated with any existing RL algorithms for LLM optimization. Unlike Process-based Reward Models (PRMs) Uesato et al. (2022) that require expensive step-level annotations, START utilizes existing outcome-based Reward Models. It requires only a minor modification to the loss calculation (a simple token-level mask), making it compatible with most modern RL frameworks like DAPO or other open-source GRPO implementations. This makes START an out-of-the-box solution for researchers seeking to boost thinking capabilities in general domains without complex engineering.

4 Experiment

4.1 Setup

We utilize Qwen3-1.7B as the seed model for all training iterations. To provide reliable and scalable feedback, we employ ArmoRM Wang et al. (2024), an 8B-parameter multi-reward model that outputs a scalar score for a single response, as our reward model. We experiment with GRPO and DAPO using ExpertQA, UltraFeedback and AlpacaEval, focusing on general question answering and instruction following with actual human instructions.

To further benchmark the efficacy of our method against specialized reasoning-alignment techniques, we include GRPO-MA Wang et al. (2025) as a baseline. GRPO-MA is designed to address gradient coupling and unstable advantage estimation in standard GRPO by employing a multi-answer generation strategy. Specifically, for each generated thinking trace, the model samples multiple independent answers. The advantage of a thinking trace is then calculated based on the aggregated performance of its associated answers, decoupling the quality of the reasoning process from the stochasticity of a single response.

For each configuration, the baseline consists of the same seed model trained under the identical RL algorithm but without the two-phase separation. All baselines are trained until convergence to ensure a fair comparison. All experiments are executed on 64 AMDEPYC9374F32-core processors, 8 NVIDIA GeForce RTX 4090 GPUs, and 1TB of RAM to facilitate parallelized computation. More details about datasets, models, algorithms are provided in the Appendix A.

4.2 Main Result

The primary experimental results are summarized in Table 3 and the reward convergence is visualized in Figure 3. We analyze the performance of START across two critical dimensions: holistic performance and standalone thinking utility.

Table 3: **Main results of START on ExpertQA.** Win Rates are calculated head-to-head against the post-trained GRPO baseline.

Config.	**Post-trained**		T_{post} + A_{pre}	
	Reward	Win Rate	Reward	Win Rate
GRPO	0.2182	—	0.1849	—
GRPO-MA	0.2140	34.08%	0.1859	54.14%
+START	**0.2201**	**59.24%**	**0.1876**	**68.15%**

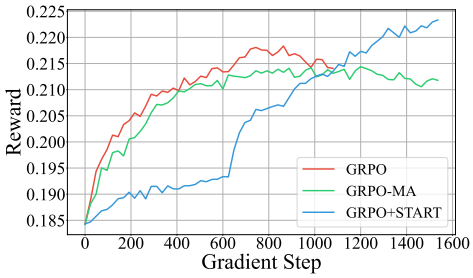


Figure 3: **Reward curves on ExpertQA.**

The most significant finding is that START successfully overcomes the thinking stagnation identified in Section 2. As shown in the T_{post}+A_{pre} column of Table 3, when pairing the post-trained thinking traces with a frozen base answering head, GRPO+START achieves a remarkable 68.15% win rate against the vanilla GRPO baseline. These results provide direct empirical evidence that by masking the answering gradients in Phase I, START effectively forces the model to allocate its optimization capacity toward evolving its internal reasoning logic.

Our results further demonstrate that cultivating a superior thinking engine leads to a significantly higher performance ceiling for the final model. In the holistic *Post-trained* evaluation, GRPO+START achieves a win rate of 59.24% over vanilla GRPO and reaches a higher reward of approximately 0.220. As illustrated in Figure 3, the reward curve for GRPO+START (blue line) exhibits a steady ascent during the first 600 steps of Phase I and a subsequent surge in Phase II, rapidly surpassing both the GRPO and GRPO-MA baselines. By first establishing a more robust logical foundation in Phase I, the model can more effectively align its answering in Phase II, resulting in a final output that is not only stylistically preferred but also cognitively deeper.

We also compare START against GRPO-MA. While GRPO-MA shows a moderate improvement in standalone thinking utility (54.14% win rate in the T_{post}+A_{pre} setting), it fails to match the gains of START (68.15%) and eventually reaches a lower reward plateau in the final post-trained model (as shown in Figure 3). While GRPO-MA samples multiple answers per thought to provide a more stable and accurate estimate of the thought's value, its optimization remains joint. In the context of GQA, where the coupling between thinking and results is inherently weak, the model still finds it cheaper to optimize the answer tokens to capture the stable reward signal rather than evolving the complex logic required for high-quality thinking. Also, for a fixed total sampling budget ($G = K \times M$), GRPO-MA needs to sample multiple answers per thought (M), thus reduces the number of unique thinking paths (K) explored and reduces the diversity of final answers. This compression of the exploration space may explains why GRPO-MA slightly underperform standard GRPO baseline in certain GQA benchmarks.

Table 4: **Comprehensive results across different datasets, models, and algorithms.** Win Rates are calculated head-to-head against the respective vanilla post-trained base algorithms.

Setting (Dataset / Model / Algorithm)	Vanilla Reward	START Reward	START Win Rate (vs. Vanilla)
ExpertQA / Qwen3-1.7B / DAPO	0.2352	**0.2385**	65.29%
UltraFeedback / Qwen3-1.7B / GRPO	0.1658	**0.1695**	61.25%
AlpacaEval / Qwen3-1.7B / GRPO	0.1642	**0.1672**	65.75%
ExpertQA / Hunyuan-1.8B / GRPO	0.2111	**0.2170**	69.43%
ExpertQA / Qwen3-4B / GRPO	0.2224	**0.2296**	68.98%

Beyond the primary GRPO results on ExpertQA, we extended our evaluation to include the DAPO algorithm, the UltraFeedback dataset and AlpacaEval dataset to further test the versatility of our approach. These supplementary experiments consistently demonstrate that START yields significant performance gains across varying RL frameworks and data distributions, effectively validating its robustness and broad generalization capabilities in facilitating thinking evolution. More experiment results can be found in Appendix C.

4.3 START Evolves Genuine Thinking Capabilities

Table 5: **Results of Phase I.** Decoupled contributions on ExpertQA (Qwen-1.7B). Win rates (WR) are relative to the base model.

Configuration	Reward	Δ R	Win Rate
Base	0.1842	—	—
T_{post} + A_{pre}	0.1888	**+0.0046**	69.11%
T_{pre} + A_{post}	0.1897	+0.0055	78.98%
Post-trained	0.1927	+0.0085	84.71%

Table 6: **Cross-task generalization.** Models trained on *ExpertQA* tested on *AlpacaEval*. START shows superior OOD generalization.

Method	Config.	Reward	Win Rate
Base	Qwen3-1.7B	0.1539	—
Direct RL	T_{post}+A_{pre}	0.1537	48.52%
	Post-trained	0.1589	63.89%
START	T_{post}+A_{pre}	**0.1551**	**54.11%**
	Post-trained	**0.1607**	**74.32%**

To further understand the dynamics of the START framework, we analyze the training characteristics and performance gains of Phase I (Thinking Evolution) in Table 5.

The results of the T_{post}+A_{pre} configuration provide direct evidence of thinking evolution. By simply replacing the base model's thinking traces with those generated after Phase I training, the model achieves a significant reward increase of $+0.0046$ and a win rate of 69.11%. This confirms that Phase I, through its gradient-masking mechanism, successfully forces the model to optimize its internal thinking independently of adjustments in the final answer. Importantly, we find that the thinking quality developed in Phase I merely degrade when the answering gradient is unmasked in Phase II. Instead, the improved thinking process serves as a robust foundation, allowing the answering head to extract more accurate insights and achieve a massive leap in performance. Moreover, as shown in our training curves in Figure 3, this masking strategy ensures a smooth and steady convergence of the thinking process, despite the loose coupling between thinking traces and final answers.

Moreover, the results presented in Table 6 highlight the robust cross-task generalization of the thinking processes evolved via START. When evaluated on the unseen AlpacaEval benchmark, the decoupled thinking component (T_{post}+A_{pre}) of the START-trained model achieves a 54.11% win rate, significantly outperforming the 48.52% win rate of the direct RL baseline. This performance gap provides empirical evidence that START effectively overcomes the thinking stagnation phenomenon identified earlier with Phase I. This further underscores that the improvements derived from START are not merely localized to training-set specific information but represent a fundamental enhancement in the model's underlying thinking engine, allowing it to provide a more effective logical foundation even in out-of-distribution scenarios.

5 Related Work

5.1 Thinking Models and RLVR

In recent years, the focus of Large Language Models (LLMs) research has shifted from simple next-token prediction toward the development of complex reasoning capabilities. Central to this evolution is the emergence of the *Thinking-Answer* paradigm, where models generate a Chain-of-Thought (CoT) Wei et al. (2022); Wang et al. (2023b) as an intermediate reasoning step before producing a final response.

The advent of Reinforcement Learning from Verifiable Rewards (RLVR) Guo et al. (2025); Liu et al. (2024) has significantly pushed the boundaries. Modern reasoning models, such as OpenAI o1(OpenAI, 2024) and DeepSeek-R1, have demonstrated that RL can substantially enhance reasoning performance in tasks with verifiable outcomes, such as mathematics, coding, and formal logic Lightman et al. (2023); Cobbe et al. (2021); Hendrycks et al. (2021); Chen (2021). In these domains, the environment provides objective and precise reward signals, such as the correctness of a mathematical result or the pass rate of code, allowing the model to evolve its reasoning strategies through large-scale self-play and trajectory search. However, current RLVR research remains largely confined to these rule-based, vertical domains. Whether the reasoning capabilities acquired through RLVR can seamlessly generalize to General Question Answering (GQA) tasks remains an open question Huan et al. (2025).

5.2 LLMs in General QA

The pursuit of human-level performance on General Question Answering (GQA) remains a central objective in the evolution of Large Language Models (Bai et al., 2022). To align models with human preferences in these non-verifiable domains, recent advancements such as *Rubrics as Rewards* (Gunjal et al., 2026) demonstrate that extending reinforcement learning beyond verifiable domains through reward models is effective for enhancing GQA performance.

Recently, the *Thinking-Answer* paradigm has been integrated into these general-purpose applications, with frontier models like DeepSeek-R1 offering dedicated thinking modes for everyday scenarios. Despite this industrial push, scholarly investigation into the specific mechanics and efficacy of thinking processes for non-verifiable general tasks is still in its infancy. A prominent exception is *Thinking LLMs* (Wu et al., 2025), which introduced Thought Preference Optimization (TPO) to train models to think in general domains without direct human thought supervision. However, while RL has proven effective for GQA alignment, there remains a critical gap in understanding whether the profound thinking abilities elicited by RLVR in reasoning tasks naturally transfer and evolve within the GQA domain.

6 Conclusion

In this work, we have investigated the untapped potential of internal thinking processes in GQA. By introducing a Cross-Generation evaluation framework, we revealed a significant disparity between domains: while thinking processes in reasoning tasks serve as a definitive performance multiplier, their efficacy in GQA is significantly lower. Moreover, we highlights that: unlike the transformative gains observed in reasoning-heavy tasks through RLVR, direct reinforcement learning on GQA tasks often fails to stimulate substantive thinking evolution. Base on the hypothesize that this failure stems from a "loose coupling" between thinking and answering in general tasks, we proposed START. By isolating the thinking phase during the initial stages of training, START forces the model to focus on the cognitive quality of its reasoning manifold. Our experimental results across multiple benchmarks and algorithms consistently demonstrate the effectiveness of START. Ultimately, we hope this exploration encourages further research into how internal thought processes can be more effectively cultivated to serve as a functional engine for broader AI applications.

AI USE STATEMENT

In this work, we used generative AI tools to aid or polish the writing of the manuscript and to assist with research execution (specifically, software code editing). We have not used generative AI tools for retrieval or discovery of related work, for research ideation, to draft sections of the paper, to generate synthetic datasets, or to prove mathematical claims; these tasks are not applicable to this work, and the research ideation, experimental design, core implementation, and analysis were carried out entirely by the authors. We have reviewed all AI-assisted work. Specifically, all LLM-generated code suggestions were manually inspected and verified for correctness by the authors, and all LLM-edited text was cross-checked against the authors' original content to ensure fidelity. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide the following resources to support the reproducibility of this work. The seed model (*Qwen3-1.7B*), reward model (*ArmoRM-Llama3-8B-v0.1*), and the GRPO and DAPO algorithms used in our experiments are described in Section 4.1; the proposed START framework, including the two-phase training paradigm and gradient masking formulation, is detailed in Section 3. The computational environment—64 AMD EPYC 9374F 32-core processors, 8 NVIDIA GeForce RTX 4090 GPUs, and 1 TB of RAM—is reported in Section 4. Detailed descriptions of the four datasets (MATH, AlpacaEval 2.0, ExpertQA, and UltraFeedback), the data splits, the sampling parameters across model families, and the full set of hyperparameters for GRPO and DAPO are documented in Appendices A and B. All reinforcement learning pipelines are implemented using the *verl* library within a containerized Docker environment (`verlai/verl:app-verl0.5-transformers4.55.4-vllm0.10.0-mcore0.13.0-te2.2`); complete training scripts, evaluation code, and configuration files are provided as Supplementary Material, and the full codebase will be open-sourced upon acceptance of this paper.

ETHICS STATEMENT

This work does not involve human subjects, sensitive personal data, or potentially harmful applications. The proposed method focuses on improving reinforcement learning training for general question answering and does not raise any ethics concerns.

REFERENCES

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022.

Mark Chen. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

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

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Bingxiang He, Wei Zhu, Yuan Ni, Guotong Xie, Ruobing Xie, Yankai Lin, et al. Ultrafeedback: Boosting language models with scaled ai feedback. In *ICML*, 2024.

Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization. In *ICML*, 2023.

Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean M. Hendryx. Rubrics as rewards: Reinforcement learning beyond verifiable domains. In *ICLR*, 2026.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. In *NeurIPS*, 2021.

Maggie Huan, Yuetai Li, Tuney Zheng, Xiaoyu Xu, Seungone Kim, Minxin Du, Radha Poovendran, Graham Neubig, and Xiang Yue. Does math reasoning improve general llm capabilities? understanding transferability of llm reasoning. *arXiv preprint arXiv:2507.00432*, 2025.

Seungone Kim, Jay Shin, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Ryan Shin, Sungdong Kim, James Thorne, Minjoon Seo, et al. Prometheus: Inducing fine-grained evaluation capability in language models. In *ICLR*, 2024.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *SOSP*, 2023.

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Alpacaeval: An automatic evaluator of instruction-following models. `https://github.com/tatsu-lab/alpaca_eval`, 5 2023.

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

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

Chris Yuhao Liu, Liang Zeng, Yuzhen Xiao, Jujie He, Jiacai Liu, Chaojie Wang, Rui Yan, Wei Shen, Fuxiang Zhang, Jiacheng Xu, and Yang Liu. Skywork-reward-v2: Scaling preference data curation via human-AI synergy. In *ICLR*, 2026.

Chaitanya Malaviya, Subin Lee, Sihao Chen, Elizabeth Sieber, Mark Yatskar, and Dan Roth. ExpertQA: Expert-curated questions and attributed answers. In *NAACL*, 2024. URL `https://openreview.net/forum?id=hhC3nTgfOv`.

MiniMax Team. Minimax m2.5: Built for real-world productivity. `https://www.minimax.io/news/minimax-m25`, 2026. Accessed: 2026-04-30.

OpenAI. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

Josh OpenAI, Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley Series in Probability and Statistics. Wiley, 1994. ISBN 978-0-47161977-2. doi: 10.1002/9780470316887. URL `https://doi.org/10.1002/9780470316887`.

Rajkumar Ramamurthy, Prithviraj Ammanabrolu, Kianté Brantley, Jack Hessel, Rafet Sifa, Christian Bauckhage, Hannaneh Hajishirzi, and Yejin Choi. Is reinforcement learning (not) for natural language processing: Benchmarks, baselines, and building blocks for natural language policy optimization. In *ICLR*, 2023.

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

Prasann Singhal, Tanya Goyal, Jiacheng Xu, and Greg Durrett. A long way to go: Investigating length correlations in rlhf. In *COLM*, 2024.

Joar Skalse, Nikolaus Howe, Dmitrii Krasheninnikov, and David Krueger. Defining and characterizing reward gaming. In *NeurIPS*, 2022.

Tencent. Hunyuan-1.8b-instruct. `https://github.com/Tencent-Hunyuan/Hunyuan-1.8B`, 8 2025.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel Bowman. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. In *NeurIPS*, 2023.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. Solving math word problems with process-and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022.

Haoxiang Wang, Wei Xiong, Tengyang Xie, Han Zhao, and Tong Zhang. Interpretable preferences via multi-objective reward modeling and mixture-of-experts. In *EMNLP*, 2024.

Hongcheng Wang, Yinuo Huang, Sukai Wang, Guanghui Ren, and Hao Dong. Grpo-ma: Multi-answer generation in grpo for stable and efficient chain-of-thought training. *arXiv preprint arXiv:2509.24494*, 2025.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In *ACL*, 2023a.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *ICLR*, 2023b.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, 2022.

Tianhao Wu, Janice Lan, Weizhe Yuan, Jiantao Jiao, Jason E Weston, and Sainbayar Sukhbaatar. Thinking llms: General instruction following with thought generation. In *ICML*, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

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

A Datasets and Hyperparameters

In this section, we provide a detailed overview of the datasets utilized in our study and the specific configurations used for evaluation and training. We check datasets' licenses properly and our uses are consistent with their intended use. All datasets are well-known and open-source, additional check of personal or offensive content is not necessary. All datasets are in English.

A.1 MATH Dataset

The MATH dataset Hendrycks et al. (2021) is a widely recognized benchmark designed to evaluate the mathematical problem-solving capabilities of large language models across various subjects, including algebra, geometry, and calculus. Problems in this dataset are categorized into five difficulty levels. To provide a more rigorous assessment of the model's thinking depth and avoid performance saturation, we focus specifically on the Level 5 subset, which contains the most challenging problems. We randomly sampled 2,000 problems for the training set and a separate, distinct set of 200 problems for the test set. To elicit structured reasoning traces, we add the following prompt to the original question:

Prompt Template

```
Let's think step by step and output the final answer within \boxed{}.
```

For verifiable reward calculation, we employ the answer extraction and normalization functions provided by the verl library Sheng et al. (2024) to ensure consistent and accurate scoring against ground-truth solutions.

A.2 AlpacaEval 2.0 Benchmark

AlpacaEval 2.0 Li et al. (2023) is an automatic evaluator for general question answering, designed to simulate human preferences across a diverse set of 805 prompts. These prompts span various real-world categories, including creative writing, coding assistance, and general information seeking. As recommended, We report the Length-controlled (LC) Win Rate as our main performance indicator. This metric is specifically designed to mitigate the length bias-a common issue where evaluators favor longer responses regardless of actual quality-by adjusting the win rate based on the relative length of the model's output versus the reference.

To balance computational efficiency with evaluation accuracy, we utilize the *alpaca_eval_vllm_llama3_70b_fn* as the evaluator. This evaluator is highly ranked in terms of human agreement while offering the distinct advantage of local deployment via the vLLM framework Kwon et al. (2023).

During the evaluation of the 805 samples, we observed that a negligible number of instances-typically 2 to 3 samples per run-encountered parsing errors or failed to produce a readable score from the evaluator. Given the extremely low frequency of these occurrences (less than 0.4% of the total set), they do not statistically impact the final calculated win rate.

A.3 ExpertQA Dataset

ExpertQA Malaviya et al. (2024) is a high-quality benchmark designed to evaluate the performance of large language models on complex, domain-specific questions verified by experts. The dataset spans a wide array of academic and professional fields, requiring both factual accuracy and nuanced reasoning.

We utilize the official main set *data/r2_compiled_anon.jsonl* file. This version contains the anonymized, compiled data from the second round of expert reviews, ensuring a high standard of reference quality and expert-level alignment. To maintain a consistent evaluation framework while ensuring the model is exposed to a diverse set of expert queries during training, 90% of the samples were randomly selected to serve as the training corpus for optimizing the thinking traces and responses. The remaining 10% of the data was reserved for testing.

A.4 UltraFeedback Dataset

UltraFeedback Cui et al. (2024) is a large-scale, high-quality preference dataset designed to align language models with human intentions. It provides a diverse range of prompts and multifaceted feedback across dimensions such as instruction-following, truthfulness, and honesty.

To maintain training stability and efficiency, we utilize a curated version of the dataset "trl-lib/ultrafeedback-prompt" from Hugging Face. This version specifically excludes samples with excessively long prompts, ensuring that the context window is focused on the interaction between the thinking trace and the final answer rather than processing outlier input lengths.

We randomly sampled 2,000 instances from the training split to serve as the training corpus and 200 instances from the test split to measure the final performance and the effectiveness of the thinking evolution on unseen prompts.

A.5 Sampling Parameters

In our experiments, we use large language models of three series. For both training and testing phases, we strictly adhere to the sampling parameters recommended as best practices for each model family to ensure optimal performance and representative behavior. Table 7 summarizes the specific configurations used for the thinking and answering generation.

Parameter	Qwen3	R1-Distill	Hunyuan
Temperature	0.6	0.6	0.7
Top-p	0.95	0.95	0.8
Top-k	20	20	20
Min-p	0	0	0
Do Sample	True	True	True
Repetition Penalty	1.0	1.0	1.05

Table 7: Sampling parameters used across different model families.

A.6 Reward Model Configuration

For all experiments involving continuous reward signals in this study, we utilize ArmoRM-Llama3-8B-v0.1 as the primary reward model. ArmoRM Wang et al. (2024) is a state-of-the-art reward model based on the Llama-3-8B architecture, designed to align large language models with human preferences. Unlike traditional reward models that provide a single scalar output, ArmoRM is a multi-objective reward model trained on a diverse set of preference datasets. It is capable of evaluating multiple dimensions of response quality, such as helpfulness, truthfulness, and safety. In our implementation, we use the overall score provided by the model. This score represents a holistic assessment of the response quality by aggregating the various attribute-specific rewards into a single scalar value.

B Reproduction Details and RL Environments

To ensure the reproducibility of our findings, we provide the following details regarding the reinforcement learning framework and training environment.

All reinforcement learning training and evaluation pipelines are implemented using the verl library Sheng et al. (2024). We have included the complete training scripts, evaluation code, and configuration files in the Supplementary Material. The full codebase will be open-sourced upon the acceptance of this paper to facilitate further research into thinking evolution.

B.1 env

To maintain consistency and avoid version-related performance variations, all experiments were conducted within a standardized containerized environment. We utilize the following Docker im-

age provided by the verl team: *verlai/verl:app-verl0.5-transformers4.55.4-vllm0.10.0-mcore0.13.0-te2.2*. This environment includes optimized versions of the Transformers library, vLLM for high-throughput inference, and fsdp for scalable training.

B.2 GRPO

The specific hyperparameter configurations for our GRPO experiments are detailed in Table 8. Almost all hyperparameters are set as official recommendation. Hyperparameter tuning is not necessary.

Parameter	**Value**
Training Setup	
Save frequency	per epoch
Test/eval frequency	per epoch
Training batch size	64
Max prompt length	256 tokens
Max response length	3,328 tokens
Model Configuration	
Model dtype	`bfloat16`
Gradient checkpointing	`True`
Remove padding	`True`
Max think tokens	2,047
Max answer tokens	1,280
Optimization	
Learning rate	1×10^{-5}
PPO mini-batch size	16
PPO micro-batch / GPU	2
KL Regularization	
KL loss enabled	`True`
KL coefficient	0.001
KL type	`low_var_kl`
KL in reward	`False`
Entropy & Exploration	
Entropy coefficient	0.0
Rollout samples / prompt (n)	6
Validation sampling	`True`
Rollout Engine & Reward	
Log prob micro-batch size	16
Reward estimator	GRPO

Table 8: GRPO training hyperparameters and configuration details.

B.3 DAPO

The implementation of DAPO is based on the standard recipes provided within the verl library, and the specific hyperparameter configurations are detailed in Table 9. Almost all hyperparameters are set as official recommendation. Hyperparameter tuning is not necessary.

B.4 GRPO-MA

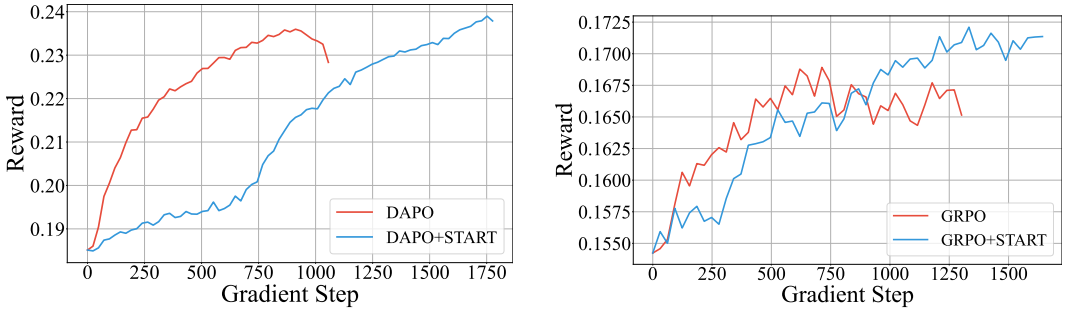
For GRPO-MA, We modified the sampling engine within the verl library to support a two-tier generation strategy, generate multiple independent answers for each thinking trace and calculate advantage independently.

Parameter	Value
Training Setup	
Save frequency	per epoch
Test/eval frequency	per epoch
Training batch size	64
Generation batch size	64
Max prompt length	256 tokens
Max response length	3,328 tokens
Model Configuration	
Model dtype	`bfloat16`
Gradient checkpointing	`True`
Remove padding	`True`
Max think tokens	2,047
Max answer tokens	1,280
Optimization	
Learning rate	1×10^{-5}
LR warmup steps	10
Weight decay	0.1
Gradient clipping norm	1.0
Loss aggregation mode	`token-mean`
PPO mini-batch size	16
PPO micro-batch / GPU	2
Dynamic batch size	`True`
KL Regularization	
KL loss enabled	`False`
KL in reward	`False`
DAPO-Specific Policy Clipping	
Clip ratio lower bound	0.20
Clip ratio upper bound	0.28
Clip ratio curvature (c)	10.0
Entropy & Exploration	
Entropy coefficient	0.0
Rollout samples / prompt (n)	6
Validation sampling	`True`
Rollout Engine (vLLM)	
Log prob micro-batch size	16
Dynamic batch size (rollout)	`True`
Group Filtering (DAPO)	
Enable filter groups	`False`
Max generation batches	10
Filtering metric	`acc`
Overlong Buffer Reward	
Enable overlong buffer	`True`
Buffer length	2,560 tokens
Penalty factor	1.0

Table 9: DAPO training hyperparameters and configuration details.

To maintain a fair comparison with standard GRPO in terms of total sampled tokens, we only adjusted the group composition. While the base RL hyperparameters (learning rate, KL coefficient, etc.) remain identical to those in Table 8, the sampling structure is: 2 thinking traces per prompt, 3 answers per thinking trace, $2 * 3 = 6$ total samples per prompt.

C Additional Experimental Results



(a) Reward curve using DAPO

(b) Reward curve on UltraFeedback

Figure 4: **Additional reward curves across different datasets and algorithms.** For baseline model, we terminate the training process when an obvious convergence trend is observed.

C.1 Generalization to Alternative Reward Models

Configuration	Reward	Δ R	Win Rate
Base	17.81	—	—
$T_{\text{post}} + A_{\text{pre}}$	18.35	+0.54	50.77%
$T_{\text{pre}} + A_{\text{post}}$	23.48	+5.67	93.63%
Post-trained	24.10	+6.29	94.27%
START(Ours)	**25.19**	**+7.38**	**65.75%**

Table 10: **Robustness check with an alternative reward model.** To verify the consistency of the thinking-answering decoupling effect, we replicate the experiments on ExpertQA using a different reward model. The results confirm that START consistently improves the thinking process where standard RL fails. Win Rate is calculated against the *Post-trained* model; others are against *Base*.

To evaluate the robustness of our findings across different feedback signals, we replicate the experiments on ExpertQA using an alternative reward model, Skywork-Reward-V2-Qwen3-8B Liu et al. (2026). As shown in Table 10, the thinking stagnation phenomenon persists even with this different reward signal: while direct RL yields a substantial reward improvement in the answering phase ($T_{\text{pre}} + A_{\text{post}}$ achieving $\Delta R = +5.67$), the gain from the evolved thinking process alone remains nearly negligible ($T_{\text{post}} + A_{\text{pre}}$ achieving only $\Delta R = +0.54$). This consistency suggests that thinking stagnation is an inherent challenge of reinforcement learning in general domains, rather than an artifact of a specific reward model. Crucially, START effectively mitigates this issue by decoupling the training phases, achieving a superior reward of 25.19 (+7.38) and a significant win rate of 65.75% over the standard post-trained baseline. These results confirm that START consistently fosters more substantive thinking evolution across varying evaluation frameworks.

C.2 Results on Reasoning Benchmarks

We have conducted experiments on the MATH benchmark and observed that START provides only a marginal improvement compared to vanilla GRPO. As in Table 11, this result further validates our central thesis: in reasoning tasks with verifiable rewards, direct reinforcement learning is already highly effective at stimulating thinking evolution while GQA tasks suffers from thinking stagnation.

Configuration	Accuracy	Δ Acc
Base	55.00%	—
$T_{\text{post}} + A_{\text{pre}}$	67.05%	**+12.05**
$T_{\text{pre}} + A_{\text{post}}$	63.00%	+8.00
Post-trained	67.70%	+12.70

Table 11: **Result of START on MATH.**

C.3 Investigation into the Encoding Hypothesis

A potential concern regarding the efficacy of START is whether the model achieves higher rewards by merely encoding or pre-computing the final response within the thinking traces, rather than evolving genuine cognitive capabilities. If the thinking process primarily consists of verbatim fragments or a near-complete draft of the final answer, the subsequent answering phase would function as a redundant transcriber rather than a logical synthesizer. To empirically test this hypothesis, we utilize a teacher model(minimax-m2.5 MiniMax Team (2026)) to assign a Thinking-Encoding Score. This metric specifically evaluates the density of direct answer-relevant information-such as premature conclusions or linguistic mirroring of the final response-within the thinking trace. A higher score suggests that the model is utilizing the thinking process as a simple buffer to bypass the logical complexity of the task.

As summarized in Table 12, our analysis reveals that the Thinking-Encoding Score actually decreased following the START training process, dropping from 6.8790 (Base) to 6.3376 (after START). This reduction provides strong evidence that START does not incentivize the model to encode final-answer tokens prematurely in the thinking process. Instead, by treating the answer generation as a fixed component of the environment during Phase I , START forces the model to utilize its internal workspace for higher-level cognitive management. These include strategic behaviors such as user intent speculation and structural output planning, which serve as a more robust and informative foundation for the answering phase rather than a simple repetition of results.

Configuration	Thinking-Encoding Score (↓)
Base (Qwen3-1.7B)	6.8790
START(Ours)	**6.3376**

Table 12: **Evaluation of answer-redundancy in thinking traces.** The Thinking-Encoding Score quantifies the extent to which the thinking process merely replicates the final answer. A lower score indicates a more abstract and strategic reasoning process.

D Exploring the Emergence of Meta-Context Modeling

To understand how thinking traces evolve under START, we conduct a subsequent quantitative analysis. In general-purpose tasks, where logical rigor is often implicit, we explore whether the utility of thinking manifests through the discovery of Meta-Contexts-potential hidden variables such as user persona, intent, and structural preferences that, while not explicitly stated, may be essential for high-quality responses Wang et al. (2023a).

Specifically, the START-trained model's thinking traces appear to function as a strategic workspace where these hidden dimensions are more explicitly modeled:

- **User Needs Identification:** The model often identifies a candidate audience (e.g., a student or someone interested in law), seemingly calibrating its internal technicality accordingly.
- **Structural Output Planning:** We observe instances where the model pre-defines a structural schema (e.g., listing each right with support from legal frameworks), potentially guiding the final response toward a more deliberate organizational logic.

As shown in Table 13, there is a notable shift in the occurrence of these patterns. While vanilla GRPO identifies user needs in 54.78% of cases, this frequency rises to 90.45% in the START-trained model. Similarly, proactive structural planning is observed in 92.36% of the traces.

Pattern	GRPO	GRPO-MA	+START
User Needs	54.78%	61.78%	90.45%
Structural Output	61.15%	75.99%	92.36%

Table 13: **Quantitative comparison of thinking patterns.** We report the frequency of meta-context dimensions (*User Needs* and *Structural Output*) identified within the thinking traces across the test set.

While these patterns alone may not fully account for the final performance gains, their systematic emergence suggests that START encourages the model to explore the observed modeling of meta-contexts. This emergence provides a plausible window into how independent thinking optimization might reshape the model's approach to general-domain instructions, moving beyond simple retrieval toward a more context-aware internal process. Fig 5 is an example case.

Instruction: What are the fundamental rights of the accused?

Base Model (Qwen3-1.7B)

...I need to figure out... I think I have a list, but I need to organize them properly...

Vanilla GRPO

...First, the accused has the right to a fair trial... I need to make sure the answer is comprehensive but not too technical... so clarity is important. Let me organize these points in a logical order.

GRPO+START

...start by recalling key principles... **[Structural Planning] I should structure the answer clearly, listing each right with support from legal frameworks...** I need to make sure the language is accessible and not too technical, **[User Needs] so the user, whether they're a student or someone interested in law, can understand the points without getting lost in jargon**. Finally, a concluding statement that reinforces the value of these rights...

Improvements: Demonstrates meta-contexts like audience needs and structural design.

Figure 5: Evolution of thinking patterns.

E Analysis of Thinking-Answer Coupling across Domains

To explore the coupling disparities between different task categories, we conduct exploratory experiments on the MATH and ExpertQA datasets, representing reasoning-intensive and general question answering domains, respectively. The objective of these experiments is to preliminary explore the coupling characteristics between thinking traces and final responses across different task types.

E.1 Sampling Method

To formalize the observation of coupling disparities, we define a nested sampling and scoring framework. For a given instruction x, we first sample a set of m distinct thinking traces $\mathcal{T} = \{T_j\}_{j=1}^{m}$.

Subsequently, for each fixed thinking trace T_j, we sample n independent response sequences $\{A_{j,k}\}_{k=1}^{n}$. For each complete generation path $s_{j,k} = [T_j, A_{j,k}]$, we obtain a quality score $R_{j,k}$. Specifically, for the MATH dataset, $R_{j,k}$ is a binary verifiable reward based on the correctness of the final answer; for ExpertQA, $R_{j,k}$ is a continuous scalar provided by a reward model.

Follow GRPO-MA, we define the Thinking Score as the expected score of all possible answers derived from a specific thought. This serves as a measure of the inherent value of a thinking trace, independent of the subsequent answering stochasticity. Based on the nested sampling framework, for each thinking trace T_j, the thinking score S_j is defined as:

$$S_j = \mathbb{E}_{A\sim\pi(\cdot|x,T_j)}[r(x, T_j, A)] \approx \frac{1}{n}\sum_{k=1}^{n} R_{j,k} \tag{1}$$

This score represents the potential of the thinking trace T_j to steer the model toward a correct or high-quality response. In our experiments, we set $m = 8$ and $n = 8$, resulting in 64 complete trajectories for each instruction x to ensure a statistically robust estimation.

E.2 Strong Coupling in Reasoning Tasks

To provide a concrete measure of coupling in the reasoning domain, we first analyze the outcome consistency for each fixed thinking trace on the MATH dataset. Since MATH provides verifiable binary rewards (correct or incorrect), we define the **Minority Outcome Ratio** γ to quantify the stochasticity of the answering phase relative to the thinking process.

For a given instruction x and a specific thinking trace T_j, let $n_{j,1}$ and $n_{j,0}$ denote the number of correct and incorrect answers among the n samples, respectively. The ratio γ is calculated as:

$$\gamma = \mathbb{E}_{x,T_j}\left[\frac{\min(n_{j,1}, n_{j,0})}{n}\right] \tag{2}$$

where the expectation is taken over all sampled thinking traces across the dataset. Our empirical analysis reveals that γ is merely **0.0469**.

This strikingly low value aligns with the intuitive nature of mathematical reasoning: the correctness of a solution is almost entirely determined by the logical integrity of the preceding thinking process. In this "strong coupling" scenario, the answering phase acts as a nearly deterministic transducer; once a correct logical path is established in T, the probability of generating an incorrect final answer A is negligible. Conversely, a logical error in T typically precludes a correct A. Thus, the minority outcome ratio of 0.0469 is a direct reflection of the strong coupling in reasoning tasks, where the quality of the thinking trace leaves little room for stochastic fluctuation in the final answer.

E.3 Loose Coupling in General QA

Unlike reasoning tasks with binary outcomes, general-purpose tasks like ExpertQA involve continuous quality scores. To analyze the coupling in this domain, we utilize the metrics of **Answering Fluctuation** (σ_{answer}) and **Thinking Fluctuation** (σ_{thinking}) as defined below. For each instruction x, based on the $m \times n$ sampled trajectories and their corresponding reward scores $R_{j,k}$, we compute:

- **Answering Fluctuation (σ_{answer})**: This represents the average degree of quality variation that occurs during the response generation phase for a fixed thought.:

$$\sigma_{\text{answer}} = \text{mean}_j\left(\text{std}_k(R_{j,k})\right) \tag{3}$$

  where $S_j = \frac{1}{n}\sum_{k=1}^{n} R_{j,k}$ is the thinking score for the j-th trace.

- **Thinking Fluctuation (σ_{thinking})**: This represents the variation in potential quality across different thinking paths. It measures how much the "direction of thought" actually influences the expected reward:

$$\sigma_{\text{thinking}} = \text{std}_j(S_j) \tag{4}$$

  where $S_j = \frac{1}{n}\sum_{k=1}^{n} R_{j,k}$ is the thinking score defined above.

We then define the **Coupling Ratio** $\rho = \sigma_{\text{thinking}}/\sigma_{\text{answer}}$. A ratio $\rho < 1.0$ implies that the quality signal from thinking is frequently overshadowed by the fluctuations in the answering phase, indicating a state of "loose coupling."

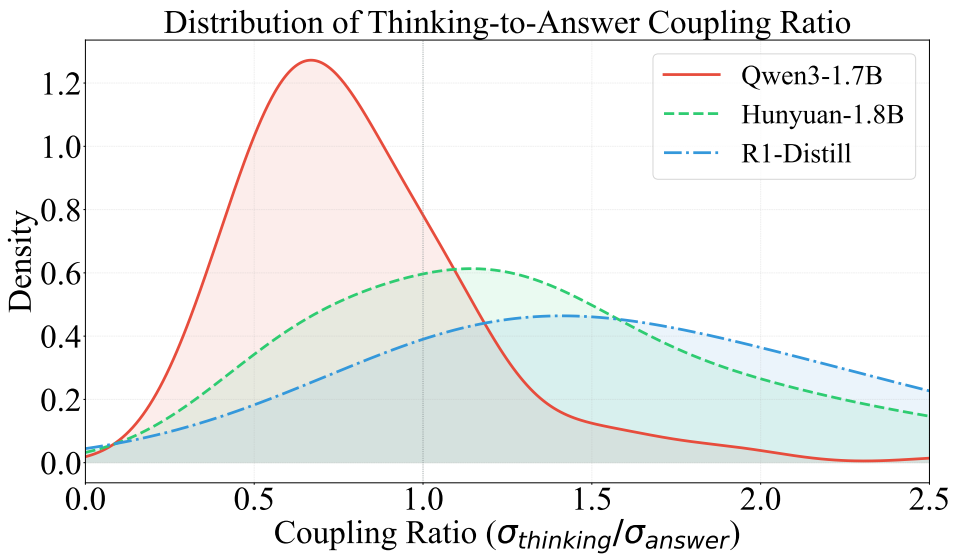


Figure 6: **Distribution of thinking-to-answer Coupling Ratio on ExpertQA.** The red curve (Qwen3-1.7B) exhibits a sharp peak at $Ratio \approx 0.65$.

Based on these metrics, we visualize the distribution of the Coupling Ratio ρ across the ExpertQA dataset in Figure 6. These empirical results provide a preliminary exploration of the loose coupling inherent in general tasks, revealing a starkly different dynamic compared to the consistency observed in MATH.

In this general-purpose domain, the quality fluctuation during the answering phase is remarkably high-frequently comparable to or even exceeding the fluctuation attributed to the thinking process. For the majority of samples across the evaluated models, the density peaks remain predominantly below or near the critical $\rho = 1.0$ line. This indicates that, unlike the deterministic nature of reasoning tasks, the thinking trace in GQA does not consistently dictate the final result. Instead, the final output quality is heavily influenced by stylistic, structural, or linguistic variances in the answering phase, even when the underlying reasoning remains the same.

While models such as R1-Distill show a relative right-shift compared to smaller models like Qwen3-1.7B, it is important to note that none of the models achieve a state of universal strong coupling in the general domain. Even for stronger models, a significant portion of the distribution resides in the loose coupling zone ($\rho < 1.0$). This demonstrates that the presence of loose coupling is a pervasive characteristic of GQA tasks across models of varying scales.

F PROMPTS

Prompt for Answer-Redundancy Detection

```
You are evaluating whether a model's answer is merely a restatement of its
thinking process.
## Question
{question}
## Thinking
{thinking}
## Answer
{answer}
## Task
Score from 0 to 10 how much the Answer is merely a restatement of the
Thinking.
- 0: The answer contains substantial new information, synthesis, or structure
not present in the thinking.
- 5: The answer partially restates the thinking but also adds meaningful
content.
- 10: The answer is almost entirely a verbatim or near-verbatim restatement
of the thinking, adding nothing beyond what was already said.
Respond with a single integer between 0 and 10. No explanation.
```

Prompt for User Intent and Identity Speculation Detection

```
Please evaluate whether the provided 'thinking' content from a large
language model contains speculation about the user's identity, role,
persona, or hidden intent and psychology.
Definitions: *User Identity or Intent Speculation involves:
The model explicitly attempts to infer who the user is, their professional
background, their level of expertise, or their specific role (e.g., "The
user seems to be a student," "The user is likely a developer," "Assuming
the user is a non-expert").
The model analyzes the prompt to guess the user's hidden intent, underlying
motivation, or psychological state (e.g., "The user might be trying to test
my safety filters," "The user seems frustrated," or "The underlying goal
is likely to bypass a restriction").
To do so, I will give you the instructions (prompts) given to the model,
and the thinking process of the model.
All inputs should be Python dictionaries.
Here is the prompt: { "instruction": "{{instruction}}", }
Here is the thinking process of the model: { "thinking": "{{context}}" }
Requirements for your reply:
Output ONLY the character '1' (if user identity, intent, or psychological
speculation is present) or '0' (if no such speculation is detected).
DO NOT include any other text, explanation, punctuation, or code blocks.
The output must be a single integer.
```

Prompt for Formatting and Stylistic Planning Detection

```
Please evaluate whether the provided 'thinking' content from a large language model contains any mention of how to structure, style, or present the final answer.
Definitions: Format or Tone Specification (Output 1 if any of these exist):
Structural/Planning Intent: Any mention of how the response should be organized or what it should contain (e.g., "I should cover...", "include factors", "mention X and Y", "list the steps", "start by...", "give examples").
Tone/Style/Vibe: Any mention of the desired feel or level of detail (e.g., "be comprehensive", "be confident", "keep it simple", "make it professional", "avoid being too technical", "be concise").
Target Goal: Any statement starting with "The answer should be..." or "I need to make sure the response is..." followed by an adjective (e.g., "comprehensive", "clear", "helpful").
Input Data:
Prompt: {{instruction}}
Thinking Process: {{context}}
Requirements for your reply:
Output ONLY the character '1' (if ANY mention is found) or '0' (if none).
DO NOT include any other text.
```