Thinking with Nothinking Calibration: A New In-Context Learning Paradigm in Reasoning Large Language Models

Anonymous ACL submission

Abstract

Reasoning large language models (RLLMs) have recently demonstrated remarkable capabilities through structured and multi-step reasoning. While prior research has primarily focused on improving their training and inference strategies, their potential for in-context learning (ICL) remains largely underexplored. To fill this gap, we propose *Thinking with Nothinking Calibration (JointThinking)*, a new ICL paradigm that prompts the model to generate two answers in parallel: one in *Thinking* mode and the other in *Nothinking* mode. A second round of *Thinking* is triggered only when the two initial responses are inconsistent, using a single prompt with two different answers. Extensive experiments across multiple reasoning benchmarks demonstrate that *JointThinking* significantly outperforms few-shot chain-of-thought (CoT), thinking twice and majority voting. Moreover, it achieves comparable in-distribution performance to training-based SOTA reasoning method, while substantially outperforming on out-of-distribution tasks. We further conduct a systematic analysis of the calibration mechanism, showing the importance of structural thinking diversity and the benefits of consistency check. Additionally, we observe that the performance gap between actual and ideal reasoning narrows as model size increases in the second thinking, indicating the strong scalability of our approach. Finally, we discuss current limitations and outline promising directions for future ICL research in RLLMs.

1 Introduction

Reasoning large language models (RLLMs), such as OpenAI-o3 (OpenAI, 2024), DeepSeek-R1 (Guo et al., 2025), and Qwen3 (Yang et al., 2025), mark a fundamental shift in model output paradigms by explicitly structuring the inference process. These models significantly improve performance on complex reasoning tasks by increasing inference-time compute (Snell et al., 2024). The inference process of RLLMs typically consists of two stages: a "Thinking" phase where the model generates reasoning chains through iterative exploration, reflection, and verification; and a "Final Solution" phase presenting the refined reasoning and answer. Such structured reasoning behavior is often acquired through reinforcement learning (RL), as noted by (Guo et al., 2025). Despite their strong reasoning capabilities, recent studies have identified a phenomenon known as "Overthinking" (Cuadron et al., 2025; Chen et al., 2024), where RLLMs continue reasoning beyond correct solutions due to uncertainty, adding unnecessary complexity and errors, particularly on simpler problems where traditional large language models (LLMs) may perform better.

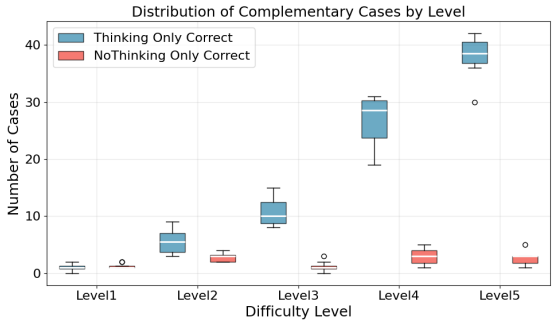

Figure 1: Comparison of R1-7B using *Thinking* and *Nothinking* on MATH500. Each mode reliably compensates for the other's failures across all difficulty levels.

In the broader LLM landscape, in-context learning (ICL) remains a highly active research area due to its flexibility and effectiveness. While prior work on RLLMs has primarily focused on improving training and inference strategies, relatively little attention has been paid to their ICL capabilities. A recent study by (Ge et al., 2025) investigates the effects of Zero-CoT and Few-shot CoT on RLLMs, suggesting that few-shot CoT improves reasoning performance. However, our empirical results reveal that the improvement is limited. We hypothesize

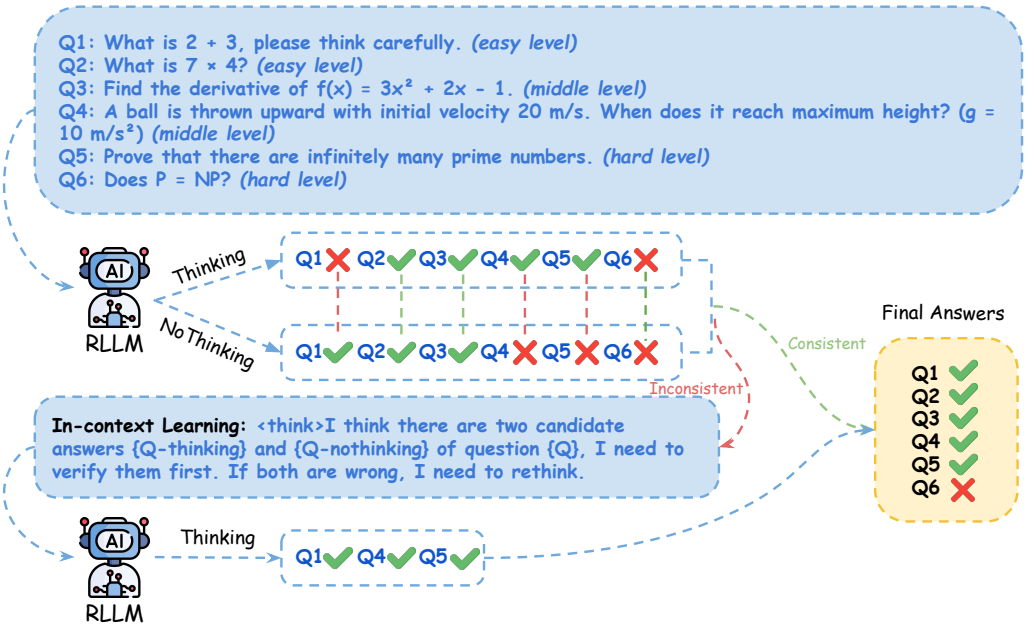

Figure 2: *Thinking with Nothinking Calibration (JointThinking)*. Given one question, the reasoning large language model generates two outputs in parallel with *Thinking* and *Nothinking* modes. If answers are inconsistent, the model will perform a second round of reasoning in the *Thinking* mode, using the two inconsistent answers as context (the "second thinking").

that RL enhances the model's intrinsic ability to discover optimal reasoning paths, thereby reducing reliance on explicit prompt-based guidance. Indeed, recent work (Li et al., 2025b) has also shown that explicit CoT reasoning can sometimes degrade instruction-following accuracy by omitting the constraints of instruction in the prompt.

Given these observations, we aim to design an ICL paradigm tailored to the strengths of RLLMs, i.e., their flexible and adaptive reasoning capabilities, while minimizing reliance on prompt design. The *Thinking* mode performs well on complex problems but is prone to overthinking and making mistakes on simpler ones. In contrast, the *Nothinking* mode (Ma et al., 2025a), while less effective on challenging questions, achieves higher accuracy on easier ones (Zhang et al., 2025a; Lou et al., 2025). Through an initial analysis of the MATH500 dataset in Figure 1, we identify a complementary pattern that each mode reliably compensates for the other's failures across all difficulty levels.

Building on these insights, we propose a novel ICL framework for RLLMs called *Thinking with Nothinking Calibration* (or *JointThinking*). Given a question, the model generates two responses in parallel: one using the *Thinking* mode and the other using the *Nothinking* mode. If both responses are consistent, the answer from the *Thinking* mode is adopted. Otherwise, both answers are fed into the prompt for a second round of *Thinking* to resolve the disagreement (see Figure 2). Extensive experiments show that our method significantly outperforms single-mode reasoning, few-shot CoT, thinking twice and majority voting baselines across various models and mathematical benchmarks. Moreover, it achieves performance comparable to training-based state-of-the-art (SOTA) reasoning enhancement method under in-distribution tasks, while demonstrating remarkable superiority in out-of-distribution scenarios.

We further validate the necessity of incorporating different reasoning modes and the consistency check mechanism. Mixed-mode calibration consistently yields lower error rates than using a single mode, while consistency check to selectively trigger second thinking not only reduces computational overhead but can also achieve better performance than always performing second thinking. Additionally, we find a clear scaling trend in the second-thinking stage: as model size increases, the gap between the generated answers and the ideal solutions progressively narrows, with our method in some cases even surpassing the ideal calibration performance. Finally, we explore current limitations of ICL in RLLMs and highlight promising future research directions. In summary, our main

contributions are:

- We introduce *Thinking with Nothinking Calibration* (or *JointThinking*), a novel ICL paradigm for RLLMs that leverages the complementary strengths of *Thinking* and *Nothinking* modes.
- With extensive experiments, we demonstrate that *JointThinking* significantly outperforms existing baseline approaches and exhibits strong generalization to out-of-distribution scenarios.
- We systematically analyze the benefits of mixed-mode calibration and the consistency check mechanism. We further investigate how model scaling influences second-thinking performance.

2 Related Work

In-Context Learning In-context learning (ICL) was first proposed by (Brown et al., 2020), which refers to the capability of large language models (LLMs) to perform tasks solely by being provided with examples or instructions within the input prompt, without any updates to their internal parameters. The ICL capabilities of LLMs have been shown to improve further through self-supervised or supervised training (Wei et al., 2023). Since then, an increasing number of studies have focused on this simple yet effective paradigm. Some works have studied how to design good demonstrations (Tang and Dong, 2024; Peng et al., 2024; Wang et al., 2025a) and how to build effective ICL-based applications (Meade et al., 2023; Buoso et al., 2024). Other works have focused on theoretical perspectives to analyze why ICL works for LLMs. (Wies et al., 2023) establishes a PAC-based theoretical framework for ICL, showing that in-context learning primarily involves task identification rather than task learning. (Zhang et al., 2025b) shows that each prompt defines a unique trajectory through the answer space, and the choice of trajectory is crucial for task performance and future navigation within the space during CoT reasoning. Models of different scales exhibit distinct ICL behaviors, with smaller models being more robust to noise while larger models tend to over-attend to irrelevant information. (Shi et al., 2024).

Reasoning Large Language Models The emergence of test-time scaling RL methods, such as GRPO (Shao et al., 2024), significantly enhances the reasoning capabilities of large language models for complex problems, especially mathematical problems. These trained models are referred to as reasoning large language models (RLLMs) (Guo et al., 2025; Yang et al., 2025), which first generate an extensive chain of thought (CoT), often involving techniques such as backtracking and verification, before arriving at a final explanation and answer. For the training stage, many studies explore how to effectively train models with suitable-length thinking processes, including reinforcement learning with length reward design (Zhang et al., 2025a; Aggarwal and Welleck, 2025; Lou et al., 2025) and supervised fine-tuning with variable-length CoT data (Yu et al., 2024; Ma et al., 2025b; Yu et al., 2025). Works exploring the inference stage mainly focus on sampling multiple outputs and aggregating them in parallel, including Best-of-N related methods (Sun et al., 2024; Wang et al., 2025b; Ma et al., 2025a) and search-guided related methods (Li et al., 2025a; Liao et al., 2025). Additionally, (Ge et al., 2025) conducts a comprehensive analysis examining the impact of few-shot CoT on RLLMs from the ICL perspective, demonstrating its positive influence on reasoning performance.

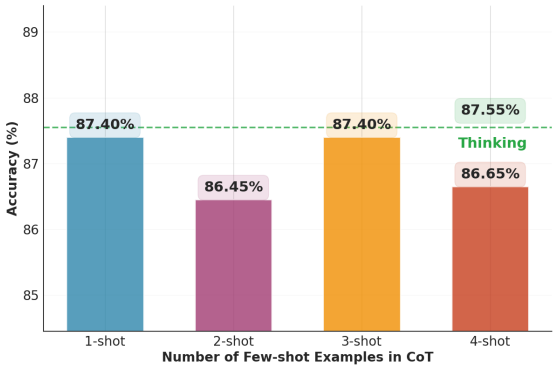

Figure 3: Effect of varying the number of few-shot CoT examples on MATH500 accuracy using R1-7B. Results are averaged over four runs. Few-shot CoT yields no improvement over direct thinking, and the performance is unrelated to the number of few-shot examples.

3 Method

3.1 Is Few-Shot CoT Still Effective?

Few-shot CoT prompting has demonstrated strong effectiveness for traditional LLMs, consistently improving performance across a variety of tasks (Wei et al., 2022). However, does this approach remain effective for RLLMs? To explore this, we conduct a preliminary analysis. As shown in Figure 3, few-shot CoT offers no improvement over direct *Thinking* (represented by the green horizontal

Model	Method	Datasets (Pass@1)				Average
		GSM8K(1319)	MATH500(500)	AIME24(30)	AMC23(40)	
R1-1.5B	Thinking	81.80	78.20	23.33	66.25	62.40
	Nothinking	77.94	79.00	19.17	68.75	61.22
	Few-shot	81.73	71.20	22.08	54.69	57.43
	Majority Voting	83.93	81.60	26.67	70.62	65.71
	Thinking Twice	82.87	78.00	30.83	70.31	65.50
	JointThinking (**Ours**)	**84.23**	**82.80**	**35.00**	**72.41**	**68.61**
R1-7B	Thinking	87.79	87.60	47.50	86.88	77.44
	Nothinking	85.90	74.60	22.92	59.38	60.70
	Few-shot	84.91	87.00	35.83	73.13	70.22
	Majority Voting	89.99	**88.00**	50.00	87.50	78.87
	Thinking Twice	90.07	87.00	50.42	86.88	78.59
	JointThinking (**Ours**)	**91.05**	87.80	**51.25**	**88.44**	**79.64**
R1-14B	Thinking	89.99	88.20	63.75	90.00	82.99
	Nothinking	90.14	74.00	30.42	72.81	66.84
	Few-shot	89.08	88.40	58.33	89.69	81.38
	Majority Voting	91.66	89.00	65.00	90.31	83.99
	Thinking Twice	93.71	89.20	60.00	92.19	83.78
	JointThinking (**Ours**)	**93.93**	**89.80**	**65.42**	**93.75**	**85.73**
R1-32B	Thinking	92.80	86.20	65.42	92.50	84.23
	Nothinking	92.72	78.20	39.58	68.97	69.87
	Few-shot	90.98	88.20	53.75	90.62	80.89
	Majority Voting	93.93	88.60	66.67	93.13	85.58
	Thinking Twice	95.07	88.80	69.58	93.75	86.80
	JointThinking (**Ours**)	**96.29**	**89.80**	**70.83**	**94.69**	**87.90**
Qwen3-8B	Thinking	94.69	87.80	59.17	86.56	82.06
	Nothinking	92.19	80.80	29.58	73.36	68.98
	Few-shot	94.47	89.80	57.50	88.12	82.47
	Majority Voting	94.92	88.60	65.00	90.00	84.63
	Thinking Twice	94.62	89.60	64.58	88.75	84.39
	JointThinking (**Ours**)	**95.00**	**91.20**	**71.25**	**92.81**	**87.57**
Qwen3-14B	Thinking	94.77	89.80	62.92	92.50	85.00
	Nothinking	92.49	84.00	27.50	74.76	69.69
	Few-shot	95.30	90.00	53.75	92.50	82.89
	Majority Voting	95.22	90.20	65.00	92.50	85.73
	Thinking Twice	94.77	91.80	70.00	96.25	88.21
	JointThinking (**Ours**)	**95.45**	**92.59**	**71.67**	**96.88**	**89.15**

Table 1: Accuracy (%) of *JointThinking* and no-training baselines across different models and benchmarks, and for AIME24 and AMC23 the results are averaged by 8 individual runs. The best and second results are **bolded** and underlined, respectively. Prompt template of different methods can be found in Appendix A.

dashed line), and its performance remains largely unaffected by the number of examples provided. This observation aligns with expectations, as current RL training pipelines for RLLMs are primarily designed to enhance internal reasoning capabilities, thereby reducing reliance on explicit instructional prompts such as few-shot demonstrations.

3.2 Thinking with Nothinking Calibration

Inspired by the complementary behavior in Figure 1, where regardless of problem difficulty, there consistently exist cases where the model fails under the *Thinking* mode but succeeds under the *Nothinking* mode, and vice versa, we propose a novel ICL framework to leverage this structured discrepancy, as shown in Figure 2.

3.2.1 Thinking Mode and Nothinking Mode

Most RLLMs today follow a common generation paradigm, where the model first produces a reasoning chain enclosed by special tokens `<|beginning of thinking|>` and `<|end of thinking|>`, followed by the final solution steps and answer.

Let M denote the reasoning model, with <BOT> and <EOT> denoting the special tokens for the beginning and end of the thinking process, respectively. Given an input question q, we define two different generation modes:

(i) *Thinking* mode: The model generates a full reasoning trajectory beginning with <BOT>:

$$r_{\text{thinking}} = M(q + \langle\text{BOT}\rangle) \quad (1)$$

(ii) *NoThinking* mode: The model is prompted to skip the thinking process and directly generate the final output:

$$r_{\text{nothinking}} = M(q + \langle\text{BOT}\rangle + s_{\text{skip}} + \langle\text{EOT}\rangle) \quad (2)$$

where s_{skip} is set to "Okay, I think I have finished thinking." following (Ma et al., 2025a). This

Method	In-distribution Datasets			Out-of-distribution Datasets	
	GSM8K	MATH500	AIME24	MMLU-Pro	GPQA
R1-1.5B(Base Model)					
AdapThink(δ=0.05)	83.1	82.0	31.0	35.77	32.89
JointThinking	**84.2**	**82.8**	**35.0**	**45.21**	**39.78**
R1-7B(Base Model)					
AdapThink(δ=0.05)	91.0	**92.0**	**55.6**	57.07	51.23
JointThinking	**91.1**	87.8	51.3	**66.79**	**57.49**

Table 2: Comparison between the SOTA training-based reasoning enhancement method (*AdapThink*) and our proposed ICL-based approach (*JointThinking*) on both in-distribution (ID) and out-of-distribution (OOD) datasets. Two methods achieve comparable performance on ID datasets, while *JointThinking* significantly outperforms *AdapThink* on OOD datasets, demonstrating superior generalization ability. The better results are **bolded**.

prompt design effectively forces the model to skip the thinking stage and proceed directly to the final solution. Full prompt templates are provided in Appendix A.

3.2.2 Second Thinking

We introduce a consistency check mechanism to determine when a second round of thinking is needed. Let $A(r)$ be a function that extracts the final answer from a response r. The consistency check $\mathrm{Con}(q)$ is defined as:

$$\mathrm{Con}(q) = \begin{cases} 1, & \text{if } A(r_\mathrm{t}) = A(r_\mathrm{n}) \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

where $r_t = r_{\text{thinking}}$ and $r_n = r_{\text{nothinking}}$. Based on this check, the final answer $\mathrm{R}(q)$ is selected as:

$$\mathrm{R}(q) = \begin{cases} A(r_t), & \text{if } \mathrm{Con}(q) = 1 \\ \mathrm{ST}(q, A(r_t), A(r_n)), & \text{if } \mathrm{Con}(q) = 0 \end{cases} \tag{4}$$

and ST denotes the second thinking processing. If the two answers are consistent, the response from the *Thinking* mode is directly adopted as the final output. Otherwise, both answers are incorporated into a second-thinking prompt to elicit a refined answer. A brief overview of the prompt is provided below, with full details available in Appendix A.

$$r_{\mathrm{ST}} = M(q + \langle \mathrm{BOT} \rangle + s_{\mathrm{ST}}) \tag{5}$$

where r_{ST} is the final answer from the second thinking process and s_{ST} represents "I think there are two candidate answers r_{thinking} and $r_{\text{nothinking}}$ for this question, I need to verify them first. If both are wrong, I need to rethink.".

Importantly, we find that increasing prompt complexity does not improve second-thinking performance. And the instruction should be placed immediately after <BOT>, marking the beginning of the reasoning process, rather than before <BOT>, as is typically done in traditional prompt engineering (an empirical analysis is discussed in Section 5.3). For clarity, we include the complete algorithm of the *JointThinking* pipeline in Appendix B.

4 Experiments

4.1 Setup

Models We conduct experiments using six models from the DeepSeek-R1-Distill-Qwen series: 1.5B, 7B, 14B, and 32B variants (Guo et al., 2025), and from Qwen3 series: 8B and 14B variants (Yang et al., 2025). All of them are SOTA open-source models and widely adopted in the community. All models are deployed on 2 × A800 80G GPUs.

Methods We compare our proposed method, *JointThinking*, with five no-training baselines. The *Thinking* and *Nothinking* modes are defined in Section 3.2.1. For few-shot CoT (Ge et al., 2025) we choose the number of examples to 3 in terms of Figure 3. Since *JointThinking* involves at most three inference passes, majority voting is set to voting@3 for a fair comparison. Thinking twice is a simple two-step thinking process, where the second *Thinking* is always triggered and uses the answer of the first *Thinking* as context. Both few-shot CoT and majority voting are in *Thinking* mode.

Datasets Evaluation is conducted on four mathematical reasoning benchmarks: GSM8K (Cobbe et al., 2021), MATH500 (Lightman et al., 2023), AIME2024 (AI-MO, 2024), and AMC2023 (AI-MO, 2023), covering a wide range of difficulty levels from elementary school word problems to high school competition-level mathematics. All models are configured with a 16K context window to support the generation of long reasoning chains.

Following the official guidance, decoding parameters are set as follows: for the *Thinking* mode, temperature is 0.6 and top-p is 0.95; for the *Nothinking* mode, temperature is 0.7 and top-p is 0.8.

4.2 Main Results

As demonstrated in Table 1, the proposed *JointThinking* method consistently outperforms all no-training methods across all models. It achieves average performance gains ranging from 0.77 to 2.94 over the second-best baseline and from 2.20 to 6.21 compared to the standard single *Thinking* mode. The results also indicate that the *Thinking* mode generally outperforms the *Nothinking* mode, with the performance gap becoming more pronounced on challenging datasets such as AIME2024, where the difference reaches up to 35.42 on Qwen3-14B. However, on easier datasets, the performance of the two modes tends to converge (e.g., 92.80 vs. 92.72 for R1-32B on GSM8K), and in some cases, *Nothinking* even slightly outperforms *Thinking* (e.g., 79.00 vs. 78.20 for R1-1.5B on MATH500).

While few-shot CoT remains one of the most commonly used ICL strategies for traditional LLMs, it demonstrates limited effectiveness on RLLMs. Nonetheless, it yields modest gains on few benchmarks for larger models such as Qwen3-14B on GSM8K and R1-32B on MATH500. In contrast, majority voting and thinking twice remain strong no-training methods for RLLMs. The former exhibits relatively superior performance on smaller model sizes, whereas the latter tends to be more effective on larger models. However, thinking twice shows instability, occasionally performing on par with or even worse than single *Thinking* (e.g., 94.77 vs. 94.77 for Qwen3-14B on GSM8K, and 60.00 vs. 63.75 for R1-14B on AIME24). Our proposed *JointThinking* consistently outperforms both baselines, demonstrating that simply increasing the number of reasoning attempts can't improve performance stably and efficiently. We show one detailed case study of our method in Appendix C.

4.3 Generalizability

In-context learning (ICL) provides inherent generalization advantages over training-based paradigms, supported by both theoretical analysis (Hosseini et al., 2022; Zhang et al., 2025c) and empirical evidence (Li et al., 2024; Lampinen et al., 2025). In this section, we compare our proposed method with *AdaptThink* (Wan et al., 2025), a SOTA training-based reasoning enhancement approach. *AdaptThink* introduces a reinforcement learning strategy that enables RLLMs to dynamically select the optimal thinking mode according to problem difficulty, thereby effectively mitigating overthinking. We evaluate both methods on in-distribution (ID) and out-of-distribution (OOD) scenarios.

Dataset	Error Rate (ER)		ΔER
	Thinking with Thinking	Thinking with Nothinking	
GSM8K	4.90%	4.06%	0.84%
MATH500	5.23%	3.68%	1.55%
AIME24	1.67%	1.25%	0.42%
AMC23	0.63%	0.00%	0.63%

Table 3: Error rate across different datasets based on R1-7B. For each dataset, we sample 8 outputs for computation. More details are provided in Appendix D.1.

For ID evaluation, we follow the *AdaptThink* paper and use GSM8K, MATH500, and AIME24 datasets. For OOD evaluation, we consider two challenging benchmarks: MMLU-Pro (Wang et al., 2024) and GPQA (Rein et al., 2024). MMLU-Pro is a professional-level subset of MMLU targeting advanced subject-specific reasoning, while GPQA is a challenging, expert-validated benchmark requiring multi-step reasoning in scientific domains.

As illustrated in Table 2, *AdaptThink* and *JointThinking* achieve comparable performance on ID datasets, with *JointThinking* consistently outperforming *AdaptThink* across all three benchmarks for R1-1.5B. The most notable difference arises in the OOD setting. On both MMLU-Pro and GPQA, *JointThinking* significantly outperforms *AdaptThink*, improving from 35.77 to 45.21 and from 32.89 to 39.78 on R1-1.5B, and from 57.07 to 66.79 and from 51.23 to 57.49 on R1-7B, respectively. These results highlight the strong generalization capability of *JointThinking*, enabling robust performance across diverse unseen domains.

4.4 Calibration Analysis

In this subsection, we provide a systematic explanation and analysis of the calibration mechanism underlying our proposed method. As illustrated in Figure 1, we observe a complementary relationship between the *Thinking* and *Nothinking* modes: one often succeeds where the other fails. We hypothesize this arises from fundamental structural differences between the two reasoning modes. The *Nothinking* mode relies more on intuition, rapidly recalling answers from training patterns but some-

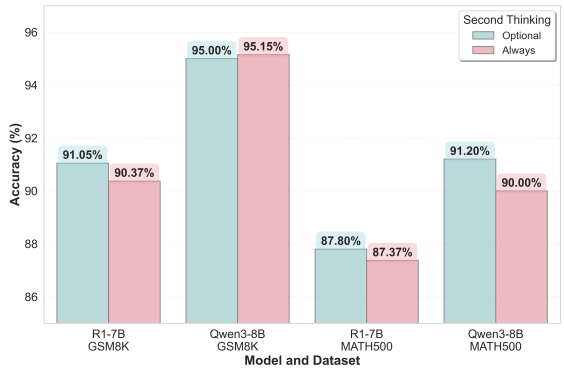

Figure 4: Performance comparison between optional and always second thinking. The optional trigger setting yields better results with less computational overhead.

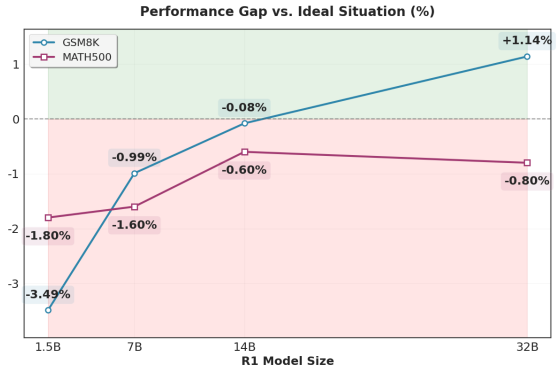

Figure 5: Scaling trend of second-thinking performance. The reduction in performance gap from the ideal situation indicates the strong scalability of *jointthinking*.

times overlooking key details. Conversely, the *Thinking* mode emphasizes deliberate reasoning through iterative exploration, backtracking, and verification, though it may overthink problems.

To empirically verify this structural thinking divergence, we evaluate the effectiveness of cross-mode calibration by using both *Thinking* and *Nothinking* modes to calibrate outputs generated by the *Thinking* mode. We introduce a metric called **Error Rate (ER)** to measure calibration performance. Let $\mathcal{D}$ denote a dataset with ground-truth labels, and let $\mathcal{C} \subseteq \mathcal{D}$ represent the subset of examples for which both modes produce consistent answers. The Error Rate is defined as:

$$\text{ER} = \frac{|\{q \in \mathcal{C} : A(r_{\text{thinking}}) \neq \text{GT}(q)\}|}{|\mathcal{C}|} \quad (6)$$

where $\text{GT}(q)$ denotes the ground-truth answer to question q, and $A(r)$ extracts the final answer from a response r. Inconsistent answers naturally signal uncertainty and can be escalated to a second round of reasoning using the same model or even more capable RLLMs for refinement. However, when an incorrect answer is mistakenly deemed reliable due to agreement between modes, it may be finalized prematurely, preventing further correction. Therefore, a lower ER reflects more effective calibration, indicating that the process effectively preserves correct answers while reducing errors.

As shown in Table 3, we sample 8 outputs per example using R1-7B across all datasets to reduce variance and report the average results using equation (6). Across all benchmarks, calibrating *Thinking* outputs with the *Nothinking* mode consistently outperforms calibration with the same mode. Notably, we observe absolute reductions in error rates ranging from 0.42% to 1.55%, underscoring the benefits of calibrating across structurally different reasoning processes. These results support our hypothesis that the structural difference between reasoning modes enables more effective calibration.

5 Further Analysis and Discussion

5.1 The Necessity of Consistency Check

The primary advantage of the consistency check mechanism in the second stage of our method lies in efficiency, as it is only applied to the most uncertain cases, keeping additional inference latency minimal. For example, on GSM8K with R1-32B, second thinking is required in just 6% of cases. More calculation details can be found in Appendix D.2. Furthermore, as shown in Figure 4, selectively triggering second thinking via consistency check can even outperform always applying it, since repeated thinking risks turning correct answers into incorrect ones. This result highlights that naively increasing reasoning iterations may degrade model performance, underscoring the effectiveness of our pipeline's design.

5.2 Scaling Trend in Second Thinking

The previous analysis primarily focuses on the overall performance of *JointThinking*. In this section, we shift our attention to the second thinking stage and investigate how its effectiveness evolves as model size increases. In our approach, when the *Thinking* and *Nothinking* modes generate inconsistent answers, a new prompt is constructed and guides the model for a second round of inference using the *Thinking* mode. This procedure is referred to as the second thinking. Within these inconsistent cases, we identify two different scenarios:

Dataset	Methods		
	Think -ing	JointThinking before <think>	JointThinking after <think>
R1-7B			
GSM8K	87.79	89.61	91.05
MATH500	87.60	87.20	87.80
AIME24	47.50	50.00	51.25
AMC23	86.88	87.50	88.44
Qwen3-8B			
GSM8K	94.69	95.00	95.00
MATH500	87.80	90.00	91.20
AIME24	59.17	62.08	71.25
AMC23	86.56	90.31	92.81

Table 4: Effect of prompt placement in the second thinking stage of *JointThinking*. More experiments across different models can be found in Appendix E.

Scenario 1: Both modes produce incorrect answers which are different each other.

Scenario 2: One mode produces a correct answer while the other does not.

Ideally, in Scenario 2, the second thinking stage should consistently recover the correct answer. However, due to structural misalignment between reasoning modes or inherent randomness in decoding, this ideal is not always achieved. Figure 5 illustrates the performance gap between actual second thinking results and the ideal upper bound (i.e., perfect recovery in Scenario 2) on GSM8K and MATH500, across models of increasing size. Interestingly, we observe a clear upward trend in second thinking performance as model size increases. Notably, the 32B model even surpasses the ideal baseline by 1.14% on GSM8K, suggesting that the RLLM can arrive at the correct answer through second thinking even when all candidate answers are incorrect. This result highlights the strong scalability of our approach, which we attribute to enhanced reflective capabilities of larger models.

5.3 Current Limitations and Future Directions

In the final part of this paper, we summarize several key findings, to discuss current limitations and suggest directions for future research. The success of LLMs is largely attributed to their strong ICL capabilities. However, this strength transfers less effectively to RLLMs. This is perhaps expected, as most RL algorithms for training RLLMs focus on solving complex mathematical problems and simply generating long reasoning chains. Such training often employs simplified prompts that lack diversity and rich information, which may inadvertently reduce the model's sensitivity to external guidance.

Our findings in Table 1 emphasize this observation, where few-shot CoT demonstrates limited effectiveness on RLLMs compared to expectations. To further investigate this limitation, we conduct an ablation study by explicitly changing the position of the prompt relative to the reasoning token in the second thinking stage. Specifically, we compare two configurations: (1) placing the prompt, consisting of the original question and two inconsistent candidate answers, **before** the `<think>` token, treating it as a general instruction, and (2) placing the prompt **after** this special token, integrating it as part of the model's reasoning process. The results reported in Table 4 show that the first setting, resembling standard prompting, provides limited benefit and even degrades performance on MATH500 with R1-7B. In contrast, the second setting consistently leads to improved performance. These findings reveal a fundamental shortcoming of current RLLMs: insufficient instruction-following capabilities hinder the effective use of ICL.

Building on these insights, we suggest two directions for future research: (1) On the training side, RL strategies should be designed not only to improve reasoning performance but also to preserve or even strengthen the model's ability to follow instructions; (2) On the inference side, future work may explore how to better leverage the structural differences between reasoning modes and integrate them more effectively for enhanced reasoning. We hope our findings inspire further exploration into improving the ICL capabilities of RLLMs.

6 Conclusion

In this work, we introduce *Thinking with Nothinking Calibration (JointThinking)*, a new ICL paradigm for RLLMs that leverages the structural difference between *Thinking* and *Nothinking* modes. Experimental results demonstrate that *JointThinking* significantly outperforms existing no-training approaches, and exhibits stronger generalization compared with training-based reasoning enhancement. We further provide a systematic analysis of the calibration mechanism, underscoring the importance of incorporating different reasoning modes and consistency check mechanism. Finally, we investigate how model scaling influences second-thinking performance, highlighting the strong scalability of our method.

7 Limitations

While our approach demonstrates strong effectiveness, several limitations should be acknowledged. First, due to computational constraints for RLLM inference with 16K context window, our experiments are limited to model size up to 32B. The experiment results demonstrate that our proposed method works across different models and sizes, and the scaling trend of the second thinking also suggests its strong scalability. Second, we focus exclusively on *Thinking* and *Nothinking* modes, without exploring broader reasoning mode combinations. However, this limitation largely stems from the constrained set of reasoning modes currently supported by existing RLLMs rather than our framework design. Third, our consistency check mechanism relies on exact or near-exact answer matching to trigger the second round of thinking. It works well for mathematical reasoning benchmarks in our experiments, while for more open-ended problems the semantic similarity consistency check could be considered.

References

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

AI-MO. 2023. Ai-mo/aimo-validation-amc. https://huggingface.co/datasets/AI-MO/aimo-validation-amc. 2023.

AI-MO. 2024. Ai-mo/aimo-validation-aime. https://huggingface.co/datasets/AI-MO/aimo-validation-aime. 2024.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. *Preprint*, arXiv:2005.14165.

Davide Buoso, Luke Robinson, Giuseppe Averta, Philip Torr, Tim Franzmeyer, and Daniele De Martini. 2024. Select2plan: Training-free icl-based planning through vqa and memory retrieval. *Preprint*, arXiv:2411.04006.

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, and 1 others. 2024. Do not think that much for 2+ 3=? on the overthinking of o1-like llms. *arXiv preprint arXiv:2412.21187*.

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

Alejandro Cuadron, Dacheng Li, Wenjie Ma, Xingyao Wang, Yichuan Wang, Siyuan Zhuang, Shu Liu, Luis Gaspar Schroeder, Tian Xia, Huanzhi Mao, and 1 others. 2025. The danger of overthinking: Examining the reasoning-action dilemma in agentic tasks. *arXiv preprint arXiv:2502.08235*.

Yuyao Ge, Shenghua Liu, Yiwei Wang, Lingrui Mei, Lizhe Chen, Baolong Bi, and Xueqi Cheng. 2025. Innate reasoning is not enough: In-context learning enhances reasoning large language models with less overthinking. *arXiv preprint arXiv:2503.19602*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Arian Hosseini, Ankit Vani, Dzmitry Bahdanau, Alessandro Sordoni, and Aaron Courville. 2022. On the compositional generalization gap of in-context learning. In *Proceedings of the Fifth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 272–280, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Andrew K Lampinen, Arslan Chaudhry, Stephanie CY Chan, Cody Wild, Diane Wan, Alex Ku, Jörg Bornschein, Razvan Pascanu, Murray Shanahan, and James L McClelland. 2025. On the generalization of language models from in-context learning and finetuning: a controlled study. *arXiv preprint arXiv:2505.00661*.

Chuanhao Li, Chenchen Jing, Zhen Li, Mingliang Zhai, Yuwei Wu, and Yunde Jia. 2024. In-context compositional generalization for large vision-language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17954–17966, Miami, Florida, USA. Association for Computational Linguistics.

Peiji Li, Kai Lv, Yunfan Shao, Yichuan Ma, Linyang Li, Xiaoqing Zheng, Xipeng Qiu, and Qipeng Guo. 2025a. Fastmcts: A simple sampling strategy for data synthesis. *arXiv preprint arXiv:2502.11476*.

Xiaomin Li, Zhou Yu, Zhiwei Zhang, Xupeng Chen, Ziji Zhang, Yingying Zhuang, Narayanan Sadagopan, and Anurag Beniwal. 2025b. When thinking fails: The pitfalls of reasoning for instruction-following in llms. *arXiv preprint arXiv:2505.11423*.

Baohao Liao, Yuhui Xu, Hanze Dong, Junnan Li, Christof Monz, Silvio Savarese, Doyen Sahoo, and Caiming Xiong. 2025. Reward-guided speculative decoding for efficient llm reasoning. *arXiv preprint arXiv:2501.19324*.

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

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

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*.

Nicholas Meade, Spandana Gella, Devamanyu Hazarika, Prakhar Gupta, Di Jin, Siva Reddy, Yang Liu, and Dilek Hakkani-Tür. 2023. Using in-context learning to improve dialogue safety. *arXiv preprint arXiv:2302.00871*.

OpenAI. 2024. Introducing o3 and o4 mini. https://openai.com/index/introducing-o3-and-o4-mini/. Accessed: 2025-06-30.

Keqin Peng, Liang Ding, Yancheng Yuan, Xuebo Liu, Min Zhang, Yuanxin Ouyang, and Dacheng Tao. 2024. Revisiting demonstration selection strategies in in-context learning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9090–9101, Bangkok, Thailand. Association for Computational Linguistics.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. 2024. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

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

Zhenmei Shi, Junyi Wei, Zhuoyan Xu, and Yingyu Liang. 2024. Why larger language models do in-context learning differently? *Preprint*, arXiv:2405.19592.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*.

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

Yiming Tang and Bin Dong. 2024. Demonstration notebook: Finding the most suited in-context learning example from interactions. *Preprint*, arXiv:2406.10878.

Xu Wan, Wei Wang, Wenyue Xu, Wotao Yin, Jie Song, and Mingyang Sun. 2025. Adapthink: Adaptive thinking preferences for reasoning language model. *arXiv preprint arXiv:2506.18237*.

Xubin Wang, Jianfei Wu, Yichen Yuan, Deyu Cai, Mingzhe Li, and Weijia Jia. 2025a. Demonstration selection for in-context learning via reinforcement learning. *Preprint*, arXiv:2412.03966.

Yiming Wang, Pei Zhang, Siyuan Huang, Baosong Yang, Zhuosheng Zhang, Fei Huang, and Rui Wang. 2025b. Sampling-efficient test-time scaling: Self-estimating the best-of-n sampling in early decoding. *arXiv preprint arXiv:2503.01422*.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, and 1 others. 2024. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Jerry Wei, Le Hou, Andrew Lampinen, Xiangning Chen, Da Huang, Yi Tay, Xinyun Chen, Yifeng Lu, Denny Zhou, Tengyu Ma, and 1 others. 2023. Symbol tuning improves in-context learning in language models. *arXiv preprint arXiv:2305.08298*.

Noam Wies, Yoav Levine, and Amnon Shashua. 2023. The learnability of in-context learning. *Advances in Neural Information Processing Systems*, 36:36637–36651.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

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

Method	Prompt Template
Thinking	Problem: {problem} \n Please put your final answer in the format boxed{} <|Assistant|>\n Answer:<think>\n
Nothinking	Problem: {problem} \n Please put your final answer in the format boxed{} <|Assistant|>\n Answer:<think>\n Okay I have finished thinking.\n</think>\n
Few-shot	Problem: {example problem}\n Answer: {example answer}\n\n ... (more few-shot cot examples) Problem: {problem} \n Please put your final answer in the format boxed{} <|Assistant|>\n Answer:<think>\n
Thinking Twice	Problem: {problem} \n Please put your final answer in the format boxed{} <|Assistant|>\n Answer:<think>\n I think there is one candidate answer {thinking_answer} for this question. I need to first verify it. If it is wrong, I need to rethink step by step and avoid making the same mistake.
JointThinking	Problem: {problem} \n Please put your final answer in the format boxed{} <|Assistant|>\n Answer:<think>\n I think there are two candidate answers {thinking_answer} and {nothinking_answer} for this question. One of them is correct or both are wrong. I need to first verify them. If both are wrong, I need to rethink step by step and avoid making the same mistake.

Table 5: Methods and their corresponding prompt templates for main results.

Zhaojian Yu, Yinghao Wu, Yilun Zhao, Arman Cohan, and Xiao-Ping Zhang. 2025. Z1: Efficient test-time scaling with code. *arXiv preprint arXiv:2504.00810*.

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

Xiang Zhang, Juntai Cao, Jiaqi Wei, Chenyu You, and Dujian Ding. 2025b. Why prompt design matters and works: A complexity analysis of prompt search space in llms. *Preprint*, arXiv:2503.10084.

Xingxuan Zhang, Haoran Wang, Jiansheng Li, Yuan Xue, Shikai Guan, Renzhe Xu, Hao Zou, Han Yu, and Peng Cui. 2025c. Understanding the generalization of in-context learning in transformers: An empirical study. In *The Thirteenth International Conference on Learning Representations*.

A Prompts

Table 5 presents detailed prompt templates for different methods in Table 1. In the section 3.2.1 when we introduce our method we use special tokens `<|beginning of thinking|>` and `<|end of thinking|>` to explain our prompt engineering in general. In our experiments, we employ the R1-series and Qwen3-series models, with specific thinking special tokens `<think>` and `</think>` instead. In particular, for the Qwen3 it natively supports switching between the thinking and nothinking modes by adding special tokens `\think` or `\nothink` after the prompt, so we follow this setting in Thinking, Nothinking and Few-shot scenarios. Notably, in the *JointThinking* pipeline, it will first use general thinking mode and nothinking mode to generate answers with their corresponding prompt templates. Only in the second thinking the *JointThinking* prompt structure shown in the last row of the table will be used. Additionally, all instructions are provided through the user prompt, without using system prompt.

B Algorithm Description

The complete pipeline of our method *JointThinking* is summarized in Algorithm 1.

C Case Study

To further illustrate our proposed method, we present a detailed case study based on R1-32B model. We select one problem from the AMC23 datasets that are wrong in single thinking mode, but are solved correctly by *JointThinking*.

Under the thinking mode, the model arrives at an incorrect final answer. This error stems from a flawed reasoning process: although the model correctly removes the 7.5% sales tax by dividing the total budget by 1.075, it then mistakenly treats the resulting value as the discounted price before applying the 20% reduction. In reality, the value obtained after removing the tax already reflects both the 20% discount and the tax. As a result, when the model attempts to "reverse" the discount by dividing again

Algorithm 1 *Thinking with NoThinking Calibration (JointThinking)*

Require: Question q, Model M
Ensure: Final answer a_{final}

```
Generate thinking response: r_thinking ← M(q + ⟨BOT⟩)
Generate nothinking response: r_nothinking ← M(q + ⟨BOT⟩ + s_skip + ⟨EOT⟩)
Extract answers: a_1 ← A(r_thinking), a_2 ← A(r_nothinking)
if a_1 = a_2 then
    a_final ← a_1
else
    Construct second thinking prompt: p_second_thinking ← Prompt_second_thinking(q, r_thinking, r_nothinking)
    Generate final response: r_final ← M(p_second_thinking)
    a_final ← A(r_final)
end if
return a_final
```

Pure Question from AMC23

Carlos went to a sports store to buy running shoes. Running shoes were on sale, with prices reduced by 20% on every pair of shoes. Carlos also knew that he had to pay a 7.5% sales tax on the discounted price. He had \$43. What is the original (before discount) price of the most expensive shoes he could afford to buy?

Figure 6: One example question from AMC23 dataset.

by 0.8, it overcompensates and produces an underestimated final answer (i.e., \$49.94 instead of the correct \$50.00). Notably, the question does not trigger any reflection or revision during the model's thinking phase. Although the subsequent computational steps after thinking processing are mathematically sound, the incorrect answer of thinking part still leads the model to the wrong conclusion.

In the nothinking mode, the model provides a clean and well-structured step-by-step solution with correct answer.

In the *JointThinking*, the reasoning model starts thinking with two candidate answers. We observe that the model initially approaches the problem in a relatively independent manner, treating the task care. The presence of two plausible candidate answers in the prompt introduces uncertainty, prompting the model to engage in more cautious and deliberate reasoning. Interestingly, after arriving at the correct answer, the model proceeds to critically examine the alternative option, evaluating its plausibility and attempting to justify or refute it (the relevant sentences are underlined with red color). This process triggers a distinctive reasoning pattern characterized by iterative reflection and verification. Ultimately, the model selects the correct answer and provides a possible explanation for why the model is able to get the other incorrect candidate answer.

Thinking Mode

<think>
First, let's determine the maximum amount Carlos can spend after accounting for the sales tax. He has \$43, and the sales tax rate is 7.5%. To find the amount before tax, we divide \$43 by 1.075, which gives approximately \$39.95.

Next, we need to find the original price of the shoes before the 20% discount. Since the discounted price is \$39.95, we divide this by 0.80 to reverse the 20% reduction. This calculation results in approximately \$49.94.

Therefore, the original price of the most expensive shoes Carlos could afford is approximately \$49.94.
</think>

To determine the original price of the most expensive shoes Carlos could afford, let's follow these steps:

1. Understand the Problem:
   - Sale Discount: 20% off the original price.
   - Sales Tax: 7.5% on the discounted price.
   - Carlos's Budget: \$43.
2. Let's Define Variables:
   - Let P be the original price of the shoes.
   - The discounted price after 20% off is $0.80P$.
   - The final price after adding 7.5% tax is $0.80P \times 1.075$.
3. Set Up the Equation:
$$0.80P \times 1.075 \leq 43$$
4. Solve for P:
$$0.80P \times 1.075 = 43$$
$$0.86P = 43$$
$$P = \frac{43}{0.86} \approx 49.94$$
5. Conclusion:
   The original price of the most expensive shoes Carlos could afford is approximately \$49.94.

$$\boxed{49.94}$$

Figure 7: The output of R1-32B in Thinking mode for selected question.

Nothinking Mode

To determine the original price of the most expensive shoes Carlos could afford, we'll follow these steps:

1. Understand the Discounts and Taxes:
   - Discount: The shoes are discounted by 20%. This means Carlos pays 80% of the original price.
   - Sales Tax: After the discount, there's a 7.5% sales tax applied to the discounted price.
2. Let P be the original price of the shoes.
3. Calculate the Discounted Price:
$$\text{Discounted Price} = P \times (1 - 0.20) = 0.80P$$
4. Calculate the Price After Tax:
$$FinalPrice = DiscountedPrice \times (1 + 0.075) = 0.80P \times 1.075 = 0.86P$$
5. Set Up the Inequality Based on Carlos's Budget:
$$0.86P \leq 43$$
6. Solve for P:
$$P \leq \frac{43}{0.86} = 50$$

Conclusion: The original price of the most expensive shoes Carlos could afford is $\boxed{50}$ dollars.

Figure 8: The output of R1-32B in Nothinking mode for selected question.

JointThinking

<think>
So, let's start.

First, the problem says Carlos went to a sports store to buy running shoes. The running shoes were on sale with a 20% discount on every pair. Then, Carlos has to pay a 7.5% sales tax on the discounted price. He had \$43. So, the question is, what is the original price of the most expensive shoes he could afford?

Alright, so I need to find the original price, let's call it P. Then, after a 20% discount, the price becomes P minus 20% of P, which is $0.8P$. Then, he has to pay 7.5% tax on that discounted price, so the total price he has to pay is $0.8P$ multiplied by 1.075.

So, the equation is $0.8P \times 1.075 =$ total price he has to pay. And this total price should be less than or equal to \$43 because he can't spend more than that.

So, 0.8×1.075 is equal to 0.86. So, $0.86P \leq 43$.
To find P, divide both sides by 0.86. So, $P \leq 43/0.86$.
Calculating 43 divided by 0.86. Hmm, 0.86 times 50 is 43, because 0.86×50 is 43. So, $P \leq 50$. So, the original price must be less than or equal to \$50.

But wait, the question is asking for the original price of the most expensive shoes he could afford. So, if the original price is \$50, he can buy it because the total is exactly \$43. So, the answer should be \$50.

But why is 49.94 another candidate? Maybe someone thought that the total should be less than \$43, so they might have rounded down? Let me see.

If the original price is \$49.94, then 20% off is $49.94 \times 0.2 = 9.988$, so the discounted price is $49.94 - 9.988 = 39.952$. Then, 7.5% tax on 39.952 is 39.952×1.075. Let me calculate that.

First, 39.952×1.075. Let's compute $39.952 \times 1 = 39.952$. Then, 39.952×0.075. Let me calculate $39.952 \times 0.07 = 2.79664$, and $39.952 \times 0.005 = 0.19976$. Adding those together, $2.79664 + 0.19976 = 2.9964$. So, total tax is 2.9964. Adding to the discounted price: $39.952 + 2.9964 = 42.9484$, which is approximately \$42.95. So, that's under \$43.

Therefore, if the original price is \$49.94, he would spend about \$42.95, which is under his \$43. So, why is \$50 the correct answer? Because if the original price is \$50, after discount and tax, it's exactly \$43, which he can afford. So, is \$50 the correct answer? Or is \$49.94 the maximum he can spend without exceeding his budget?

Wait, the problem says he had \$43. So, he can spend up to \$43. So, if the total cost is exactly \$43, he can buy it. So, \$50 is okay because the total is exactly \$43. So, \$50 is the maximum original price he can afford.

.....

So, why did I get confused earlier? Because I incorrectly subtracted 10.002 from 50.01 as 39.999, but it's actually 40.008. So, that was my mistake.
Therefore, the correct answer is \$50.

But wait, let's confirm with $P = 50$.
Discount: $50 \times 0.2 = 10$, so discounted price is 40.
Tax: $40 \times 1.075 = 43$.
So, total is exactly \$43, which he can afford.
If P is just a little over 50, say 50.01, the total is just over 43, which he can't afford.
Therefore, the maximum original price is \$50.

But wait, in the initial calculation, I thought the total was $0.86P$, and $43/0.86 = 50$. So, that's consistent.
Therefore, the correct answer is \$50.

So, why was 49.94 a candidate? Maybe someone thought that the total should be strictly less than \$43, so they might have rounded down, but in reality, since \$50 gives exactly \$43, which he can afford, the answer is \$50.

Therefore, the original price is \$50.
</think>

Carlos went to a sports store to buy running shoes that were on sale with a 20% discount. After the discount, a 7.5% sales tax was applied. He had \$43 to spend. To find the original price (P) of the most expensive shoes he could afford:
1.Discount Calculation: 20% off means the price becomes 80% of the original, so $0.8P$.
2.Tax Calculation: 7.5% tax on the discounted price: $0.8P \times 1.075 = 0.86P$.
3. Total Cost: $0.86P \leq 43$.
4. Solve for P: $P \leq \frac{43}{0.86} = 50$.

Verification:
1.For $P = 50$: Discounted price is $50 \times 0.8 = 40$. Tax adds $40 \times 0.075 = 3$, totaling 43, which Carlos can afford.
2.For $P = 50.01$: The total exceeds \$43, so it's unaffordable.

Thus, the most expensive shoes Carlos can afford originally cost $\boxed{50}$ dollars.

854

D Experiment Details

In this section, we will provide more supplementary computation details for tables used in the main text of the paper.

D.1 The Computation Details of Error Rate

In the Table 8, we illustrate the computation details of error rate for Table 3. Error rate denotes the proportion of incorrect answers among consistent answers in our method, referring to the result of dividing the number of answers in Both Correct by the number of answers in Both Incorrect & Equiv in terms of table content. In order to ensure the validity of the conclusion, we repeat the experiment 8 times on R1-7B and take the average score as the final experimental result. The performance of thinking with nothinking calibration consistently outperforms thinking with thinking calibration for all datasets, showing that calibration capability from different modes is better than the same modes.

D.2 The Consistency-checking Analysis of JointThinking in the Main Results

Table 6 shows the consistency analysis of answers' correctness after the first stage of *JointThinking*, where reasoning large language model generates answers with thinking mode and nothinking mode in parallel. For each question, there are four possible categories: Both Correct and Both Incorrect (the answers from two different modes are both correct & incorrect), Only Thinking Correct and Only Nothinking Correct (for two answers, only the one from thinking & nothinking mode is correct, and the other is incorrect). These four categories are mutually exclusive and their sum is the total amount of the dataset. Both Incorrect & Equiv is one subset of Both Incorrect which means although two answers are consistent, they are both wrong.

In our method, when generating answers from thinking mode and nothinking mode, the answers that fall into the categories of Both Correct and Both Incorrect & Equiv will be considered as the final answers to output, and those from Both Incorrect & Equiv are error answers after consistency check. Others that don't pass the consistency check will be assembled into prompt with original question for the second thinking.

E Additional Results of Prompt Placement

In order to study the effect of prompt placement in the second thinking of our proposed method, we conduct a comprehensive ablation study across different models (R1-series and Qwen3-series), sizes (7B, 8B, 14B), and different datasets (GSM8K, MATH500, AIME24, AMC23). For all running, we follow the experiment setting in Section 4, where we set the temperature to 0.6, top-p to 0.95 for the thinking mode and temperature to 0.7, top-p to 0.8 for the nothinking mode. Everything is the same, only the position of the prompt is different. *JointThinking* before <think> means the prompt is put before the `<|beginning of thinking|>` special token as the normal instruction, and *JointThinking* after <think> means the prompt is put after the `<|beginning of thinking|>` special token as the stater of the long cot in the reasoning processing. As shown in Table 7, the experiments illustrate that placing the prompt after the special token significantly outperforms the baseline and before the special token, which is one of the keys to the effectiveness of our method. We also discover that in some cases *JointThinking* before <think> is even worse than the baseline, showing that insufficient instruction following ability limits the performance of prompt engineering.

Model	Thinking&NoThinking (The First Stage)	Datasets			
		GSM8K(1319)	MATH500(500)	AIME24(30)	AMC23(40)
R1-1.5B	Both Correct	950	363	5	25
	Both Incorrect	162	77	18	7
	Both Incorrect & Equiv	73	21	1	1
	Only Thinking Correct	129	28	5	6
	Only NoThinking Correct	78	32	2	2
R1-7B	Both Correct	1069	361	6	25
	Both Incorrect	105	53	15	4
	Both Incorrect & Equiv	51	18	0	0
	Only Thinking Correct	89	77	9	9
	Only NoThinking Correct	56	9	0	2
R1-14B	Both Correct	1136	359	12	29
	Both Incorrect	79	48	10	4
	Both Incorrect & Equiv	46	18	1	1
	Only Thinking Correct	51	82	7	7
	Only NoThinking Correct	53	11	1	0
R1-32B	Both Correct	1192	369	13	28
	Both Incorrect	64	47	11	2
	Both Incorrect & Equiv	42	26	0	0
	Only Thinking Correct	32	62	6	8
	Only NoThinking Correct	31	22	0	2

Table 6: The analysis of consistency in the first stage of *JointThinking* for main results (one group).

Dataset	Methods		
	Thinking (Baseline)	JointThinking before <think>	JointThinking after <think>
		R1-7B	
GSM8K	87.79	89.61	91.05
MATH500	87.60	87.20	87.80
AIME24	47.50	50.00	51.25
AMC23	86.88	87.50	88.44
		R1-14B	
GSM8K	89.99	92.34	93.93
MATH500	88.20	87.00	89.80
AIME24	63.75	63.33	65.42
AMC23	90.00	87.50	93.75
		Qwen3-8B	
GSM8K	94.69	95.00	95.00
MATH500	87.80	90.00	91.20
AIME24	59.17	62.08	71.25
AMC23	86.56	90.31	92.81
		Qwen3-14B	
GSM8K	94.77	95.38	95.45
MATH500	89.80	91.58	92.59
AIME24	62.92	61.67	71.67
AMC23	92.50	94.06	96.88

Table 7: Effect of prompt placement in the second thinking of our proposed method across different model sizes and datasets.

Method			GSM8K	MATH500	AIME24	AMC23
Thinking&Thinking	Exp1	Both Correct	1099	421	10	34
		Both Incorrect & Equiv	64	29	0	0
	Exp2	Both Correct	1101	423	11	32
		Both Incorrect & Equiv	71	25	0	0
	Exp3	Both Correct	1103	425	13	33
		Both Incorrect & Equiv	63	25	1	0
	Exp4	Both Correct	1107	425	12	33
		Both Incorrect & Equiv	60	24	0	0
	Exp5	Both Correct	1095	430	9	32
		Both Incorrect & Equiv	56	23	0	0
	Exp6	Both Correct	1098	424	10	33
		Both Incorrect & Equiv	67	27	2	1
	Exp7	Both Correct	1098	418	9	33
		Both Incorrect & Equiv	69	28	1	1
	Exp8	Both Correct	1106	420	10	34
		Both Incorrect & Equiv	67	28	0	0
Thinking&Nothinking	Exp1	Both Correct	1069	350	4	21
		Both Incorrect & Equiv	51	23	0	0
	Exp2	Both Correct	1054	347	6	22
		Both Incorrect & Equiv	57	17	0	0
	Exp3	Both Correct	1049	369	9	28
		Both Incorrect & Equiv	60	11	1	0
	Exp4	Both Correct	1074	361	4	23
		Both Incorrect & Equiv	50	18	0	0
	Exp5	Both Correct	1061	354	5	25
		Both Incorrect & Equiv	45	17	1	0
	Exp6	Both Correct	1070	351	6	24
		Both Incorrect & Equiv	53	20	0	0
	Exp7	Both Correct	1048	360	4	20
		Both Incorrect & Equiv	59	20	0	0
	Exp8	Both Correct	1065	355	8	22
		Both Incorrect & Equiv	53	21	1	0
Error Rate	Thinking&Thinking		4.90%	5.23%	1.67%	0.63%
	Thinking&Nothinking		**4.06%**	**3.68%**	**1.25%**	**0.00%**

Table 8: Error rate computation details across different datasets based on the R1-7B. Thinking&Thinking means using thinking results to calibrate thinking results, and Thinking&Nothinking means using nothinking results to calibrate thinking results, as known as *JointThinking*.