Meta Thinker: Thinking What AI Thinks

Junyu Guo*
University of California, Berkeley

Shangding Gu
University of California, Berkeley

Costas Spanos
University of California, Berkeley

Javad Lavaei
University of California, Berkeley

Abstract

Large Language Models (LLMs) have shown remarkable ability to tackle complex reasoning problems across diverse domains. Yet, their performance often hinges on adopting an appropriate reasoning strategy tailored to the problem structure. Existing paradigms such as Chain-of-Thought (CoT) and Tree-of-Thought (ToT) have substantially improved reasoning quality, but a fundamental question remains: *how can we automatically determine the optimal reasoning style for a given task?* This work addresses the challenge of *adaptive reasoning strategy selection* in LLMs. We investigate whether LLMs can autonomously identify and apply the most suitable thinking style across different problem types. Through comprehensive experiments spanning five representative reasoning paradigms, we analyze their relative performance on mathematical, logical, and commonsense reasoning benchmarks. Building on these insights, we propose a *meta-thinking prompt algorithm* that enables LLMs to dynamically select—or even synthesize—reasoning styles based on contextual inputs. Our approach improves both accuracy and token efficiency, demonstrating the potential for self-adaptive reasoning in LLMs. This study represents a step toward more flexible and context-aware, reasoning systems. The implementation is publicly available at `https://github.com/JamesJunyuGuo/meta_thinker_LLM`.

1 Introduction

Large Language Models (LLMs) have demonstrated impressive capabilities across a wide spectrum of tasks, ranging from natural language reasoning to agent-based tool use. Among these, *mathematical reasoning* has emerged as a particularly important frontier, where advanced models such as GPT-4 [Achiam et al., 2023] and DeepSeek-R1 [Guo et al., 2025] exhibit strong problem-solving performance. These models are capable of tackling high-level mathematical problems when directly provided with questions.

Despite these advances, LLMs still struggle to consistently generate high-quality responses across diverse scenarios. They often *overthink*, producing unnecessarily verbose explanations [Chen et al., 2024, Sui et al., 2025]. In other cases, while an LLM may already possess the latent ability to solve the problem, it fails due to insufficiently targeted instructions [An et al., 2024, Zhang et al., 2024]. To address this, recent work has explored structured reasoning paradigms, such as Chain-of-Thought (CoT) [Wei et al., 2022] and Tree-of-Thought (ToT) [Yao et al., 2023], which encourage LLMs to follow explicit reasoning trajectories that can improve accuracy. However, with state-of-the-art models like GPT-o1 [Jaech et al., 2024] can already solve relatively simple reasoning problems without the need for additional token-intensive reasoning. This observation motivates the need for adaptive approaches: ideally, LLMs should produce concise and direct answers for straightforward problems,

*Corresponding authors: `junyuguo24@berkeley.edu`

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: MATH-AI.

while adopting more elaborate reasoning strategies only when the task demands it (see [Fang et al., 2025]). Under this setting, we pose the central research question:

> *How can LLMs be equipped to autonomously select the most appropriate reasoning style for a given problem?*

2 Related Works

Thinking Styles in LLM Reasoning. Recent advances in LLM reasoning center on structured *thinking styles*. Chain-of-Thought (CoT) prompting [Wei et al., 2022] demonstrated that step-by-step reasoning significantly improves performance on mathematical and logical tasks. Subsequent work expanded this concept: Tree-of-Thought (ToT) [Yao et al., 2023] enabled parallel exploration of reasoning paths, while Algorithm-of-Thought (AoT) [Sel et al., 2023], Sketch-of-Thought (SoT) [Aytes et al., 2025], and Chain-of-Draft (CoD) [Xu et al., 2025] incorporated algorithmic priors, concise sketches, and iterative drafting. Other approaches use high-level templates to enhance structured reasoning [Gao et al., 2024, Yasunaga et al., 2023, Yang et al., 2025]. However, these methods typically employ a fixed, pre-selected reasoning style. In contrast, our work focuses on *meta-thinking*, enabling LLMs to dynamically select or synthesize optimal reasoning strategies for each problem.

Benchmarking LLM Reasoning. Researchers have developed benchmarks to evaluate reasoning strategies across diverse domains. Mathematical reasoning is assessed using GSM8K [Cobbe et al., 2021] and MATH, while CommonsenseQA [Talmor et al., 2018] and LogiQA [Liu et al., 2020] target commonsense and logical inference. Complex reasoning is measured through Game of 24 [Guo et al., 2025] and multi-hop QA datasets [Yang et al., 2018]. These benchmarks reveal how prompting styles affect performance, but typically evaluate styles in isolation. Our work complements this research by proposing a unified framework for benchmarking *adaptive* style selection across diverse tasks.

3 Methodology

3.1 Thinking Style Categories

Instruction Guidance. Methods like Chain-of-Thought (CoT) [Wei et al., 2022] and ReMA [Wan et al., 2025] use strategic prompt design to guide LLMs. These approaches provide high-level instructions (e.g., "Think step by step") that help models decompose complex problems, leveraging their inherent capabilities through targeted activation. Related work [Yasunaga et al., 2023] also explores using LLMs to self-generate effective instructions.

In-Context Learning. Approaches such as Chain-of-Draft (CoD) [Xu et al., 2025] and Sketch-of-Thought (SoT) [Aytes et al., 2025] use demonstration examples to shape LLM outputs into specific formats. Particularly effective for symbolic reasoning, these methods enable models to learn reasoning patterns from few-shot examples and generalize to novel problems. This direction is extended in [Gao et al., 2024, Yang et al., 2025] through high-level reasoning templates.

Search-Based Approaches. Tree-of-Thought (ToT) [Yao et al., 2023] and Algorithm-of-Thought (AoT) [Sel et al., 2023] model reasoning as graph exploration, using multiple sampling iterations to systematically navigate solution spaces. These methods excel in combinatorial optimization and puzzle-solving tasks. Recent work [Meng et al., 2024] shows that integrating classical search algorithms (e.g., A*) with LLMs can further enhance performance.

3.2 Comparison Between Different Styles

We evaluate five representative reasoning styles:

Chain of Thought (CoT) [Wei et al., 2022]: Step-by-step decomposition for multi-step reasoning.

Chain of Draft (CoD) [Xu et al., 2025]: Condensed, symbolic reasoning using few-shot examples.

Sketch of Thought (SoT) [Aytes et al., 2025]: Two-stage approach that retrieves few-shot examples based on question type.

Tree of Thought (ToT) [Yao et al., 2023]: Tree-search maintaining parallel reasoning paths using pruning.

Algorithm of Thought (AoT) [Sel et al., 2023]: Backtracking search that explores alternative solution paths.

3.3 Beyond Single Style: Choosing the Optimal Strategy

Formalization of Meta-Thinking. We formalize the process of reasoning-style selection as a meta-level decision problem. Given an input task $x \in \mathcal{X}$ and a finite set of candidate reasoning styles $\mathcal{S} = \{s_1, s_2, \ldots, s_K\}$, each style corresponds to a reasoning policy $\pi_{s_i}(y \mid x)$ that produces an output sequence y. A *meta-thinker* $\mathcal{M}_\theta$ aims to learn a mapping $\mathcal{M}_\theta : \mathcal{X} \to \mathcal{S}$ that selects the most suitable reasoning style s^* for a given task. The optimal style is determined by maximizing an expected utility function that balances reasoning accuracy and efficiency: $s^* = \arg\max_{s \in \mathcal{S}} \mathbb{E}_{y \sim \pi_s(\cdot|x)} \big[\alpha \cdot \mathrm{Acc}(y, x) - \beta \cdot \mathrm{Cost}(y)\big]$, where $\alpha, \beta > 0$ are weighting coefficients for accuracy and token usage cost. Intuitively, the meta-thinker decides whether a task requires deeper reasoning (e.g., multi-step symbolic derivation) or a concise direct answer, thus dynamically balancing reasoning depth and efficiency. In our framework, this mapping is realized through prompting rather than explicit parameter training—allowing zero-shot adaptation across diverse reasoning tasks.

Similar to [Yang et al., 2025] and [Gao et al., 2024], our approach employs a high-level meta-reasoning template to guide the LLM. However, a key distinction of our method is its focus on fostering *explicit style awareness* within the reasoning process. Rather than simply providing instructions to solve the problem directly, we prompt the LLM to first analyze the problem type, consider its own capabilities, and then *autonomously select* the most appropriate reasoning style from a predefined set (e.g., CoT, ToT, AoT, SoT, CoD). Crucially, our prompt design also encourages the LLM to *innovate beyond* this predefined set. If the available styles are deemed insufficient for the problem at hand, the model is guided to synthesize a novel, hybrid, or adapted reasoning strategy. This capacity for strategy innovation aligns with the core objective of meta-learning: to rapidly adapt and generalize to new challenges based on prior experience. This approach mitigates a critical limitation of static instruction methods: their high dependency on problem type, which often restricts general applicability. By enabling the model to dynamically choose or even create its reasoning strategy, our method achieves greater flexibility and robustness across diverse and novel tasks. We provide two concrete examples of our meta-reasoning prompt templates in Appendix C, demonstrating how they guide the model through this style-selection and innovation process.

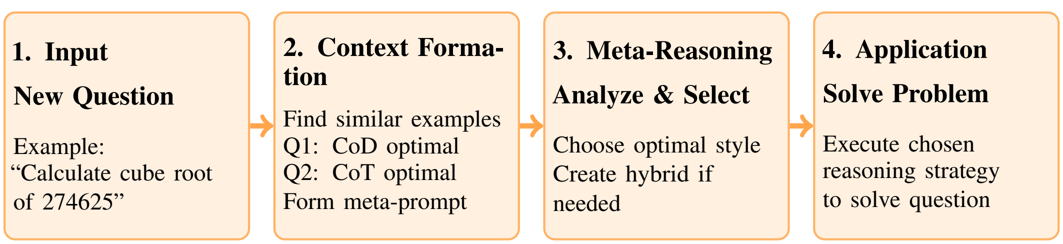


Figure 1: Meta-selection process for adaptive LLM reasoning.

Algorithm 1 Meta-Style Selection during Inference

Require: Input question x, candidate styles $\mathcal{S} = \{s_1, \ldots, s_K\}$

Compute $h_x = f_{\mathrm{enc}}(x)$

Estimate $p(s \mid x) = \mathrm{Softmax}(W h_x)$

$s^* \leftarrow \arg\max_{s \in \mathcal{S}} p(s \mid x)$

Query LLM with $\pi_{s^*}(x)$ to get y_{s^*}

return y_{s^*} (answer) and its reasoning trace

Algorithmic Overview. Algorithm 1 summarizes the meta-thinking procedure used during inference. The meta-thinker operates as a lightweight controller on top of the base LLM: it predicts which reasoning style is most appropriate for the given task and routes the query accordingly.

Formally, each input question x is first encoded into a continuous task representation $h_x = f_{\text{enc}}(x)$, where $f_{\text{enc}}(\cdot)$ is a pretrained encoder that captures the semantic and structural features of the problem. A learnable projection matrix $W \in \mathbb{R}^{K \times d}$ transforms h_x into a K-dimensional score vector, one for each reasoning style in $\mathcal{S} = \{s_1, \ldots, s_K\}$. The style prior $p(s \mid x)$ is obtained by applying a softmax over these scores: $p(s \mid x) = \mathrm{Softmax}(W h_x)$, which represents the probability that reasoning style s is optimal for task x. The controller then selects the most probable style $s^* = \arg\max_s p(s \mid x)$ and queries the base LLM using the corresponding reasoning-style prompt.

4 Experiments

4.1 Meta-Thinker: Adaptive Reasoning Style Selection

We begin by evaluating the baseline performance of individual reasoning styles across various tasks to establish a foundation for comparison. This analysis helps identify the strengths and limitations of each static approach. We then evaluate our Meta-Thinker paradigm across five diverse reasoning datasets: LogiQA, CommonsenseQA, GSM8k, Game24, and AIME. The core of our approach involves presenting the LLM with five distinct reasoning style candidates and prompting it to autonomously select the most appropriate style for each given problem. We present our result here in Table 1, with further results on other open source models attached in Appendix A.

Dataset Construction. To systematically evaluate adaptive reasoning, we constructed a multi-domain benchmark covering five representative datasets: GSM8K, LogiQA, CommonsenseQA, Game24, and AIME. Each dataset emphasizes different cognitive skills—arithmetic decomposition, logical deduction, commonsense inference, combinatorial search, and high-level mathematical reasoning. For every question x_i, we generated responses $\{y_{i,s}\}_{s \in \mathcal{S}}$ using five reasoning styles: CoT, CoD, SoT, ToT, and AoT. Each response was paired with the corresponding ground-truth label y_i^* and annotated with metrics for correctness, token length, and reasoning-step count.

Formally, let $D = \{(x_i, y_i^*)\}_{i=1}^N$ denote the full task set. For each instance, we compute a style-specific utility score $U_{i,s} = \lambda \cdot \mathbf{1}[y_{i,s} = y_i^*] - (1-\lambda)\frac{\mathrm{len}(y_{i,s})}{\max_{s'} \mathrm{len}(y_{i,s'})}$, where $\lambda \in [0,1]$ controls the trade-off between accuracy and efficiency. The optimal reasoning style label for that instance is then $s_i^* = \arg\max_{s \in \mathcal{S}} U_{i,s}$. The resulting dataset $\{(x_i, s_i^*)\}_{i=1}^N$ serves as a supervision source for analyzing meta-selection behavior and fine-tuning the meta-thinker under the SFT framework. Additional details on filtering, annotation, and balancing across domains are provided in Appendix D.

To further enhance adaptability, we extend our evaluation to scenarios where the LLM is encouraged to generate novel reasoning strategies when the predefined styles are deemed insufficient. While this capability for style innovation shows promise, we observe its most pronounced effects primarily on the Game24 dataset, with more limited impact on other tasks in our current evaluation. We present the sample response in Appendix B.

Table 1: Qwen 72B: accuracy (%)

Dataset	CoT	SoT	CoD	ToT	AoT	Ours
aime	32.32	30.16	33.93	31.42	30.52	33.87
commonsenseqa	86.20	88.74	85.80	57.00	66.40	92.70
game24	33.60	26.00	30.00	25.80	34.00	43.56
gsm8k	91.16	89.90	91.08	46.06	66.69	89.67
logiqa	36.60	61.80	45.60	28.40	21.80	52.56

4.2 Discussion

Finding: Task-Dependent Efficiency of Reasoning Styles

Search-based methods (AoT, ToT) excel in open-ended problems (e.g., Game24) through systematic multi-path exploration, albeit at high computational cost. Conversely, concise methods (SoT, CoD) achieve maximal efficiency on well-structured tasks (e.g., LogiQA) by producing accurate, abbreviated responses with significantly lower token consumption.

A qualitative analysis of responses reveals distinct failure and success modes inherent to each reasoning style. On the open-ended Game24 task, AoT succeeded by dynamically exploring diverse arithmetic operations. In contrast, ToT failed due to an unpromising initial branch from which it could not recover efficiently, highlighting a critical sensitivity of tree-search algorithms to early path selection. Non-search methods (CoT, CoD, SoT) failed on this task, defaulting to non-strategic, brute-force enumeration rather than guided exploration.

Conversely, on the structured LogiQA task, concise methods demonstrated substantial efficiency gains: CoD and SoT produced responses 16% and 94% shorter than CoT, respectively, while maintaining competitive accuracy. This efficiency stems from their distinct operational mechanisms:

Finding: Limitations of Supervised Fine-Tuning for Style Selection

SFT fails to equip LLMs with robust reasoning style selection capabilities. Even when provided with explicit rationales for optimal style choices, models default to shallow statistical patterns rather than developing a genuine understanding of strategy-task alignment.

We fine-tuned models on a curated dataset from GSM8K, LogiQA, and CommonsenseQA (500 questions per dataset). For each question, we collected responses generated under four distinct reasoning styles: CoT, CoD, ToT, and SoT. The optimal style was automatically determined by selecting the correct response (verified against ground truth) with the lowest token consumption. We trained separate models both with and without rationales explaining why a particular style was optimal for each instance. At inference time, the fine-tuned model was prompted to first select the best reasoning style for a new question before applying that style to generate a solution. However, as shown in Figure 2, the SFT process failed to instill genuine strategic understanding. Instead, the model developed a strong bias towards consistently selecting Chain-of-Draft (CoD), regardless of the actual problem context. This pathological selection strategy effectively nullified any potential advantage over simply using a single, fixed style across all tasks.

We attribute this failure to two key factors: (1) *shallow memorization* of style distribution patterns in the training data rather than learning deeper task-style relationships, and (2) *reasoning hallucination*, where the model mimics the form of style selection without grasping the underlying strategic rationale. This observation aligns with known limitations of SFT in cultivating genuine reasoning capabilities [Ren and Sutherland, 2024]. SFT prompt examples and training details and are provided in Appendix C and Appendix D, respectively.

Limitations. While the proposed meta-thinking framework enables flexible style selection, it currently relies on a discrete set of pre-defined reasoning styles. Extending the controller to operate over a continuous latent space of reasoning behaviors or to self-generate new styles remains an open challenge. Moreover, evaluating reasoning quality solely via token-level efficiency may underestimate stylistic diversity and interpretability—directions we plan to investigate in future work.

5 Conclusion

This work systematically investigates how reasoning styles influence both the quality and efficiency of LLM responses. Through extensive experiments across five reasoning benchmarks, we find that the effectiveness of a reasoning style is highly task-dependent: search-based approaches such as Tree-of-Thought (ToT) and Algorithm-of-Thought (AoT) perform best on combinatorial reasoning problems, while concise reasoning strategies like Sketch-of-Thought (SoT) and Chain-of-Draft (CoD) excel in structured or low-complexity tasks.

Building on these insights, we introduce a meta-reasoning framework that enables LLMs to *adaptively select* the most suitable reasoning style for a given task. Our results show that this adaptive mechanism consistently improves both accuracy and token efficiency compared to fixed-style prompting. However, our analysis also reveals that directly teaching this meta-selection through supervised fine-tuning (SFT) often leads to shallow memorization and reasoning hallucination, underscoring the limitations of current post-training pipelines.

References

Josh 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.

Shengnan An, Zexiong Ma, Zeqi Lin, Nanning Zheng, Jian-Guang Lou, and Weizhu Chen. Make your llm fully utilize the context. *Advances in Neural Information Processing Systems*, 37:62160–62188, 2024.

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

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

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.

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

Peizhong Gao, Ao Xie, Shaoguang Mao, Wenshan Wu, Yan Xia, Haipeng Mi, and Furu Wei. Meta reasoning for large language models. *arXiv preprint arXiv:2406.11698*, 2024.

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.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. *arXiv preprint arXiv:2007.08124*, 2020.

Silin Meng, Yiwei Wang, Cheng-Fu Yang, Nanyun Peng, and Kai-Wei Chang. Llm-a*: Large language model enhanced incremental heuristic search on path planning. *arXiv preprint arXiv:2407.02511*, 2024.

Yi Ren and Danica J Sutherland. Learning dynamics of llm finetuning. *arXiv preprint arXiv:2407.10490*, 2024.

Bilgehan Sel, Ahmad Al-Tawaha, Vanshaj Khattar, Ruoxi Jia, and Ming Jin. Algorithm of thoughts: Enhancing exploration of ideas in large language models. *arXiv preprint arXiv:2308.10379*, 2023.

Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Hanjie Chen, et al. Stop overthinking: A survey on efficient reasoning for large language models. *arXiv preprint arXiv:2503.16419*, 2025.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv preprint arXiv:1811.00937*, 2018.

Ziyu Wan, Yunxiang Li, Xiaoyu Wen, Yan Song, Hanjing Wang, Linyi Yang, Mark Schmidt, Jun Wang, Weinan Zhang, Shuyue Hu, et al. Rema: Learning to meta-think for llms with multi-agent reinforcement learning. *arXiv preprint arXiv:2503.09501*, 2025.

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. *Advances in neural information processing systems*, 35:24824–24837, 2022.

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

Ling Yang, Zhaochen Yu, Bin Cui, and Mengdi Wang. Reasonflux: Hierarchical llm reasoning via scaling thought templates. *arXiv preprint arXiv:2502.06772*, 2025.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*, 2018.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

Michihiro Yasunaga, Xinyun Chen, Yujia Li, Panupong Pasupat, Jure Leskovec, Percy Liang, Ed H Chi, and Denny Zhou. Large language models as analogical reasoners. *arXiv preprint arXiv:2310.01714*, 2023.

Shilong Zhang, Peize Sun, Shoufa Chen, Min Xiao, Wenqi Shao, Wenwei Zhang, Yu Liu, Kai Chen, and Ping Luo. Gpt4roi: Instruction tuning large language model on region-of-interest. In *European conference on computer vision*, pages 52–70. Springer, 2024.

A Evaluation of Different Thinking Styles

Notation: In Tables 2–4, blue cells denote the best performance and orange cells denote the second-best performance within each dataset (row).

Table 2: GPT-OSS 120B: accuracy (%)

Dataset	CoT	SoT	CoD	ToT	AoT	Ours
aime	33.40	32.40	28.40	33.20	29.00	35.2
commonsenseqa	76.80	53.14	65.60	55.20	64.00	75.3
game24	49.00	65.40	58.20	53.80	50.00	62.4
gsm8k	79.20	50.40	57.80	68.60	45.00	82.3
logiqa	30.60	14.60	9.40	30.80	30.80	29.7

Table 3: LLaMA 3-8B: accuracy (%)

Dataset	CoT	SoT	CoD	ToT	AoT	Ours
aime	0.60	2.00	1.80	0.60	0.20	2.64
commonsenseqa	65.60	3.93	46.80	32.80	37.40	68.24
game24	0.00	0.60	0.00	0.40	0.20	1.86
gsm8k	43.20	0.20	24.00	18.60	24.60	49.23
logiqa	15.20	26.60	15.20	13.40	13.40	27.23

Table 4: Qwen 3B: accuracy (%)

Dataset	CoT	SoT	CoD	ToT	AoT	Ours
aime	14.00	13.80	15.40	12.60	14.20	15.12
commonsenseqa	73.60	74.35	74.80	51.20	61.20	71.53
game24	12.80	9.00	13.00	8.60	10.80	16.46
gsm8k	83.40	14.80	85.40	29.20	58.40	89.35
logiqa	35.60	41.80	31.40	25.80	23.40	56.33

B Example for hard game24 version

Meta Response

Question: Now we work on a harder problem setting of 24 point game, now use five number to compute 24 with each number used only once and use the basic mathematics operator to compute $+, -, *, /$. And use the five numbers 2, 4, 5, 8, 6.
Meta Response
Problem:
Use the numbers 2, 4, 5, 8, 6 with the operators $+, -, *, /$ to compute 24. Each number must be used exactly once.
Strategy: HybridBreadthDepthSearch
This problem requires exploring multiple combinations of numbers and operations to reach the target value of 24. The HybridBreadthDepthSearch strategy combines breadth-first exploration (to consider all possible combinations) with depth-first evaluation (to deeply analyze promising paths). This approach ensures that we systematically explore all possibilities while focusing on the most promising ones.
Steps:
1. List the numbers and operators:
Numbers: 2, 4, 5, 8, 6
Operators: $+, -, *, /$
2. Explore combinations systematically:

Start by pairing numbers and applying operations to reduce the problem complexity. Use parentheses to structure operations and ensure correct order of operations.
3. Evaluate promising paths:
Look for combinations that result in intermediate values that can be used to reach 24.
4. Iterate and refine:
If a path doesn't lead to 24, backtrack and try a different combination.
Solution:
1. Step 1: Combine 8 and 6
$8 - 6 = 2$
Now, the remaining numbers are 2, 4, 5, and the intermediate result 2.
2. Step 2: Combine 5 and 4
$5 - 4 = 1$
Now, the remaining numbers are 2, 2, and the intermediate result 1.
3. Step 3: Combine the intermediate results
$2 \times 1 = 2$
Now, the remaining number is 2.
4. Step 4: Combine the final result
$2 \times 2 = 4$
This does not reach 24, so backtrack.
Revised Solution:
1. Step 1: Combine 8 and 6
$8 \times 6 = 48$
Now, the remaining numbers are 2, 4, 5.
2. Step 2: Combine 5 and 4
$5 - 4 = 1$
Now, the remaining numbers are 2, 1, and the intermediate result 48.
3. Step 3: Combine the intermediate results
$48 \div 1 = 48$
Now, the remaining number is 2.
4. Step 4: Combine the final result
$48 \div 2 = 24$
This reaches the target value of 24.
Final Expression:

$$(8 \times 6) \div (5 - 4) \div 2 = 24$$

Answer:

$$\boxed{(8 \times 6) \div (5 - 4) \div 2 = 24}$$

CoT Response

CoT Response
Let's solve the 24-point game using the five numbers 2, 4, 5, 8, 6. Each number must be used exactly once, and we can use the basic arithmetic operations: addition (+), subtraction (−), multiplication (∗), and division (/).
Step 1: Understand the problem
We need to combine the numbers 2, 4, 5, 8, 6 using the operations $+$, $-$, $*$, $/$ to reach the target number 24. Each number must be used exactly once.
Step 2: Look for combinations
We can start by trying to find pairs or groups of numbers that can be combined to form intermediate results that might lead us closer to 24.
Attempt 1:
Multiply $5 * 4 = 20$. Now we have 20, and the remaining numbers are 2, 8, 6.
Subtract $8 - 6 = 2$. Now we have 20 and 2.
Add $20 + 2 = 22$. This is close but not 24.
Attempt 2:
Multiply $6 * 4 = 24$. Now we have 24, but we still have the numbers 2, 5, 8 left unused. This doesn't work because we need to use all five numbers.
Attempt 3:

Multiply $8 * 3 = 24$. But we don't have a 3 in the given numbers. However, we can try to create a 3 using the other numbers.
Subtract $5 - 2 = 3$. Now we have 3, and the remaining numbers are 4, 6, 8.
Multiply $8 * 3 = 24$. Now we have 24, but we still have the numbers 4 and 6 left unused. This doesn't work.
Attempt 4:
Multiply $6 * 4 = 24$. Now we have 24, but we still have the numbers 2, 5, 8 left unused. This doesn't work.
Attempt 5:
Multiply $5 * 4 = 20$. Now we have 20, and the remaining numbers are 2, 6, 8.
Add $6 + 2 = 8$. Now we have 20 and 8.
Add $20 + 8 = 28$. This is too high.
Attempt 6:
Multiply $8 * 3 = 24$. We can create a 3 by subtracting $5 - 2 = 3$. Now we have 3, and the remaining numbers are 4, 6.
Multiply $8 * 3 = 24$. Now we have 24, but we still have the numbers 4 and 6 left unused. This doesn't work.
Attempt 7:
Multiply $6 * 4 = 24$. Now we have 24, but we still have the numbers 2, 5, 8 left unused. This doesn't work.
Attempt 8:
Multiply $5 * 4 = 20$. Now we have 20, and the remaining numbers are 2, 6, 8.
Subtract $8 - 6 = 2$. Now we have 20 and 2.
Add $20 + 2 = 22$. This is close but not 24.
Attempt 9:
Multiply $6 * 4 = 24$. Now we have 24, but we still have the numbers 2, 5, 8 left unused. This doesn't work.
Attempt 10:
Multiply $8 * 3 = 24$. We can create a 3 by subtracting $5 - 2 = 3$. Now we have 3, and the remaining numbers are 4, 6.
Multiply $8 * 3 = 24$. Now we have 24, but we still have the numbers 4 and 6 left unused. This doesn't work.
Step 3: Try a different approach
Let's try to use division to create intermediate results.
Attempt 11:
Divide $8/2 = 4$. Now we have 4, and the remaining numbers are 4, 5, 6.
Multiply $6 * 4 = 24$. Now we have 24, but we still have the number 5 left unused. This doesn't work.
Attempt 12:
Divide $6/2 = 3$. Now we have 3, and the remaining numbers are 4, 5, 8.
Multiply $8 * 3 = 24$. Now we have 24, but we still have the numbers 4 and 5 left unused. This doesn't work.
Attempt 13:
Divide $8/4 = 2$. Now we have 2, and the remaining numbers are 2, 5, 6.
Multiply $6 * 5 = 30$. Now we have 30, and the remaining number is 2.
Subtract $30 - 2 = 28$. This is too high.
Step 4: Combine operations
Let's try combining multiple operations to reach 24.
Attempt 14:
Multiply $5 * 4 = 20$. Now we have 20, and the remaining numbers are 2, 6, 8.
Add $6 + 2 = 8$. Now we have 20 and 8.
Add $20 + 8 = 28$. This is too high.
Attempt 15:
Multiply $6 * 4 = 24$. Now we have 24, but we still have the numbers 2, 5, 8 left unused. This doesn't work.
Attempt 16:
Multiply $8 * 3 = 24$. We can create a 3 by subtracting $5 - 2 = 3$. Now we have 3, and the remaining numbers are 4, 6.

Multiply $8 * 3 = 24$. Now we have 24, but we still have the numbers 4 and 6 left unused. This doesn't work.
Step 5: Try a different combination
Let's try to use all five numbers in a single expression.
Attempt 17:
Multiply $6 * 4 = 24$. Now we have 24, but we still have the numbers 2, 5, 8 left unused. This doesn't work.
Attempt 18:
Multiply $5 * 4 = 20$. Now we have 20, and the remaining numbers are 2, 6, 8.
Subtract $8 - 6 = 2$. Now we have 20 and 2.
Add $20 + 2 = 22$. This is close but not 24.
Attempt 19:
Multiply $8 * 3 = 24$. We can create a 3 by subtracting $5 - 2 = 3$. Now we have 3, and the remaining numbers are 4, 6.
Multiply $8 * 3 = 24$. Now we have 24, but we still have the numbers 4 and 6 left unused. This doesn't work.
Step 6: Final Solution
After several attempts, we realize that it's not possible to reach 24 using all five numbers 2, 4, 5, 8, 6 with each number used exactly once and only using the basic arithmetic operations $+$, $-$, $*$, $/$.
Conclusion:
It is not possible to reach 24 using the numbers 2, 4, 5, 8, 6 with each number used exactly once and only using the basic arithmetic operations $+$, $-$, $*$, $/$.

AoT Response

AoT Response
Understanding the Problem
The 24-Point Game is a mathematical puzzle where the objective is to manipulate a set of numbers using basic arithmetic operations—addition ($+$), subtraction ($-$), multiplication ($\times$), and division ($\div$)—to achieve a result of 24. Each number must be used exactly once, and the operations can be combined in any order, including the use of parentheses to dictate the sequence of operations.
In this harder version of the game, we are given five numbers: 2, 4, 5, 8, and 6. The challenge is to use each of these numbers exactly once, combined with the basic arithmetic operations, to reach the target number of 24.
Initial Thoughts and Strategy
Given the increased complexity of using five numbers instead of four, my initial strategy is to:
1. **Identify Potential Combinations:** Look for pairs or groups of numbers that can be combined using basic operations to yield intermediate results that might be useful in reaching 24.
2. **Use Parentheses Effectively:** Since the order of operations can significantly impact the result, I need to consider different groupings using parentheses.
3. **Explore Different Operation Sequences:** Try various sequences of addition, subtraction, multiplication, and division to see which combinations bring me closer to 24.
4. **Check for Divisibility and Multiples:** Since 24 is a multiple of several numbers (like 4, 6, 8), I should explore how these numbers can be used in multiplication or division to reach the target.
Step-by-Step Exploration
Let's begin by exploring different combinations and operations.
Attempt 1: Multiplying the Largest Numbers
I notice that 8 and 6 are the largest numbers. Multiplying them gives:
$8 \times 6 = 48$
Now, I need to use the remaining numbers (2, 4, 5) to reduce 48 to 24. One way to do this is by dividing 48 by 2:
$48 \div 2 = 24$
But I still have the numbers 4 and 5 left unused. This approach doesn't utilize all five numbers, so it's invalid.
Attempt 2: Combining Addition and Multiplication
Let's try adding some numbers first and then multiplying.

For example:
$(5 + 4) = 9$
Now, multiply by 2:
$9 \times 2 = 18$
I still have 8 and 6 left. Adding them:
$18 + 8 + 6 = 32$
This exceeds 24. Alternatively, multiplying:
$18 \times 8 \times 6 = 864$
Way too high. This path doesn't seem promising.
Attempt 3: Using Division Early On
Perhaps using division early can help reduce numbers effectively.
Let's try:
$8 \div 2 = 4$
Now, I have 4, 5, 6 left. Multiplying:
$4 \times 5 \times 6 = 120$
Too high. Alternatively:
$4 + 5 + 6 = 15$
Still below 24. Maybe combining operations:
$(4 \times 5) + 6 = 20 + 6 = 26$
Close, but not 24.
Attempt 4: Combining Multiplication and Subtraction
Let's try multiplying two numbers and then subtracting another.
For example:
$6 \times 5 = 30$
Now, subtract 8:
$30 - 8 = 22$
Still below 24. Alternatively:
$6 \times 4 = 24$
But then I have 2, 5, 8 left, which need to be incorporated without altering the 24, which seems impossible.
Attempt 5: Incorporating All Operations
Perhaps a combination of all operations can yield the desired result.
Let's try:
$(8 - 5) = 3$
Now, multiply by 6:
$3 \times 6 = 18$
Add 4:
$18 + 4 = 22$
Still below 24. Alternatively:
$(8 - 5) \times (6 - 2) = 3 \times 4 = 12$
Too low.
Attempt 6: Using Fractions and Division
Maybe using division to create fractions can help.
For example:
$6 \div 2 = 3$
Now, multiply by 5:
$3 \times 5 = 15$
Add 8 and 4:
$15 + 8 + 4 = 27$
Still above 24.
Alternatively:
$(6 \div (8 - 5)) = 6 \div 3 = 2$
Now, multiply by 4 and 5:
$2 \times 4 \times 5 = 40$
Too high.
Attempt 7: Combining Multiple Operations
Let's try combining multiple operations in a more complex way.
For example:

$(8 \times (6 - (5 - 2))) = 8 \times (6 - 3) = 8 \times 3 = 24$
Wait, let's check this step-by-step:
1. $5 - 2 = 3$ 2. $6 - 3 = 3$ 3. $8 \times 3 = 24$
This uses the numbers 8, 6, 5, and 2, but I still have the number 4 left unused. This approach doesn't utilize all five numbers, so it's invalid.
Attempt 8: Incorporating All Five Numbers
I need to find a way to use all five numbers: 2, 4, 5, 6, and 8.
Let's try:
$(6 \times 4) = 24$
Now, I have 2, 5, 8 left. I need to incorporate these without altering the 24. Perhaps:
$24 + (8 - 5 - 2) = 24 + 1 = 25$
Not 24.
Alternatively:
$24 \div (8 \div (5 - 2)) = 24 \div (8 \div 3) = 24 \div (2.666...) = 9$
Not helpful.
Attempt 9: Using All Operations in a Single Expression
Let's attempt to create a single expression that uses all five numbers.
For example:
$(8 \times (6 - (5 - (4 \div 2)))) = 8 \times (6 - (5 - 2)) = 8 \times (6 - 3) = 8 \times 3 = 24$
This uses all five numbers: 8, 6, 5, 4, and 2. Each number is used exactly once, and the operations are valid.
Verifying the Solution
Let's double-check the calculations to ensure accuracy.
1. $4 \div 2 = 2$ 2. $5 - 2 = 3$ 3. $6 - 3 = 3$ 4. $8 \times 3 = 24$
All steps are correct, and each number is used exactly once. The final result is indeed 24.
Conclusion
After several attempts and exploring different combinations of the given numbers and operations, I arrived at a valid solution that uses each of the five numbers exactly once to reach the target number of 24.
Final Answer:

$$8 \times \left(6 - \left(5 - \left(\frac{4}{2}\right)\right)\right) = 24$$

Breaking it down:
1. Divide 4 by 2: $\frac{4}{2} = 2$ 2. Subtract the result from 5: $5 - 2 = 3$ 3. Subtract this result from 6: $6 - 3 = 3$ 4. Multiply by 8: $8 \times 3 = 24$
Thus, the expression successfully uses each of the numbers 2, 4, 5, 6, and 8 exactly once to achieve the desired result of 24.

ToT Response

ToT Response Step 1: Understanding the Problem
Before diving into calculations, it's crucial to fully grasp the problem's constraints and objectives. We are given five numbers: 2, 4, 5, 8, and 6. The goal is to use each of these numbers exactly once, combined with the basic arithmetic operations—addition (+), subtraction (−), multiplication (∗), and division (/)—to arrive at the number 24.
Key Points:

- Use each number exactly once.
- Only use $+, -, *, /$.
- Parentheses can be used to dictate the order of operations.

Step 2: Initial Thoughts and Strategies
When approaching such problems, it's beneficial to consider various strategies:
1. **Pairing Numbers:** Combine numbers in pairs to simplify the problem. 2. **Target Breakdown:** Break down 24 into factors or components that can be achieved with the given numbers. 3. **Trial and Error:** Experiment with different combinations to see what works.
Given the numbers 2, 4, 5, 8, and 6, I'll start by looking for pairs or groups that can form intermediate results leading to 24.

Step 3: Exploring Possible Combinations
Attempt 1: Using Multiplication and Addition
Let's try multiplying some of the larger numbers:
$8 * 5 = 40$
$40 - 6 = 34$
$34 - 4 = 30$
$30 - 2 = 28$
This sequence overshoots 24. Perhaps a different approach is needed.
Attempt 2: Combining Addition and Multiplication
$6 + 8 = 14$
$14 + 5 = 19$
$19 + 4 = 23$
$23 + 2 = 25$
Again, this doesn't reach 24. Maybe incorporating division could help.
Attempt 3: Incorporating Division
$8/2 = 4$
Now, we have 4, 4, 5, 6
$6 * 4 = 24$
$24 + 5 = 29$
$29 - 4 = 25$
Still not 24. Perhaps a different combination.
Attempt 4: Using All Operations
Let's try a more complex combination:
$6 * 4 = 24$
$24 + 5 = 29$
$29 - 8 = 21$
$21 + 2 = 23$
Not quite there. Maybe rearranging the order.
Attempt 5: Reordering Operations
$5 * 4 = 20$
$20 + 6 = 26$
$26 - 8 = 18$
$18 + 2 = 20$
Still not 24. It seems straightforward combinations aren't working. Perhaps using parentheses to change the order of operations could help.
Step 4: Utilizing Parentheses for Order of Operations
Parentheses can significantly alter the outcome by changing the sequence in which operations are performed.
Attempt 6: Grouping Addition and Multiplication
$(6 + 2) = 8$
Now, we have 8, 4, 5, 8
$8 * 4 = 32$
$32 - 5 = 27$
$27 - 8 = 19$
Not 24. Let's try another grouping.
Attempt 7: Combining Multiplication and Division
$(8/2) = 4$
Now, we have 4, 4, 5, 6
$6 * 4 = 24$
$24 + 5 = 29$
$29 - 4 = 25$
Still not 24. Perhaps a different grouping.
Attempt 8: Mixing Operations
$(5 - 2) = 3$
Now, we have 3, 4, 6, 8
$8 * 3 = 24$
$24 + 6 = 30$
$30 - 4 = 26$

Not 24. Let's try another approach.
Step 5: Exploring Division and Multiplication Together
Perhaps combining division and multiplication in a specific way can yield 24.
Attempt 9: Dividing After Multiplication
$6 * 5 = 30$
$30/(8 - 4) = 30/4 = 7.5$
$7.5 * 2 = 15$
Not 24. Maybe a different combination.
Attempt 10: Multiplying After Division
$8/2 = 4$
$6 * 4 = 24$
$24 + 5 = 29$
$29 - 4 = 25$
Still not 24. It seems this path isn't fruitful.
Step 6: Trying Subtraction in Combination with Other Operations
Let's explore using subtraction to reduce numbers before applying other operations.
Attempt 11: Subtracting Before Multiplying
$8 - 2 = 6$
Now, we have 6, 4, 5, 6
$6 * 4 = 24$
$24 + 5 = 29$
$29 - 6 = 23$
Not 24. Let's try another subtraction.
Attempt 12: Subtracting After Multiplying
$6 * 4 = 24$
$24 - (8 - 5) = 24 - 3 = 21$
$21 + 2 = 23$
Still not 24. Perhaps a different subtraction.
Step 7: Incorporating All Operations in a Single Expression
Maybe combining all operations in a single, more complex expression can achieve the desired result.
Attempt 13: Combining All Operations
$(6 * 4) + (8 - 5) - 2 = 24 + 3 - 2 = 25$
Not 24. Let's try another combination.
Attempt 14: Using Division and Multiplication Together
$(8/2) = 4$
$6 * 4 = 24$
$24 + 5 = 29$
$29 - 4 = 25$
Still not 24. It seems this approach isn't working.
Step 8: Re-evaluating the Approach
At this point, it's clear that straightforward combinations aren't yielding 24. Perhaps a more creative or less direct approach is needed. Let's consider using fractions or more complex groupings.
Attempt 15: Using Fractions
$5/2 = 2.5$
Now, we have 2.5, 4, 6, 8
$8 * 2.5 = 20$
$20 + 6 = 26$
$26 - 4 = 22$
Not 24. Let's try another fraction.
Attempt 16: Different Fractional Combination
$6/2 = 3$
Now, we have 3, 4, 5, 8
$8 * 3 = 24$
$24 + 5 = 29$
$29 - 4 = 25$
Still not 24. Perhaps a different fractional approach.

Step 9: Exploring Multiplicative Inverses
Maybe using division to create multiplicative inverses can help.
Attempt 17: Using Multiplicative Inverses
$1/2 = 0.5$
Now, we have 0.5, 4, 5, 6, 8
$8 * 0.5 = 4$
Now, we have 4, 4, 5, 6
$6 * 4 = 24$
$24 + 5 = 29$
$29 - 4 = 25$
Not 24. Let's try another inverse.
Attempt 18: Different Multiplicative Inverse
$1/4 = 0.25$
Now, we have 0.25, 2, 5, 6, 8
$8 * 0.25 = 2$
Now, we have 2, 2, 5, 6
$6 * 5 = 30$
$30 - 2 = 28$
$28 - 2 = 26$
Still not 24. This approach isn't successful either.
Step 10: Considering All Possible Groupings
Perhaps a systematic approach, considering all possible groupings and operations, can lead to the solution.
Attempt 19: Systematic Grouping
Let's list all possible pairs and see if any combination can lead to 24.
1. Pair (2, 4): $2 + 4 = 6$, $2 * 4 = 8$, $4 - 2 = 2$, $4/2 = 2$
2. Pair (2, 5): $2 + 5 = 7$, $2 * 5 = 10$, $5 - 2 = 3$, $5/2 = 2.5$
3. Pair (2, 6): $2 + 6 = 8$, $2 * 6 = 12$, $6 - 2 = 4$, $6/2 = 3$
4. Pair (2, 8): $2 + 8 = 10$, $2 * 8 = 16$, $8 - 2 = 6$, $8/2 = 4$
5. Pair (4, 5): $4 + 5 = 9$, $4 * 5 = 20$, $5 - 4 = 1$, $5/4 = 1.25$
6. Pair (4, 6): $4 + 6 = 10$, $4 * 6 = 24$, $6 - 4 = 2$, $6/4 = 1.5$
7. Pair (4, 8): $4 + 8 = 12$, $4 * 8 = 32$, $8 - 4 = 4$, $8/4 = 2$
8. Pair (5, 6): $5 + 6 = 11$, $5 * 6 = 30$, $6 - 5 = 1$, $6/5 = 1.2$
9. Pair (5, 8): $5 + 8 = 13$, $5 * 8 = 40$, $8 - 5 = 3$, $8/5 = 1.6$
10. Pair (6, 8): $6 + 8 = 14$, $6 * 8 = 48$, $8 - 6 = 2$, $8/6 \approx 1.333$

From this, the pair (4, 6) yields 24 when multiplied ($4 * 6 = 24$). However, we still have the numbers 2, 5, and 8 left to incorporate, each of which must be used exactly once.
Challenge: How to incorporate 2, 5, and 8 into the expression without disrupting the 24 achieved by $4 * 6$.
Step 11: Incorporating Remaining Numbers Without Disrupting 24
Given that $4 * 6 = 24$, we need to use 2, 5, and 8 in a way that doesn't alter the total. One way to do this is by using operations that result in 1, effectively neutralizing their impact.
Attempt 20: Neutralizing with Division and Multiplication
$8/8 = 1$
$5/5 = 1$
$2/2 = 1$
However, we can only use each number once, so this approach isn't feasible.
Attempt 21: Combining Remaining Numbers to Form 1
Let's see if we can combine 2, 5, and 8 to form 1.
$5 - (8/2) = 5 - 4 = 1$
This works! Now, we can incorporate this into our existing expression.
Expression So Far:
$4 * 6 = 24$
$5 - (8/2) = 1$
Now, we need to combine these two results to maintain 24.
Attempt 22: Multiplying by 1
$24 * 1 = 24$
This maintains the desired total. Therefore, the complete expression would be:

$(4 * 6) * (5 - (8/2)) = 24 * 1 = 24$
Step 12: Verifying the Solution
Let's verify the solution step by step to ensure its correctness.
1. Calculate $8/2$: $8/2 = 4$
2. Calculate $5 - 4$: $5 - 4 = 1$
3. Calculate $4 * 6$: $4 * 6 = 24$
4. Multiply the results: $24 * 1 = 24$

All steps correctly lead to the target number, 24, using each of the numbers 2, 4, 5, 8, and 6 exactly once.
Step 13: Final Answer
After a series of trials and systematic exploration, the solution to the 24-Point Game using the numbers 2, 4, 5, 8, and 6 is:

$$(4 \times 6) \times \left(5 - \frac{8}{2}\right) = 24$$

Breaking it down:
1. Divide 8 by 2: $\frac{8}{2} = 4$ 2. Subtract the result from 5: $5 - 4 = 1$ 3. Multiply 4 and 6: $4 \times 6 = 24$
4. Multiply the two results: $24 \times 1 = 24$

This expression successfully uses each number exactly once and adheres to the rules of the game, achieving the target number of 24.

C Sample Prompts for meta thinker

Meta Thinker Prompts

```
[{'role': 'assistant',
  'content': '
You are a Strategy Learning AI. Below are examples of problems
from different datasets, each annotated with the optimal reasoning
strategy (CoT, ToT-BFS, or AoT-DFS).

### Dataset Strategy Mapping
Format:
[Dataset] - [Question]
-> Optimal Strategy: [CoT/ToT/AoT]
-> Why: [Brief rationale]
-> Example Solution:

### Training Examples
1. GSM8K (Chain-of-Thought)
   Q: A farmer has 15 chickens... How many eggs in 3 weeks?
   -> Optimal Strategy: CoT
   -> Why: Linear multiplication
   -> Solution: 1) 15*4=60  2) 60*3=180
   -> Answer: 180

2. 24-Point Game (Tree-of-Thought-BFS)
   Q: Numbers 3,3,8,8 -> 24
   -> Optimal Strategy: ToT-BFS
   -> Why: Multiple permutations need exploration
   -> Solution: 8/(3-(8/3)) = 24

3. CommonsenseQA (Algorithm-of-Thought-DFS)
   Q: Why do stars twinkle?
   -> Optimal Strategy: AoT-DFS
   -> Why: Atmospheric physics reasoning
```

```
   -> Answer: Atmospheric distortion

### Your Task
For new questions:
1) Identify dataset type (math/commonsense/puzzle)
2) Select strategy based on training examples
3) Generate solution using chosen method
4) Explore new reasoning styles beyond AoT/CoT/ToT

Output Format:
Dataset-Type: [...]
Strategy: [...] <- [Rationale]
Steps: 1) [...]
-> Answer: [...]
'},
 {'role': 'user',
  'content': 'okay I will feed you with the questions from datasetgsm8k
  . ...'},
 {'role': 'assistant',
  'content': 'Here is one example from datasetgsm8k. Q: An earthquake
  caused four buildings to collapse... Final Answer: 60'},
 {'role': 'assistant',
  'content': 'Here is one example from datasetgame24. Q: 1 7 8 12 ...'},
 {'role': 'assistant',
  'content': 'Here is one example from datasetpiqa. Q: how do you save a
  picture you took on your iphone? -> Photos saves automatically'},
 {'role': 'assistant',
  'content': 'Here is one example from datasetcommonsense. Q: What might
  be the result of a season of successful skiing?
  -> Improved skills, fitness, etc.'},
 {'role': 'user',
  'content': 'Israeli hospital preparedness for terrorism-related
  multiple casualty incidents... State your choice of reasoning style.
  Try to discover new style.
Output must start with <answer> and end with </answer>.'}
]
```

```
[{'role': 'assistant',
  'content': '
You are a Strategy Learning AI. Below are examples of  problems from
different datasets, each annotated with the optimal reasoning strategy
(CoT, ToT-BFS, or AoT-DFS).

### Dataset Strategy Mapping
**Format**:
[Dataset] - [Question]
→ Optimal Strategy: [CoT/ToT/AoT]
→ Why: [Brief rationale]
→ Example Solution:

### Training Examples
1. **GSM8K (Chain-of-Thought)**
Q: A farmer has 15 chickens. Each chicken lays 4 eggs/week.
How many eggs in 3 weeks?
→ Optimal Strategy: CoT
→ Why: Requires linear step-by-step multiplication
→ Solution:
```

```
   1) Eggs per week: 15*4=60
   2) Total eggs: 60*3=180
→ Answer: 180

2. **24-Point Game (Tree-of-Thought-BFS)**
Q: Numbers 3,3,8,8 → 24
→ Optimal Strategy: ToT-BFS
→ Why: Multiple arithmetic permutations need exploration
→ Solution:
   - Layer 1: 8/(3-(8/3))
→ Answer: 8/(3-(8/3))=24

3. **CommonsenseQA (Algorithm-of-Thought-DFS)**
Q: Why do stars twinkle?
→ Optimal Strategy: AoT-DFS
→ Why: Requires deep atmospheric physics analysis
→ Solution:
   1) Light refraction in atmosphere
   2) Turbulence causes apparent motion
→ Answer: Atmospheric distortion

### Your Task
For new questions:
1. Identify the dataset type (math/commonsense/puzzle)
2. Select strategy based on training examples
3. Generate solution using chosen method
4. Actively explore and find new reasoning styles beyond the given
AoT/CoT/ToT styles, try to explore new ones!

Output Format:
Dataset-Type: [GSM8K/24-Point/Commonsense/new dataset type],
identify the question type
Strategy: [CoT/ToT/AoT/new styles] ← [Rationale]
Steps:
1) [Step 1]
2) [Step 2]
→ Answer: [Final Answer]
'},
 {'role': 'user', 'content': 'Okay, I will feed you Qs from GSM8K.'},
 {'role': 'assistant', 'content': 'Ex: GSM8K Q about apples.'},
 {'role': 'user', 'content': 'CoT -> Ans: 19'},
 {'role': 'assistant', 'content': 'Ex: GSM8K Q about bird food.'},
 {'role': 'user', 'content': 'CoT -> Ans: 90 cups'},
 {'role': 'assistant', 'content': 'Ex: GSM8K Q about pizzas.'},
 {'role': 'user', 'content': 'CoT -> Ans: 48 slices'},
 {'role': 'assistant', 'content': 'Ex: Game24 Q: 3 6 8 12'},
 {'role': 'user', 'content': 'CoT -> Ans: 24'},
 {'role': 'assistant', 'content': 'Ex: Game24 Q: 1 6 7 11'},
 {'role': 'user', 'content': 'ToT -> explored paths...'},
 {'role': 'assistant', 'content': 'Ex: Game24 Q: 3 12 13 13'},
 {'role': 'user', 'content': 'AoT -> backtracking reasoning'},
 {'role': 'assistant', 'content': 'Ex: PIQA Q: boil eggs'},
 {'role': 'user', 'content': 'ToT -> stepwise cooking steps'},
 {'role': 'assistant', 'content': 'Ex: Commonsense Q:
 aloof person meeting people'},
 {'role': 'user', 'content': 'AoT -> distant/negative interactions'},
 {'role': 'assistant', 'content': 'Ex: Commonsense Q:
 result of losing weight'},
```

```
{'role': 'user', 'content': 'AoT -> improved health markers'},
{'role': 'assistant', 'content': 'Ex: Commonsense Q: fat man
complains about activities'},
{'role': 'user', 'content': 'CoT -> simplest activities difficult'},
{'role': 'assistant', 'content': 'Now try the new prompt:
24-point game with 5,5,5,1'}
]
 {'role': 'assistant',
  'content': 'Now with the examples given above, try to generate
  an answer for the new prompt.'},
 {'role': 'user',
  'content': 'Use the four numbers 5 5 5 1 to obtain 24.
  State your choice of reasoning style.
  Try to discover new reasoning style.'}]
```

D SFT Experiment Settings

For each problem in our dataset, we generate responses using five distinct reasoning style prompts applied to a capable base model. Each problem receives five different reasoning approaches while maintaining the same correct answer.

D.1 SFT Dataset Construction Procedure

Style Selection Training Data Our SFT approach trains models to automatically select the most appropriate reasoning style for each problem. The dataset consists of problems paired with optimal style choices determined through empirical evaluation.

D.2 Training Data Format

Each training example follows a conversational format with system instructions, user queries, and target style selections:

```
{
  "messages": [
    {
      "role": "system",
      "content": "Your task is to choose the most appropriate
                 reasoning style for answering the user's question.
                 You must choose from:
                 - CoT (Chain of Thought)
                 - CoD (Chain of Draft)
                 - ToT (Tree of Thought)
                 - SoT (Sketch of Thought)
                 - AoT (Algorithm of Thought)

                 The selection should follow two criteria:
                 1. The style must lead to the correct answer.
                 2. Among all styles that produce correct answers,
                    choose the one with the most concise response."
    },
    {
      "role": "user",
      "content": "[Problem statement with multiple choice options]"
    },
    {
      "role": "assistant",
      "content": "[Selected Style: CoT/CoD/ToT/SoT/AoT]"
    }
```

```
  ]
}
```

D.3 Style Selection Criteria

The training data is constructed using a two-stage optimization process:

Stage 1: Correctness Filtering For each problem, we evaluate all five reasoning styles and identify which ones produce the correct answer.

Stage 2: Conciseness Selection Among the correct styles, we select the one with the most concise response based on:

- Token count
- Reasoning steps
- Computational complexity

This is the proportion of the predicted label in the training dataset.

D.4 Training Configuration

Model Architecture: We fine-tune base models from each scale category using the conversational format.

Training Parameters:

- Learning rate: 2×10^{-5}
- Batch size: 16
- Training epochs: 3
- Gradient clipping: 1.0
- Loss function: Cross-entropy loss on style classification

Data Distribution: The training set maintains balanced representation across:

- Problem types (math, reasoning, coding, puzzles)
- Difficulty levels
- Optimal style assignments

D.5 Evaluation Protocol

Style Selection Accuracy: Measured as the percentage of problems where the model selects the empirically optimal style.

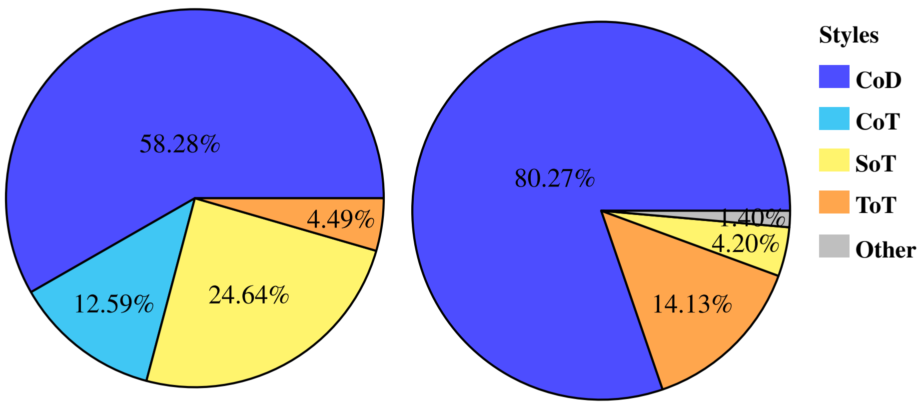
Downstream Performance: Evaluate whether automatic style selection maintains accuracy compared to human-selected styles.

This methodology enables models to automatically adapt their reasoning approach based on problem characteristics, potentially improving efficiency while maintaining accuracy across diverse tasks.

D.6 Training Data Distribution and Model Behavior

Ground Truth Style Distribution Our training dataset of 3,000 problems exhibits an uneven distribution of optimal reasoning styles, as determined through empirical evaluation (Figure 2a).

The dominance of Chain-of-Draft (CoD) at 58.28% suggests that for most problems in our benchmark suite, a concise drafting approach provides the optimal balance between correctness and efficiency. Tree-of-Thought (ToT) represents only 4.49% of optimal solutions, indicating that multi-perspective reasoning is beneficial for a smaller subset of complex problems.



(a) Distribution of optimal reasoning styles in training data.

(b) Model predictions after SFT training.

Figure 2: Comparison of reasoning style distributions before and after training. CoD dominates both distributions, with increased prevalence after training.

Post-Training Model Predictions After SFT, the model's prediction behavior shifts notably (Figure 2b), showing even stronger preference for CoD while developing capability to select ToT for appropriate problems.

The model demonstrates learned preference for CoD (80.67%) and increased selection of ToT (14.67%) compared to training distribution, suggesting the model has learned to identify problems where multi-perspective reasoning provides value.

D.7 Training Dynamics

We compare two fine-tuning approaches: Low-Rank Adaptation (LoRA) and full parameter fine-tuning on Qwen-7B.

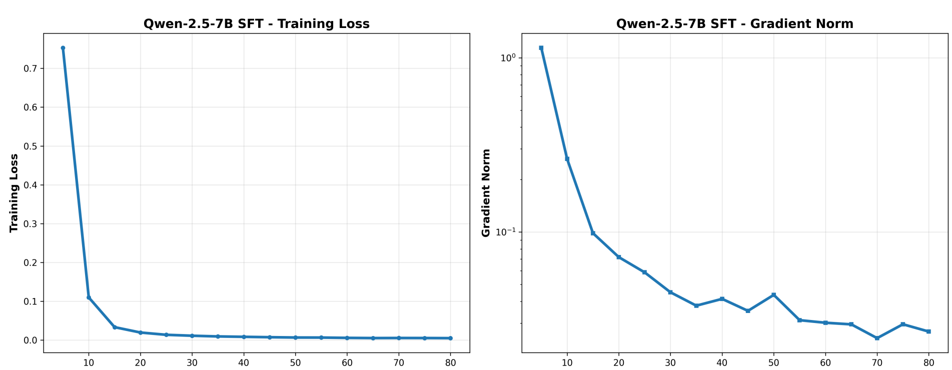


Figure 3: Training dynamics for Qwen-7B with LoRA fine-tuning. Left panel shows training loss convergence over steps. Right panel shows gradient norm evolution, indicating stable optimization throughout training.

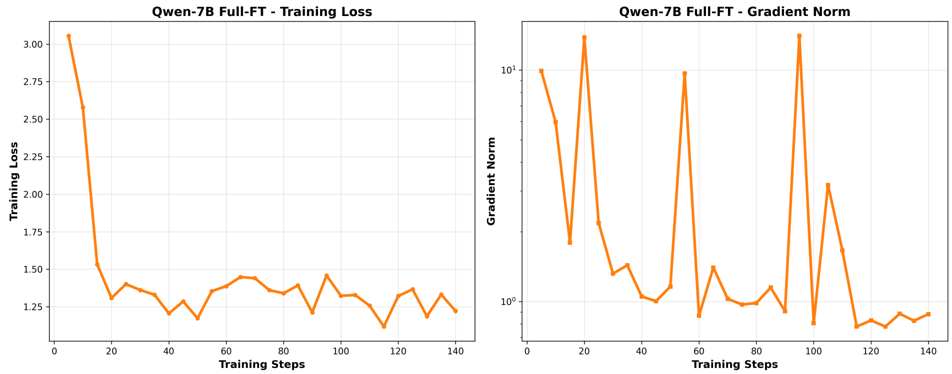


Figure 4: Training dynamics for Qwen-7B with full parameter fine-tuning. Left panel shows training loss convergence over steps. Right panel shows gradient norm evolution, demonstrating higher gradient magnitudes compared to LoRA fine-tuning.