Socratic-MCTS: Test-Time Visual Reasoning by Asking the Right Questions

David Acuna[†♣] **Ximing Lu**[†‡♣] **Jaehun Jung**[†‡♣] **Hyunwoo Kim**[†♣]
Amlan Kar[†§] **Sanja Fidler**[†§] **Yejin Choi**[†]

[†]NVIDIA [‡]University of Washington [§]University of Toronto

{dacunamarrer, ximingl, jaehunj, hyunwook, yejinc}@nvidia.com

Abstract

Recent research in vision-language models (VLMs) has centered around the possibility of equipping them with implicit long-form chain-of-thought reasoning—akin to the success observed in language models—via distillation and reinforcement learning. But what about the non-reasoning models already trained and deployed across the internet? Should we simply abandon them, or is there hope for a search mechanism that can elicit hidden knowledge and induce long reasoning traces—*without* any additional training or supervision? In this paper, we explore this possibility using a Monte Carlo Tree Search (MCTS)-inspired algorithm, which injects subquestion–subanswer pairs into the model's output stream. We show that framing reasoning as a search process—where subquestions act as latent decisions within a broader inference trajectory—helps the model "connect the dots" between fragmented knowledge and produce extended reasoning traces in non-reasoning models. We evaluate our method across three benchmarks and observe consistent improvements. Notably, our approach yields a 2% overall improvement on MMMU-PRO, including a significant 9% gain in Liberal Arts.

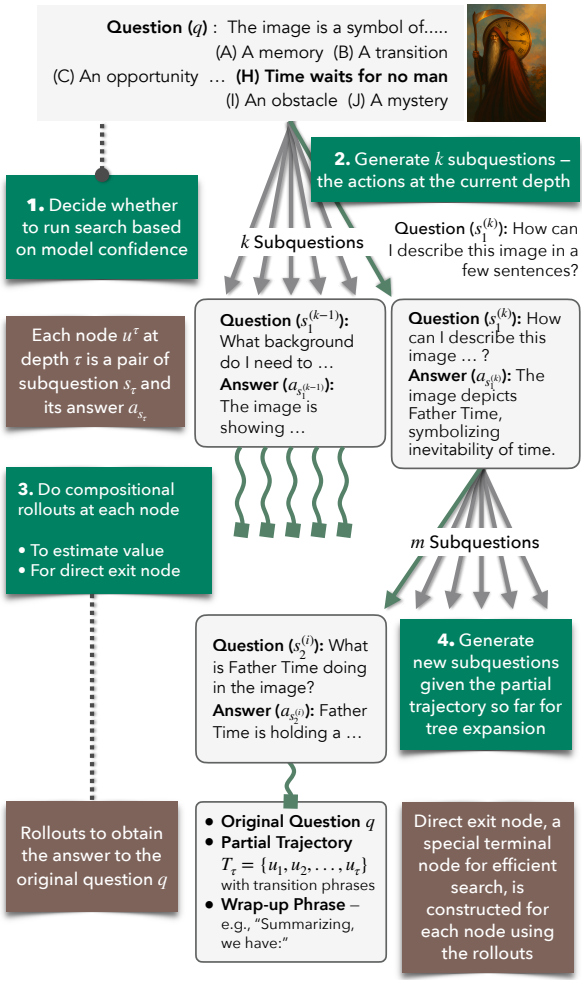


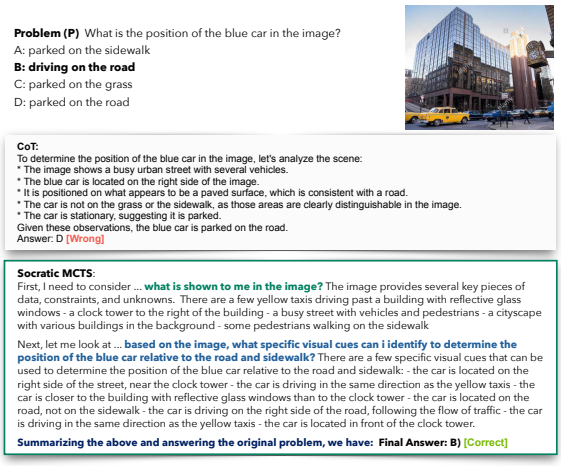
Figure 1: **Socratic-MCTS Overview.** In Socratic-MCTS, *actions* are defined as *subquestions*, and each *node state* consists of a *subquestion–subanswer* pair. During search, rollouts are performed by preconditioning the model on the accumulated reasoning trajectory in a compositional manner. To structure this trajectory and enable faster rollouts, we use *transition phrases* (e.g., "First, I need to consider...") and conclude with a *wrap-up phrase* (e.g., "Summarizing, we have:"), which cue the model to complete the reasoning and produce a final answer. We estimate value through internal agreement and incorporate early-exit and selective search mechanisms to adaptively reduce computational overhead—all without external supervision.

1 Introduction

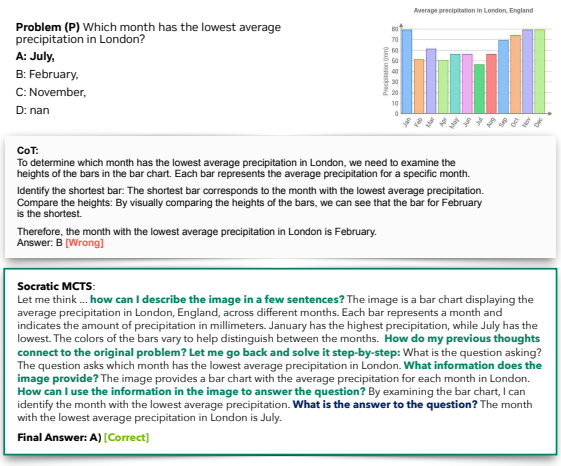
Recent state-of-the-art large language models (LLMs) have demonstrated remarkable capabilities in complex reasoning tasks (Chen et al., 2024b; DeepSeek-AI et al., 2025; OpenAI et al., 2024). A key driver of this progress has been the emergence of long chain-of-thought (CoT) reasoning through post-training with techniques such as reinforcement learning and distillation (DeepSeek-AI et al., 2025; Lu et al., 2025; Muennighoff et al., 2025). Inspired by this success, much of the recent focus in vision-language modeling has shifted toward mirroring these advances, attempting to induce reasoning capabilities primarily through reinforcement learning

♣ First co-authors.

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 24159–24172
November 4-9, 2025



(a) Socratic-MCTS on a logical reasoning question.



(b) Socratic-MCTS on a chart question.

Figure 2: **Qualitative comparison on MMStar.** We show Socratic-MCTS responses on two multimodal questions from the benchmark, comparing qualitatively against the *CoT prompting* baseline. Socratic-MCTS allows the model to uncover relevant knowledge, verify intermediate steps, and synthesize final answers coherently.

and distillation (Du et al., 2025; Liao et al., 2025).

But what about the many existing non-reasoning VLMs already trained and publicly available? As of this writing, the vast majority of open-weight VLMs fall into this category (Chen et al., 2024a; Liao et al., 2025; Bai et al., 2025). Should we simply abandon them, or is there hope for mechanisms that can repurpose these models—eliciting structured reasoning without additional training or supervision? While at first glance this may seem unfeasible, these models have been trained on vast internet-scale data and may possess latent knowledge and reasoning potential that conventional prompting fails to activate. Can we unlock reasoning in non-reasoning VLMs by eliciting hidden, belated knowledge inaccessible through standard "step-by-step" CoT prompting?

In this work, we propose **Socratic-MCTS**, a test-time algorithm that frames reasoning in non-reasoning models as a structured search problem—requiring no fine-tuning, no supervision, and no architectural modifications. At its core, Socratic-MCTS introduces a simple but powerful abstraction: the subquestion–subanswer pair. By framing reasoning as a search process where subquestions represent latent decisions within a broader inference trajectory, we enable models to "connect the dots" between fragmented pieces of knowledge. By searching over semantically meaningful chunks, rather than individual lines, Socratic-MCTS strikes a middle ground between free-form generation and conventional tree search—producing coherent long reasoning traces that progressively move toward the final solution. Our key insight is that longer CoT could emerge at test-time from the deliberate exploration of which subquestions to ask, and in what order—injecting structure into the model's output stream to guide it toward correct answers. Socratic-MCTS formalizes this process through an MCTS-inspired framework that estimates and incorporates early-exit mechanisms to adaptively reduce computational overhead —all without relying on external supervision. We evaluate Socratic-MCTS across three benchmarks and find that it consistently improves performance, notably achieving a significant 9% gain in Liberal Arts categories on the MMMU-PRO benchmark.

2 Preliminaries

Consider a model $\mathcal{M}$ that, when given a question $q := \{I, \hat{q}\}$, where I is an image and $\hat{q}$ is a text prompt of a multiple-choice question, generates both an intermediate reasoning **trajectory** T and a final solution a. Formally, given an input question $q \in \mathcal{Q}$, the model $\mathcal{M}$ produces a pair $(T, a) := \mathcal{M}(q)$, where $T \in \mathcal{T}$ denotes the chain of reasoning (or "thoughts"), and $a \in \mathcal{A}$ denotes the final answer to q. Each reasoning trajectory T consists of a **set of core units of reasoning**, denoted $T := \{u^1, u^2, \ldots, u^\tau\}$. Each u^τ represents an individual reasoning step. In most settings, u^τ corresponds to a sentence or logical operation—e.g., a mathematical step—and is often delineated in model outputs via sentence colons.

In our framework, each atomic reasoning unit is represented as $u^\tau := \{s_\tau, a_{s\tau}\}$, where s_τ is a subquestion and $a_{s\tau}$ is the corresponding answer. To facilitate segmentation between reasoning units, we append **transition phrases**—natural language cues such as "First, let me think ..." at the beginning of each thought u^i. These structure the trajectory allowing coherent composition.

Monte Carlo Tree Search Monte Carlo Tree Search (MCTS) is a sample-based tree-search algorithm that has proved effective on complex reasoning tasks. In our case, MCTS operates on a rooted, directed tree whose nodes correspond to partial reasoning trajectories. We represent a node at depth τ as $\mathbf{n}_\tau := \langle q,\ T_\tau,\ N,\ W,\ Q \rangle$ where (i) q is the original query, (ii) $T_\tau = \{u^1, \ldots, u^\tau\}$ is the sequence of core reasoning units generated so far (with $u^0 := \varnothing$ for the root), (iii) $N \in \mathbb{N}$ is the visit count, (iv) $W \in \mathbb{R}$ is the cumulative reward returned by completed roll-outs that passed through the node, and (v) $Q := W/N$ is its empirical value estimate. An edge from $\mathbf{n}_\tau$ to $\mathbf{n}_{\tau+1}$ is defined by an action. MCTS operates in four iterative phases: selection, expansion, simulation, and backpropagation. These are repeated until a predefined computational budget is reached. We refer the reader to Browne et al. (2012) for details.

3 Socratic MCTS

We adapt MCTS to a *Socratic* setting in which *actions* are **subquestions** and each **node state** is the pair $u^\tau := \{s_\tau, a_{s\tau}\}$ comprising the current subquestion and its answer. This is inspired by the Socratic method of reasoning, as the model explicitly and progressively decomposes the problem by asking and answering intermediate questions. Thus, unlike previous MCTS approaches that implicitly sample tokens or steps as actions, we explicitly define subquestions as structured dynamic actions, aligning MCTS with the Socratic method (Farnsworth, 2021). We emphasize that our algorithm operates at test time and does not rely on supervision. Instead, it leverages compositional rollouts and internal agreement to guide the search.

3.1 Explicit Subquestions as Actions

Previous work on applying MCTS to LLMs typically leaves the action space implicit—the next token or thought sampled from the model (Guan et al., 2025; Luo et al., 2024). Instead, we define an action at depth τ as a self-contained subquestion $s_{\tau+1}$ that (i) relates to the original query q and (ii) decomposes the task into a tractable sub-problem. This explicit representation allows searching over semantically meaningful pieces of knowledge while preserving goal-directedness. Specifically, we obtain a finite number of subquestions k_q: $\{s_{\tau+1}, \ldots, s_{\tau+k_q}\} \sim \mathcal{M}_s(\cdot \mid q, T_\tau, \mathrm{p}_{\text{sub}})$, where $\mathcal{M}_s$ is the *subquestion policy* and the prompt p_{sub} instructs the model to *ask* rather than *answer*. Each sampled subquestion defines an edge from $\mathbf{n}_\tau$ to a new child $\mathbf{n}_{\tau+1}$. The corresponding answer is obtained from an *answer policy*: $a_{s\tau+1} \sim \mathcal{M}_a(\cdot \mid I, s_{\tau+1})$, which answers the subquestion *in isolation*. Empirically, we found this decoupling crucial in multimodal models, preventing propagating errors or contaminating the answer. The new node's state is then $u^{\tau+1} := \{s_{\tau+1}, a_{s\tau+1}\}$. We make the transition from u^τ to $u^{\tau+1}$ natural by appending a transition phrase such as "Next, let me ..." drawn from a list.

For simplicity, we set $\mathcal{M}_s = \mathcal{M}_a = \mathcal{M}$ in our experiments, and set $temp = 0.6$, deferring the exploration of heterogeneous or multi-agent policies within the Socratic MCTS framework to future work. Notably, any agent—including a different model or a human—could be placed in either role.

3.2 Navigating the Reasoning Tree

Guided Selection via UCT. Starting at the root $\mathbf{n}_0$, we repeatedly choose the child that maximises $\mathrm{UCT}(\mathbf{n}_{\tau+1}) = Q + c\sqrt{\frac{\ln N_{\text{parent}}}{N_{\tau+1}}}$, until a leaf is reached. The constant $c > 0$ controls the exploration–exploitation balance. Following common practice (Kocsis and Szepesvári, 2006), we set the exploration constant $c = 1.4$.

Expanding with Socratic Questions. If the current leaf node is non-terminal and not fully expanded, we generate up to k_q new sub-questions and their answers as described above, initializing each child with $N = W = 0$. A node is considered *fully expanded* when all of its sampled subquestions have been explored (i.e., all available actions have been taken). A node is considered *terminal* if one of two conditions holds: (1) the model fails to generate an answer for a proposed action (e.g., empty answer), or (2) the generation degenerates (e.g., repetitive outputs). In all experiments, $k_q = 6$ for the first tree level and $k_q = 3$ for the rest.

Compositional Rollouts for Self-consistency. We perform rollouts to estimate a node's value by composing transition phrases, subquestions, and their

answers along the reasoning path. Concretely, we precondition the model on the current partial trajectory $T_{\tau+1}$. We additionally concatenate a *wrap-up transition phrase* (e.g., "Summarizing, we have:") that cues the model to complete the reasoning and produce a final answer. Formally, we generate K completions by preconditioning the model on $T_{\tau+1}$ followed by K distinct wrap-up transition phrases: $a^{(1)}, \ldots, a^{(K)} \sim \mathcal{M}\big(\cdot \mid q, T_{\tau+1}, \text{wrap}^{(k)}\big)$, where $\text{wrap}^{(k)}$ denotes the k-th natural language wrap-up phrase. Notably, this procedure enables efficient rollouts—requiring only a few newly generated tokens—as the model is preconditioned to produce a final response. We also observe that diverse wrap-up phrases yield variability in the response, which is necessary for computing internal agreement. We set $K = 8$ in all our experiments.

Internal Agreement for Value Estimation. We rely on the model's internal consistency as a proxy reward signal. Specifically, each of the sampled answer $a^{(k)}$ is parsed using a lightweight heuristic that extracts a canonical choice (e.g., a multiple-choice label), and penalizes overly verbose generations. Formally, let the set of unique extracted answers from all rollouts be $\mathcal{A} := \{\hat{a}^{(1)}, \ldots, \hat{a}^{(K)}\}$, and let $w^{(k)} := \text{score}(a^{(k)}) \in [0, 1]$ be a normalized confidence weight assigned by the heuristic. For example, if $a^{(k)}$ is degenerate—i.e., it reaches the maximum generation length without producing a valid answer—then its weight $w^{(k)} = 0$. We compute the value estimate via weighted majority voting: $V := \arg\max_{a \in \mathcal{A}} \sum_{k=1}^{K} \mathbf{1}\big[\hat{a}^{(k)} = a\big] \cdot w^{(k)}$. That is, we select the answer that accumulates the highest total confidence across the K rollouts.

Special Direct-Exit Nodes. While we encourage the algorithm to explore deeper chains of reasoning when beneficial, we also want to allow it to finalize an answer early if further composition fails to improve the value estimate. To support this, we allow each node to optionally include a *direct exit node*. This is a special terminal node that includes the terminal answer a in the reasoning trajectory—obtained from compositional rollouts at that node as explained above. To mitigate an unintended bias introduced by this special node, we omit the exploration term when computing the UCT score on them.

Selective Search Based on Model Confidence. We observe that for certain problems, the model exhibits high confidence in its initial direct answer obtained through vanilla sampling. To handle these cases efficiently, we first estimate the model's confidence in its initial answer and introduce a hyperparameter-controlled early-exit threshold that skips the tree-search algorithm altogether when exceeded (we used 0.9). We explore two approaches for confidence estimation: taking the maximum answer token probability and performing majority voting through sampling. Empirically, we did not observe a major difference between them. Future work might explore other adaptive schemes.

Method	MMMU-Pro			MMStar
	Liberal Arts	STEM+B	Overall	Overall
GPT-4o (051324)	-	-	0.540	0.647
Claude 3.5-Sonnet	-	-	0.550	0.651
Gemini-1.5-Pro	-	-	0.494	0.591
LLaVA-OneVision-72B	-	-	0.380	0.658
Qwen2-VL-72B	-	-	0.492	0.683
InternVL2-Llama3-76B	-	-	0.419	0.674
InternVL2.5-78B				
+ Direct	0.538	**0.507**	0.517	0.692
+ CoT	0.544	0.479	0.506	0.689
+ Least-to-Most	0.296	0.276	0.280	0.486
+ Socratic MCTS (Ours)	**0.628**	0.492	**0.537**	**0.711**

Table 1: Performance across different reasoning benchmarks. Socratic MCTS consistently outperforms direct, CoT and LtM baselines across all benchmarks. We evaluate InternVL-78B Direct, CoT, LtM. Others results are for reference taken from their respective reports. As such, entries not reported by the authors appear as -.

4 Experiments

Setup. We evaluate InternVL-78B (Chen et al., 2024d) using default hyperparams, and compare it against non-reasoning SoTA models GPT-4o (240513) and Claude 3.5-Sonnet, with results from (Chen et al., 2024d). We benchmark three baselines: (1) Direct Answer, (2) CoT prompting, using specific prompts from (Yue et al., 2024b) and greedy decoding. We further implemented a multimodal version of Least-To-Most (LtM). For LtM, we recursively decompose the original question up to a depth of 3 and solve the resulting subquestions in a bottom-up fashion. We evaluate on MMMU-PRO (Yue et al., 2024b), a more challenging variant of MMMU (Yue et al., 2024a), under a 10-option multiple-choice format. We also assess performance on MMStar (Chen et al., 2024c) (1,500 visually grounded, leakage-controlled samples) and MathVista (Lu et al., 2023) (English-only, multiple-choice subset of `test-mini`) for visual math reasoning. We report single runs with 40 iterations for MMMU-PRO and 20 for others.

Main Results. Socratic-MCTS consistently outperforms all baselines (Tab 1 and 2) , with strong

Method	MathVista (mini-eng)
+ Direct	0.740
+ CoT	0.763
+ Least-to-Most	0.471
+ **Socratic MCTS (Ours)**	**0.782**

Table 2: Performance on MathVista mini further filtered to include only multiple-choice questions in English.

gains in tasks requiring less symbolic reasoning. In MMMU-Pro, it notably improves accuracy by 9% in Liberal Arts. We define *Liberal Arts* as Art, Art Theory, Design, Economics, Geography, History, Literature, Music, Psychology, and Sociology, and all remaining subjects as *STEM+B*. Surprisingly, across all benchmarks, we find that in VLMs, decomposition via prompting —as in LtM— underperforms both direct and CoT, underscoring the limitations of prompting and the fundamental differences between non-reasoning VLMs and LLMs.

5 Additional Results

Additional Model. In the supplementary material, Table 3, we show the experimental results of using Socratic-MCTS on an additional model, Llama-3.2-90B-vision. As with InternVL-78B, we observe consistent notable gains of 2% on MMMU-Pro and 1.5% on MMStar.

Comparison with Self-Consistency (majority voting). In Table 4 supplementary material, we also report InternVL-2.5-78B accuracy on MMMU-PRO using (i) sampling-based CoT + self-consistency with different numbers of generations and (ii) our Socratic MCTS. Socratic MCTS outperforms CoT + self-consistency in all the experiments highlighting the strength of our search-based decomposition. We emphasize a majority-vote layer could just as easily be placed on top of Socratic MCTS. We leave this promising extension of Socratic MCTS for future work.

6 Related Work

Prior work has explored question decomposition for LLMs as a direct prompting method (Radhakrishnan et al., 2023; Zhou et al., 2022; Khot et al., 2022; Jung et al., 2022; Liao et al., 2024a,b), mainly in text-only domains. In contrast, we integrate decomposition into a tree search process, allowing models to explore and compose subquestions dynamically, with a focus on vision. Recent adaptations of MCTS to VLMs (Yao et al., 2024; Wu et al., 2025; Wang et al., 2025; Xie et al., 2024) either use MCTS as part of a training loop or in a small dataset to generate high-level reasoning templates. Closer to our setting, Hao et al. (2023) use MCTS and frame subquestions as actions in math tasks. Our work differs in both domain and methodology: we target non-reasoning VLMs, introduce preconditioning in rollouts and early-exit mechanisms, and operate in the multimodal domain—producing long reasoning traces that require grounded visual understanding. Concurrently, with our work (Zhang et al., 2025) proposes an MCTS-style test-time algorithm and define a "step" as a sub-question plus its answer. While both approaches employ an MCTS-style test-time algorithm, several key differences remain in terms of generation strategy, reward and efficiency and scope of the task.

7 Conclusions

We introduce Socratic-MCTS, a test-time algorithm that frames reasoning as structured search over subquestion–subanswer pairs. Evaluations across multiple benchmarks show consistent performance gains, particularly on non-symbolic tasks—validating that fragmented knowledge can be elicited to produce long reasoning traces in "non-reasoning" VLMs, without additional training.

8 Limitations

While our approach demonstrates promising improvements, it comes with several limitations. First, non-autoregressive reasoning is inherently less GPU-efficient, making structured search methods like MCTS slower in practice. Without significant advances in compute efficiency, such structured approaches may remain less practical than simpler methods like majority voting with brute-force scaling. Second, although internal agreement serves as a useful value signal in the absence of supervision, we observed that frozen VLMs tend to be overconfident in their outputs—often ignoring chain-of-thought cues, regardless of how they are preconditioned. Encouraging faithfulness of the CoT and output diversity in such models remains an open research challenge. Finally, due to the computational cost of search, we did not explore hyperparameter tuning and multi-agent settings. These choices may offer further performance gains and warrant future investigation.

References

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, and 8 others. 2025. Qwen2.5-vl technical report. *Preprint*, arXiv:2502.13923.

Cameron B Browne, Edward Powley, Daniel Whitehouse, Simon M Lucas, Peter I Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. 2012. A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in games*, 4(1):1–43.

Guiming Hardy Chen, Shunian Chen, Ruifei Zhang, Junying Chen, Xiangbo Wu, Zhiyi Zhang, Zhihong Chen, Jianquan Li, Xiang Wan, and Benyou Wang. 2024a. Allava: Harnessing gpt4v-synthesized data for a lite vision-language model. *Preprint*, arXiv:2402.11684.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Conghui He, Jiaqi Wang, Feng Zhao, and Dahua Lin. 2024b. Sharegpt4v: Improving large multi-modal models with better captions. In *European Conference on Computer Vision*, pages 370–387. Springer.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, and 1 others. 2024c. Are we on the right way for evaluating large vision-language models? *arXiv preprint arXiv:2403.20330*.

Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, and 1 others. 2024d. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling. *arXiv preprint arXiv:2412.05271*.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, and 181 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Yifan Du, Zikang Liu, Yifan Li, Wayne Xin Zhao, Yuqi Huo, Bingning Wang, Weipeng Chen, Zheng Liu, Zhongyuan Wang, and Ji-Rong Wen. 2025. Virgo: A preliminary exploration on reproducing o1-like mllm. *Preprint*, arXiv:2501.01904.

Ward Farnsworth. 2021. *The Socratic method: a practitioner's handbook*. Godine Boston.

Xinyu Guan, Li Lyna Zhang, Yifei Liu, Ning Shang, Youran Sun, Yi Zhu, Fan Yang, and Mao Yang. 2025. rstar-math: Small llms can master math reasoning with self-evolved deep thinking. *arXiv preprint arXiv:2501.04519*.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. 2023. Reasoning with language model is planning with world model. *arXiv preprint arXiv:2305.14992*.

Jaehun Jung, Lianhui Qin, Sean Welleck, Faeze Brahman, Chandra Bhagavatula, Ronan Le Bras, and Yejin Choi. 2022. Maieutic prompting: Logically consistent reasoning with recursive explanations. *Preprint*, arXiv:2205.11822.

Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. 2022. Decomposed prompting: A modular approach for solving complex tasks. *arXiv preprint arXiv:2210.02406*.

Levente Kocsis and Csaba Szepesvári. 2006. Bandit based monte-carlo planning. In *European conference on machine learning*, pages 282–293. Springer.

Yuan-Hong Liao, Sven Elflein, Liu He, Laura Leal-Taixé, Yejin Choi, Sanja Fidler, and David Acuna. 2025. Longperceptualthoughts: Distilling system-2 reasoning for system-1 perception. *arXiv preprint arXiv:2504.15362*.

Yuan-Hong Liao, Rafid Mahmood, Sanja Fidler, and David Acuna. 2024a. Can feedback enhance semantic grounding in large vision-language models? *Preprint*, arXiv:2404.06510.

Yuan-Hong Liao, Rafid Mahmood, Sanja Fidler, and David Acuna. 2024b. Reasoning paths with reference objects elicit quantitative spatial reasoning in large vision-language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17028–17047, Miami, Florida, USA. Association for Computational Linguistics.

Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. 2023. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. *arXiv preprint arXiv:2310.02255*.

Ximing Lu, Seungju Han, David Acuna, Hyunwoo Kim, Jaehun Jung, Shrimai Prabhumoye, Niklas Muennighoff, Mostofa Patwary, Mohammad Shoeybi, Bryan Catanzaro, and 1 others. 2025. Retro-search: Exploring untaken paths for deeper and efficient reasoning. *arXiv preprint arXiv:2504.04383*.

Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Meiqi Guo, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, and 1 others. 2024. Improve mathematical reasoning in language models by automated process supervision. *arXiv preprint arXiv:2406.06592*.

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

OpenAI, :, Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, Alex Iftimie, Alex Karpenko, Alex Tachard Passos, Alexander Neitz, Alexander Prokofiev, Alexander Wei, Allison Tam, and 244 others. 2024. Openai o1 system card. *Preprint*, arXiv:2412.16720.

Ansh Radhakrishnan, Karina Nguyen, Anna Chen, Carol Chen, Carson Denison, Danny Hernandez, Esin Durmus, Evan Hubinger, Jackson Kernion, Kamilė Lukošiūtė, and 1 others. 2023. Question decomposition improves the faithfulness of model-generated reasoning. *arXiv preprint arXiv:2307.11768*.

Xiyao Wang, Zhengyuan Yang, Chao Feng, Hongjin Lu, Linjie Li, Chung-Ching Lin, Kevin Lin, Furong Huang, and Lijuan Wang. 2025. Sota with less: Mcts-guided sample selection for data-efficient visual reasoning self-improvement. *arXiv preprint arXiv:2504.07934*.

Jinyang Wu, Mingkuan Feng, Shuai Zhang, Ruihan Jin, Feihu Che, Zengqi Wen, and Jianhua Tao. 2025. Boosting multimodal reasoning with mcts-automated structured thinking. *arXiv preprint arXiv:2502.02339*.

Yuxi Xie, Anirudh Goyal, Wenyue Zheng, Min-Yen Kan, Timothy P Lillicrap, Kenji Kawaguchi, and Michael Shieh. 2024. Monte carlo tree search boosts reasoning via iterative preference learning. *arXiv preprint arXiv:2405.00451*.

Huanjin Yao, Jiaxing Huang, Wenhao Wu, Jingyi Zhang, Yibo Wang, Shunyu Liu, Yingjie Wang, Yuxin Song, Haocheng Feng, Li Shen, and 1 others. 2024. Mulberry: Empowering mllm with o1-like reasoning and reflection via collective monte carlo tree search. *arXiv preprint arXiv:2412.18319*.

Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, and 1 others. 2024a. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9556–9567.

Xiang Yue, Tianyu Zheng, Yuansheng Ni, Yubo Wang, Kai Zhang, Shengbang Tong, Yuxuan Sun, Botao Yu, Ge Zhang, Huan Sun, and 1 others. 2024b. Mmmu-pro: A more robust multi-discipline multimodal understanding benchmark. *arXiv preprint arXiv:2409.02813*.

Congzhi Zhang, Jiawei Peng, Zhenglin Wang, Yilong Lai, Haowen Sun, Heng Chang, Fei Ma, and Weijiang Yu. 2025. Vrest: Enhancing reasoning in large vision-language models through tree search and self-reward mechanism. *arXiv preprint arXiv:2506.08691*.

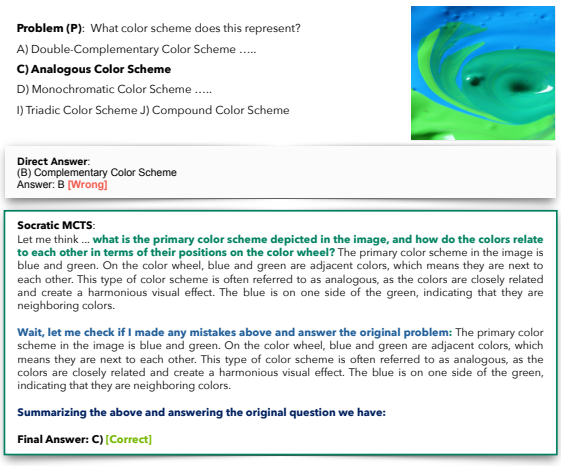
Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and 1 others. 2022. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*.

A Supplementary Material

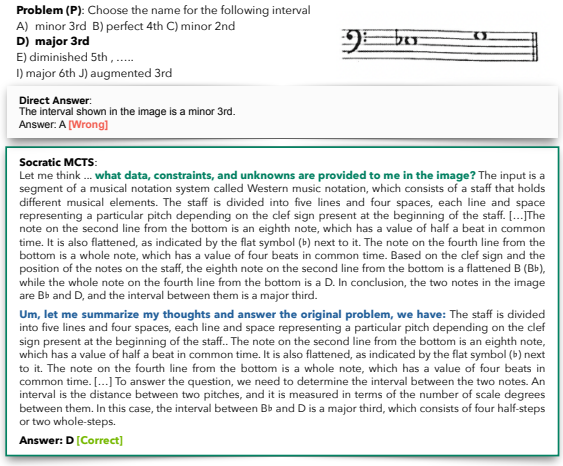
This appendix provides additional materials to support and illustrate the main findings of the paper. It is organized as follows:

- **Section B** presents additional qualitative examples illustrating how Socratic-MCTS generates structured reasoning traces across various tasks and benchmarks. These cases complement the main results and provide further insight into the interpretability of the method.
- **Section C** discusses notable failure cases where the model struggles, highlighting limitations of the current approach.
- **Section D** provides the full text prompts used for subquestion generation and evaluation.
- **Section E** provides additional experimental results.

B Qualitative Results



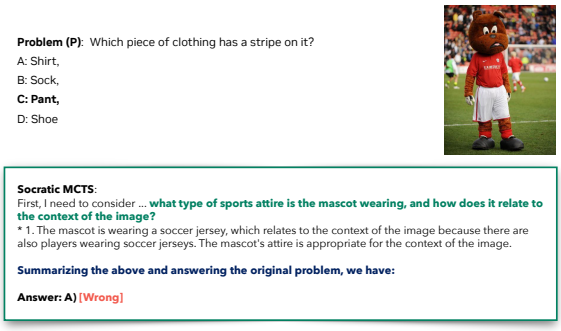
(a) Socratic-MCTS on a multimodal color theory question.



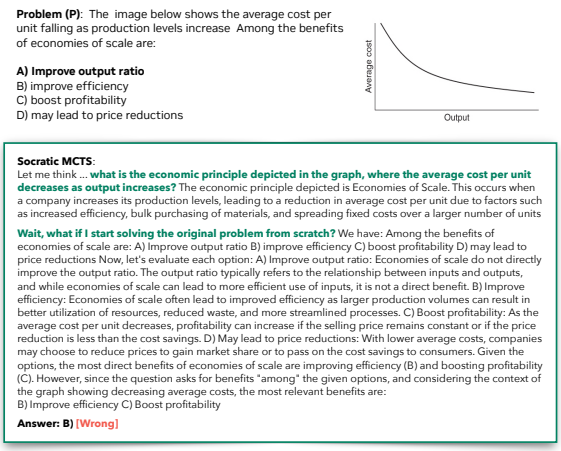
(b) Socratic-MCTS on a music question.

Figure 3: **Qualitative comparison on MMMU-PRO.** We show Socratic-MCTS responses on two multimodal questions from the benchmark, comparing qualitatively against the best-performing baseline (*direct prompting*). Subquestion–subanswer composition allows the model to uncover relevant knowledge, verify intermediate steps, and synthesize final answers coherently.

C Failure cases



(a) Socratic-MCTS failure case on MMStar.



(b) Socratic-MCTS failure case on MMMU-Pro

Figure 4: **Failure cases of Socratic-MCTS** We show failure cases of Socratic-MCTS on MMStar and MMMU-Pro.

D Full text prompts and phrases used

```
starting_out_phrases = [
    "Let me think ...",
    "First, I need to consider ...",
]
```

Figure 5: **`starting_out_phrases`: phrases used to begin the reasoning trajectory.**

```
transition_phrases = [
    "Next, let me look at ...",
    "Moving on, I wonder ... ",
    "That leads me to the next point ...",
    "Expanding on that ...",
    "So what does this mean for ...",
    "Now, I need to think about ... "
]
```

Figure 6: **`transition_phrases`: phrases used to signal intermediate reasoning steps.**

```
wrap_up_phrases = [
    "Summarizing the above and answering the original problem, we have:",
    "Wait, let me check if I made any mistakes above and answer the original problem:",
    "Okay, wrapping up any remaining calculations and answering the original problem, we have:",
    "Now, solving the original problem, we have:",
    "Um, let me summarize my thoughts and answer the original problem, we have:",
    "How do my previous thoughts connect to the original problem? Let me go back and solve it:",
    "Next, let me evaluate the above information recognizing that some of it may be incorrect:",
    "So, let me summarize the above information and directly answer the original problem:",
    "Wait, what if I start solving the original problem from scratch? We have:",
    "Wait, what if I start solving step-by-step the original problem from scratch? We have:",
    "Let's verify step-by-step the above information and answer the original problem:",
]
```

Figure 7: **`wrap_up_phrases`: phrases used to conclude and produce the final answer.**

```
---
** Problem: $#$problem$#$
---
** Intermediate Reasoning: $#$partialreasoning$#$
---
Context: You have been provided with:
* An image
* A problem related to that image
* Some intermediate reasoning steps that outline partial progress or initial thinking.
Your Goal:
- Formulate a single follow-up question that moves the reasoning process closer to solving the
↪  original problem.
Guidelines:
- Use the Socratic method by focusing on the next logical step in the problem-solving process.
Your question might:
- Clarify the core issue
- Challenge hidden assumptions
- Identify missing information or data
- Refine or expand upon a hypothesis
- Break the problem down into more manageable parts
- Probe deeper into the analysis or experimentation
- Prompt reevaluation or checking for errors
- Encourage synthesis to form a conclusion
- Spark reflection or consideration of next steps
Constraints:
- Ensure the question is directly relevant to the problem.
- Ensure the question is directly relevant to the intermediate reasoning steps.
- Ensure the question is asked in first-person perspective.
- Make the question open-ended enough to encourage deeper thinking, but specific enough to be
↪  actionable.
- If necessary use domain-specific terminology and aim to retrive domain-specific knowledge.
- Avoid simply restating the information already provided or present in the intermediate
↪  reasoning steps; instead, aim to advance the user's understanding or resolution of the
↪  problem.

Your response should be of the following format: 'Question: $Q' (without quotes) where Q is
↪  your proposed follow up question. - do not write anything other than 'Question: $Q'
↪  (without quotes) where Q is your proposed question."

Assistant: Question:
```

Figure 8: **Text prompt used to generate follow up subquestions.**

```
---
** Problem: $#$problem$#$
---
Context: You have been provided with:
* An image
* A multiple choice question problem related to that image
Your Goal:
- Formulate a single follow-up question that moves the reasoning process closer to solving the
↪  original problem.
Guidelines:
- Use the Socratic method by focusing on the next logical step in the problem-solving process.
Your question might:
- Clarify the core issue
- Challenge hidden assumptions
- Clarify or ask to describe parts of the image that are relevant to solve the problem
- Identify missing information or data
- Refine or expand upon a hypothesis
- Break the problem down into more manageable parts
- Probe deeper into the analysis or experimentation
- Prompt reevaluation or checking for errors
- Encourage synthesis to form a conclusion
- Spark reflection or consideration of next steps
Constraints:
- Ensure the question is directly relevant to the problem.
- Ensure the quesiton is asked in first-person perspective.
- Ensure the question is directly relevant to the intermediate reasoning steps.
- Ensure the question is asked in first-person perspective.
- Make the question open-ended enough to encourage deeper thinking, but specific enough to be
↪  actionable.
- If necessary use domain-specific terminology and aim to retrive domain-specific knowledge.
- Avoid simply restating the information already provided or present in the intermediate
↪  reasoning steps; instead, aim to advance the user's understanding or resolution of the
↪  problem.
Your response should be of the following format: 'Question: $Q' (without quotes) where Q is
↪  your proposed follow up question. - do not write anything other than 'Question: $Q'
↪  (without quotes) where Q is your proposed question."
Assistant: Question:
```

Figure 9: **Zero-Shot text prompt used to generate initial subquestions.**

```
$#$problem$#$
Answer with the option letter from the given choices directly. The last line of your response
↪  should be of the following format: 'Answer: $LETTER' (without quotes) where LETTER is one
↪  of options.
```

Figure 10: **Text prompt from (Yue et al., 2024a) used to evaluate Direct.**

```
$#$problem$#$
Answer the preceding multiple choice question. The last line of your response should be of the
↪  following format: 'Answer: $LETTER' (without quotes) where LETTER is one of options. Think
↪  step by step before answering.
```

Figure 11: **Text prompt from (Yue et al., 2024a) used to evaluate zero-shot CoT.**

E Additional Results

Additional Model. Search-based methods are computationally expensive. Therefore, we concentrated on one of the best open-source non-reasoning VLM available at the time of this writing (i.e. InternVL-78B). That said, in table 3, we show the experimental results of using Socratic-MCTS on an additional model, Llama-3.2-90B-vision. For this experiment, we use the MMMU-PRO 4-options setting and randomly select a subset of 1000 problems from MMstar.As with InternVL-78B, we observe consistent notable gains of 2% on MMMU-Pro and 1.5% on MMStar.

Llama-3.2-90B	Overall Acc (MMMU-PRO, 4 opts)	Overall Acc (MMstar, 1000)
+ CoT	0.584	0.594
+ Socratic-MCTS	**0.603**	**0.608**

Table 3: Performance comparison of Llama-3.2-90B with different reasoning methods.

Comparison with CoT+Self-Consistency (majority voting) Table 4 reports InternVL-2.5-78B accuracy on MMMU-PRO using (i) sampling-based CoT + self-consistency (majority vote, t=0.6) with different numbers of generations and (ii) our Socratic MCTS. We also added Direct greedy decoding (the best single generation baseline) for reference. Socratic MCTS outperforms CoT + self-consistency in all the experiments highlighting the strength of our search-based decomposition. We emphasize a majority-vote layer could just as easily be placed on top of Socratic MCTS (i.e. running Socratic MCTS multiple times for the same problem and selecting the most common answer). We leave this promising extension of Socratic MCTS for future work.

# Generations	Method	Accuracy
1	Direct (greedy)	0.517
8	CoT + Self-Consistency	0.509
16	CoT + Self-Consistency	0.517
32	CoT + Self-Consistency	0.522
1	**Socratic MCTS (ours)**	**0.537**

Table 4: Comparison across different methods and numbers of generations.