
Mulberry: Empowering MLLM with o1-like Reasoning and Reflection via Collective Monte Carlo Tree Search

Huanjin Yao^{2,3,*}, Jiaxing Huang^{1,*}, Wenhao Wu³, Jingyi Zhang¹, Yibo Wang², Shunyu Liu¹,

Yingjie Wang¹, Yuxin Song³, Haocheng Feng³, Li Shen⁴, Dacheng Tao^{1}

¹ Nanyang Technological University ² Tsinghua University ³ Baidu Inc. ⁴ Sun Yat-sen University

* Equal Contribution  Corresponding Author

Abstract

In this work, we aim to develop an MLLM that understands and solves questions by learning to create each intermediate step of the reasoning involved till the final answer. To this end, we propose Collective Monte Carlo Tree Search (CoMCTS), a new learning-to-reason method for MLLMs, which introduces the concept of collective learning into “tree search” for effective and efficient reasoning-path searching and learning. The core idea of CoMCTS is to leverage collective knowledge from multiple models to collaboratively conjecture, search and identify effective reasoning paths toward correct answers via four iterative operations including Expansion, Simulation and Error Positioning, Backpropagation, and Selection. Using CoMCTS, we construct Mulberry-260k, a multimodal dataset with a tree of rich, explicit and well-defined reasoning nodes for each question. With Mulberry-260k, we perform collective SFT to train our model, Mulberry, a series of MLLMs with o1-like step-by-step Reasoning and Reflection capabilities. Extensive experiments demonstrate the superiority of our proposed methods on various benchmarks. Code is available at <https://github.com/HJYao00/Mulberry>.

1 Introduction

“What I cannot create, I do not understand.”

—Richard Feynman

Multimodal large language models (MLLMs) embody the essence of this dictum, which understand the world by learning to create expected responses to multimodal inputs such as images and text. While MLLMs have recently shown significant progress in straightforward tasks [1, 2], they often experience obviously increased failures on complex tasks requiring in-depth reasoning [3]. Feynman’s dictum might be the perfect metaphor of such failures of MLLMs, as we should only be able to work something out if we can create and have a firm understanding of each step of the reasoning involved. However, current MLLMs predominantly operate in a simple “direct prediction” mode [4], *i.e.*, generating brief, final answers to questions with little explicit and well-defined intermediate reasoning steps.

In this work, we aim to develop an MLLM that understands and solves questions by learning to create each intermediate step of the reasoning involved till the final answer. Recent advances in NLP, such as OpenAI o1 [5], have shown great potential in enabling LLM to learn to reason and tackle complex language tasks [6]. The core design of these advances lies in AlphaGo-like “tree search”: they employ tree search methods, like MCTS [7], to bootstrap an LLM itself to build a tree

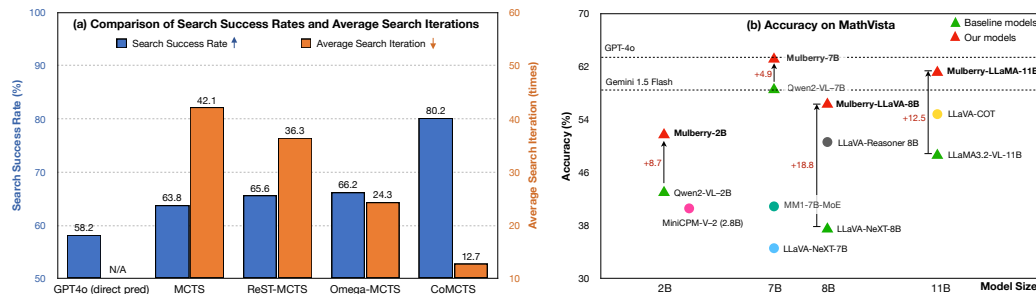


Figure 1: **(a)** Our CoMCTS shows great superiority in search effectiveness and efficiency against other tree search methods. **(b)** Our Mulberry, trained on CoMCTS-searched data, outperforms most open-sourced MLLMs and achieves competitive results against closed-source ones, showing outstanding abilities in step-by-step reasoning and reflection.

of intermediate thoughts, explore effective reasoning paths, and leverage these paths to teach model to reason step-by-step.

An intuitive idea is to directly apply these “tree search” methods to search effective reasoning paths for MLLMs, which, however, does not work well. As shown in Figure 1, we believe this is largely attributed to several search challenges for MLLMs. (1) *Search Effectiveness*: Traditional MCTS methods [7, 8, 9, 10] generally work by self-bootstrapping while current MLLMs are typically trained with little explicit and well-defined intermediate reasoning steps, making these search methods often trapped in homogeneous low-quality nodes within the reasoning space of a single MLLM, ultimately leading to low search success rates. (2) *Search Efficiency*: Traditional MCTS methods typically expand and explore only one reasoning node per search iteration, which advance a single step each time and demand massive iterations, making them inefficient for computation-intensive MLLMs.

To tackle these challenges, we propose Collective Monte Carlo Tree Search (CoMCTS), a new learning-to-reason method for MLLMs, which introduces the concept of collective learning into “tree search” for effective and efficient reasoning-path searching and learning. The core idea of CoMCTS is to leverage collective knowledge to collaboratively conjecture, search and identify effective reasoning paths toward correct answers. Specifically, CoMCTS searches effective reasoning paths iteratively, and in each iteration, it leverages collective knowledge from multiple MLLMs to jointly (a) expand diverse and complementary candidate subsequent reasoning nodes till the end from a given start node, (b) simulate reasoning outcomes, position error candidate nodes and prune them along with their child nodes, (c) backpropagate to update the score and visit count of each reasoning node in a bottom-up manner, and (d) select the leaf reasoning node with the highest Upper Confidence Bound value as next start node.

In this way, our CoMCTS achieves effective and efficient reasoning search. (1) The joint expansion mechanism enables CoMCTS to concatenate reasoning trajectories from multiple MLLMs via iterative search, ultimately constructing an unified reasoning tree comprising diverse and complementary reasoning nodes. Thus, it allows reasoning-path search not only within the reasoning space of a given MLLM itself but also among those of others, benefiting from the synergy of multiple MLLMs while avoiding being trapped in homogeneous low-quality nodes within the reasoning space of a single MLLM itself. (2) The joint simulation and error positioning mechanism enables CoMCTS to, in each search iteration, skip multiple intermediate steps and select the last correct step as the next start node, largely reducing search time while maintaining search effectiveness. Here, collective knowledge is also crucial as it is often challenging for a model to recognize and position errors made by itself while relatively easy by using other models.

Furthermore, we extend our CoMCTS for reflective reasoning-path search. Based on the unified reasoning tree constructed by CoMCTS, which provides both positive and negative reasoning nodes, we identify and integrate negative sibling nodes into effective reasoning paths to build the reflective reasoning path that includes a transition from a negative reasoning node to a positive one. By learning from reflective reasoning paths, MLLMs can perform appropriate step-wise reflection, dynamically calibrating their reasoning trajectory from an erroneous node toward a correct one during long-chain reasoning. Here, collective knowledge facilitates reflective reasoning-path search by providing a rich set of diverse positive and negative reasoning nodes.

Using our CoMCTS, we search effective and reflective reasoning paths for a set of multimodal inputs, and construct Mulberry-260k, a Multimodal learning-to-Reason-and-Reflect dataset with a tree of rich, explicit and well-defined reasoning nodes for each question. With Mulberry-260k, we perform collective supervised fine-tuning to train our model, Mulberry, a series of Multimodal LLMs with o1-like step-by-step Reasoning and Reflection capabilities.

The main contributions of this work are fourfold. **First**, we introduce the concept of collective learning into MCTS, and propose CoMCTS which leverages collective knowledge to collaboratively conjecture, search and identify effective and reflective reasoning paths for MLLMs, significantly improving search effectiveness and efficiency. To the best of our knowledge, this is the first work that explores collective learning with MCTS for MLLMs. **Second**, we construct Mulberry-260k that provides a valuable resource for advancing research in step-by-step reasoning and reflection in MLLMs. **Third**, we develop Mulberry, a series of MLLMs with outstanding capabilities in step-by-step reasoning and reflection. **Fourth**, extensive experiments demonstrate the superiority of our proposed methods on various benchmarks.

2 Related Work

2.1 Multimodal Large Language Model

MLLMs [1, 2, 11, 12, 13, 14, 15, 16] have made notable advancements in general vision-language understanding, enabling them to interpret visual semantics across various domains. Recent studies [17, 3] explore MLLM reasoning and reveal that directly employing CoT prompt to derive the final answer may result in limited gains or even degradation. In addition, some studies [18, 19] introduce plan-based CoT prompting to guide models to generate intermediate information for predicting final answers. Recent advances [4] attempt structured reasoning with a planed flow of certain pre-defined stages, enhancing the CoT capabilities [15] of MLLMs. Differently, this paper, for the first time, introduces the concept of “tree search” into MLLM reasoning and proposes a novel CoMCTS technique to search effective and reflective reasoning paths to train our Mulberry, a series of MLLMs with outstanding capabilities in step-by-step reasoning and reflection.

2.2 Large Language Model Reasoning

LLM reasoning methods can be broadly categorized into three types, *i.e.*, prompt-based, plan-based and learning-based reasoning. Prompt-based methods, like Chain-of-Thought (CoT) [20], mimic human reasoning by providing a few hand-crafted, step-by-step solutions as references. Plan-based methods, such as Tree/Graph-of-thought [21, 22], predict multiple reasoning paths in a tree or graph manner and take consistent units of thought for thoughtful decision-making. Learning-based reasoning methods, represented by GPTo1, Star [23], Iter-MCTS [6] and ReST-MCTS [24], first employ tree search approaches [25], like MCTS, to bootstrap an LLM itself to build a tree of intermediate thoughts, explore effective reasoning paths, and leverage these paths to train model to reason step-by-step.

2.3 Monte-Carlo Tree Search

Monte-Carlo Tree Search (MCTS) is a powerful search paradigm for complex decision making problems and has been extensively explored across diverse fields, including games [26, 27], robotics [28, 29], theorem proving [30], matrices multiplication [31], etc. For instance, AlphaGo [26] introduces deep learning into MCTS, achieving superhuman results in board and video games [26, 27]. Besides, [32, 33] explore MCTS for path finding and train timetabling problems, while [34] integrates MCTS into physics-informed planning networks for robot control. In this work, we propose CoMCTS that enables effective and reflective reasoning-path searching and learning on MLLMs.

2.4 Collective Learning

Collective learning, also known as Co-training, aims to harness collective intelligence of multiple individuals to improve learning outcomes. This concept originates in early pioneering studies [35, 36, 37], which utilize collective knowledge to address data insufficiency issues in classification learning. Recent advances introduce collective learning into deep neural networks for efficient and effective

deep learning. For example, [38, 39] employ collective knowledge from multiple classifiers to predict more accurate pseudo-labels for semi-supervised classification; [40] utilizes collective knowledge from multiple discriminators to enhance image discrimination and generation; and [41] leverages the synergy of multiple models for reinforcement learning.

3 Method

We first present CoMCTS that introduces collective learning into “tree search” for effective and efficient reasoning-path searching and learning. We then illustrate the extension of CoMCTS for reflective reasoning-path search, and describe data construction and model training using CoMCTS.

3.1 CoMCTS for effective reasoning

The core idea of CoMCTS is to leverage collective knowledge to collaboratively conjecture, search and identify effective reasoning nodes in an iterative manner, aiming to find effective reasoning paths leading to correct answers.

We denote a policy model as π , initialized by a pre-trained MLLM. We leverage collective knowledge from a group of MLLMs $\{\pi_1, \pi_2, \dots, \pi_K\}$ to jointly search and learn effective reasoning paths. Given a multimodal input question Q (e.g., a text instruction with an image, $Q = \{\text{text}, \text{image}\}$), each model π can generate a sequence of intermediate reasoning states toward the final answer $(s_1, s_2, s_3, \dots, s_M) \sim \pi_\theta(\cdot|Q)$ via autoregressive next token prediction. We define the intermediate reasoning state at step m as s_m and the state generated by model π_k at step m as s_m^k . Each reasoning step consists of one or a few sentences.

CoMCTS algorithm begins at the root node, i.e., either the start of a response or an incomplete response, and performs reasoning-path search via a certain number of iterations, where each iteration comprises four key operations: (a) Expansion, (b) Simulation and Error Positioning, (c) Backpropagation, and (d) Selection, as elaborated below.

(a) Expansion. The goal of this operation in CoMCTS is to expand the current leaf reasoning node (if it is not a terminal node) to integrate new subsequent candidate reasoning nodes. Given the current leaf node s_m^k (i.e., the node selected by Operation (d) Selection or the root node), CoMCTS utilizes collective knowledge from a group of MLLMs, $\{\pi_1, \pi_2, \dots, \pi_K\}$, to jointly expand a set of diverse and complementary candidate reasoning paths $S_{\text{candidate}} = \cup_{j=1}^K S_{\text{candidate}}^j$ in parallel till terminal node:

$$S_{\text{candidate}}^j \sim \pi_j(\cdot|Q, \text{Parent}(s_m^k), s_m^k), \quad (1)$$

where $\text{Parent}(s_m^k)$ returns all parent nodes of s_m^k and $(\text{Parent}(s_m^k), s_m^k)$ denotes the current reasoning path from the root node to s_m^k . $S_{\text{candidate}}^j = \{s_i^j\}$ stands for a potential reasoning path generated by model π_j starting from s_m^k .

(b) Simulation and Error Positioning. In this operation, CoMCTS utilizes collective knowledge from $\{\pi_1, \pi_2, \dots, \pi_K\}$ to jointly estimate the potential value of child nodes $s_i^j \in S_{\text{candidate}}$ (added in Operation (a)), and considers low-score nodes as erroneous reasoning nodes, and positions and filters out them along with their child nodes:

$$R(s_i^j) = \frac{1}{K} \sum_{l=1}^K \pi_l(\cdot|\text{prompt}_{\text{eval}}, Q, \text{Parent}(s_i^j), s_i^j) \quad (2)$$

$$S_{\text{candidate}}^* = \{s_i^j \in S_{\text{candidate}} | R(s_i^j) \geq t\} \quad (3)$$

where $R(s_i^j)$ denotes a reasoning node evaluation function that uses the prompt, $\text{prompt}_{\text{eval}}$, to request a group of MLLMs, $\{\pi_1, \pi_2, \dots, \pi_K\}$, to jointly evaluate the candidate reasoning node s_i^j . t is a threshold and discontinued reasoning nodes in $S_{\text{candidate}}^*$ are automatically removed following the error node removal in Eq.(3).

(c) Backpropagation. Given the new reasoning tree expanded and simulated using collective knowledge in Operations (a)-(b), CoMCTS performs a bottom-up update from the leaf nodes back to the root node. Each node s along the newly expanded path in the reasoning tree updates its statistics,

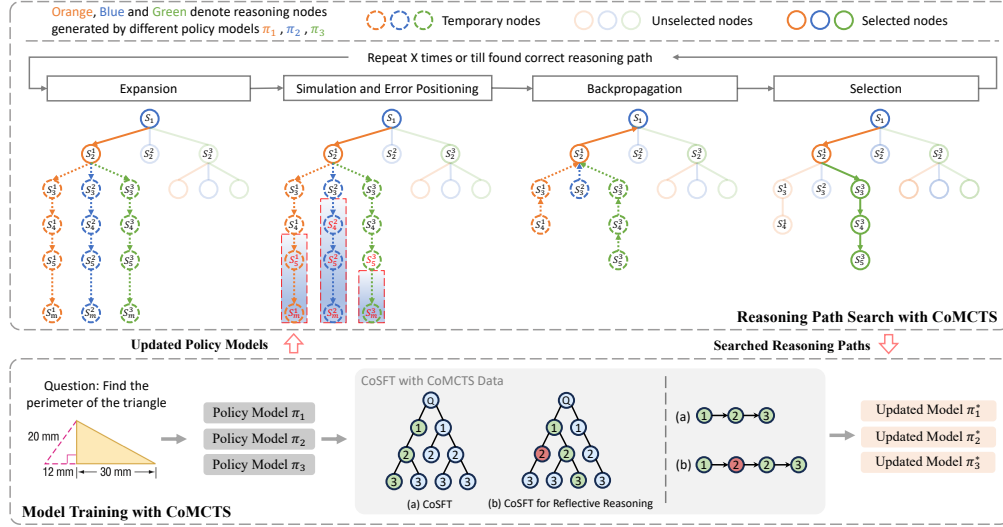


Figure 2: **Overview.** Our CoMCTS trains Mulberry with two alternating phases. *In top part*, CoMCTS searches reasoning paths iteratively, and in each iteration, it utilizes collective knowledge from multiple MLLMs to jointly (a) expand diverse and complementary candidate subsequent reasoning nodes till the end from a given start node, (b) simulate reasoning outcomes, position error candidate nodes and prune them along with their child nodes, (c) backpropagate to update the score and visit count of each reasoning node in a bottom-up manner, and (d) select the leaf reasoning node with the highest UCB value as next start node. *In bottom part*, we train the model to learn from the reasoning trees constructed by CoMCTS.

including visit count N and node value V :

$$V(s) \leftarrow \frac{N(s) \cdot V(s) + \sum_{s_l \in \text{Child}(s)} R(s_l)}{N(s) + \text{CountChild}(S_{\text{candidate}}^*, s)}, \quad (4)$$

$$N(s) \leftarrow N(s) + \text{CountChild}(S_{\text{candidate}}^*, s), \quad (5)$$

where $\text{Child}(s)$ returns all the child nodes of s , and $\text{CountChild}(S_{\text{candidate}}^*, s)$ is a child node counting function that calculates the number of child nodes of s in $S_{\text{candidate}}^*$.

(d) Selection. Following Operations (a), (b) and (c), CoMCTS traverses the updated reasoning tree to select the next starting node. This selection is guided by the Upper Confidence Bound (UCB) value, which balances search exploration and exploitation. The UCB value of a node s is computed using the node reward value $V(s)$ and the visit count $N(s)$. Among the candidate nodes $s \in S_{\text{candidate}}^*$, the one with the highest UCB value is chosen as the starting node s_m^{k*} for next search iteration:

$$s_m^{k*} = \arg \max_{s \in S_{\text{candidate}}^*} V(s) + c \cdot \sqrt{\frac{\log N(\hat{s})}{1 + N(s)}} \quad (6)$$

where c stands for a constant which controls the level of exploration. $\hat{s}$ denotes the parent node of s .

CoMCTS. These four operations, *i.e.*, (a) Expansion, (b) Simulation and Error Positioning, (c) Backpropagation and (d) Selection, are repeated for a pre-defined number of iterations or until correct reasoning paths are found. This iterative process allows CoMCTS to construct a question-dependent reasoning tree S with the correct reasoning path Y , and ultimately form a multimodal learning-to-reason data triplet $\{Q, Y, S\}$. By applying our CoMCTS to a set of multimodal questions, we can construct a collection of multimodal learning-to-reason data triplets, which provide a tree of rich, explicit and well-defined reasoning nodes toward the final answer for each question and enable MLLMs to learn to reason step-by-step.

3.2 CoMCTS for reflective reasoning

In this subsection, we extend CoMCTS for reflective reasoning-path search. Based on the unified reasoning tree constructed by CoMCTS, *i.e.*, $\{Q, Y, S\}$, which provides both positive and negative

reasoning nodes, we identify and integrate negative sibling nodes into effective reasoning paths to build the reflective reasoning path that includes a transition from a negative reasoning node to a positive one.

Identifying negative sibling node. Given the effective reasoning path Y , we identify the negative sibling reasoning node for $s \in Y$ using UCB:

$$s_{\text{neg}} = \arg \min_{s_l \in \text{Sibling}(s)} \text{UCB}(s_l) - \text{UCB}(s), \quad \forall s \in Y, \quad (7)$$

where $\text{Sibling}(s)$ returns all the sibling nodes of s , *i.e.*, the nodes on the same hierarchical level under the same parent node of s . $\text{UCB}(s) = V(s) + c \cdot \sqrt{\frac{\log N(\bar{s})}{1+N(s)}}$ as in Eq. 6.

Constructing reflective reasoning path. Based on Eq. 7, we randomly sample a reasoning node $s \in Y$ with its negative sibling node s_{neg} , and concatenate them with a reflection prompt to form a reflection trajectory, *i.e.*, $(s_{\text{neg}}, \text{prompt}_{\text{reflect}}, s)$. We then use a function $\text{Replace}(\cdot)$ that replaces $s \in Y$ with $(s_{\text{neg}}, \text{prompt}_{\text{reflect}}, s)$ to convert Y into the reflective reasoning path Y_{reflect} :

$$Y_{\text{reflect}} = \text{Replace}(Y, s, (s_{\text{neg}}, \text{prompt}_{\text{reflect}}, s)), \quad (8)$$

where $\text{prompt}_{\text{reflect}}$ denotes a reflection prompt, such as “The previous reasoning step is wrong and let’s rethink it again.” Then, we can integrate the reflective reasoning path Y_{reflect} into our data as a quadruplet $\{Q, Y, Y_{\text{reflect}}, S\} \in D$.

Collective Supervised Fine-Tuning (CoSFT). Given $(Q, Y) \in \mathcal{D}$, we apply standard SFT objective to train our MLLM to learn from D constructed by CoMCTS:

$$\mathcal{L}_{\text{CoSFT}}(\pi_k) = \sum_{(Q, Y) \in \mathcal{D}} \log \pi_k(Y|Q), \quad (9)$$

where $Y = \{s\}$ denotes the effective reasoning path that includes a sequence of reasoning nodes collectively conjectured, searched and identified by a group of MLLMs.

CoSFT for reflective reasoning. Given a question and its reasoning tree $(Q, S) \in \mathcal{D}$ constructed by CoMCTS, we randomly sample a reflective reasoning path Y_{reflect} from S as in Eqs.7-8, and conduct CoSFT for reflective reasoning:

$$\mathcal{L}_{\text{CoSFT-Re}}(\pi_k) = \sum_{(Q, Y_{\text{reflect}}) \in \mathcal{D}} \log \pi_k(Y_{\text{reflect}}|Q), \quad (10)$$

where $Y_{\text{reflect}} = \{s\}$ denotes the reflective reasoning path that includes an additional step-wise reflection trajectory.

The goal of $\mathcal{L}_{\text{CoSFT}}$ and $\mathcal{L}_{\text{CoSFT-Re}}$ is to maximize the log probability of effective and reflective reasoning path Y and Y_{reflect} over a tree of reasoning nodes S generated by CoMCTS. In addition, $\mathcal{L}_{\text{CoSFT-Re}}$ enables to leverage the negative information during CoMCTS search process by learning to calibrate negative reasoning nodes.

3.3 Training with Collective MCTS

Using CoMCTS, we search effective and reflective reasoning paths for a set of multimodal input questions, and construct Mulberry-260k, a multimodal learning-to-reason-and-reflect dataset with a tree of rich, explicit and well-defined reasoning nodes for each question, *i.e.*, a set of quadruplets $\{Q, Y, Y_{\text{reflect}}, S\} \in D$. To learn collective knowledge from Mulberry-260k, we perform collective SFT to train our model. Mulberry, a series of Multimodal LLMs with o1-like step-by-step Reasoning and Reflection capabilities.

Algorithm 1 Training Mulberry with CoMCTS

Input: a set of policy models $\{\pi_1, \pi_2, \dots, \pi_K\}$ initialized by different MLLMs; a set of multimodal questions D_Q

for $i = 1$ to MaxEpoch **do**

Reasoning Tree Search using CoMCTS:

for $Q \in D_Q$ **do**

Collective Monte Carlo tree search:

$\{Q, Y, S\} = \text{CoMCTS}(\{\pi_1, \pi_2, \dots, \pi_K\}; Q)$

if found an effective reasoning path then

Search and find Y_{reflect} from S

Add $\{Q, Y, Y_{\text{reflect}}, S\}$ into D

Remove Q from D_Q

Model Training with CoMCTS Reasoning Trees:

for $k = 1$ to K **do**

for $(Q, Y, Y_{\text{reflect}}, S) \in D$ **do**

Supervised Fine-Tuning:

Optimize π_k via $\mathcal{L}_{\text{CoSFT}}(\pi_k)$ and

$\mathcal{L}_{\text{CoSFT-Re}}(\pi_k)$

Output: Trained policy models $\{\pi_1, \pi_2, \dots, \pi_K\}$

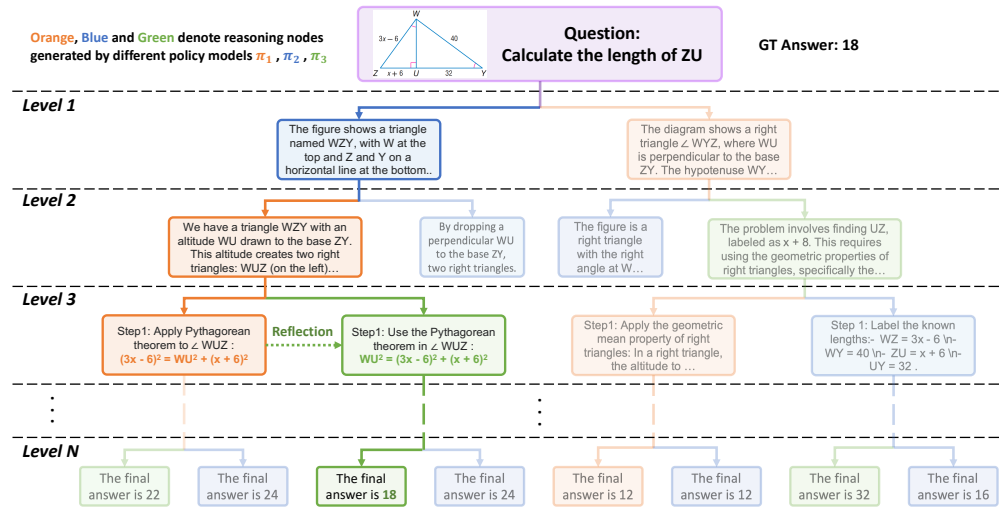


Figure 3: Qualitative illustration of reasoning tree searched by CoMCTS with rich, explicit, well-defined reasoning nodes.

4 Experiments

In this section, we first introduce our CoMCTS-generated dataset, Mulberry-260K, including its sources, construction, and analysis in Sec. 4.1, and provide implementation details in Sec. 4.2. We then present the main results in Sec. 4.3, demonstrating the effectiveness of the searched data (*i.e.*, Mulberry-260K) and the trained models (*i.e.*, Mulberry). In Sec. 4.4, we perform comprehensive ablation studies on the impact of effective and reflective reasoning data and the contributions of collective knowledge sources. Sec. 4.5 discusses the effectiveness and efficiency of CoMCTS with other tree search methods.

4.1 Dataset

The Sources of Raw Data. To construct a comprehensive and general-purpose tree-based reasoning dataset, we collect 260K raw multimodal input questions (*i.e.*, a text task instruction with an image as an input question) from a wide range of domains, covering General Multimodal Understanding, Mathematics, Figure Understanding, Realworld Understanding, Science, Medical Image Understanding, etc. The specific data sources are provided in the Appendix I.

Reasoning Data Construction. As detailed in Sec. 3 and Algorithm 1 and visually illustrated in Figures 2 and 3, we employ our CoMCTS to search effective and reflective reasoning paths for a set of raw multimodal input questions as collected from the mentioned “The Sources of Raw Data”, ultimately constructing our dataset, Mulberry-260K. Note we only sample 15K data for reflective reasoning training to avoid overabundance of reflection data.

4.2 Implementation Detail

We implement collective learning in CoMCTS with four models, including GPT-4o, Qwen2-VL-7B, LLaMA-3.2-11B-Vision-Instruct, and Qwen2-VL-72B, to construct Mulberry-260K. In CoMCTS, we set maximum search iteration to 20 and threshold t in Eq. 3 to 0. In each iteration, each model generates one candidate reasoning path to balance search exploration and exploitation. We adopt four popular MLLMs as baselines, conducting experiments on Qwen2-VL-7B and LLaMA-3.2-11B-Vision-Instruct to examine the effectiveness of CoMCTS, and on Qwen2-VL-2B and LLaVA-NeXT-8B to study the generalization of CoMCTS-searched data. Training details are in the Appendix B.

4.3 Main Results

To examine the effectiveness of searched data (*i.e.*, Mulberry-260K) and trained models (*i.e.*, Mulberry), we conduct extensive experiments with four powerful baseline models, and comprehensively benchmark our Mulberry with various state-of-the-arts, including general and reasoning-based

Table 1: **Main Results.** To examine the effectiveness of the searched data (*i.e.*, Mulberry-260K) and the trained models (*i.e.*, Mulberry), we conduct extensive experiments with four powerful baseline models, and comprehensively benchmark our Mulberry with various state-of-the-arts, including general and reasoning-based MLLMs.

Method	MathVista	MMStar	MMMU	ChartQA	DynaMath	HallBench	MM-Math	MME _{sum}	AVG
<i>Closed-Source Model</i>									
GPT-4o [42]	63.8	63.9	69.1	85.7	63.7	55.0	31.8	2329	64.5
Claude-3.5 Sonnet [43]	67.7	62.2	68.3	90.8	64.8	55.0	-	1920	-
<i>Open-Source Model</i>									
MM-1.5-7B [44]	47.6	-	41.8	78.6	-	-	-	1861	-
Idefics3-LLaMA3-8B [45]	58.4	55.9	46.6	74.8	-	-	-	1937	-
InternVL2-8B [46]	58.3	61.5	51.8	83.3	39.7	-	-	2210	-
MiniCPM-V-2.6-8B [47]	60.6	57.5	49.8	-	-	48.1	-	2348	-
DeepSeek-VL2-MOE-4.5B [48]	62.8	61.3	51.1	86.0	-	-	-	2253	-
<i>Reasoning Model</i>									
LLaVA-CoT-11B [4]	54.8	57.6	-	-	-	47.8	-	-	-
LLaVA-Reasoner-8B [3]	50.6	54.0	40.0	83.0	-	-	-	-	-
Insight-V-8B [49]	49.8	57.4	42.0	77.4	-	-	-	2069	-
LLaVA-NeXT-8B [50]	37.5	42.1	41.7	69.5	22.7	33.4	0.6	1957	39.7
Mulberry-LLaVA-8B	56.3	54.5	43.0	79.5	34.1	47.5	18.9	2021	50.7 ^{11↑}
Llama-3.2-11B-V-Ins. [51]	48.6	49.8	41.7	83.4	34.3	40.3	4.1	1787	45.8
Mulberry-Llama-11B	61.1	58.5	45.6	83.5	37.2	48.9	18.7	2035	53.3 ^{7.5↑}
Qwen2-VL-2B [2]	43.0	48.0	41.1	73.5	24.9	41.7	1.0	1872	42.5
Mulberry-2B	51.7	51.3	42.0	77.7	30.0	44.9	13.9	2013	47.9 ^{5.4↑}
Qwen2-VL-7B [2]	58.2	60.7	54.1	83.0	42.1	50.6	5.9	2327	54.7
Mulberry-7B	63.1	61.3	55.0	83.9	45.1	54.1	23.7	2396	58.9^{4.2↑}

MLLMs. The evaluation spans 8 widely used and challenging datasets, covering the fields ranging from general and mathematical reasoning to hallucination and multi-disciplinary understanding and reasoning in Tab. 1.

Comparison with baselines. We first conduct experiments on baselines Qwen2-VL-7B and LLaMA-3.2-11B-Vision-Instruct that are involved in collective learning of CoMCTS for joint reasoning-path conjecture, search and identification. Trained with jointly-searched Mulberry-260k data, Mulberry-7B and Mulberry-11B bring clear performance improvements against their baselines, *i.e.*, +4.2% over Qwen2-VL-7B and +7.5% over LLaMA-3.2-11B-Vision-Instruct averaged on 8 benchmarks, validating the effectiveness of CoMCTS. On the other hand, we examine the generalization of Mulberry-260k by applying it to train other models not involved in CoMCTS, such as Qwen2-VL-2B and LLaVA-NeXT-8B. Trained with Mulberry-260k, Mulberry-2B and Mulberry-8B enhance their baselines with +5.4% and +11.0% gains, respectively, averaged on 8 benchmarks, demonstrating the generalization of CoMCTS.

Comparison with reasoning-response models. We benchmark Mulberry with various state-of-the-art reasoning-response models. Using the same base model LLaVA-NeXT-8B [50], our Mulberry outperforms LLaVA-Reasoner-8B and Insight-V-8B by +5.7% and +6.5% on mathematical benchmark MathVista, and by +3.0% and +1.0% on multi-disciplinary benchmark MMMU, respectively. Besides, Mulberry-11B surpasses LLaVA-COT-11B by +6.3% on MathVista under the same baseline LLaMA-3.2-11B-Vision-Instruct. The great superiority of Mulberry is largely attributed to our CoMCTS, which conducts collective tree search and generates rich, explicit and well-defined reasoning nodes with flexible numbers of steps.

Comparison with state-of-the-arts. We compare our Mulberry with popular state-of-the-art models, both open-source and closed-source. The results in Tab. 1 show that Mulberry, trained on CoMCTS-searched data, outperforms most open-sourced MLLMs and achieves competitive results against closed-source ones, demonstrating outstanding abilities in step-by-step reasoning and reflection.

4.4 Ablation Study

Ablation Study on CoMCTS. We conduct ablation studies with powerful GPT-4o as the baseline over 1K samples from Geo3K [52] and GeoQA-Plus [53], as shown in Tab. 2. As the core of CoMCTS, we examine how each model in the collective learning group contributes to the overall tree search performance. Tab. 2 reports the Search Success Rates, and baseline GPT-4o works not

Table 2: **Ablation Study on CoMCTS.** We study how each model in CoMCTS collective learning contribute to overall tree search performance in Search Success Rate (S.S.R.).

Direct Pred	CoMCTS					S.S.R.
GPT-4o	GPT-4o	Qwen2-VL-7B	Llama3.2-11B	Qwen2-VL-72B		
✓						58.2
	✓					63.8
	✓	✓				66.2
	✓	✓	✓			69.7
	✓	✓	✓	✓		80.2

very well without tree search. It shows that CoMCTS with only GPT-4o improves the performance to 63.8%, largely because our tree search designs like expansion, simulation and error positioning can work even without using collective knowledge. Besides, progressively involving more models into CoMCTS consistently improves the search performance, even with small models like Qwen2-VL-7B (*i.e.*, +2.4%), demonstrating the effectiveness of CoMCTS in capturing useful collective knowledge not only with large models but also from small models. In final, the inclusion of all four models in the proposed CoMCTS performs clearly the best, *i.e.*, 80.2%, validating the effectiveness of collective learning on reasoning tree search.

Ablation Study on Mulberry. We train Mulberry with effective and reflective reasoning data searched by CoMCTS and study their respective contributions to overall reasoning performance. Results in Tab. 3 on MathVista show that incorporating reflection data enhances performance by 0.8%, demonstrating the complementarity of effective and reflective reasoning data searched by CoMCTS.

4.5 Discussion

Comparison with other tree search methods. We compare our CoMCTS with other tree search methods in search effectiveness and efficiency, including the baseline “GPT-4o direction prediction”, “traditional MCTS [7]”, “ReST-MCTS [24]” that enhances MCTS by introducing partial search, and “Omega-MCTS [9]” that improves MCTS by designing binary search. Tab. 4 shows the results in search success

Table 4: **Comparison with other tree search methods.** “GPT-4o (direct)” refers to baseline without tree search. CoMCTS outperforms in search effectiveness and efficiency.

Methods	Search Success Rate ↑	Average Search Iteration ↓
GPT4o (direct)	58.2	-
MCTS	63.8	42.1
ReST-MCTS	65.6	36.3
Omega-MCTS	66.2	24.3
CoMCTS	80.2	12.7

rate and average search iteration that indicate search effectiveness and efficiency respectively. We can observe that existing tree search methods improve GPT-4o with limited gains. One main reason lies in that traditional MCTS methods generally work by self-bootstrapping and often get trapped in homogeneous low-quality nodes within the reasoning space of a single MLLM. On the other hand, CoMCTS shows great superiority in search effectiveness and efficiency, thanks to the joint expansion mechanism in CoMCTS that allows reasoning-path search not only within the reasoning space of a given MLLM itself but also among those of others, benefiting from the synergy of multiple MLLMs while avoiding being trapped within the reasoning space of a single MLLM.

5 Conclusion

This paper presents CoMCTS, a new learning-to-reason approach for MLLMs, which introduces the concept of collective learning into “tree search” for effective and efficient reasoning-path searching and learning. Based on the CoMCTS, we search effective and reflective reasoning paths for a set of multimodal inputs, and construct Mulberry-260k, a multimodal learning-to-reason-and-reflect dataset with a tree of rich, explicit and well-defined reasoning nodes for each question. Using Mulberry-260k, we train our model, Mulberry, a series of MLLMs with o1-like step-by-step Reasoning and Reflection capabilities. Furthermore, extensive experiments, ablation studies and discussion demonstrate the superiority of our proposed methods on various benchmarks. We hope that CoMCTS along with Mulberry-260k and Mulberry will provides valuable resources and offer new insights for multimodal MCTS search and reasoning.

Table 3: **Ablation Study on Mulberry.** As Mulberry is trained with effective and reflective reasoning data searched by CoMCTS, we study their respective contributions.

Methods	MathVista
Baseline	43.0
w/o Reflection Data (+245K)	50.9
w/ Reflection Data (+15K)	51.7

References

- [1] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2024.
- [2] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.
- [3] Ruohong Zhang, Bowen Zhang, Yanghao Li, Haotian Zhang, Zhiqing Sun, Zhe Gan, Yinfei Yang, Ruoming Pang, and Yiming Yang. Improve vision language model chain-of-thought reasoning. *arXiv preprint arXiv:2410.16198*, 2024.
- [4] Guowei Xu, Peng Jin, Li Hao, Yibing Song, Lichao Sun, and Li Yuan. Llava-o1: Let vision language models reason step-by-step. *arXiv preprint arXiv:2411.10440*, 2024.
- [5] OpenAI. Introducing openai o1, 2024.
- [6] Yuxi Xie, Anirudh Goyal, Wenyue Zheng, Min-Yen Kan, Timothy P Lillicrap, Kenji Kawaguchi, and Michael Shieh. Monte carlo tree search boosts reasoning via iterative preference learning. *arXiv preprint arXiv:2405.00451*, 2024.
- [7] Rémi Coulom. Efficient selectivity and backup operators in monte-carlo tree search. In *International conference on computers and games*, pages 72–83. Springer, 2006.
- [8] Zhenting Qi, Mingyuan Ma, Jiahang Xu, Li Lyna Zhang, Fan Yang, and Mao Yang. Mutual reasoning makes smaller llms stronger problem-solvers. *arXiv preprint arXiv:2408.06195*, 2024.
- [9] Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, Jiao Sun, et al. Improve mathematical reasoning in language models by automated process supervision. *arXiv preprint arXiv:2406.06592*, 2024.
- [10] Mengdi Liu, Xiaoxue Cheng, Zhangyang Gao, Hong Chang, Cheng Tan, Shiguang Shan, and Xilin Chen. Protinvtree: Deliberate protein inverse folding with reward-guided tree search. *arXiv preprint arXiv:2506.00925*, 2025.
- [11] Haoyu Lu, Wen Liu, Bo Zhang, Bingxuan Wang, Kai Dong, Bo Liu, Jingxiang Sun, Tongzheng Ren, Zhuoshu Li, Hao Yang, et al. Deepseek-vl: towards real-world vision-language understanding. *arXiv preprint arXiv:2403.05525*, 2024.
- [12] Huanjin Yao, Wenhao Wu, Taojiannan Yang, YuXin Song, Mengxi Zhang, Haocheng Feng, Yifan Sun, Zhiheng Li, Wanli Ouyang, and Jingdong Wang. Dense connector for mllms. *arXiv preprint arXiv:2405.13800*, 2024.
- [13] Jingyi Zhang, Jiaxing Huang, Huanjin Yao, Shunyu Liu, Xikun Zhang, Shijian Lu, and Dacheng Tao. R1-vl: Learning to reason with multimodal large language models via step-wise group relative policy optimization. *arXiv preprint arXiv:2503.12937*, 2025.
- [14] Huanjin Yao, Qixiang Yin, Jingyi Zhang, Min Yang, Yibo Wang, Wenhao Wu, Fei Su, Li Shen, Minghui Qiu, Dacheng Tao, et al. R1-sharevl: Incentivizing reasoning capability of multimodal large language models via share-grpo. *arXiv preprint arXiv:2505.16673*, 2025.
- [15] Jingyi Zhang, Jiaxing Huang, Sheng Jin, and Shijian Lu. Vision-language models for vision tasks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.
- [16] Huanjin Yao, Ruifei Zhang, Jiaxing Huang, Jingyi Zhang, Yibo Wang, Bo Fang, Ruolin Zhu, Yongcheng Jing, Shunyu Liu, Guanbin Li, et al. A survey on agentic multimodal large language models. *arXiv preprint arXiv:2510.10991*, 2025.
- [17] Xiang Yue, Tianyu Zheng, Yuansheng Ni, Yubo Wang, Kai Zhang, Shengbang Tong, Yuxuan Sun, Botao Yu, Ge Zhang, Huan Sun, et al. Mmmu-pro: A more robust multi-discipline multimodal understanding benchmark. *arXiv preprint arXiv:2409.02813*, 2024.
- [18] Chancharik Mitra, Brandon Huang, Trevor Darrell, and Roei Herzig. Compositional chain-of-thought prompting for large multimodal models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14420–14431, 2024.
- [19] Bozhi Luan, Hao Feng, Hong Chen, Yonghui Wang, Wengang Zhou, and Houqiang Li. Textcot: Zoom in for enhanced multimodal text-rich image understanding. *arXiv preprint arXiv:2404.09797*, 2024.

- [20] 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.
- [21] 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, 2024.
- [22] Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, et al. Graph of thoughts: Solving elaborate problems with large language models. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 38, pages 17682–17690, 2024.
- [23] Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. Star: Bootstrapping reasoning with reasoning. Advances in Neural Information Processing Systems, 35:15476–15488, 2022.
- [24] Dan Zhang, Sining Zhou, Ziniu Hu, Yisong Yue, Yuxiao Dong, and Jie Tang. Rest-mcts*: Llm self-training via process reward guided tree search. arXiv preprint arXiv:2406.03816, 2024.
- [25] Sungjin Park, Xiao Liu, Yeyun Gong, and Edward Choi. Ensembling large language models with process reward-guided tree search for better complex reasoning. arXiv preprint arXiv:2412.15797, 2024.
- [26] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. nature, 550(7676):354–359, 2017.
- [27] Weirui Ye, Shaohuai Liu, Thanard Kurutach, Pieter Abbeel, and Yang Gao. Mastering atari games with limited data. Advances in neural information processing systems, 34:25476–25488, 2021.
- [28] Graeme Best, Oliver M Cliff, Timothy Patten, Ramgopal R Mettu, and Robert Fitch. Dec-mcts: Decentralized planning for multi-robot active perception. The International Journal of Robotics Research, 38(2-3):316–337, 2019.
- [29] Tuan Dam, Georgia Chalvatzaki, Jan Peters, and Joni Pajarinen. Monte-carlo robot path planning. IEEE Robotics and Automation Letters, 7(4):11213–11220, 2022.
- [30] Guillaume Lample, Timothee Lacroix, Marie-Anne Lachaux, Aurelien Rodriguez, Amaury Hayat, Thibaut Lavril, Gabriel Ebner, and Xavier Martinet. Hypertree proof search for neural theorem proving. Advances in neural information processing systems, 35:26337–26349, 2022.
- [31] Alhussein Fawzi, Matej Balog, Aja Huang, Thomas Hubert, Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Francisco J R Ruiz, Julian Schrittwieser, Grzegorz Swirszcz, et al. Discovering faster matrix multiplication algorithms with reinforcement learning. Nature, 610(7930):47–53, 2022.
- [32] Yelisey Pitanov, Alexey Skrynnik, Anton Andreychuk, Konstantin Yakovlev, and Aleksandr Panov. Monte-carlo tree search for multi-agent pathfinding: Preliminary results. In International Conference on Hybrid Artificial Intelligence Systems, pages 649–660. Springer, 2023.
- [33] Feiyu Yang. An integrated framework integrating monte carlo tree search and supervised learning for train timetabling problem. arXiv preprint arXiv:2311.00971, 2023.
- [34] Harshil Vagadia, Mudit Chopra, Abhinav Barnawal, Tamajit Banerjee, Shreshth Tuli, Souvik Chakraborty, and Rohan Paul. Phyplan: Compositional and adaptive physical task reasoning with physics-informed skill networks for robot manipulators. arXiv preprint arXiv:2402.15767, 2024.
- [35] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In Proceedings of the eleventh annual conference on Computational learning theory, pages 92–100, 1998.
- [36] Shiliang Sun and Feng Jin. Robust co-training. International Journal of Pattern Recognition and Artificial Intelligence, 25(07):1113–1126, 2011.
- [37] Shipeng Yu, Balaji Krishnapuram, Rómer Rosales, and R Bharat Rao. Bayesian co-training. The Journal of Machine Learning Research, 12:2649–2680, 2011.
- [38] Siyuan Qiao, Wei Shen, Zhishuai Zhang, Bo Wang, and Alan Yuille. Deep co-training for semi-supervised image recognition. In Proceedings of the european conference on computer vision (eccv), pages 135–152, 2018.

- [39] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 3723–3732, 2018.
- [40] Kaiwen Cui, Jiaxing Huang, Zhipeng Luo, Gongjie Zhang, Fangneng Zhan, and Shijian Lu. Genco: Generative co-training for generative adversarial networks with limited data. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 36, pages 499–507, 2022.
- [41] Jakob Foerster, Ioannis Alexandros Assael, Nando De Freitas, and Shimon Whiteson. Learning to communicate with deep multi-agent reinforcement learning. Advances in neural information processing systems, 29, 2016.
- [42] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. arXiv preprint arXiv:2410.21276, 2024.
- [43] Anthropic. Claude 3.5 sonnet, 2024.
- [44] Haotian Zhang, Mingfei Gao, Zhe Gan, Philipp Dufter, Nina Wenzel, Forrest Huang, Dhruvi Shah, Xianzhi Du, Bowen Zhang, Yanghao Li, et al. Mm1. 5: Methods, analysis & insights from multimodal llm fine-tuning. arXiv preprint arXiv:2409.20566, 2024.
- [45] Hugo Laurençon, Andrés Marafioti, Victor Sanh, and Léo Tronchon. Building and better understanding vision-language models: insights and future directions. In Workshop on Responsibly Building the Next Generation of Multimodal Foundational Models, 2024.
- [46] Zhe Chen, Weiyun Wang, Hao Tian, Shenglong Ye, Zhangwei Gao, Erfei Cui, Wenwen Tong, Kongzhi Hu, Jiapeng Luo, Zheng Ma, et al. How far are we to gpt-4v? closing the gap to commercial multimodal models with open-source suites. arXiv preprint arXiv:2404.16821, 2024.
- [47] Yuan Yao, Tianyu Yu, Ao Zhang, Chongyi Wang, Junbo Cui, Hongji Zhu, Tianchi Cai, Haoyu Li, Weilin Zhao, Zhihui He, et al. Minicpm-v: A gpt-4v level mllm on your phone. arXiv preprint arXiv:2408.01800, 2024.
- [48] Zhiyu Wu, Xiaokang Chen, Zizheng Pan, Xingchao Liu, Wen Liu, Damai Dai, Huazuo Gao, Yiyang Ma, Chengyue Wu, Bingxuan Wang, et al. Deepseek-vl2: Mixture-of-experts vision-language models for advanced multimodal understanding. arXiv preprint arXiv:2412.10302, 2024.
- [49] Yuhao Dong, Zuyan Liu, Hai-Long Sun, Jingkan Yang, Winston Hu, Yongming Rao, and Ziwei Liu. Insight-v: Exploring long-chain visual reasoning with multimodal large language models. arXiv preprint arXiv:2411.14432, 2024.
- [50] Bo Li, Kaichen Zhang, Hao Zhang, Dong Guo, Renrui Zhang, Feng Li, Yuanhan Zhang, Ziwei Liu, and Chunyuan Li. Llava-next: Stronger llms supercharge multimodal capabilities in the wild, May 2024.
- [51] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. arXiv preprint arXiv:2407.21783, 2024.
- [52] Pan Lu, Ran Gong, Shibiao Jiang, Liang Qiu, Siyuan Huang, Xiaodan Liang, and Song-Chun Zhu. Inter-gps: Interpretable geometry problem solving with formal language and symbolic reasoning. In The 59th Annual Meeting of the Association for Computational Linguistics (ACL), 2021.
- [53] Jiaqi Chen, Jianheng Tang, Jinghui Qin, Xiaodan Liang, Lingbo Liu, Eric P Xing, and Liang Lin. Geoqa: A geometric question answering benchmark towards multimodal numerical reasoning. arXiv preprint arXiv:2105.14517, 2021.
- [54] Levente Kocsis and Csaba Szepesvári. Bandit based monte-carlo planning. In European conference on machine learning, pages 282–293. Springer, 2006.
- [55] Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyang Luo, Zhangchi Feng, and Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. arXiv preprint arXiv:2403.13372, 2024.
- [56] Tianyu Yu, Yuan Yao, Haoye Zhang, Taiwan He, Yifeng Han, Ganqu Cui, Jinyi Hu, Zhiyuan Liu, Hai-Tao Zheng, Maosong Sun, et al. Rllhf-v: Towards trustworthy mllms via behavior alignment from fine-grained correctional human feedback. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 13807–13816, 2024.

- [57] Shiyin Lu, Yang Li, Qing-Guo Chen, Zhao Xu, Weihua Luo, Kaifu Zhang, and Han-Jia Ye. Ovis: Structural embedding alignment for multimodal large language model. [arXiv preprint arXiv:2405.20797](#), 2024.
- [58] Chuyang Zhao, YuXin Song, Junru Chen, Kang Rong, Haocheng Feng, Gang Zhang, Shufan Ji, Jingdong Wang, Errui Ding, and Yifan Sun. Octopus: A multi-modal llm with parallel recognition and sequential understanding. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.
- [59] Pan Zhang, Xiaoyi Dong, Yuhang Zang, Yuhang Cao, Rui Qian, Lin Chen, Qipeng Guo, Haodong Duan, Bin Wang, Linke Ouyang, Songyang Zhang, Wenwei Zhang, Yining Li, Yang Gao, Peng Sun, Xinyue Zhang, Wei Li, Jingwen Li, Wenhai Wang, Hang Yan, Conghui He, Xingcheng Zhang, Kai Chen, Jifeng Dai, Yu Qiao, Dahua Lin, and Jiaqi Wang. Internlm-xcomposer-2.5: A versatile large vision language model supporting long-contextual input and output, 2024.
- [60] Hugo Laurençon, Léo Tronchon, Matthieu Cord, and Victor Sanh. What matters when building vision-language models? [arXiv preprint arXiv:2405.02246](#), 2024.
- [61] Yanwei Li, Yuechen Zhang, Chengyao Wang, Zhisheng Zhong, Yixin Chen, Ruihang Chu, Shaoteng Liu, and Jiaya Jia. Mini-gemini: Mining the potential of multi-modality vision language models. [arXiv preprint arXiv:2403.18814](#), 2024.
- [62] Zhuosheng Zhang, Aston Zhang, Mu Li, Hai Zhao, George Karypis, and Alex Smola. Multimodal chain-of-thought reasoning in language models. [arXiv preprint arXiv:2302.00923](#), 2023.
- [63] Hao Shao, Shengju Qian, Han Xiao, Guanglu Song, Zhuofan Zong, Letian Wang, Yu Liu, and Hongsheng Li. Visual cot: Advancing multi-modal language models with a comprehensive dataset and benchmark for chain-of-thought reasoning. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024.
- [64] Jiahui Gao, Renjie Pi, Jipeng Zhang, Jiacheng Ye, Wanjun Zhong, Yufei Wang, Lanqing Hong, Jianhua Han, Hang Xu, Zhenguo Li, et al. G-llava: Solving geometric problem with multi-modal large language model. [arXiv preprint arXiv:2312.11370](#), 2023.
- [65] Minjoon Seo, Hannaneh Hajishirzi, Ali Farhadi, Oren Etzioni, and Clint Malcolm. Solving geometry problems: Combining text and diagram interpretation. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1466–1476, 2015.
- [66] Jiaqi Chen, Tong Li, Jinghui Qin, Pan Lu, Liang Lin, Chongyu Chen, and Xiaodan Liang. Unigeo: Unifying geometry logical reasoning via reformulating mathematical expression. [arXiv preprint arXiv:2212.02746](#), 2022.
- [67] Ke Wang, Junting Pan, Weikang Shi, Zimu Lu, Mingjie Zhan, and Hongsheng Li. Measuring multimodal mathematical reasoning with math-vision dataset. [arXiv preprint arXiv:2402.14804](#), 2024.
- [68] Mehran Kazemi, Hamidreza Alvari, Ankit Anand, Jialin Wu, Xi Chen, and Radu Soricut. Geomverse: A systematic evaluation of large models for geometric reasoning. [arXiv preprint arXiv:2312.12241](#), 2023.
- [69] Wenhao Shi, Zhiqiang Hu, Yi Bin, Junhua Liu, Yang Yang, See-Kiong Ng, Lidong Bing, and Roy Ka-Wei Lee. Math-llava: Bootstrapping mathematical reasoning for multimodal large language models. [arXiv preprint arXiv:2406.17294](#), 2024.
- [70] Kushal Kalle, Brian Price, Scott Cohen, and Christopher Kanan. Dvqa: Understanding data visualizations via question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5648–5656, 2018.
- [71] Minesh Mathew, Dimosthenis Karatzas, and CV Jawahar. Docvqa: A dataset for vqa on document images. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 2200–2209, 2021.
- [72] Samira Ebrahimi Kahou, Vincent Michalski, Adam Atkinson, Ákos Kádár, Adam Trischler, and Yoshua Bengio. Figureqa: An annotated figure dataset for visual reasoning. [arXiv preprint arXiv:1710.07300](#), 2017.
- [73] Nitesh Methani, Pritha Ganguly, Mitesh M Khapra, and Pratyush Kumar. Plotqa: Reasoning over scientific plots. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1527–1536, 2020.
- [74] Ahmed Masry, Do Xuan Long, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. Chartqa: A benchmark for question answering about charts with visual and logical reasoning. [arXiv preprint arXiv:2203.10244](#), 2022.

- [75] Minesh Mathew, Viraj Bagal, Rubèn Tito, Dimosthenis Karatzas, Ernest Valveny, and CV Jawahar. Infographicvqa. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pages 1697–1706, 2022.
- [76] Yilun Zhao, Yunxiang Li, Chenying Li, and Rui Zhang. Multihier: Numerical reasoning over multi hierarchical tabular and textual data. arXiv preprint arXiv:2206.01347, 2022.
- [77] Fuxiao Liu, Kevin Lin, Linjie Li, Jianfeng Wang, Yaser Yacoob, and Lijuan Wang. Mitigating hallucination in large multi-modal models via robust instruction tuning. In The Twelfth International Conference on Learning Representations, 2023.
- [78] Pan Lu, Liang Qiu, Jiaqi Chen, Tony Xia, Yizhou Zhao, Wei Zhang, Zhou Yu, Xiaodan Liang, and Song-Chun Zhu. Iconqa: A new benchmark for abstract diagram understanding and visual language reasoning. arXiv preprint arXiv:2110.13214, 2021.
- [79] Pan Lu, Liang Qiu, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, Tanmay Rajpurohit, Peter Clark, and Ashwin Kalyan. Dynamic prompt learning via policy gradient for semi-structured mathematical reasoning. arXiv preprint arXiv:2209.14610, 2022.
- [80] Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2901–2910, 2017.
- [81] Adam Dahlgren Lindström and Savitha Sam Abraham. Clevr-math: A dataset for compositional language, visual and mathematical reasoning. arXiv preprint arXiv:2208.05358, 2022.
- [82] Zhuowan Li, Xingrui Wang, Elias Stengel-Eskin, Adam Kortylewski, Wufei Ma, Benjamin Van Durme, and Alan L Yuille. Super-clevr: A virtual benchmark to diagnose domain robustness in visual reasoning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 14963–14973, 2023.
- [83] Jason J Lau, Soumya Gayen, Asma Ben Abacha, and Dina Demner-Fushman. A dataset of clinically generated visual questions and answers about radiology images. Scientific data, 5(1):1–10, 2018.
- [84] Xiaoman Zhang, Chaoyi Wu, Ziheng Zhao, Weixiong Lin, Ya Zhang, Yanfeng Wang, and Weidi Xie. Pmc-vqa: Visual instruction tuning for medical visual question answering. arXiv preprint arXiv:2305.10415, 2023.
- [85] Aniruddha Kembhavi, Minjoon Seo, Dustin Schwenk, Jonghyun Choi, Ali Farhadi, and Hannaneh Hajishirzi. Are you smarter than a sixth grader? textbook question answering for multimodal machine comprehension. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 4999–5007, 2017.
- [86] Aniruddha Kembhavi, Mike Salvato, Eric Kolve, Minjoon Seo, Hannaneh Hajishirzi, and Ali Farhadi. A diagram is worth a dozen images. In Computer Vision—ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14, pages 235–251. Springer, 2016.
- [87] Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Taffjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. Advances in Neural Information Processing Systems, 35:2507–2521, 2022.
- [88] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In Proceedings of the IEEE international conference on computer vision, pages 2425–2433, 2015.
- [89] Dustin Schwenk, Apoorv Khandelwal, Christopher Clark, Kenneth Marino, and Roozbeh Mottaghi. Askvqa: A benchmark for visual question answering using world knowledge. In European conference on computer vision, pages 146–162. Springer, 2022.
- [90] Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards vqa models that can read. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 8317–8326, 2019.
- [91] Danna Gurari, Qing Li, Abigale J Stangl, Anhong Guo, Chi Lin, Kristen Grauman, Jiebo Luo, and Jeffrey P Bigham. Vizwiz grand challenge: Answering visual questions from blind people. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 3608–3617, 2018.
- [92] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 6904–6913, 2017.

Appendix

We provide theorem statement and assumptions in Section A, implementation details in Section B, more discussions in Section C and additional experiments in Section D. In addition, we discuss the relation to other studies in Section E, provide the analysis of reasoning training data in Section F and qualitative results in Section G. Then, we provide the used prompts in Section H and the sources of the raw data in Section I. In the end, we provide the analysis with error bars in Section J and discussion about the limitations in Section K.

A Theorem Statement and Assumptions

Theorem A.1 (CoMCTS Consistency). *Let $\{\pi_1, \pi_2, \dots, \pi_K\}$ be a collection of models used in CoMCTS. Suppose the following conditions hold:*

1. *The search space is finite, or after pruning below a threshold t , it remains effectively traversable.*
2. *The return μ_s of each node s lies in a bounded interval $[a, b] \subset \mathbb{R}$.*
3. *Nonzero generation probability and unbiased evaluation:*
 - (a) *For each model π_k , there exists $\epsilon_k > 0$ such that the probability of generating an optimal branch is at least ϵ_k .*
 - (b) *The collective evaluation function*

$$R(s) = \frac{1}{K} \sum_{l=1}^K \pi_l(\cdot \mid \text{prompt}_{eval}, Q, \text{Parent}(s), s)$$

has bounded bias, with controllable variance, approximating the true node value μ_s .

4. *UCB Parameter: In the Selection phase, a UCB strategy with $c > 0$ is used.*
5. *Repeated-Visit Opportunity: All candidate nodes above threshold t can be revisited with positive probability; thus, as $T \rightarrow \infty$, each node not pruned is sampled infinitely often if it remains promising.*

Under these assumptions, as $T \rightarrow \infty$, the CoMCTS algorithm visits the optimal path infinitely often, and the value estimate $V(s)$ converges to the true value μ_s for each node.

Proof of Theorem A.1

Lemma A.2 (UCB Consistency [54]). *In the multi-armed bandit setting with bounded i.i.d. rewards, the UCB1 strategy*

$$\text{UCB}(k) = \bar{X}_k + c \sqrt{\frac{\ln n}{n_k}}$$

ensures that each suboptimal arm is pulled only finitely often, while the optimal arm is sampled infinitely often. Consequently, $\bar{X}_k \rightarrow \mu_k$ as $n \rightarrow \infty$.

Lemma A.3 (Ensemble Near-Unbiasedness and Variance Control). *Suppose $\hat{\mu}_1, \dots, \hat{\mu}_K$ are unbiased (or bounded-bias) estimators of some μ , each with bounded variance. Then their average remains close to μ in expectation and can significantly reduce variance compared to a single estimator.*

The proof proceeds by showing four key properties:

1. *Optimal children appear in the tree due to nonzero expansion probability ($\epsilon_k > 0$).*
2. *Optimal nodes are not pruned, via Hoeffding's inequality demonstrating that $R(s^*) > t$ with high probability if $\mu_{s^*} > t$.*
3. *UCB selection guarantees repeated visits for discovered optimal nodes.*
4. *Value estimates converge to the true values μ_s under repeated sampling.*

Optimal Children Appear in the Tree

Let s^* be a child node on the unknown optimal path. Each model π_k has probability $\epsilon_k > 0$ of generating s^* during *Expansion*. If these events are approximately independent across k , the joint probability of including s^* is

$$p_{\text{expansion}} = 1 - \prod_{k=1}^K (1 - \epsilon_k) > 0.$$

Hence, after enough expansions, s^* almost surely appears in the CoMCTS search tree.

Optimal Nodes Survive Threshold-Based Pruning

Suppose s^* is truly optimal, with $\mu_{s^*} > t$. In the *Simulation and Error Positioning* step, CoMCTS computes

$$R(s^*) = \frac{1}{K} \sum_{l=1}^K \pi_l \left(\cdot \mid \text{prompt}_{\text{eval}}, Q, \text{Parent}(s^*), s^* \right).$$

Let each individual score be X_l within a bounded interval. Under near-unbiasedness (Lemma A.3),

$$\mathbb{E}[X_l] \approx \mu_{s^*}, \quad \mathbb{E}[R(s^*)] \approx \mu_{s^*}.$$

Define $\delta = \mu_{s^*} - t > 0$. Then

$$\Pr[R(s^*) < t] = \Pr\left[\frac{1}{K} \sum_{l=1}^K X_l < t\right] = \Pr\left[\sum_{l=1}^K X_l - K \mu_{s^*} \leq -K \delta\right].$$

By Hoeffding's inequality, for X_l and approximate independence, there holds

$$\Pr\left[\sum_{l=1}^K X_l < K t\right] \leq \exp(-K \delta^2).$$

Therefore, a truly optimal node s^* with $\mu_{s^*} > t$ is pruned with exponentially small probability, implying s^* remains in the tree almost surely.

UCB Selection Guarantees Repeated Visits

Nodes not pruned enter the *Selection* stage, using

$$\text{UCB}(s) = V(s) + c \sqrt{\frac{\ln(N(\hat{s}))}{1 + N(s)}},$$

where $\hat{s}$ is the parent of s . By analogy to multi-armed bandits (Lemma A.2), if s^* is the highest-value child, it gets chosen infinitely often. Hence $N(s^*) \rightarrow \infty$ as $T \rightarrow \infty$.

Value Estimates Converge

When a node s returns score $R(s)$, backpropagation updates:

$$\begin{aligned} V(\hat{s}) &\leftarrow \frac{N(\hat{s}) V(\hat{s}) + \sum_{s_l \in \text{Child}(\hat{s})} R(s_l)}{N(\hat{s}) + \text{CountChild}(S_{\text{candidate}}^*, \hat{s})}, \\ N(\hat{s}) &\leftarrow N(\hat{s}) + \text{CountChild}(S_{\text{candidate}}^*, \hat{s}). \end{aligned}$$

As $N(s^*) \rightarrow \infty$, the empirical average $V(s^*)$ converges to μ_{s^*} by standard arguments (e.g. the law of large numbers for bounded returns). Ancestor nodes on the optimal path also receive increasingly accurate updates, ensuring $V(s) \rightarrow \mu_s$.

Combining all above results, we conclude that CoMCTS inherits the same consistency guarantees as standard MCTS, with optimal paths ultimately visited infinitely often and $V(s) \rightarrow \mu_s$.

B Training Details

B.1 Implementation Details

CoMCTS implementation Details. We implement collective learning in CoMCTS with four models, including GPT-4o, Qwen2-VL-7B, LLaMA-3.2-11B-Vision-Instruct, and Qwen2-VL-72B, to construct Mulberry-260K. In CoMCTS, we set maximum search iteration to 20 and threshold t in Eq. 3 to 0. In each iteration, each model generates one candidate reasoning path to balance search exploration and exploitation. In CoMCTS, we set the temperature to 0.9 for all models to balance creativity and determinism. For all other hyperparameters, we use the same default values as in the respective base models.

Model training details. In this paper, we use Mulberry-260K data searched by CoMCTS to train the Mulberry series models based on four popular MLLMs, including both the search models included in CoMCTS (*i.e.*, Qwen2-VL-7B and LLaMA-3.2-11B-Vision-Instruct) and the models not within the search (*i.e.*, Qwen2-VL-2B and LLaVA-NeXT-8B). We use the LLaMA-factory training framework [55] to train our models, using 8 NVIDIA H100 GPUs.

For the training scheme, all models are optimized for 2 epoch with the AdamW optimizer and a cosine learning schedule. We fine-tune the models with a batch size of 128 and employ DeepSpeed ZeRO-3 strategy to optimize memory consumption. A unified learning rate of $1e-5$ is used for LLaVA-NeXT-8B and LLaMA-3.2-11B-Vision-Instruct, while for Qwen2-VL-2B and Qwen2-VL-7B, the learning rates are adjusted to $2e-5$ and $5e-6$, respectively. As the number of tokens in the responses is significantly greater than that of direct simple answers, the training time increases accordingly, as detailed in Tab. 5.

B.2 Training Loss Curve

We provide more in-depth training details regarding the training loss curve in Fig. 4. Using the Mulberry-260K dataset for collective supervised fine-tuning, consistent and stable training is achieved across all four models, with the training loss steadily decreasing, demonstrating the training stability of Mulberry-260K searched by CoMCTS.

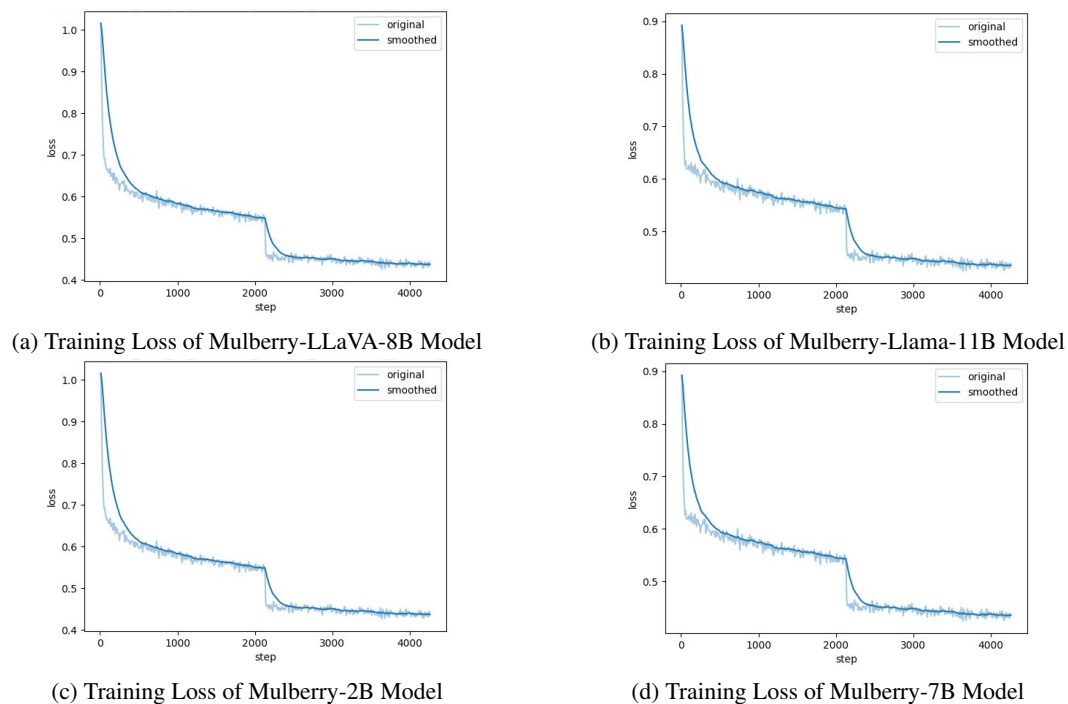


Figure 4: Training loss curve.

Table 5: **Training time.** We provide the training time for Mulberry series models trained on Mulberry-260K data generated by CoMCTS. The experiments are conducted on 8 NVIDIA H100 GPUs, using Llama-Factory framework [55].

Metrics	Mulberry-LLaVA-8B	Mulberry-Llama-11B	Mulberry-2B	Mulberry-7B
Training Time (hours)	12.5	11.2	4.5	6.0

C More Discussions

C.1 Detailed Comparison with the Existing CoT Training Dataset

We provide detailed comparison of Mulberry-260K with the existing CoT training dataset (i.e., LLaVA-Reasoner and LLaVA-CoT-100K) in Table 6. Table 6 compares key attributes such as dataset size (number of images and QA pairs), average token length of questions and responses, and the inclusion of reasoning and reflection. As shown, Mulberry-260K not only includes both reasoning and reflection, but also provides longer and more informative responses, underscoring its value as a CoT training resource.

Specifically, unlike LLaVA-Reasoner, which generates CoT reasoning by directly prompting GPT-4o, Mulberry-260K employs the proposed CoMCTS search, producing step-by-step reasoning paths that are not only longer but also feature more precise intermediate steps, benefiting from collective expansion and error positioning in CoMCTS.

Different to LLaVA-CoT, which uses structured reasoning with a fixed flow of certain pre-defined stages, Mulberry-260K leverages the proposed CoMCTS to dynamically search for both reasoning and reflection data, resulting in more detailed and logically coherent intermediate steps.

Furthermore, thanks to CoMCTS, Mulberry-260K is the first multimodal step-by-step reasoning dataset equipped with reflective capabilities, fostering deeper exploration of reflection in MLLMs.

Table 6: Detailed comparison with the existing CoT training dataset.

Datasets	Images	QA-Pairs	Avg Question token	Avg Response token	Reasoning	Reflection
LLaVA-Reasoner	~404K	~404K	31.4	122.0	Y	N
LLaVA-CoT-100K	~100K	~255K	23.2	212.4	Y	N
Mulberry-260K	~260K	~260K	31.9	234.5	Y	Y

C.2 Correctness of the Generated Reasoning Paths

As discussed in Limitation, neither our method nor advanced reasoning models (e.g., o1, QvQ, DeepSeek-R1) can ensure every step is 100% correct. Similar limitations are noted in QvQ. Here, we provide the manual evaluation of 100 reasoning paths. As shown in Table 7, our CoMCTS yields a much lower error rate in reasoning paths compared to MCTS.

Table 7: Manual evaluation of the correctness of the CoMCTS-generated reasoning paths.

Methods	Reasoning Path Error Rate
MCTS	42%
CoMCTS	18%

C.3 Training MLLMs on CoMCTS vs. Other Tree Search Data.

We examine the quality of the CoMCTS generated data by providing the experiments using data generated from standard MCTS. As shown in Table 8, models are trained on 26K samples searched by MCTS and CoMCTS, respectively. By incorporating collective knowledge, CoMCTS generates more accurate reasoning paths, leading to a 2.1% performance improvement.

Table 8: Training MLLMs on CoMCTS vs. other Tree Search data.

Methods	MathVista
LLaMA-V-11B	48.6
MCTS	50.3
CoMCTS	52.4

D Additional Experiments

In this section, we include additional experiments on CoMCTS and Mulberry models. Unless otherwise specified, the experiments in this section are based on the Qwen2-VL [2] model.

D.1 More Detailed Ablation Study on CoMCTS

We provide a more detailed analysis of search success rate in the CoMCTS ablation study in Table 9, studying the direct predictions of search models and their performance in CoMCTS. We can observe that due to the model’s limited reasoning capability, directly performing reasoning prediction with a single model yields poor performance. When using only Qwen2-VL-7B, the success rate of direct predictions is just 37.4%, and even the powerful closed-source model GPT-4o achieves only 58.2% accuracy in direct reasoning predictions. However, by leveraging tree search and collective knowledge, our proposed CoMCTS significantly improves the search success rate to 80.2% with four search models, demonstrating the effectiveness of CoMCTS.

Table 9: **Ablation Study on CoMCTS.** We study how each model in CoMCTS collective learning contribute to overall tree search performance in Search Success Rate (S.S.R.).

Direct Prediction				CoMCTS				S.S.R.
GPT-4o	Qwen2-VL-7B	Llama3.2-11B	Qwen2-VL-72B	GPT-4o	Qwen2-VL-7B	Llama3.2-11B	Qwen2-VL-72B	
✓								58.2
	✓							37.4
		✓						35.4
			✓					48.3
				✓				63.8
				✓	✓			66.2
				✓	✓	✓		69.7
				✓	✓	✓	✓	80.2

D.2 Parameter Studies

Threshold t . We conduct a parameter study on the threshold t of Equation 3 in CoMCTS, analyzing its impact on search success rate and average search iterations, as shown in Table 10. In the CoMCTS tree, the node value V represents the quality of this reasoning step, and the threshold t determines whether the node is removed based on its value, thereby regulating the overall quality of nodes. Setting t to a higher value, *i.e.*, $t = 0.2$, enhances the quality at each step but also slows down the expansion of nodes. As the quality of each node improves, search models are more likely to find the correct path within the collective search space, leading to an increase in search success rate. However, as node expansion slows, the average search iterations increase significantly. Setting t to a smaller value, *i.e.*, $t = -0.2$, increases the likelihood of nodes containing irrelevant information, disrupting the reasoning process. This results in a slight decrease in search success rate and a modest increase in average search iterations. On the other hand, setting the threshold as zero, *i.e.*, $t = 0$, achieves a great trade-off in search success rate and average search iterations, balancing search effectiveness and efficiency.

Temperature and repetition penalty. As shown in Tables 11-12, we also conduct extra hyperparameter studies (e.g., temperature and repetition penalty), indicating our method is tolerant to these parameters.

Table 10: **Parameter Study of Threshold t .** We study the impact of different threshold values t in Equation 3 of CoMCTS on the search success rate and average search iterations.

Threshold	Search Success Rate	Average Search Iteration
-0.2	79.8	12.9
0	80.2	12.7
0.2	80.9	16.3

Table 11: **Parameter Study of Temperature.**

Temperature	Search Success Rate
0.7	80
0.8	79
0.9 (we use)	81
1.0	80
1.1	78

D.3 Discussion on Step Separators

We conduct experiments to discuss the impact of different step separators in Table 13. First, the absence of step separators results in disorganized reasoning logic, leading to a notable decline in model performance, with an accuracy of only 48.4% on MathVista. This result highlights the importance of step separators in reasoning, as they enhance MLLMs' ability to comprehend the logical structure within reasoning paths. Next, we compare the special tokens separators (*e.g.*, `<step_1>`, `<step_2>`, etc) with context separators (*e.g.*, `###`, as illustrated in Figure 7). The former requires adding special tokens to the vocabulary, while the latter relies on predefined formatting rules. The results in Table 13 show that the context separators method outperform the special token separators, achieving an accuracy of 51.7% compared to 50.6%. This performance difference may stem from models' inherent tendency to use markdown formatting (*e.g.*, `###`) to highlight answers and separate steps, which could lower the cognitive associated with learning this pattern in limited data.

D.4 Discussing the Importance of Reasoning

We discuss the importance of step-by-step reasoning in Tab. 14. Specifically, using the same 260K multimodal data, we compare the results trained with direct answers to those trained with step-by-step reasoning searched by CoMCTS. Experimental results show that training with reasoning responses can substantially enhance model performance. For example, on MathVista, it outperforms direct answer training by a margin of 4.9%. This improvement demonstrates the potential of reasoning to enhance MLLM performance.

D.5 Training with Different Proportions of Mulberry-260K

As shown in Tab. 15, we conduct the suggested studies by training models using different proportions of Mulberry-260K data over the base model LLaMA-3.2-Vision-11B-Instruct. We can observe that Mulberry consistently improves performance across all Mulberry-260K data proportions, demonstrating its effectiveness in limited-data scenarios. Besides, the performance gains increase steadily as more Mulberry-260K data is used, highlighting its strong scalability.

D.6 Comparison with Base Models Trained using Other CoT Data

As shown in Table 16, we conduct the comparison for analyzing the impact of different CoT training datasets on the same base model. Using the LLaMA3-LLaVA-NeXT-8B base model, despite using less training data, our Mulberry-LLaVA-8B outperforms LLaVA-Reasoner-8B by +3.1% in average performance, which is largely because LLaVA-Reasoner learns reasoning responses directly distilled from GPT-4o while our Mulberry learns from more effective reasoning data searched by the proposed CoMCTS.

Similarly, with the LLaMA-3.2-Vision-11B-Instruct base model, our Mulberry-LLaMA-11B achieves better results using only 100K QA pairs, significantly fewer than the 255K QA pairs used by

Table 12: **Parameter Study of Repetition Penalty.**

Repetition penalty	Search Success Rate
0.8	78
0.9	79
1.0 (we use)	81
1.1	79
1.2	76

Table 13: **Discussion on Step Separators.** We study the impact of different reasoning step separators on model performance.

Methods	MathVista Accuracy
w/o Separators	48.4
Special Token Separators	50.6
Context Separators	51.7

Table 14: **Discussing the Importance of Reasoning.** We compared the results of models trained on direct answer data with those trained on Mulberry-260K reasoning data.

Benchmark	Baseline	Direct Answers	Step-by-step Reasoning
MathVista	43.0	46.8	51.7

LLaVA-CoT-11B. When scaling our dataset to 260K QA pairs, Mulberry-LLaMA-11B outperforms LLaVA-CoT-11B by +2.7% on average, further demonstrating the advantages of CoMCTS-searched reasoning paths over the pre-defined structured reasoning methods in LLaVA-CoT.

E Relation to Other Studies

E.1 Relation to Other MCTS Methods

We summarize the contributions of CoMCTS and highlight the differences compared to previous MCTS methods [24, 9, 7]: (a) This paper presents CoMCTS, a new learning-to-reason approach for MLLMs, which introduces the concept of collective learning into “tree search” for effective and efficient reasoning-path searching and learning. (b) The joint expansion mechanism enables CoMCTS to concatenate reasoning trajectories from multiple MLLMs via iterative search, ultimately constructing a unified reasoning tree comprising diverse and complementary reasoning nodes. This allows reasoning-path searches across multiple models, leveraging their synergy while avoiding traps in low-quality, homogeneous nodes within a single MLLM’s reasoning space. (c) In each search iteration, CoMCTS skips multiple intermediate steps and selects the last correct step as the next start node in the joint simulation and error positioning operation, largely reducing search time while maintaining search effectiveness. Additionally, collective knowledge enhances error positioning, allowing models to better identify errors through mutual validation.

E.2 Relation to Other Reasoning MLLMs

Multimodal large language models [56, 57, 58, 59, 11, 60, 61] have made remarkable progress, with recent MLLMs increasingly focusing on reasoning and intermediate steps [62, 63, 4, 3, 49]. We summarize the contributions of our models and highlight the differences compared to previous reasoning MLLMs: (a) Based on the CoMCTS, we search effective and reflective reasoning paths for multimodal inputs, and construct Mulberry-260k, a multimodal reason and reflect dataset with a tree of rich, explicit and well-defined reasoning nodes for each question. Mulberry-260k is then used to train reasoning model Mulberry. (b) Unlike pre-defined stage-based methods, CoMCTS searches for step-level reasoning data of varying lengths. Training Mulberry models on these CoMCTS-searched reasoning paths equips it with the ability to perform flexible step-by-step reasoning. (c) To the best of our knowledge, Mulberry is the first work to explore the reflective capabilities of MLLMs through tree search and collective knowledge.

Table 15: Training with different proportions of Mulberry-260K.

Models	Percentage of Mulberry-260K	QA-Pairs	MathVista	MMStar	MMMU	AVG
LLaMA-3.2-V-11B (Baseline)	-	-	48.6	49.8	41.7	46.7
	10%	~26k	52.4	53.1	42.8	49.4
Mulberry-11B	30%	~78k	54.0	56.1	43.6	51.2
	50%	~130k	58.1	57.9	44.9	53.6
	100%	~260k	61.1	58.5	45.6	55.1

Table 16: Comparison of training with different CoT datasets on the same base model.

Models	Base Model	CoT Data	QA-Pairs	MathVista	MMStar	MMMU	AVG
LLaMA3-LLaVA-NeXT-8B	-	-	-	37.5	42.1	41.7	40.4
LLaVA-Reasoner-8B	LLaMA3-LLaVA-NeXT-8B	LLaVA-Reasoner-404K	~404K	50.6	54.0	40.0	48.2 (+7.8)
Mulberry-LLaVA-8B	LLaMA3-LLaVA-NeXT-8B	Mulberry-260K	~260K	56.3	54.5	43.0	51.3 (+10.9)
LLaMA-3.2-Vision-11B-Instruct	-	-	-	48.6	49.8	41.7	46.7
LLaVA-CoT-11B	LLaMA-3.2-Vision-11B-Instruct	LLaVA-CoT-100K	~255K	54.8	57.6	44.9	52.4 (+5.7)
Mulberry-Llama-11B	LLaMA-3.2-Vision-11B-Instruct	Mulberry-100K	~100K	56.8	57.8	44.7	53.1 (+6.4)
Mulberry-Llama-11B	LLaMA-3.2-Vision-11B-Instruct	Mulberry-260K	~260K	61.1	58.5	45.6	55.1 (+8.4)

F Analysis of Reasoning Training Data

F.1 Analysis of Reasoning Data Distribution.

We analyze the CoMCTS-searched reasoning paths in Mulberry-260K by examining the distribution of reasoning steps, as shown in Figure 5. Figure 5 shows that reasoning steps predominantly fall between 6 and 8, with an average of 7.5 across the entire Mulberry-260k. For simple reasoning tasks, the chart-related subset of Mulberry-260k, reasoning steps typically ranges from 6 to 7, averaging 6.8. In contrast, for complex mathematical and logical reasoning tasks, such as the geometry-related subset of Mulberry-260k, the distribution shifts to 7 and 10 steps, with an average of 8.9. These observations highlight that CoMCTS’s collective tree search design generates effective reasoning trajectories with flexible numbers of steps, learning from which allows to train a powerful MLLM with reasoning flexibility, *i.e.*, a model can “think less and faster” when handling simple questions (*i.e.*, allocate and generate fewer intermediate reasoning steps) and “think more and slower” when tackling complex tasks (*i.e.*, allocate and generate a greater number of intermediate reasoning steps).

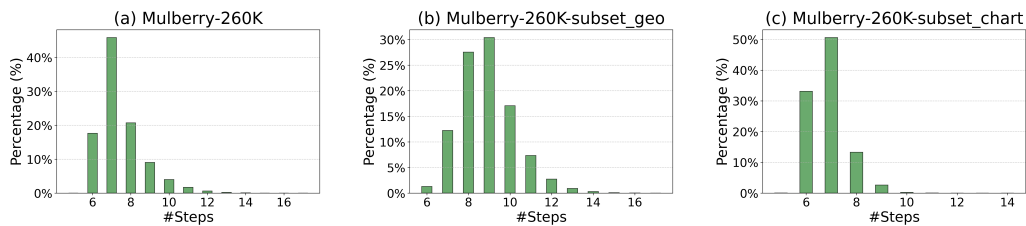


Figure 5: Distribution of reasoning steps in Mulberry-260K data.

F.2 Analysis of Effective Reasoning Training Data

We provide an example of effective reasoning data generated by CoMCTS in Figure 6. In this reasoning path, an accurate image analysis is first searched, along with reasonable rationales, allowing CoMCTS to subsequently search logical and correct step-by-step reasoning based on precise perception and rationales, ultimately deriving the correct answer. This clear reasoning largely due to CoMCTS’s collective expansion and collective error positioning, allowing search models to complement each other, refine error identification, and iteratively improve reasoning paths.

F.3 Analysis of Reflective Reasoning Training Data

We provide an example of reflective training data generated by CoMCTS in Figure 7. In this reasoning path, one search model identifies correct reasoning path after several iterations, while CoMCTS’s

collective error positioning finds a perception error at Step 1 in another path. CoMCTS then integrates the incorrect Step 1 into the correct reasoning path (*i.e.*, before its sibling nodes) with a reflective prompt to create reflective data. Here, incorrect Step 1 misjudges the proportion of "Medium Orchid", but later steps identify and correct the error. Fine-tuning with this reflective data equips the model with the ability to reassess previous steps when generating the next step, leading to more accurate reasoning and improves performance.

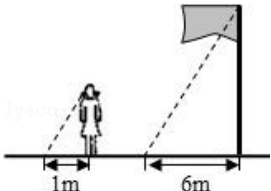
G Qualitative Results

G.1 Qualitative Comparison

We provide a qualitative comparison of LLaVA-NeXT-8B [50], Qwen2-VL-7B [2], and Mulberry-7B in Figure 8. LLaVA-NeXT-8B and Qwen2-VL-7B generate relatively short predictions without thorough thinking, leading to incorrect answers. In contrast, our Mulberry, trained with CoMCTS-searched reasoning data, creates rich, explicit and well-defined reasoning steps with comprehensive understanding, ultimately arriving at the correct answer.

G.2 Reflective Qualitative Results

We present a reflective qualitative results of Mulberry in Figure 9. Fine-tuning on reflective reasoning data, Mulberry models develop the capability to reflect on previously generated step. Here, Mulberry initially misidentifies the accuracy of the parent but, through continuous reflection, detects the error at Step 2, corrects it, and ultimately arrives at the correct answer.



Question: If Xiaohua's shadow measures 1.0 units at the same time the flagpole's shadow measures 6.0 units, and Xiaohua's height is approximately 1.6 units, what is the estimated height of the flagpole?
Choices: A: 6.6m B: 7.6m C: 8.6m D: 9.6m

Correct Answer: D

Image Description:
The image depicts Xiaohua standing next to a flagpole, with the shadows of both objects cast on the ground. Xiaohua's shadow is marked as measuring 1 meter, while the flagpole's shadow is marked as measuring 6 meters.

Rationales:
The problem provides measurements of shadows and the height of Xiaohua. Using the properties of similar triangles, the question seeks the height of the flagpole.

Let's think step by step.

Step 1:
Identify the relationship between Xiaohua's height and her shadow. The ratio of Xiaohua's height to her shadow is 1.6:1.0.

Step 2:
Use the same ratio for the flagpole to find its height. The ratio of the flagpole height to its shadow will be the same as Xiaohua's.

Step 3:
Set up the proportion using the known quantities:

$$\frac{\text{Xiaohua's Height}}{\text{Xiaohua's Shadow}} = \frac{\text{Flagpole's Height}}{\text{Flagpole's Shadow}}$$

Step 4:
Substitute in the provided values:

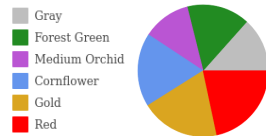
$$\frac{1.6}{1.0} = \frac{h}{6.0}$$

Step 5:
Solve for h (flagpole's height):

$$h = 1.6 \times 6.0 = 9.6$$

The final answer is: D

Figure 6: An example of effective reasoning training data.



Question: Is Medium Orchid the minimum?
Answer yes or no.

Correct Answer: Yes

Image Description:
A pie chart divided into six segments representing different colors. Each segment has a label indicating its color. The colors include Gray, Forest Green, Medium Orchid, Cornflower, Gold, and Red.

Rationales:
- The pie chart visually represents proportions of each category (color).
- The size of each slice indicates how much of the whole each category occupies.
- Since the question asks if Medium Orchid is the minimum, we need to compare its slice size against others.

Let's think step by step.

Step 1:
Identify the sizes of all slices.
- Gray: Smallest slice
- Forest Green: Second smallest slice
- **Medium Orchid: Third smallest slice**
- Cornflower: Fourth largest slice
- Gold: Fifth largest slice
- Red: Largest slice

I think I have made a mistake. Let me rethink it.

Step 1:
Identify the slice labeled "Medium Orchid" on the pie chart.

Step 2:
Visually compare the size of the Medium Orchid slice to each of the other slices (Gray, Forest Green, Cornflower, Gold, Red).

Step 3:
Determine if the Medium Orchid slice is the **smallest** among all the slices.

Step 4:
Validate the comparison by checking if any other slice is smaller than the Medium Orchid slice.

The final answer is:
Yes.

Figure 7: An example of reflective reasoning training data.



Question: Use a sector paper sheet with a central angle of 120.0 and a radius of 6.0 to roll into a conical bottomless paper cap (as shown in the picture), then the bottom perimeter of the paper cap is ()

Choices:
(A) 2 π cm; (B) 3 π cm;
(C) 4 π cm; (D) 5 π cm

Correct Answer: (C)

LLaVA: We need to find the derivative of $f(x)$ at $x=2$ and compare it with the derivative of $f(x)$ at $x=5$. We can do this by using the power rule of differentiation, which states that if $f(x) = x^n$, then $f'(x) = nx^{n-1}$. At $x=2$, $f'(2) = 2^4(n-1) = 2^4 \cdot 3 = 8$. At $x=5$, $f'(5) = 5^4(n-1) = 5^4 \cdot 3 = 125$. Therefore, the derivative of $f(x)$ at $x=2$ is larger than that at $x=5$. The correct answer is (A).

Qwen2-VL: First, let's find the circumference of the base of the cone. The circumference of the base is equal to the length of the arc of the sector paper sheet.

The formula for the circumference of a circle is $C = 2\pi r$, where r is the radius. In this case, the radius of the base of the cone is equal to the radius of the sector paper sheet, which is 6.0 cm.

So, the circumference of the base of the cone is: $C = 2\pi \times 6.0 = 12\pi$ cm.

However, the question asks for the bottom perimeter of the paper cap, which is the circumference of the base of the cone. Therefore, the bottom perimeter of the paper cap is: 12π cm. So, the correct answer is D

Mulberry:
Image Description:
A diagram shows a circular sector being transformed into a cone without its base. The sector has a central angle labeled as 120° and a given radius length of 6.

Rationales:
To find the circumference or "bottom perimeter" of the resulting cone from rolling up this sector, we need to understand that when you form a cone using such a sector, the arc length becomes the circumference of the base circle of the cone.

Let's think step by step.

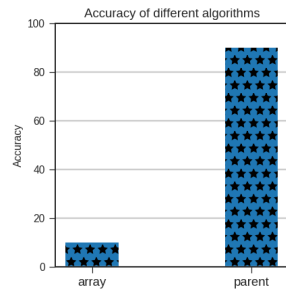
Step 1:
Calculate the arc length of the sector which forms the circumference of the cone's base. Arc Length = central angle/360 $\times$ 2 π r. Here, $r=6$ cm and central angle = 120°.

Step 2:
Substitute values into the formula:
Arc Length = (120/360) $\times$ 2 π $\times$ 6.
Simplify it further to get the actual arc length.

Step 3:
Perform calculations: Arc Length = 1/3 $\times$ 12 π = 4 π . This means the circumference of the cone's base equals 4 π .

The final answer is:
C

Figure 8: **Qualitative Comparison.** Our Mulberry, trained with CoMCTS-searched reasoning data, creates rich, explicit and well-defined reasoning steps with comprehensive understanding, ultimately arriving at the correct answer.



Question: How many algorithms have accuracies higher than 90?

Correct Answer: Zero.

Annotation: array (10), parent (90)

Image Description:
A bar chart titled "Accuracy of different algorithms" shows two bars representing algorithm accuracies. The first bar labeled "array" has a height around 10, while the second bar labeled "parent" reaches up to about 95.

Rationales:
- The x-axis represents the names of the algorithms (array and parent).
- The y-axis measures accuracy from 0 to 100.
- Each algorithm has one corresponding bar indicating its accuracy level.

Let's think step by step.

Step 1:
Identify the accuracy levels for each algorithm.
- Array: Approximately 10%
- Parent: Approximately 95%

Step 2:
Determine which algorithms exceed an accuracy of 90%.
- Only the "parent" algorithm exceeds this threshold.
I think I have made a mistake. Let me rethink it.

Step 2:
Compare these values against the threshold of 90%.

Step 3:
Conclude how many algorithms meet or exceed this criterion. Based on the comparison:
- No algorithm reaches above 90%.

The final answer is:
Zero.

Figure 9: **Reflective Qualitative Results.** Our Mulberry, trained with CoMCTS-searched reflection data, generated reflective reasoning, ultimately arriving at the correct answer.

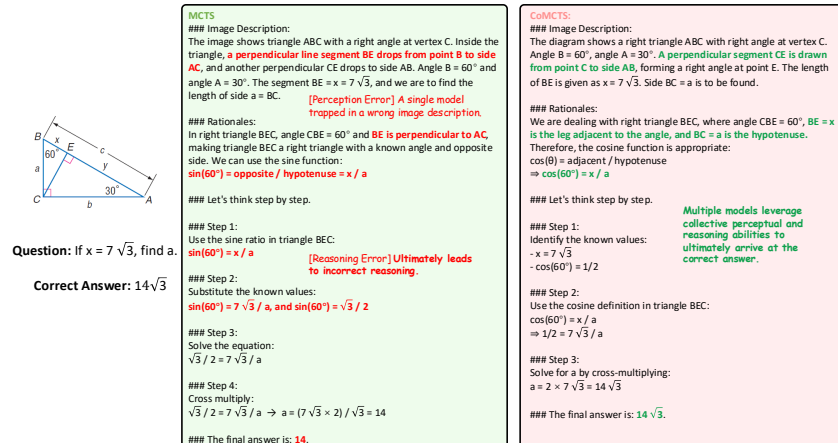


Figure 10: An example to compare CoMCTS with other tree search methods.

G.3 Examples to Compare CoMCTS with Other Tree Search Methods

We provided a qualitative comparison between the CoMCTS and single model MCTS.

When a single model performs step-by-step reasoning, an error in the initial image understanding can lead to a cascade of incorrect reasoning steps, ultimately resulting in a wrong answer. However, when multiple models are used collaboratively for reasoning, if one model fails to correctly interpret the image, others can help identify the correct steps. By cross-verifying and complementing each other's interpretations, the ensemble can jointly search for and arrive at the correct answer.

As illustrated in Figure 10, the single-model MCTS failed to correctly interpret the image and got trapped in an erroneous node, leading to a reasoning path that consistently failed to arrive at the correct answer.

H Prompt

In this section, we present the prompts involved in different operations of CoMCTS to execute various task instructions. For expansion in operation (a), we use the prompt in Figure 11 and Figure 12. The

prompt in Figure 11 is used to expand the root node for the closed-source model GPT-4o and all nodes for the open-source models. It is worth noting that, since open-source models can directly and forcefully incorporate preceding reasoning steps, allowing the model to generate subsequent steps based on previous responses, whereas closed-source models cannot. Therefore, we design specific prompts for closed-source models to expand non-root nodes. For GPT-4o, we use the prompt shown in Figure 12 to expand non-root nodes in the existing search tree and generate subsequent reasoning steps. For error positioning in operation (b), we use the prompt shown in Figure 13 to evaluate the correctness of each step. Each step is scored as correct, neutral, or incorrect, with scores of 1, 0, and -1, respectively. The evaluation results for each model are then combined through a weighted average. Finally, we use the prompt in Figure 14 to verify if the final answer from reasoning path matches the ground truth.

```

Generate an image description based on the question.
Then, provide a rationale to analyze the question.
Next, generate a step-by-step reasoning process to solve the problem. Ensure the steps are logical
and concise.
Finally, provide a concise summary of the final answer in the following format: 'The final answer is:
xxx'. If the question is multiple-choice, provide the options along with their content. If it is free-
form, directly present the final result. Do not provide any explanation.

Format your response with the following sections, separated by ###:
### Image Description:
### Rationales:
### Let's think step by step.
### Step 1:
### Step 2:
...
### The final answer is:

{QUESTION}

```

Figure 11: **The prompt for expansion.** The prompt expands root nodes in closed-source and all nodes in open-source models.

```

Generate an image description based on the question.
Then, provide a rationale to analyze the question.
Next, generate a step-by-step reasoning process to solve the problem. Ensure the steps are logical
and concise.
Finally, provide a concise summary of the final answer in the following format: 'The final answer is:
xxx'. If the question is multiple-choice, provide the options along with their content. If it is free-
form, directly present the final result. Do not provide any explanation.

Format your response with the following sections, separated by ###:
### Image Description:
### Rationales:
### Let's think step by step.
### Step 1:
### Step 2:
...
### The final answer is:

{QUESTION}

Please complete the response based on the reasoning prefix without altering its content.

Reasoning prefix: {REASONING_PREFIX}

```

Figure 12: **The prompt for expansion.** The prompt expands non-root nodes in closed-source models.

I The Sources of Raw Data

To construct a comprehensive and general-purpose tree-based reasoning dataset, we collect 260K raw multimodal input questions spanning various domain, including

- 55K Mathematical Data: From GLLaVA [64], GEOS [65], UniGeo [66], GeoQA Plus [53], Geo3K [52], MathVision [67], GeoMverse [68], and MathV360K [69]. These datasets cover a broad spectrum of mathematical problems, where solving requires extensive reasoning

```

### Question:
{QUESTION}

### Ground truth answer:
{GT_ANSWER}

### Reasoning steps:
{REASONING}

Given the question and reasoning steps listed above, along with the corresponding ground truth answer, please evaluate the correctness of the image description, rationales, and each step of the reasoning process.

Requirements:
1. Output the decision ("correct", "neutral", "incorrect") for each step following the format of "Final Decision:\nImage Description: [your decision]; Rationales: [your decision]; Let's think step by step: [your decision]; Step 1: [your decision]; Step 2: [your decision]; ...";
2. Do not provide any explanation.

```

Figure 13: The prompt for error positioning.

```

Evaluate whether the model's answer matches the correct result.

- If it does not align, respond with 'No'.
- If the model's answer aligns with the correct result, respond with 'Yes'.

Provide only 'Yes' or 'No' as the output, with no explanation.

The question is: {QUESTION}

The model's answer is: {MODEL_ANSWER}

The correct result is: {GT_ANSWER}

```

Figure 14: The prompt for evaluating the final result.

and multiple logical steps, highlighting their value in Mulberry-260K. We apply CoMCTS search to the raw data from these datasets to generate reasoning paths, which are then used to train Mulberry models, equipping them with advanced mathematical and logical reasoning skills.

- 116K Figure Understanding data: From DVQA [70], DocVQA [71], FigureQA [72], PlotQA [73], ChartQA [74], InfoVQA [75], MultiHiertt [76], and LRV-Chart [77]. These datasets cover various figure types and understanding tasks, including charts, bar graphs, pie charts, histograms, etc. Training on these data searched by CoMCTS equips the model with figure reasoning capabilities such as table computation, information retrieval, and trend analysis.
- 41K Math Word Problem Data: From IconQA [78], TabMWP [79], CLEVR [80], CLEVR-Math [81], and Super-CLEVR [82]. These datasets span various mathematical word reasoning tasks. Training on them in Mulberry-260K strengthens the model's counting, arithmetic, and logical deduction skills, enhancing its accuracy and interpretability in solving complex math word problems.
- 2K Medical Data: From VQA-RAD [83], and PMC-VQA [84]. These datasets focus on medical visual understanding and reasoning, encompassing a variety of radiology images and diseases. Leveraging this subset enhances the model's capability in medical image comprehension and diagnostic reasoning.
- 17K Science Data: From TQA [85], AI2D [86], and ScienceQA [87]. These datasets in CoMCTS plays a critical role in enhancing the model's ability to tackle complex scientific problems, perform multi-modal reasoning, and interpret scientific illustrations and textual descriptions cohesively.
- 24K Nature World QA Data: From VQA-AS [88], A-OKVQA [89], TextVQA [90], Vizwiz [91], and VQA2.0 [92]. These datasets encompass tasks involving natural scenes, textual elements, and open-ended visual question answering, challenging models to interpret complex visuals, understand embedded text, and generate accurate responses. This subset

plays a key role in enhancing the model’s ability to reason about real-world visual and textual content across diverse contexts.

J Error Bars

We provide the error bars of different Mulberry models on the MathVista benchmark, as shown in Tab. 17. We conduct five repeated evaluations to obtain the error bars, and the results indicate that the error bars fall within a narrow range.

Table 17: **Error Bars.** The experimental results are based on five repeated evaluations.

Benchmark	Mulberry-LLaVA-8B	Mulberry-Llama-11B	Mulberry-2B	Mulberry-7B
MathVista	56.3 ± 0.3	61.1 ± 0.2	51.7 ± 0.3	63.1 ± 0.2

K Limitations

Mulberry is a preliminary exploration work in o1-like MLLM, leveraging Collective Monte Carlo Tree Search to enable effective and efficient reasoning-path searching and learning. CoMCTS leverages collective knowledge to significantly improve the search success rate and efficiency of reasoning path search. By training on the reasoning data generated through CoMCTS, Mulberry has gained step-by-step reasoning capabilities, leading to a substantial improvement in overall performance. Nevertheless, certain limitations must be acknowledged.

Hallucinations in intermediate steps. Hallucinations are still prevalent in MLLMs, whether in closed or open-source models. For instance, the models may generate obvious errors in intermediate reasoning steps yet still arrive at the correct final answer in CoMCTS. Therefore, although we incorporated multiple models to better detect errors, some errors still persist in the intermediate steps because ensuring the correctness of all intermediate steps often requires human checks, which is extremely costly and unaffordable for us.

Error localization. During our experiments, we observed that models struggle to detect their own errors. To address this, CoMCTS employs multiple models to cross-check each other’s errors. However, our findings also revealed that smaller models often fail to generate effective detection responses, while larger models occasionally exhibit inaccurate error localization. Thus, inaccurate localization may impact the efficiency of the search and we recommend using larger models for error localization or exploring better prompts to enable smaller models to localize errors more accurately.

Reflective capability. Mulberry is an early-stage work in exploring reflective capabilities within the field of MLLMs. We would like to clarify that since reflective data constitutes only a small portion of the entire dataset, the Mulberry model generates reflective responses occasionally. We hope to further explore the model’s reflective capabilities in future work.

NeurIPS Paper Checklist

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [Yes]

Justification: We provide our contributions and scope both in the abstract and introduction.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [Yes]

Justification: We discuss the limitation in the paper.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: This paper does not include new theoretical results

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [\[Yes\]](#)

Justification: Yes, we provide all necessary information to reproduce the experimental results.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [\[Yes\]](#)

Justification: We will provide the anonymous code, and our code and data will be publicly available.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [\[Yes\]](#)

Justification: The training and test details are described in the paper.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [\[Yes\]](#)

Justification: We report error bars based on multiple runs on part of the benchmarks.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [\[Yes\]](#)

Justification: We provide this information in the paper.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [\[Yes\]](#)

Justification: Our research follows the NeurIPS Code of Ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [\[Yes\]](#)

Justification: There is no societal impact of the work performed.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The paper poses no such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: Yes. All assets are properly credited and used under their respective licenses.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [\[Yes\]](#)

Justification: We will provide an anonymous URL.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [\[NA\]](#)

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [\[NA\]](#)

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: This research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.