Unlocking the Mysteries of OpenAI o1: A Survey of the Reasoning Abilities of Large Language Models

Guoyin Wang▼, Shengyu Zhang♣, Tianyu Zhan♣, Zhouzhou Shen♣, Jiwei Li♣, Xueyu Hu♣, Xiaofei Sun♣, Fei Wu♣, Gelei Deng▲, Jie Zhang★, Runyi Hu▲, Tianwei Zhang▲, Xiaoya Li♦, Shuhe Wang♠ and Eduard Hovy♠

▼01.AI ♣Zhejiang University ▲Nanyang Technological University
★CFAR and IHPC, A*STAR ♦University of Washington ♠The University of Melbourne

Abstract

The release of OpenAI's o1 marks a significant milestone in AI, achieving proficiency comparable to PhD-level expertise in mathematics and coding. While o1 excels at solving complex reasoning tasks, it remains a closed-resource model, limiting its accessibility and broader application in academic and industrial contexts. Despite numerous efforts to replicate o1's results, these attempts often focus on isolated aspects of the model (e.g., training, inference), neglecting the holistic interplay between components and failing to provide a global picture of the pathways to enhance LLMs' reasoning capabilities, and replicate o1's performance. Currently, there is no systematic review of these replication efforts, nor a clear survey of the major issues that must be addressed to achieve comparable performance to o1.

In this survey, we are going to provide a systematic review of the most up-to-date state of knowledge on reasoning LLMs, helping researchers understand the current challenges and advancements in this field. We will summarize recent efforts to replicate o1's performances, and more importantly, address the key obstacles in enhancing the reasoning abilities. We will (1) review the basic concepts and techniques behind o1 and efforts in replicating o1 models; (2) detail efforts in constructing logical and structured reasoning datasets; (3) delve into important training techniques (e.g., supervised fine-tuning, reinforcement learning, DPO) that harness these datasets to ensure the model acquires robust logical reasoning and structured problem-solving capabilities; and (4) explores different inference techniques (e.g., tree-of-thoughts, critic-based approaches, self-correction strategies) in reasoning LLMs that assist in navigating the problem space and identifying efficient problem-solving paths. We will also summarize the current challenges and discuss opportunities for further improvement of reasoning LLMs.[1]

Figure 1: Word cloud of the keywords used in the titles of the papers involved in this survey.

[1]The latest update was on Jan.20, 2025.
Project page can be found at: https://github.com/ShuheSH/A-Survey-of-the-Reasoning-Abilities-of-LLMs.

1 Introduction

Large language models (LLMs) (Jiang et al., 2023; Bai et al., 2023; OpenAI, 2023; Yang et al., 2024a; Dubey et al., 2024; OpenAI, 2024a; Mistral AI, 2024; Team et al., 2024b; Liu et al., 2024a; Wake et al., 2024; Shao et al., 2024b; OpenAI, 2024b; GLM et al., 2024) have achieved remarkable performance in numerous language tasks (Sun et al., 2023b; Wang et al., 2023b; Wan et al., 2023; Sun et al., 2023c,a; Wang et al., 2023a; Sun et al., 2023d; Liu et al., 2024c; Yao et al., 2024; Wang et al., 2024d). Despite their impressive capabilities, LLMs still face significant challenges in reasoning. They struggle with tasks that require logical deduction, numerical calculations, or consistent chains of thought. Errors are observed even in simple tasks that demand multi-step thinking, highlighting gaps in how these models acquire, represent, and apply knowledge (Cobbe et al., 2021b; Wei et al., 2022; Wang and Lu, 2023; Shakarian et al., 2023; Shi et al., 2023; Chang et al., 2024; Ahn et al., 2024).

The release of OpenAI o1 (OpenAI, 2024b) marks a significant milestone in AI, particularly in enhancing its reasoning abilities. OpenAI o1 is capable of solving complex reasoning tasks and demonstrates capabilities comparable to PhD-level proficiency in math and coding. Unfortunately, o1 is a closed-resource model, which limits its accessibility and potential for broader academic and industrial use. This restricted access hinders collaborative efforts to further refine its abilities and limits the opportunity for researchers and developers to build upon its foundation. Additionally, the lack of transparency in the model's underlying architecture and training data raises concerns about bias and fairness, making it difficult to fully understand its decision-making processes.

As a result of the closed-resource nature of o1, numerous efforts have emerged to replicate o1's impressive results (Shao et al., 2024b; Mistral AI, 2024; Team, 2024b; o1 Team, 2024; Zhao et al., 2024b; Team, 2024a; DeepSeek-AI et al., 2024). o1, however, is a highly complex system, with substantial improvements across multiple AI modules, including training methodologies, inference mechanisms, datasets, and evaluation processes. Existing efforts to replicate o1 tend to focus on isolated aspects of the model, often neglecting the holistic interplay between these components, and, as a result, missing the full picture in enhancing LLMs' reasoning abilities. As a result, there is currently no systematic review of the efforts to replicate o1, and more importantly, no clear survey of the major issues that must be addressed to achieve comparable performance across all these dimensions.

In this survey, we provide a systematic review of the challenges and opportunities involved in replicating o1's performance. Specifically, we aim to identify the key obstacles in enhancing the reasoning abilities of LLMs and offer a comprehensive review of recent efforts to address them:

(1) At the **dataset** level, we will review recent efforts in constructing high-quality, diverse, and representative datasets that play a pivotal role in enhancing LLMs' reasoning capabilities. By incorporating challenging reasoning problems, well-structured scenarios, and diverse contexts, these datasets bridge the gap between the generic text data on which most LLMs are typically trained, which often lacks complex reasoning problems, and the reasoning tasks that demand advanced reasoning abilities.

(2) At the **training** level, we will delve into different training techniques that harness these datasets to ensure the model acquires robust logical reasoning and structured problem-solving capabilities. These techniques include *supervised training*, where labeled data fully guides the model's learning process; *reinforcement learning*, which optimizes decision-making through trial-and-error and reward-based feedback; and *direct preference optimization*, which aligns model outputs with user or system-defined preferences. Additionally, we will examine various adaptations and hybrid approaches of these methods, focusing on their strengths, limitations, and synergies to create a balanced and effective training framework.

(3) At the **inference** level, we will provide an in-depth review of techniques designed to identify and execute reasoning paths during the decoding process in LLMs. These techniques include *tree-of-thoughts* methods, *critic-based* approaches for evaluating and refining outputs, *self-correction* strategies to enhance solution accuracy, and other advanced methodologies aimed at improving the model's reasoning and decision-making capabilities in diverse contexts.

The rest of this survey is organized as follows:

- Section 2 provides an introduction of OpenAI O1, briefing its techniques and performances on benchmarks.

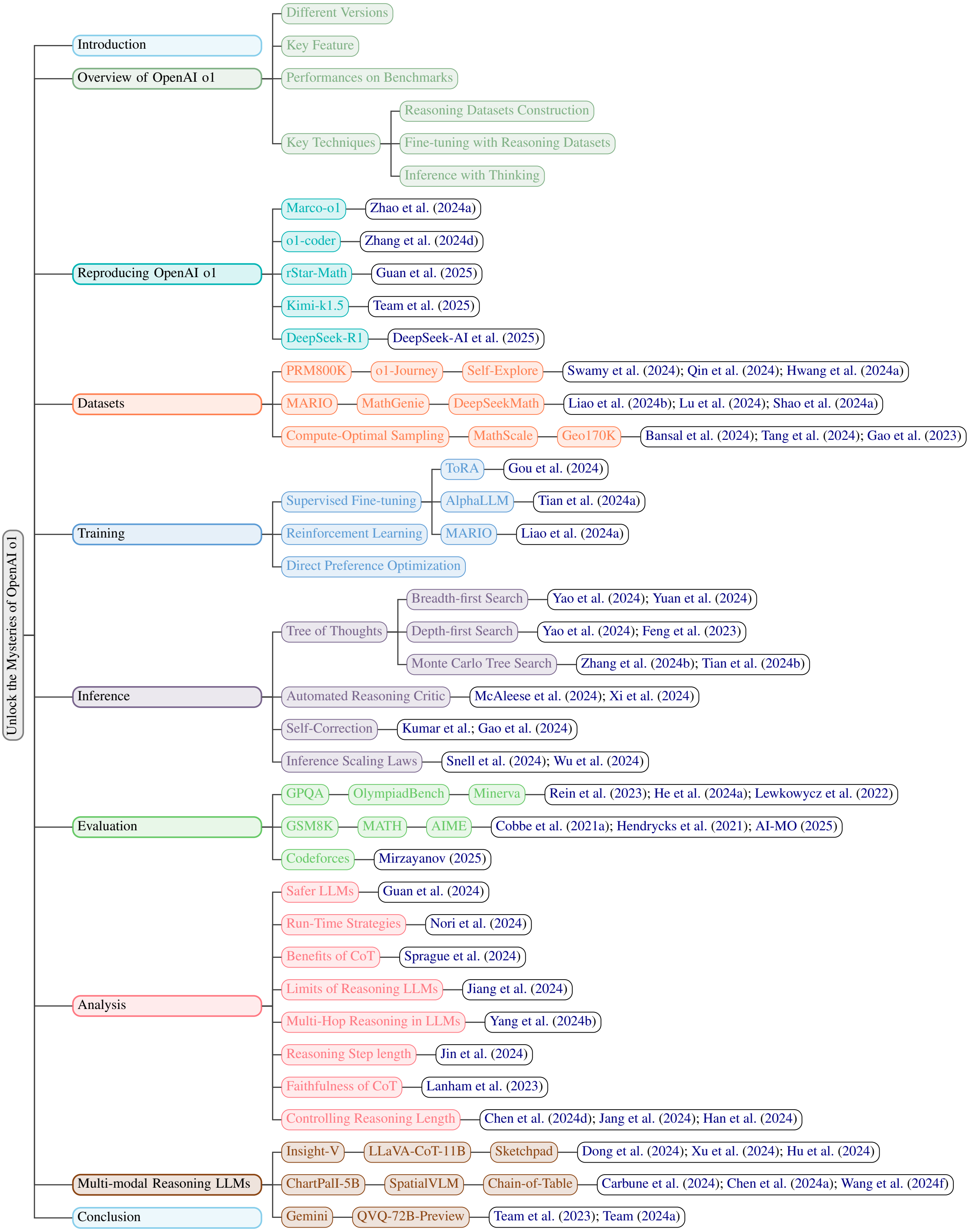


Figure 2: An overview of this survey, which covers the key aspects for replicating o1: dataset construction, training, inference strategies, and evaluation.

- Section 4 details recent efforts in constructing logical and structured reasoning datasets.
- Section 5 details training techniques that leverage reasoning datasets to build large language models with strong logical and structured reasoning capabilities.
- Section 6 explores commonly used inference techniques in reasoning LLMs that assist in navigating the problem space and identifying efficient problem-solving paths.
- Section 8 summarizes the current challenges and discusses opportunities for further improvement of reasoning LLMs.
- Section 9 briefs recent efforts in building LLMs with multi-modal reasoning abilities.
- We conclude this survey in Section 10.

2 Overview of OpenAI o1

OpenAI's o1 series represents a significant advancement in artificial intelligence, focusing on enhanced reasoning capabilities to tackle complex, multi-step tasks. Unlike previous models, o1 is designed to spend more time "thinking" before responding, making it particularly effective in areas such as science, mathematics, and programming.

2.1 Different Versions of o1

The o1 series represents a line of models, with each version designed with different model sizes, inference speeds, and prices to cater to diverse use cases. Additionally, each version has specific reasoning capabilities that allow it to perform better in various domains like mathematics, programming, and science. The following is a breakdown of the release dates, abilities, and features of the different o1 models:

- **o1-Preview**, released on September 12, 2024 offers advanced multi-step reasoning and excels in complex problem-solving tasks. It features the "chain of thought" process to enhance reasoning accuracy but requires higher computational resources and has slower inference speed.
- **o1-Mini**, released on September 12, 2024, is a faster, more cost-effective alternative to the o1-Preview, offering 80% lower costs while still providing good reasoning for tasks like coding and STEM problems. It's designed for users who prioritize speed over the depth of reasoning and is ideal for developers, students, and quick technical applications.
- **Full o1**, released on December 5, 2024, provides the highest level of reasoning power, with capabilities on par with PhD-level expertise in fields like math, science, and programming. It's the most accurate and reliable option for professionals requiring precise, multi-step analysis but comes with higher computational costs and slower performance.
- **o1-Lite**, to be released on January 2025, is a lightweight, cost-efficient version of the full o1, offering moderate reasoning abilities for general tasks at faster speeds. It's designed for small businesses, educational platforms, and individual users who need basic problem-solving capabilities at a reduced price, without the need for deep analysis or heavy computational resources.

2.2 Key Feature: Thinking before Response Generation

The key feature of the o1 model is that it allocates more time to "thinking" before generating responses: it conducts human-like "thinking" actions that produce detailed, step-by-step solutions when dealing with reasoning problems, such as:

The Given Question:
How many Rs are in "strawberry"?
Generated by o1:
There are three 'R's in the word "strawberry."
Here's the breakdown: *S - T - R - A - W - B - E - R - R - Y*

- 3rd letter: R
- 8th letter: R
- 9th letter: R

So, the letter 'R' appears three times in "strawberry."

By allocating more "thinking" time, the model can decompose complex, multi-step problems into smaller, manageable components. On one hand, this allows the o1 model to produce well-structured and coherent solutions to reasoning problems; on the other, it offers transparency into the reasoning process, enabling users to understand how the model reached its answers.

2.3 Performances on Benchmarks

o1 models have demonstrated superior performance on challenging tasks, achieving results comparable to PhD students in fields like physics, chemistry, and biology:

- AIME 2024 (Math): The o1 model achieved a score of 13.9, placing it among the top 500 students nationally and above the cutoff for the USA Mathematical Olympiad.
- GPQA (Chemistry, Physics and Biology): The o1 model surpassed the performance of recruited human experts with PhDs, becoming the first model to do so on this benchmark.
- 2024 International Olympiad in Informatics (Coding): The o1 model scored 213 points and ranked in the 49th percentile.
- Programming Contests Hosted by Codeforces (Coding): The o1 model achieved an Elo rating of 1673, performing better than 93% of competitors.

As shown in Table 1 and Figure 3, the evaluation of o1 demonstrates remarkable gains over gpt-4o across a diverse array of reasoning-intensive benchmarks, including competition math, code-generation challenges, and domain-specific question answering. On average, o1 exhibits substantially higher pass@1 and consensus@64 accuracy than its predecessor, suggesting that targeted architectural and training improvements have led to more robust reasoning capabilities., o1 achieves new state-of-the-art results in multiple categories, such as surpassing PhD-level experts on GPQA Diamond and attaining a top-500 rank on the AIME 2024 exam, which underscores its ability to rival human performance on challenging tasks.

Figure 4 presents some human evaluation results between openai o1 and gpt-4o. Interestingly, the human preference evaluations reveal that o1 does not uniformly outperform gpt-4o across all domains. While o1 demonstrates a pronounced advantage in more structured, logic-driven tasks, such as computer programming, data analysis, and mathematical calculation (where it wins over 70% of all cases), it lags slightly behind gpt-4o in more subjective or stylistic tasks like personal writing. For editing text, o1 and gpt-4o are comparable, suggesting that the model's improvements in reasoning do not necessarily translate to an equally strong edge in language polish or creative composition.

2.4 Speculated Techniques in o1

OpenAI o1 is a closed-source model, and as a result, there has been speculation surroundings the techniques behind its development. Various blogs, reports, and technical papers have attempted to deconstruct its capabilities, offering hypotheses about its underlying architecture, datasets, training methods, and reasoning processes. Some speculate that o1 leverages advanced forms of reinforcement learning and unsupervised learning, while others suggest a highly refined chain of thought approach to enhance its multi-step reasoning.

2.4.1 Constructing Reasoning Formatted Datasets

Building the training dataset is the initial step in equipping the model with the desired capabilities. For reasoning LLMs, this involves constructing a reasoning formatted dataset. Unlike standard datasets, which may focus on single-step tasks or basic understanding, reasoning formatted datasets emphasize logical progression, multi-step thinking, and structured problem-solving approaches. For example:

Example 1:
Input: What is the sum of the first 10 positive integers?
Reasoning Path:

- Step 1: The first 10 positive integers are 1, 2, 3, $\cdots$, 10.
- Step 2: The sum of a sequence can be calculated using the formula $(n \times (n+1))/2(n \times (n+1))/2(n \times (n+1))/2$.
- Step 3: Substituting $n = 10$, we get $(10 \times 11)/2 = 55(10 \times 11)/2 = 55(10 \times 11)/2 = 55$.
- Answer: The sum is 55.

Example 2:
Input: If Alice is older than Bob and Bob is older than Charlie, who is the oldest?
Reasoning Path:

- Step 1: Alice is older than Bob.
- Step 2: Bob is older than Charlie.
- Step 3: Thus, Alice is older than both Bob and Charlie.
- Answer: Alice is the oldest.

In prior research (Swamy et al., 2024), OpenAI researchers discovered that exposing LLMs to reasoning formatted data enables them to learn logical patterns and enhance their accuracy. Here, we provide a brief overview of reasoning formatted datasets. We will detail methods for constructing such datasets are discussed in Section 4.

2.4.2 Training LLMs on Reasoning Formatted Dataset

After constructing the datasets, people speculated that the next step is to fine-tune o1 on these specialized reasoning datasets. This process often involves supervised fine-tuning (SFT) (Zhang et al., 2023) and reinforcement learning (RL) (Wang et al., 2024e) techniques to optimize the o1's performance. Supervised fine-tuning is the initial step in training reasoning LLMs. This process mainly involves training the o1-like reasoning LLMs to generate complete reasoning paths in response to the given questions. Through this approach, LLMs learn to follow logical chains, produce coherent outputs, and establish a solid foundation for tackling more advanced reasoning problems and employing sophisticated techniques. For more specific andn detaied techniques, we put in Section 5.1.

Reinforcement learning is another speculated core step in fine-tuning o1. One popular route is the large-scale Reinforcement Learning from Human Feedback (RLHF) paradigm. This method uses reasoning datasets (*e.g.*, mathematical proofs or compiler feedback code traces) and employs a reward model that captures correctness, logical consistency, and sometimes per-step "process" quality. An alternative strategy uses Monte Carlo Tree Search (MCTS) to drive exploration. The model can propose and evaluate multiple partial solutions, with MCTS guiding the generation of high-quality trajectories and off-policy RL (or supervised fine-tuning) refining the policy. For more technical details regarding reinforcement learning in reasoning LLMs, we put in Section 5.2.

2.4.3 Inference with Advanced Thinking Strategies

Researchers speculate that the inference approach of o1 combines a series of reasoning techniques:

Problem breakdown which deconstructs complex problems into smaller, manageable parts (as illustrated below), which makes it easier to arrive at a correct solution;

The Given Question:
How many Rs are in "strawberry"?
Decomposition:

- Identify all the words: *S - T - R - A - W - B - E - R - R - Y*
- Identify which positions are the word R: 3rd letter, 8th letter, and 9th letter.
- Calculate: The letter 'R' appears three times

Mistake recognition and self correction which detects and rectifies errors in their reasoning, much like a human reassessing and adjusting a flawed approach. Such as when get a response "The area is $10 \times 5 = 50$" by given a question "What is the area of a triangle with a base of 10 and a height of 5?". A critic model is employed, or the o1-like reasoning LLM itself is prompted, to evaluate whether the response is accurate. If an error is identified, the o1-like reasoning LLM will immediately generate a new answer. This iterative process continues until the critic model or the LLM determines that the generated response is correct.

Solution Exploration which involves o1-like reasoning LLMs examining multiple potential solution paths before reaching a final answer, ensuring the selection of the most logical and accurate result. This process resembles a tree structure, where the input problem serves as the root node, each node represents a step in the solution, and the path from a leaf node to the root forms a complete solution path. The o1-like reasoning LLM employs various search strategies to construct this solution tree and evaluate the validity of each path, resulting in more precise and insightful outcomes.

Dataset	Metric	gpt-4o	o1-preview	o1
Competition Math AIME (2024)	cons@64	13.4	56.7	83.3
	pass@1	9.3	44.6	74.4
Competition Code CodeForces	Elo	808	1,258	1,673
	Percentile	11.0	62.0	89.0
GPQA Diamond	cons@64	56.1	78.3	78.0
	pass@1	50.6	73.3	77.3
Biology	cons@64	63.2	73.7	68.4
	pass@1	61.6	65.9	69.2
Chemistry	cons@64	43.0	60.2	65.6
	pass@1	40.2	59.9	64.7
Physics	cons@64	68.6	89.5	94.2
	pass@1	59.5	89.4	92.8
MATH	pass@1	60.3	85.5	94.8
MMLU	pass@1	88.0	90.8	92.3
MMMU (val)	pass@1	69.1	n/a	78.2
MathVista (testmini)	pass@1	63.8	n/a	73.9

Table 1: Official evaluation results of o1 on typical benchmarks (o1 Contributors, 2024).

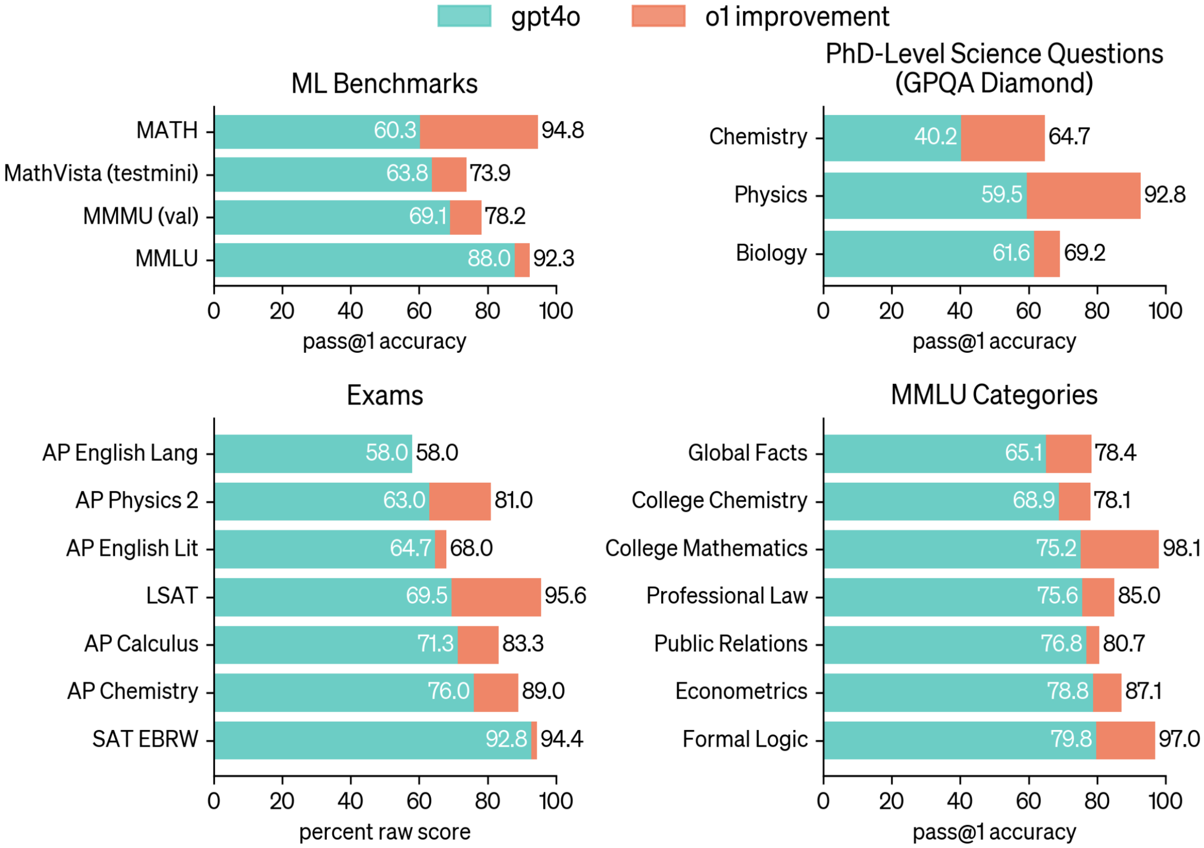


Figure 3: Official sub-category evaluation results of o1 on typical benchmarks (o1 Contributors, 2024).

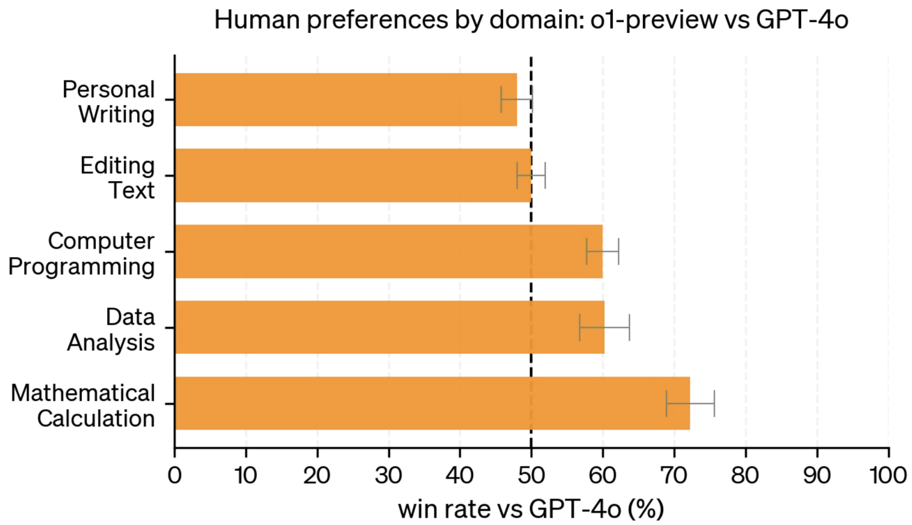


Figure 4: Official human evaluation results of o1 (o1 Contributors, 2024).

3 Recent Efforts in Reproducing OpenAI o1

Currently, many efforts have been made to replicate OpenAI's o1 or specific capabilities of o1 (such as code generation and mathematical reasoning). We have collected eight such works, among which six are open-source, three provide reports or papers, and four include comparisons with o1. For detailed information, please refer to Table 2. In the following section, we will focus on introducing the works that have provided reports or papers.

Model	Organization	# Params	Open Source	Report/Paper Available	Comparison with o1
Gemini 2.0 Flash (Google AI)	Google	-	✗	✗	✗
QVQ-72B-Preview (QwenLM, QVQ)	Alibaba	72B	✓ [1]	✗	✓
Marco-o1 (Zhao et al., 2024a)	Alibaba	7B	✓ [2]	✓ [8]	✗
Skywork o1 (o1 Team, 2024)	KUNLUN	8B	✓ [3]	✗	✗
QwQ-32B-Preview (QwenLM, QwQ)	Alibaba	32B	✓ [4]	✗	✓
o1-Coder (Zhang et al., 2024d)	北京交通大學 Beijing Jiaotong University	-	✓ [5]	✓ [9]	✗
rStar-Math (Guan et al., 2025)	Microsoft	1.5B,3B,7B	✓ [6]	✓ [10]	✓
Kimi-k1.5 (Team et al., 2025)	Moonshot AI	-	✗	✓ [11]	✓
DeepSeek-R1 (DeepSeek-AI et al., 2025)	deepseek	671B-A31B	✓ [7]	✓ [12]	✓

[1] https://huggingface.co/Qwen/QVQ-72B-Preview
[2] https://github.com/AIDC-AI/Marco-o1
[3] https://huggingface.co/Skywork/Skywork-o1-Open-Llama-3.1-8B
[4] https://huggingface.co/Qwen/QwQ-32B-Preview
[5] https://github.com/ADaM-BJTU/o1-Coder
[6] https://github.com/zhentingqi/rStar
[7] https://huggingface.co/deepseek-ai/DeepSeek-R1
[8] https://arxiv.org/pdf/2501.04519
[9] https://arxiv.org/pdf/2411.14405
[10] https://arxiv.org/pdf/2412.00154
[11] https://arxiv.org/pdf/2501.12599
[12] https://arxiv.org/pdf/2501.12948

Table 2: Overview of recent efforts in reproducing OpenAI o1. The format '671B-A31B' refers to MoE models with 141B total and 39B active parameters.

3.1 Marco-o1

Macro-o1 (Zhao et al., 2024a), developed by Alibaba, explores the generalization capabilities of the o1 model in open-ended domains lacking clear standards or quantifiable rewards, unlike disciplines with standard answers such as mathematics, physics, or coding. It employs techniques like Chain-of-Thought (CoT) fine-tuning, Monte Carlo Tree Search (MCTS), reflective processes, and advanced reasoning to address complex real-world challenges. Experimental results indicate that Macro-o1 exhibits o1-like reasoning abilities, achieving significant accuracy gains of +6.17% on the MGSM (English) dataset and +5.60% on the MGSM (Chinese) dataset, highlighting its improved reasoning performance. Additionally, it pioneers the application of large reasoning models (LRMs) in machine translation, particularly excelling in translating slang expressions, while investigating inference-time scaling laws in multilingual contexts.

As illustrated in Figure 5, Macro-o1's core idea is to first fine-tune a base LLM using a combined dataset and then perform inference with MCTS to expand the solution space. The fine-tuning dataset comprises three components: the refined Open-O1 CoT Dataset (O1, 2025), a Marco-o1 CoT Dataset generated via MCTS, and the Marco Instruction Dataset. During inference, two action strategies are applied within the MCTS framework: "step as action" for efficient exploration and "mini-step as action" (32 or 64 tokens) for finer granularity. The latter broadens the solution space by incorporating more detailed reasoning steps, enhancing the model's capacity to handle complex tasks. A reflection mechanism further improves performance by prompting the model to reevaluate its reasoning with phrases like: "Wait! Maybe I made some mistakes! I need to rethink from scratch." This self-reflection helps correct errors in difficult problems. The final solutions are selected based on calculated confidence scores, as shown in Figure 5.

3.2 o1-Coder

o1-Coder (Zhang et al., 2024d), developed by Beijing Jiaotong University, aims to evaluate the performance of OpenAI's o1 model in coding tasks by adapting it to better address programming-related problem-solving challenges. The goal is to enhance the model's capabilities through focused improvements. o1-Coder combines reinforcement learning (RL) with Monte Carlo Tree Search (MCTS) to strengthen the model's System-2 reasoning abilities. The system involves training a Test Case Generator (TCG) for standardized testing, utilizing MCTS to generate reasoning-augmented code data, and iteratively

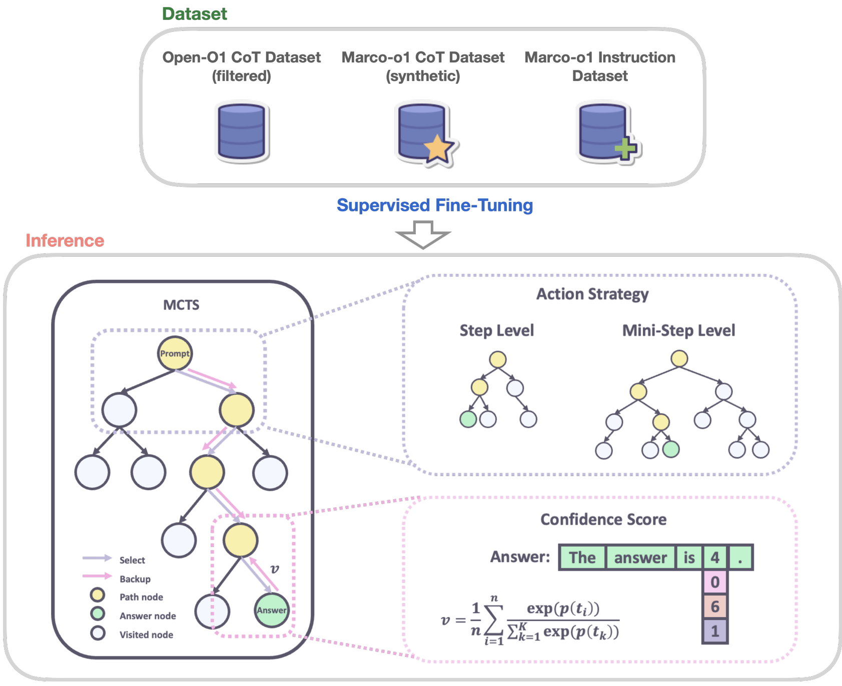


Figure 5: The overview of Marco-o1 (Zhao et al., 2024a). Macro-o1 is fine-tuned using a combination of three datasets. During the MCTS inference stage, action strategies at two levels are utilized to achieve a balance between efficiency and performance. The confidence score for each generated token t_i is calculated by applying the softmax function to its log probability along with the log probabilities of the top 5 alternative tokens. The overall reward score v for the rollout sequence is then obtained by averaging the confidence scores across all tokens. The figure is adapted from Zhao et al. (2024a).

refining the policy model to evolve from pseudocode to fully functional code. The model is still under development, and further updates along with experimental results will be shared in future versions.

As illustrated in Figure 6, the o1-Coder framework consists of six key steps: 1. The process begins by training a Test Case Generator (TCG), denoted as γ_{TCG}, to automatically create test cases based on the given problem descriptions. 2. Next, Monte Carlo Tree Search (MCTS) is applied to the original code dataset, producing a new dataset $\mathcal{D}_{\text{process}}$. This dataset incorporates reasoning processes and a validity indicator to distinguish correct from incorrect steps. 3. The dataset is then used to fine-tune the policy model π_θ, encouraging it to adopt a "think before acting" approach. 4. The reasoning data from the previous step is used to initialize a process reward model (PRM), ρ_{PRM}, which evaluates the quality of reasoning steps. 5. Both the PRM, ρ_{PRM}, and TCG, γ_{TCG}, provide rewards based on process and outcome, respectively. This enables reinforcement learning to iteratively update the policy model π_θ. 6. Finally, the updated policy model generates new reasoning data, which is used to refine the PRM ρ_{PRM}, creating a self-improving iterative cycle through steps 4, 5, and 6. This approach forms a feedback loop that enhances the model's reasoning and coding performance over time.

3.3 rStar-Math

rStar-Math (Guan et al., 2025), developed by Microsoft, demonstrates that small language models (SLMs) can match or surpass the mathematical reasoning abilities of OpenAI's o1 model, without the need for distillation from larger models. This is achieved through "deep thinking" via Monte Carlo Tree Search (MCTS), where an SLM-based math policy conducts test-time searches, guided by a process preference model (PPM) also built on SLMs. The core advancements of rStar-Math lie in three key solutions designed to overcome training challenges for the two SLMs: a code-enhanced Chain-of-Thought (CoT) data synthesis method, a PPM training framework, and a self-evolution strategy. Extensive experiments demonstrate significant improvements on the MATH benchmark. rStar-Math enhances the accuracy of Qwen2.5-Math-7B from 58.8% to 90.0% and Phi3-mini-3.8B from 41.4% to 86.4%, outperforming o1-

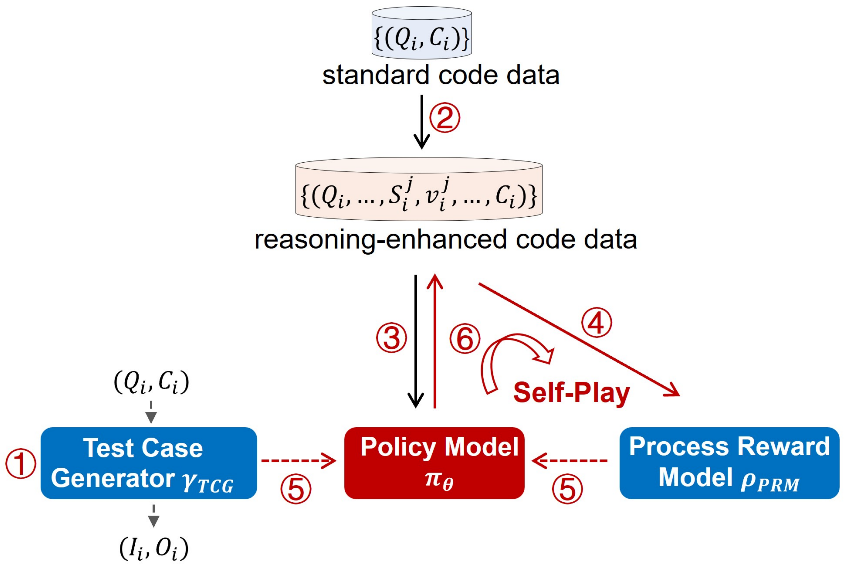


Figure 6: The overview of o1-Coder's (Zhang et al., 2024d) Self-Play+RL training framework. It integrates a Test Case Generator (TCG), Monte Carlo Tree Search (MCTS), and a Process Reward Model (PRM) to iteratively refine the policy model π_θ through reasoning-based reinforcement learning. The cycle involves generating reasoning data, updating the PRM, and enhancing the model's capability to produce high-quality reasoning code. The figure is adapted from Zhang et al. (2024d).

preview by +4.5% and +0.9%, respectively. On the USA Math Olympiad (AIME), rStar-Math successfully solves an average of 53.3% (8/15) of the problems, placing it within the top 20% of high school math students.

As shown in Figure 7, rStar-Math trains a math policy SLM and a process preference model (PPM) integrated with Monte Carlo Tree Search (MCTS) for deep thinking. The training involves three key innovations. **First**, a code-augmented CoT data synthesis method uses MCTS rollouts to generate step-by-step reasoning trajectories annotated with self-assigned Q-values. The policy SLM samples candidate nodes, producing one-step CoTs and corresponding Python code. Only nodes with successful code execution are retained, reducing errors. Q-values are assigned to each step based on its contribution, ensuring accurate reasoning trajectories. **Second**, a PPM to enable reliable prediction of reward labels for math reasoning steps. Rather than using noisy Q-values directly, the PPM distinguishes correct steps from incorrect ones using preference pairs and optimizes its scoring with a pairwise ranking loss (Ouyang et al., 2022). This improves the accuracy of stepwise reward assignment compared to traditional methods (Chen et al., 2024b). **Finally**, a four-round self-evolution framework refines the policy model and PPM. Starting with a dataset of 747k math word problems, each round uses the updated models to generate better training data. This iterative process leads to: (1) a stronger policy SLM, (2) a more reliable PPM, (3) improved reasoning trajectories, and (4) expanded data coverage for more challenging math problems.

3.4 Kimi-k1.5

Kimi-k1.5 (Team et al., 2025), developed by Moonshot AI, is a multi-modal LLM which represents a significant advancement in scaling reinforcement learning (RL). The authors introduce a novel approach by focusing on long context scaling, extending the context window of RL to 128k, and refining policy optimization methods. Unlike traditional RL frameworks that rely on complex techniques such as MCTS, value functions, and process reward models, Kimi-k1.5 establishes a streamlined and effective RL framework. The model achieves state-of-the-art reasoning performance across various benchmarks and modalities, rivaling OpenAI's o1 (see Figure 8). Additionally, the authors introduce long2short methods that utilize long-CoT techniques to enhance short-CoT models, significantly outperforming existing models like GPT-4o and Claude Sonnet 3.5 by up to 550% (see Figure 9).

The development of Kimi-k1.5 involves several stages: pretraining, vanilla supervised fine-tuning

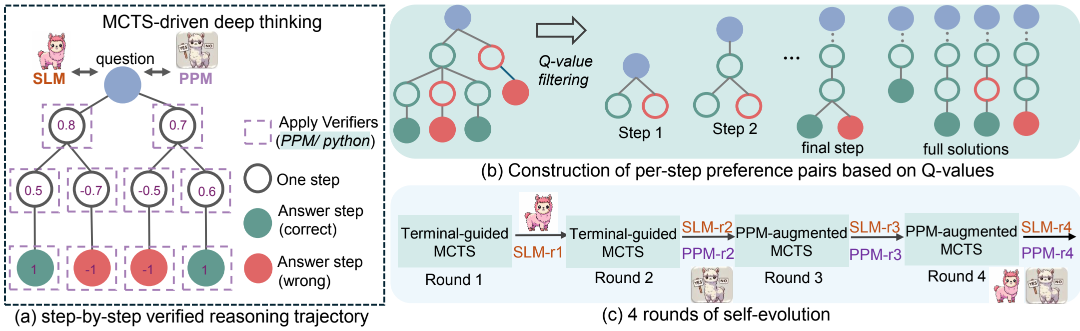


Figure 7: The overview of rStar-Math (Guan et al., 2025): (a) Star-Math uses Monte Carlo Tree Search (MCTS) for "deep thinking," where a math policy SLM guides test-time search, aided by a process reward model based on an SLM. (b) MCTS rollouts assign Q-values to intermediate steps based on their contribution: steps that help more trajectories reach the correct answer receive higher Q-values and are deemed higher quality. (c) Both the policy SLM and process reward model (PPM) are developed from scratch and continuously refined to enhance reasoning performance. The figure is adapted from Guan et al. (2025).

(SFT), long-CoT supervised fine-tuning, and reinforcement learning (RL). The primary innovation lies in the RL phase, where the authors construct a high-quality RL prompt set designed to guide the model toward robust reasoning while mitigating risks such as reward hacking and overfitting to superficial patterns. This prompt set is characterized by three key properties: diverse coverage, balanced difficulty, and accurate evaluability. During RL training, three critical strategies are employed:

- **Online Policy Mirror Descent**: A variant of this algorithm is used to optimize the training process (Abbasi-Yadkori et al., 2019; Mei et al., 2019; Tomar et al., 2020).
- **Length Penalty**: A reward mechanism is introduced to control the rapid growth of token length, thereby enhancing token efficiency.
- **Sampling Methods**: Two sampling techniques are utilized to improve training efficiency: (1) *Curriculum sampling*: This method progressively trains the model from simpler to more complex tasks, enhancing both training efficiency and model performance. (2) *Prioritized sampling*: This strategy focuses on areas where the model underperforms by sampling problematic tasks more frequently, proportional to their failure rates, thereby accelerating learning in weaker areas.

While long-CoT models demonstrate strong performance, they often require more test-time tokens compared to standard short-CoT LLMs. To address this, the authors propose four methods to transfer the reasoning capabilities of long-CoT models to short-CoT models, a challenge referred to as the long2short problem. These methods include: (1) **Model Merging**: Combining a long-CoT model with a shorter model by averaging their weights. (2) **Shortest Rejection Sampling**: Using the long-CoT model to generate multiple responses to the same question and selecting the shortest correct response for SFT. (3) **Direct Preference Optimization (DPO)**: Forming pairwise preference data using positive (shortest correct solution) and negative (longer solutions) samples for DPO training. (4) **Long2short RL**: A two-phase training approach where, after standard RL training, a model with optimal performance and token efficiency is selected for a second phase. In this phase, a length penalty is applied, and the maximum response length is reduced to encourage more concise responses.

3.5 DeepSeek-R1

DeepSeek-R1 (DeepSeek-AI et al., 2025), developed by DeepSeek, is a state-of-the-art reasoning model that achieves performance comparable to OpenAI's o1 series models. This work pioneers the use of pure reinforcement learning (RL) to enhance language model reasoning capabilities, focusing on self-evolution without relying on supervised data. The authors first train DeepSeek-R1-Zero, a model derived from DeepSeek-V3-Base, using large-scale RL without supervised fine-tuning (SFT). This preliminary model

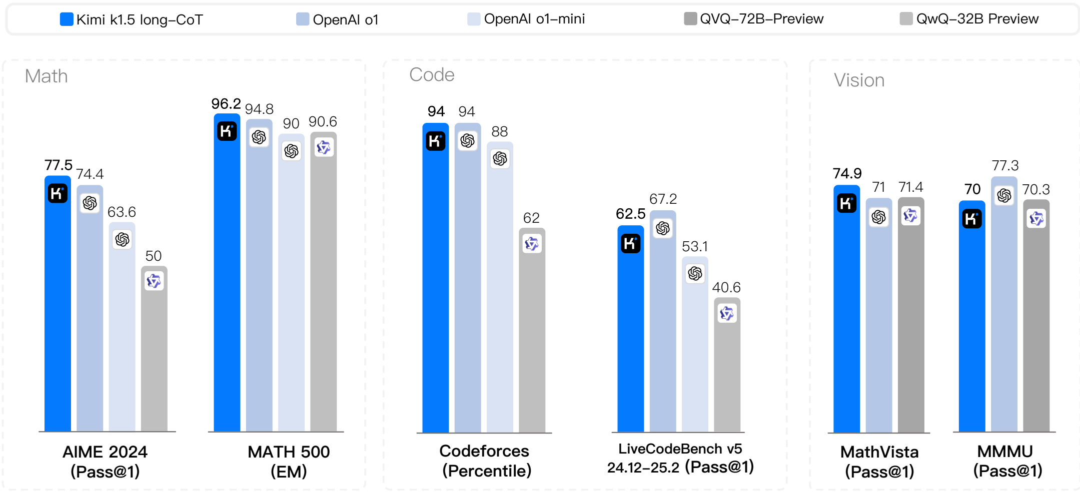


Figure 8: Results of Kimi-k1.5 (Team et al., 2025) long-CoT. The figure is adapted from Team et al. (2025).

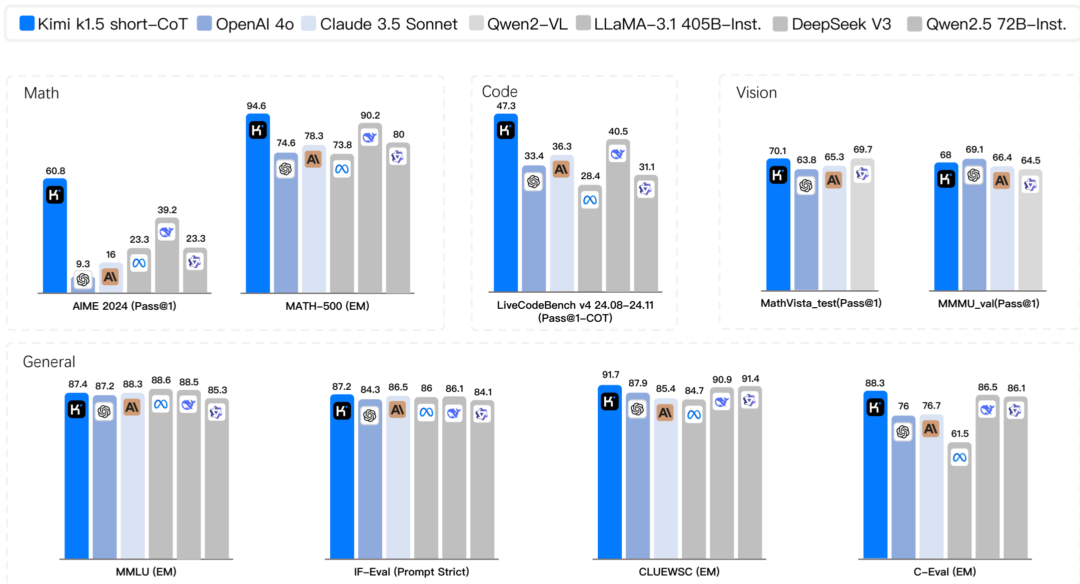


Figure 9: Results of Kimi-k1.5 (Team et al., 2025) short-CoT. The figure is adapted from Team et al. (2025).

demonstrates significant reasoning improvements, with the pass@1 score on AIME 2024 increasing from 15.6% to 71.0%. With majority voting, the score further rises to 86.7%, matching the performance of OpenAI-o1-0912. To address issues such as poor readability, language mixing, and to further boost reasoning performance, the authors introduce DeepSeek-R1. This enhanced model incorporates a small amount of cold-start data and a multi-stage training pipeline, achieving performance on par with OpenAI-o1-1217 (see Figure 10).

The RL training process for DeepSeek-R1-Zero employs Group Relative Policy Optimization (GRPO) (Shao et al., 2024b), which eliminates the need for a critic model by estimating baselines from group scores. The reward system is rule-based, consisting of two main components: accuracy rewards and format rewards. The accuracy reward evaluates the correctness of responses, while the format reward enforces the use of ‘<think>’ and ‘</think>’ tags to structure the reasoning process. During training, an intermediate version of the model exhibited an "aha moment" (see Figure 11), where it learned to allocate more time to reevaluate its initial approach, demonstrating the evolving reasoning capabilities facilitated by RL.

The training process for DeepSeek-R1 consists of two alternating stages of SFT and RL, structured as follows:

- **Initial Cold Start SFT**: The process begins with the collection of thousands of high-quality, readability-focused long CoT datasets. These datasets are used to fine-tune DeepSeek-V3-Base, establishing a robust foundation for subsequent RL training.
- **First Reasoning-oriented RL Stage**: The model undergoes large-scale reasoning-oriented RL, leveraging the same methodology applied in DeepSeek-R1-Zero to enhance its reasoning capabilities. Upon convergence, the resulting checkpoint is utilized to gather additional SFT data for the next stage.
- **Second SFT Stage**: This phase focuses on further refining the model through a combination of reasoning and non-reasoning data. For reasoning data, specialized prompts are curated, and reasoning trajectories are generated via rejection sampling using the RL checkpoint from the previous stage. For non-reasoning data, such as writing, factual QA, self-cognition, and translation, the DeepSeek-V3 pipeline is utilized, integrating portions of the DeepSeek-V3 SFT dataset. Finally, DeepSeek-V3-Base is fine-tuned for two epochs using this comprehensive dataset to ensure optimal performance across a wide range of tasks.
- **Second RL Stage for all Scenarios**: A final RL phase is conducted to align the model with human preferences, enhancing its helpfulness and harmlessness while further refining its reasoning abilities. Specifically, the model is trained using a combination of reward signals and diverse prompt distributions to ensure robust and generalizable performance.

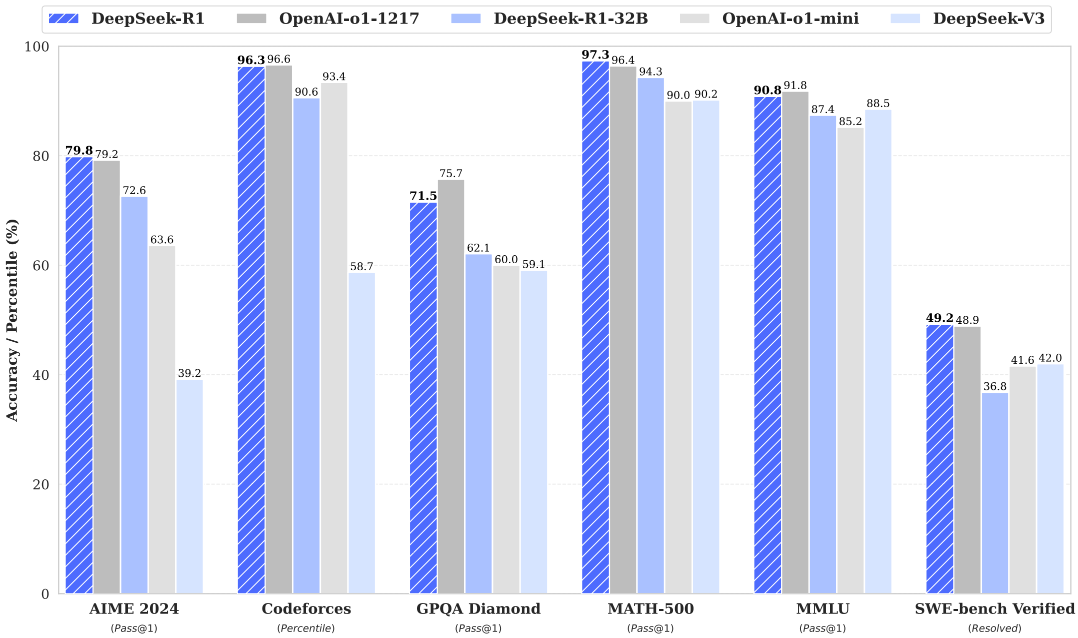


Figure 10: Results of DeepSeek-R1 (DeepSeek-AI et al., 2025). The figure is adapted from DeepSeek-AI et al. (2025).

4 Datasets

In this section, we provide a detailed overview of the construction processes for reasoning datasets that can be used in developing reasoning LLMs like OpenAI o1. These datasets are typically constructed using one or a combination of the following two strategies: the **machine-generated**, where a trained model provides feedback, responses or grades, and the **human-generated**, where humans are asked to provide feedback. Some datasets rely on a single strategy, while others combine both approaches. Datasets that we will cover here include PRM800K (Swamy et al., 2024), O1-Journey (Qin et al., 2024), Self-Explore (Hwang et al., 2024a), MARIO (Liao et al., 2024b), MathGenie (Lu et al., 2024), DeepSeekMath (Shao et al., 2024a), Question: If $a > 1$, then the sum of the real solutions of $\sqrt{a - \sqrt{a + x}} = x$ is equal to

Response: <think>
To solve the equation $\sqrt{a - \sqrt{a + x}} = x$, let's start by squaring both $\cdots$
$\left(\sqrt{a - \sqrt{a + x}}\right)^2 = x^2 \implies a - \sqrt{a + x} = x^2.$
Rearrange to isolate the inner square root term:
$(a - x^2)^2 = a + x \implies a^2 - 2ax^2 + (x^2)^2 = a + x \implies x^4 - 2ax^2 - x + (a^2 - a) = 0$
...
Wait, wait. Wait. That's an aha moment I can flag here.
Let's reevaluate this step-by-step to identify if the correct sum can be $\cdots$
We started with the equation:
$\sqrt{a - \sqrt{a + x}} = x$
First, let's square both sides:
$a - \sqrt{a + x} = x^2 \implies \sqrt{a + x} = a - x^2$
Next, I could square both sides again, treating the equation: $\cdots$
...

Figure 11: An intriguing "aha moment" observed in an intermediate version of DeepSeek-R1-Zero (DeepSeek-AI et al., 2025), where the model demonstrates the ability to rethink its approach using an anthropomorphic tone. This moment not only highlights the model's evolving reasoning capabilities but also underscores the remarkable potential and elegance of reinforcement learning in fostering advanced cognitive behaviors. The figure is adapted from DeepSeek-AI et al. (2025).

Compute-Optimal Sampling (Bansal et al., 2024), MathScale (Tang et al., 2024) and Geo170K (Gao et al., 2023). An overview of these datasets is shown in Table 3.

4.1 PRM800K

PRM800K (Swamy et al., 2024) is an open-source large-scale dataset consisting of step-level human feedback labels, created using a combination of machine-generated and human-generated strategies. It includes 800K step-level annotations drawn from 75K solutions to 12K problems sourced from the *MATH* dataset. Each dataset instance comprises two primary components: "steps" and "labels". The "steps" represent intermediate reasoning steps produced by the GPT-4 language model in a sequential format. The "labels" are human-generated annotations for each step, categorizing them as correct (positive), incorrect (negative), or ambiguous (neutral). The dataset was created through the following process:

1. Solution Generation: GPT-4 is used to generate step-by-step solutions to MATH problems.
2. Filtering: Only solutions with correct final answers were retained.
3. Human Annotation: Labelers assigned positive, negative or neutral labels to each individual step, with a particular focus on "convincing wrong-answer" solutions - high-rated but incorrect examples, as illustrated in Figure 12 - to maximize the utility of feedback.

4.2 O1-Journey

O1-Journey (Qin et al., 2024) is an open-source English reasoning dataset consisting of 677 instances, of which 327 are allocated for training. It is also created using a combination of machine-generated and human-generated strategies. Each instance in O1-Journey contains three components: "question", "answer", and "longCOT". The "question" specifies the problem or task to be solved, the "answer" provides the correct solution, and the "longCOT" offers a comprehensive reasoning process, including intermediate steps, reflections, and corrections. The details for O1-Journey construction are as follows:

Dataset	Data Source	Data Scale	Machine Generated	Human Generated	Open-source
PRM800K (Swamy et al., 2024)[1]	MATH	800K annotations	✓	✓	✓[1]
O1-Journey (Qin et al., 2024)[2]	MATH, PRM800K	677 instances	✓	✓	✓[2]
Self-Explore (Hwang et al., 2024a)	GSM8K, MATH	Model-specific	✓	✗	✗
MARIO (Liao et al., 2024b)[3]	GSM8K, MATH, MetaMath	28.8K instances	✓	✓	✓[3]
MathGenie (Lu et al., 2024)	GSM8K, MATH	170K qa pairs	✓	✗	✗
DeepSeekMath (Shao et al., 2024a)[4]	AlgebraicStack, arXiv, GitHub	120B tokens	✓	✓	✓[4]
Compute-Optimal Sampling (Bansal et al., 2024)	GSM8K, MATH, etc.	Model-specific	✓	✗	✗
MathScale (Tang et al., 2024)[5]	GSM8K, MATH	2M qa pairs	✓	✗	✓[5]
G-LLaVA (Gao et al., 2023)[6]	Geometry3K, GeoQA+	170K instances	✓	✗	✓[6]

[1] https://github.com/openai/prm800k
[2] https://github.com/GAIR-NLP/O1-Journey
[3] https://github.com/MARIO-Math-Reasoning/MARIO
[4] https://github.com/deepseek-ai/DeepSeek-Math
[5] https://github.com/XylonFu/MathScale
[6] https://github.com/pipilurj/G-LLaVA

Table 3: An overview of currently widely-adopted reasoning datasets for LLMs.

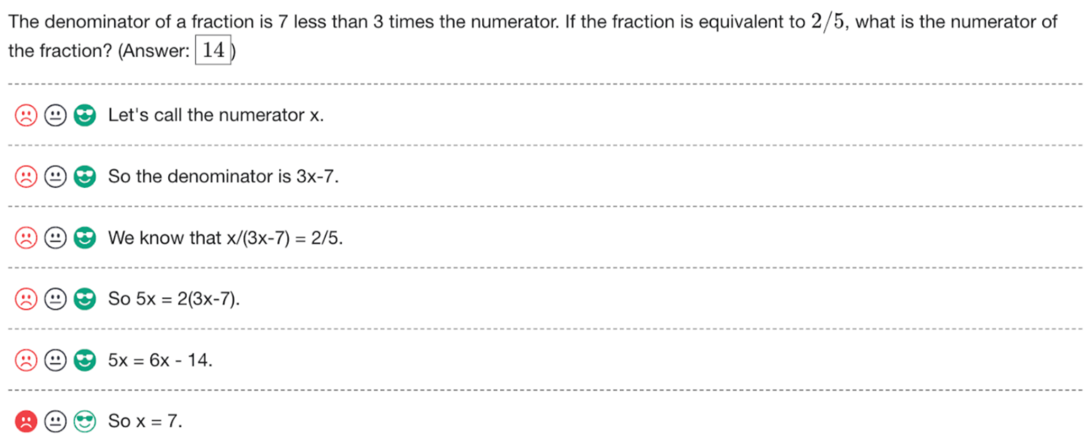


Figure 12: A screenshot of the interface used to collect feedback in *PRM800K*. The figure is borrowed from Swamy et al. (2024).

1. Reasoning Tree Construction: A pretrained policy model is used to generate reasoning trees for examples in *MATH* and *PRM800K*. The reasoning trees will be further evaluated by a reward model and incorrect trees will be discarded.

2. Reasoning Data Expansion: a multi-agent system is implemented to generate detailed reasoning data. One agent produces reasoning steps, while another critic and provides feedback. This iterative process ensures that reflections, backtracking, and revisions align with human-like thought processes.

3. Data Augmentation: Human annotators manually refine the extended reasoning data.

4.3 Self-Explore

Self-Explore (Hwang et al., 2024a) is a dataset designed to enhance the reasoning capabilities of large language models, relying solely on machine-generated data. It leverages fine-grained rewards derived from the model's generated rationales, making the data scale specific to the model. The Self-Explore process is detailed as follows:

1. Fine-tuning the Base Model: The process begins by fine-tuning a base model M_{SFT} using an initial human-curated dataset, such as *GSM8K* or *MATH*.

2. Generating Pairwise Dataset: The model then generates multiple rationales to create a dataset, denoted as $\mathcal{D}_{\text{GEN}}$. By comparing predictions to the ground truth, correct and incorrect rationales are identified, forming a pairwise dataset $\mathcal{D}_{\text{pair}}$.

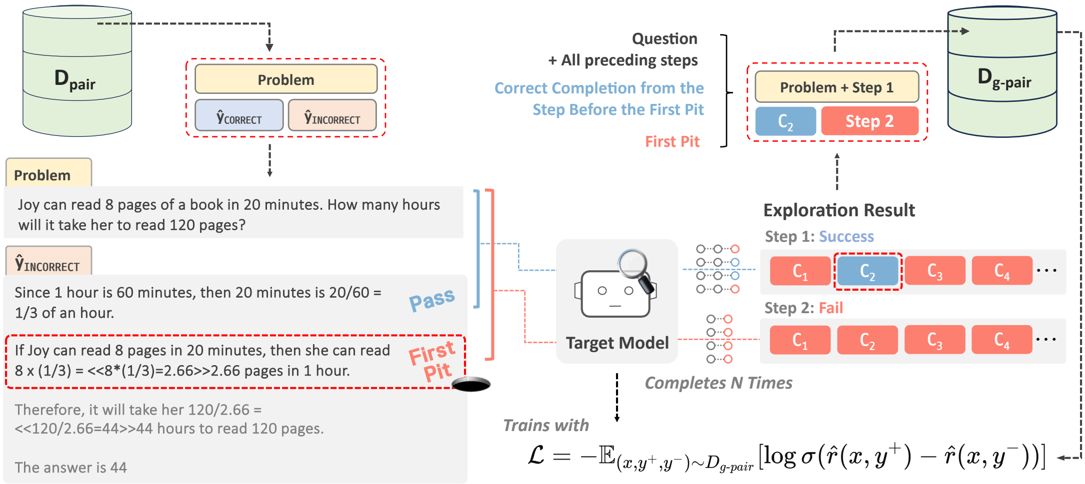


Figure 13: Overview of SELF-EXPLORE (Hwang et al., 2024a). A pairwise dataset ($\mathcal{D}_{\text{pair}}$) constructed through outcome supervision uses incorrect rationales to guide the target model in generating multiple completions at each step. If the correct answer is not produced from these completions, that step is identified as the *first pit*. By leveraging the first pit, $\mathcal{D}_{\text{pair}}$ is transformed into a granular preference dataset ($\mathcal{D}_{\text{g-pair}}$), providing more detailed learning signals during training. The figure is borrowed from Hwang et al. (2024a).

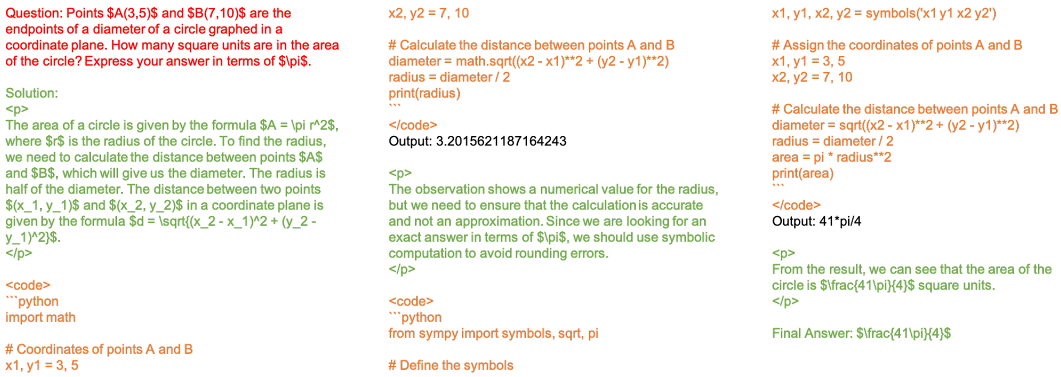


Figure 14: The dataset format of MARIO (Liao et al., 2024b). A reformatting example combines text analyses with corresponding code snippets in a single display. The figure is borrowed from Liao et al. (2024b).

3. Locating the "First Pit": A critical step involves detecting the initial flawed reasoning step, termed the "first pit," within incorrect rationales. This is determined by examining continuations from each step. If no continuation leads to the correct answer, the corresponding step is marked as the "first pit," as illustrated in Figure 13. This process produces a detailed pairwise dataset $\mathcal{D}_{\text{g-pair}}$.

4. Enhancing the Model: Finally, the $\mathcal{D}_{\text{g-pair}}$ dataset is employed for fine-grained preference learning techniques, further refining the model's reasoning capabilities.

Note that Self-Explore is not open-source.

4.4 MARIO

MARIO (Liao et al., 2024b) is an open-source mathematical reasoning dataset that integrates text analysis and code snippets, derived from the *GSM8K*, *MATH*, and *MetaMath* datasets. It comprises approximately 28.8K solutions generated through GPT-4 annotations, human reviews, and self-training. The dataset is structured in an HTML-like format, with text enclosed in <p> tags and code in <code> tags, as illustrated in Figure 14. The study employs the REACT instruction framework to ensure that GPT effectively utilizes external tools when required. The dataset was created through the following process:

1. Solution Generation for *GSM8K* and *MATH*: As depicted in Figure 15, initial solutions for these datasets were generated using a large language model and subsequently verified by humans to correct simple errors.

2. Self-Training for *MATH*: Self-training was applied to improve the coverage rate for *MATH* using an additional sampling strategy.

3. Dataset Enhancement with *MetaMath*: An additional 240K transformed questions from *MetaMath* were introduced, significantly enhancing the dataset's diversity.

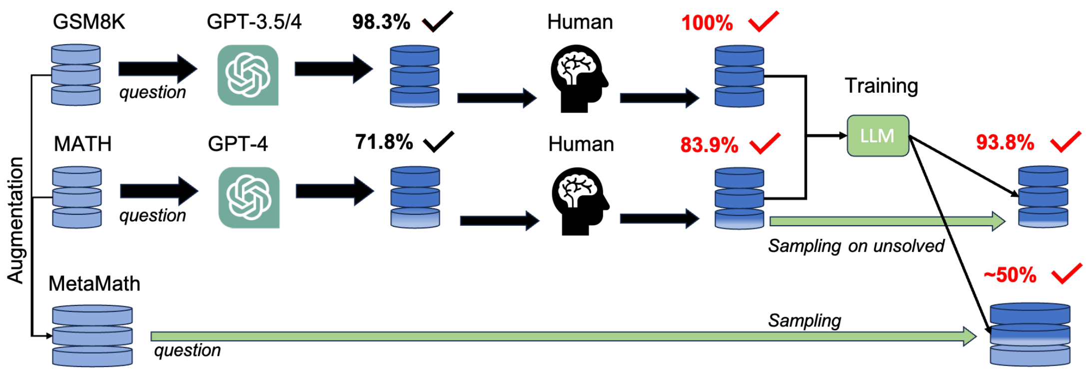


Figure 15: The data pipeline of Mario (Liao et al., 2024b). The data pipeline describes the generation process for the GSM8K and MATH datasets. GPT-4 generates initial annotations, which are then verified by humans to correct simple errors. For the MATH dataset, an extra sampling approach, based on a self-trained LLM, is used. The figure is borrowed from Liao et al. (2024b).

4.5 MathGenie

MathGenie (Lu et al., 2024) is a dataset designed to generate synthetic mathematical problems aimed at improving the mathematical reasoning skills of large language models. The resulting dataset, *MathGenieData*, comprises 170K question-solution pairs, including 110K sourced from *GSM8K* and 60K from *MATH*. This dataset is utilized for fine-tuning various pre-trained models. The framework follows a three-step process, as illustrated in Figure 16:

1. Iterative Solution Augmentation: A seed dataset of 15K math problems and solutions, derived from *GSM8K* and *MATH* is expanded using a fine-tuned LLaMA-2 70B model. Through iterative augmentation, diverse solutions are generated that significantly deviate from the originals, ensuring greater variety and broader coverage.

2. Question Back-Translation: Augmented solutions are converted into new math questions using the fine-tuned LLaMA-2 70B model. By leveraging solution constraints, the framework ensures that the newly generated questions are both reliable and meaningful.

3. Verification-Based Solution Filtering: A solution generator, fine-tuned on the LLaMA-2 70B model, produces code-integrated solutions for the newly generated questions. These solutions undergo a rigorous verification process that integrates natural language and code rationales, ensuring that only accurate solutions are retained.

Note that MathGenie is not open-source.

4.6 DeepSeekMath

The DeepSeekMath Corpus (Shao et al., 2024a) is an open-source, large-scale, high-quality mathematical reasoning dataset containing 120 billion tokens. The dataset is generated by both machines and humans. Its primary data source is *Common Crawl*, supplemented by contributions from *AlgebraicStack*, *arXiv*,

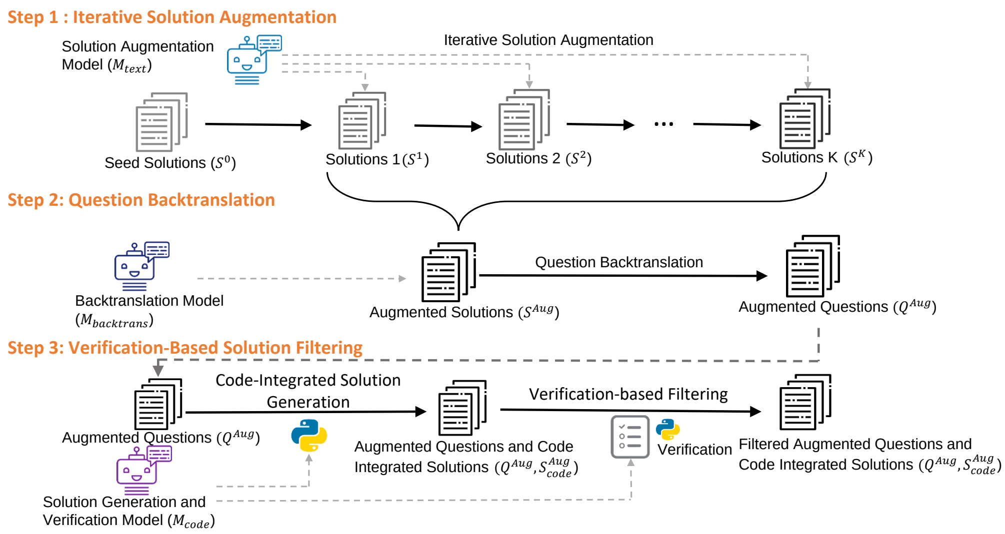


Figure 16: Framework of MathGenie (Lu et al., 2024). Step 1: The Iterative Solution Augmentation method adds more examples to human-annotated solutions in the GSM8K and MATH datasets. Step 2: Question Back-translation turns these solutions into new questions. Step 3: Verification-Based Solution Filtering selects reliable code-based solutions by generating and verifying them through a series of validation steps. The figure is borrowed from Lu et al. (2024).

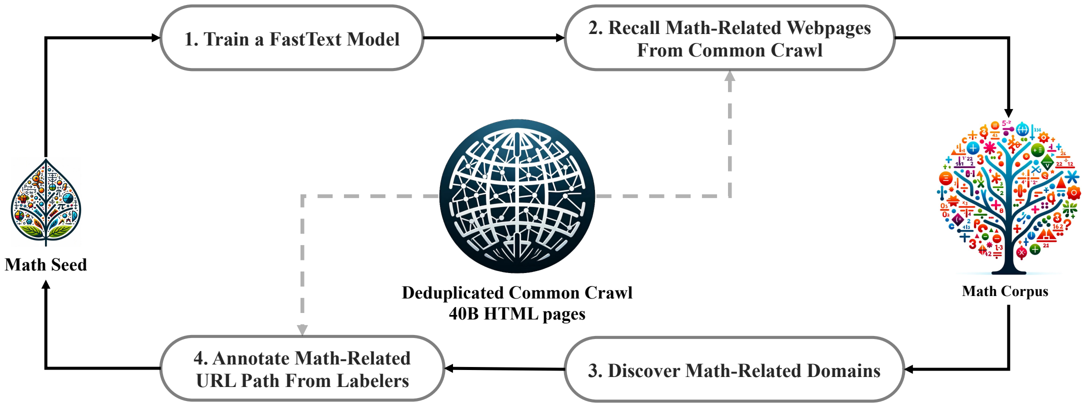


Figure 17: Pipeline of DeepSeekMath (Shao et al., 2024a). The iterative process for gathering math-related web pages from Common Crawl. The figure is borrowed from Shao et al. (2024a).

GitHub, and other natural language texts. Designed to enhance language models' mathematical reasoning capabilities, the dataset is multilingual, with a particular focus on English and Chinese mathematical content.

The dataset is constructed through an iterative process of data collection and refinement, as illustrated in Figure 17:

1. Training the Classifier: Initially, a fastText-based classifier is trained using *OpenWebMath* as positive examples and diverse web pages as negative examples. Subsequent classifier training is conducted using the collected web pages.

2. Extracting Mathematical Content: The classifier extracts additional mathematical content from *Common Crawl*, which is then refined through human annotation. To maintain quality and avoid data contamination, web pages that include questions or answers from benchmarks are filtered out.

This iterative process enhances classifier accuracy and incrementally expands the dataset.

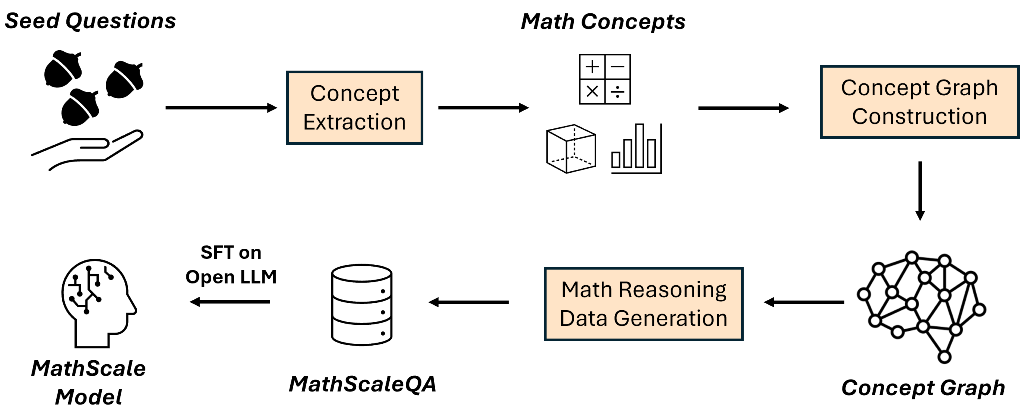


Figure 18: Overview of MathScale. MathScale starts with seed math questions and follows three steps: concept extraction, concept graph construction, and mathematical reasoning data generation. This produces the MathScaleQA dataset, which is used to train LLMs. The figure is borrowed from Tang et al. (2024).

4.7 Compute-Optimal Sampling

Compute-Optimal Sampling (Bansal et al., 2024) examines the effectiveness of synthetic data in enhancing the reasoning capabilities of language models (LMs). It compares data generated by weaker but computationally cheaper (WC) LMs with data produced by stronger yet more computationally expensive (SE) models. It uses *MATH* and *GSM8K* as seed datasets. Synthetic data is generated by sampling multiple candidate solutions for each problem using either a WC model (*e.g.*, Gemma2-9B) or an SE model (*e.g.*, Gemma2-27B) under a fixed computational budget. Candidate solutions are filtered according to the accuracy of the final answer, and the resulting data is utilized for supervised fine-tuning of LMs. The study evaluates three fine-tuning paradigms:

1. Knowledge Distillation: A student LM acquires knowledge from synthetic data generated by a teacher LM.
2. Self-Improvement: An LM improves its reasoning abilities using its own generated synthetic data.
3. Weak-to-Strong Improvement: A stronger LM improves its reasoning ability using synthetic data generated by a weaker LM.

The results indicate that models fine-tuned on WC-generated data consistently outperform those fine-tuned on SE-generated data across multiple benchmarks and fine-tuning paradigms. These findings challenge the prevailing reliance on SE models for creating synthetic data, indicating that WC models could provide a more computationally efficient approach for training sophisticated reasoning models.

4.8 MathScale

MathScale (Tang et al., 2024) is an open-source dataset designed to generate high-quality mathematical reasoning datasets using large language models such as GPT-3.5. The resulting dataset, *MathScaleQA*, is machine-generated and contains 2 million math question-answer pairs, developed through a three-step process illustrated in Figure 18:

1. Concept Extraction: GPT-3.5 identifies "topics" (*e.g.*, "Arithmetic operations") and "knowledge points" (*e.g.*, "Dot product properties") from 20,000 seed math questions. This process results in 2,000 topics and 8,000 knowledge points. The seed questions are sourced from the *MwpBench* training set, which aggregates datasets such as *GSM8K*, *MATH*, *TAL-SCQ*, *Math23k*, *Ape210k*, *GaokaoBench-Math*, and *AGIEval*.
2. Concept Graph Construction: A graph is created with nodes representing topics and knowledge points, while edges indicate their co-occurrence. A random walk algorithm is applied to sample novel combinations of topics and knowledge points.

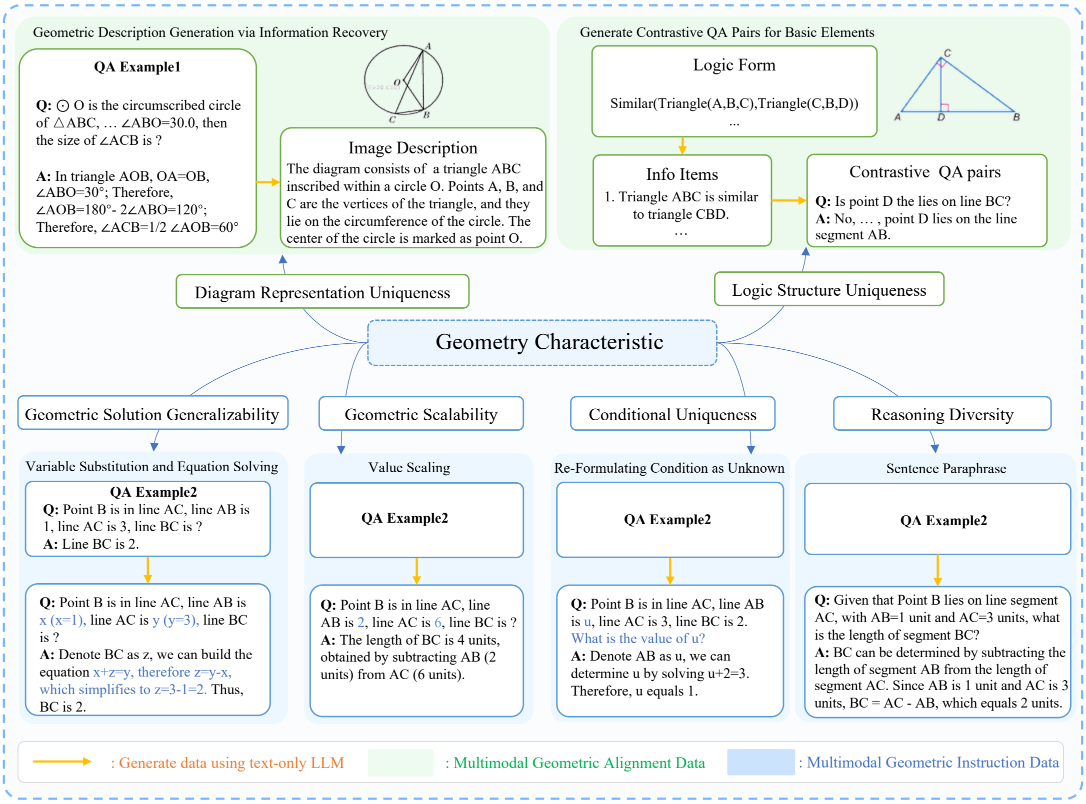


Figure 19: Framework for generating Geo170K (Gao et al., 2023). Multi-modal geometric data is generated based on the features of geometry problems. The figure is borrowed from Gao et al. (2023).

3. Data Generation: Based on the sampled combinations, GPT-3.5 generates new math questions and answers. The dataset undergoes a decontamination process to remove questions overlapping with the *MwpBench* test set.

4.9 Geo170K

Geo170K (Gao et al., 2023) is an open-source synthetic multimodal geometry dataset containing over 170K instances, including 60K image-caption pairs and 110K question-answer (QA) pairs. The dataset is designed to enhance multimodal large language models (MLLMs) in solving geometric problems by integrating textual and visual inputs. It is entirely machine-generated.

The dataset comprises two primary components: "alignment data" and "instruction data" as depicted in Figure 19. The construction of *Geo170K* follows these steps:

1. Data Generation: Existing geometry datasets (e.g., *Geometry3K* and *GeoQA+*) and ChatGPT are used to generate new data, including image descriptions and QA pairs.
2. Image Descriptions: ChatGPT generates detailed descriptions of geometric images based on QA pairs and creates contrastive QA pairs to improve the model's understanding of geometric elements.
3. Instruction Data: The dataset is expanded by generating instruction-tuning data through processes such as equation solving, value scaling, condition reformulation, and sentence paraphrasing.
4. Dataset Creation: Image descriptions, QA pairs, and instruction data are combined to form the *Geo170K* dataset.

5 Training

As discussed earlier, reasoning datasets are meticulously structured, often representing step-by-step problem-solving processes. Effectively training LLMs on such datasets requires methodologies that

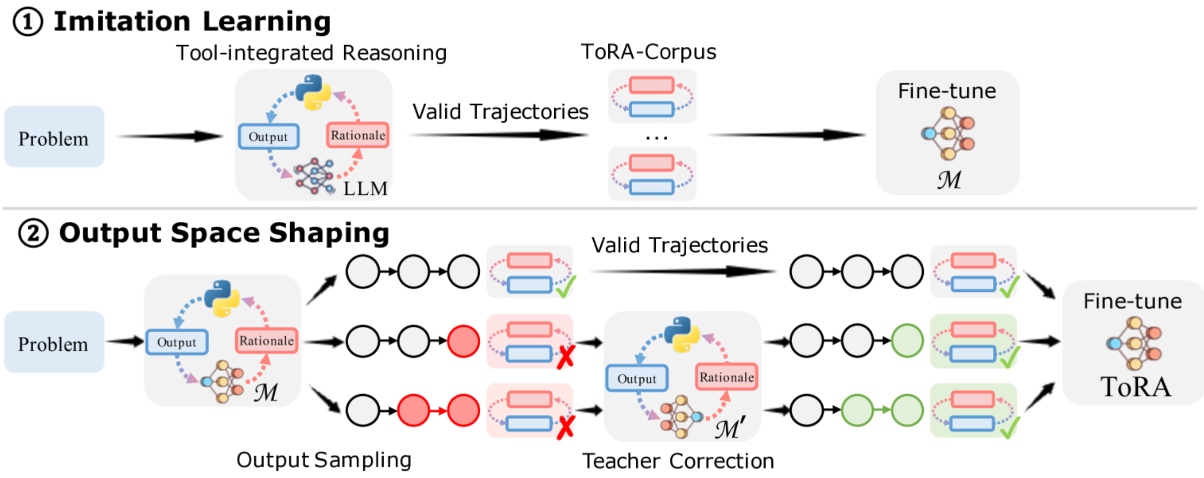


Figure 20: A two-step learning approach proposed in ToRA (Gou et al., 2024). In the first step of Imitation Learning, target LLM is prompted to tool-integrated Reasoning trajectories (TORA-CORPUS) and use this corpus to fine-tune a model. In the second step of output space shaping, a teacher model is used to maintain strong reasoning while keep diversified output trajectories. The figure is adapted from (Gou et al., 2024).

maximize the utility of each reasoning step, whether correct or erroneous. This section explores three key training paradigms designed to leverage reasoning datasets for training o1-like reasoning LLMs:

1 **Supervised Fine-tuning (SFT)**: A foundational technique that refines pre-trained LLMs by explicitly teaching structured reasoning patterns through labeled (INSTRUCTION, ANSWER) pairs.

2 **Reinforcement Learning from Human Feedback (RLHF)**: A refinement approach that aligns LLM outputs with human preferences or quality signals, further enhancing reasoning skills through iterative optimization.

3 **Direct Preference Optimization (DPO)**: A simplified alternative to RLHF that directly optimizes fine-tuned LLMs for preferred reasoning outputs without requiring intermediate reward modeling.

A summary of each of these paradigms is presented in Table 4. Below we present their methodologies, strengths, and contributions to reasoning-focused training in details.

5.1 Supervised Fine-tuning

Supervised Fine-tuning (SFT) serves as the cornerstone for developing reasoning capabilities in LLMs. By utilizing structured (*instruction*, *answer*) pairs, SFT provides explicit guidance, enabling models to learn systematic reasoning patterns and produce accurate outputs across complex reasoning tasks. The process typically begins with a pre-trained LLM, which embodies extensive general knowledge and linguistic understanding. SFT fine-tunes these models on task-specific datasets, emphasizing logical reasoning, problem-solving, and domain-specific expertise. These datasets often reflect deterministic reasoning frameworks, enabling the model to generate consistent and interpretable outputs for tasks such as mathematical problem-solving, program synthesis, and logical deduction. To further optimize performance, SFT is frequently integrated with complementary training paradigms. For instance, multi-task fine-tuning leverages diverse datasets to improve generalization, while curriculum learning structures training data to progressively increase task difficulty. This adaptability allows SFT to be tailored to specific reasoning requirements, making it a versatile and essential component of LLM training.

ToRA is a series of Tool-integrated Reasoning Agents designed to address the persistent challenges in mathematical problem-solving by combining natural language reasoning with program-based tool use (Gou et al. (2024)) . As demonstrated in Figure 20, ToRA leverages SFT on curated tool-use trajectories from mathematical datasets such as GSM8k and MATH, enabling models to interleave natural language

Paper	Key Innovation	Main Techniques
	Sec. 5.1 Supervised Fine-tuning (SFT)	
ToRA (Gou et al., 2024)	Tool-integrated Mathematical Focused Reasoning Agents	Imitation Learning, Output Space Shaping
AlphaLLM (Tian et al., 2024a)	Self Improving Training	SFT with Monte Carlo Tree Search
MARIO (Liao et al., 2024a)	Mathematical Reasoning Framework	Data Enhancement with GPT-4, Human Review, and Self-training
	Sec. 5.2 Reinforcement Learning from Human Feedback (RLHF)	
Self-Critiquing (Saunders et al., 2022a)	Fundations of Language Model Self-Critiquing	AI-assisted Human Feedback,
OVM (Yu et al., 2023)	Evaluating the Potential of Incomplete Reasoning Paths	Outcome-supervised Value Models
PPO-MCTS (Liu et al., 2024b)	Value-Guided Decoding trhough PPO	Proximal Policy Optimization, Monte Carlo Tree Search
MATH-SHEPHERD (Wang et al., 2024b)	Eliminatioin of Human Annotation	step-wise verification through MCTS
Qwen-2.5-math (Zhang et al., 2025)	Enhanced Process Reward Model	LLM-as-a-judge
Roadmap to o1 (Zeng et al., 2024)	Combination of Various Techniques to Reproduce o1	Policy Initialization, Reward Shaping, Policy Gradient
PROCESSBENCH (Zheng et al., 2024)	Benchmark for Error Identification in Mathematical Reasoning	Step-level Error Detection
PRMBENCH (Song et al., 2025)	Fine-grained Benchmark for Process Reward Models	Multi-dimensional Evaluation Benchmark
	Sec. 5.3 Direct Preference Optimization (DPO)	
CPO (Zhang et al., 2024c)	Fine-tuning CoT reasoning with ToT	Tree-of-Thoughts framework, Paired preference fine-tuning
SVPO (Chen et al., 2024c)	Step-level preferences for reasoning improvement	MCTS for step-level preferences, Value model integration
PPO-MCTS (Liu et al., 2024b)	Value-guided decoding during inference	PPO value network MCTS
Self-Explore (Hwang et al., 2024b)	Self-guided Learning with Fine-grained Rewards	First-pit Identifications
Agent Q (Putta et al., 2024)	Enhanced Agentic reasoning	MCTS with AI feedback, Offline DPO

Table 4: An overview of LLM Training Techniques (Sec. 5).

reasoning with external computational tools, including symbolic solvers and libraries. This approach harnesses the semantic analysis and planning strengths of natural language while utilizing tools for precise computation and symbolic manipulation. To overcome the limitations of traditional SFT, which often restricts the diversity of output trajectories, the authors introduce output space shaping. This technique supplements imitation learning with a broader set of self-sampled and corrected trajectories, ensuring flexibility and robustness in reasoning. The result is a suite of models that achieve significant performance improvements, including a 22% absolute gain over prior state-of-the-art WizardMath-70B on the MATH dataset. ToRA-Code-34B also surpasses GPT-4's CoT results and matches GPT-4's performance on solving problems with code, achieving an unprecedented accuracy exceeding 50% on the MATH dataset.

AlphaLLM proposed in Tian et al. (2024a), is a self-improving framework that integrates Monte Carlo Tree Search with LLMs to enhance reasoning capabilities without additional annotations. Leveraging SFT, AlphaLLM uses its imagination component to synthesize prompts, addressing data scarcity issues common in reasoning tasks. To optimize the text generation process, the authors introduce $\eta MCTS$, which formulates text generation as an options-over-Markov Decision Process (MDP) problem, improving search efficiency by reducing the search depth and employing techniques like state merging and adaptive branching. Additionally, a trio of critic models guides the search by evaluating expected rewards, node correctness, and overall trajectory outcomes. These critics dynamically utilize tools for tasks like arithmetic and code execution, ensuring accurate feedback during the search. AlphaLLM constructs high-reward trajectories using $\eta MCTS$ and integrates them into training data, forming a self-improving loop through iterative fine-tuning. Experimental results demonstrate substantial performance gains on mathematical reasoning tasks, with improvements from 57.8 to 92.0 on GSM8K and from 20.7 to 51.0 on MATH when starting from LLaMA-2-70B and WizardMath-70B-V1.0.

MARIO is a pipeline for fine-tuning LLMs on mathematical reasoning tasks using a novel dataset that integrates both text analysis and code execution (Liao et al. (2024a)). The authors address limitations in traditional fine-tuning approaches by enhancing data quality through a combination of GPT-4 annotations, human review, and self-training, correcting errors in datasets such as GSM8K and MATH. As demonstrated in Figure The proposed pipeline leverages SFT to optimize all model parameters using this curated dataset, focusing on generating precise reasoning steps alongside correct solutions. To further enhance performance, MARIO incorporates multi-task fine-tuning with LoRA, which allows the model to evaluate the validity of generated solutions through a lightweight binary classification layer while maintaining generative capabilities. This dual-role approach reduces computational costs and simplifies deployment by enabling a single model to both generate and evaluate solutions. Experimental results demonstrate significant improvements, with a 7B-parameter MARIO model achieving state-of-the-art accuracy on GSM8K and MATH datasets.

5.2 Reinforcement Learning with Human Feedback (RLHF)

Reinforcement Learning from Human Feedback (RLHF) has emerged as a critical paradigm for aligning large language models with human preferences, enabling improved reasoning and alignment capabilities. By incorporating iterative feedback and leveraging reinforcement learning techniques, RLHF enhances the models' ability to evaluate, refine, and generate outputs aligned with human expectations. Recent works have advanced this approach by introducing innovative frameworks for training reward models, integrating guided decoding, and enabling self-improvement without reliance on extensive human annotations.

5.2.1 Methodological Innovations in RLHF

Building on the foundational principles of supervised fine-tuning, RLHF takes reasoning enhancement a step further by introducing dynamic, iterative learning processes. Unlike static fine-tuning methods, RLHF relies on human or model-generated feedback to refine intermediate outputs, ensuring that models not only produce accurate results but also align with desired human preferences. This transition from fixed data-driven optimization to interactive, feedback-based refinement has enabled significant advancements in reasoning tasks. In this subsection, we delve into innovative RLHF frameworks that address challenges such as error propagation, long-horizon planning, and the integration of preference-aligned training objectives.

Self-Critiquing (Cobbe et al. (2021a)) is one of the foundational works by OpenAI, introducing verifiers to evaluate the correctness of model-generated solutions. This approach addresses a core limitation of LLMs: errors in multi-step reasoning that propagate and lead to irrecoverable mistakes. By training verifiers to rank candidate solutions during inference, the method ensures that the most accurate output is selected, achieving performance improvements equivalent to a 30-time increase in model size. The introduction of GSM8K, a dataset of 8.5K diverse grade-school math problems, has since become a standard benchmark for evaluating reasoning capabilities in mathematical challenges. Building upon previous work that introduced verifiers to assess model-generated solutions, OpenAI's subsequent research

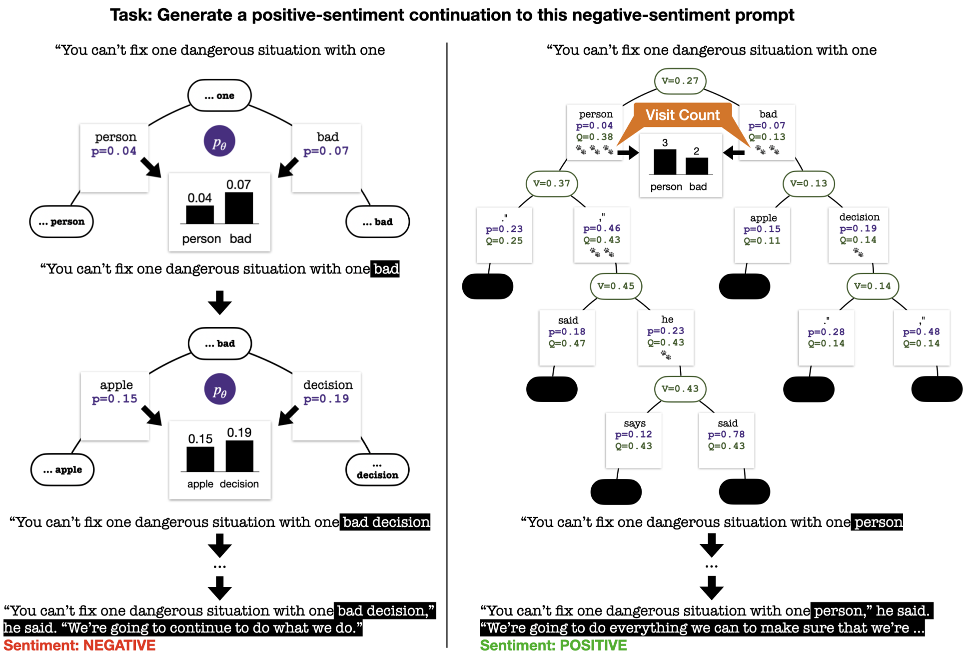


Figure 21: Left: traditional greedy decoding from a PPO policy does not satisfy the task constraint. Right: PPO+MCTS with the same PPO policy and additional value model satisfies the task constraint, where V and Q are derived from the output of the PPO value model. The figure is borrowed from (Liu et al., 2024b).

delved deeper into enhancing human evaluators' capabilities through self-critiquing models. In their 2022 study, Saunders et al. (2022a) fine-tune large language models to produce natural language critiques of their own outputs, particularly focusing on topic-based summarization tasks. This work represents a foundational step in incorporating human feedback into AI systems at scale, addressing the inherent challenges of evaluating complex outputs in tasks such as summarization. The authors fine-tune large language models (LLMs) to generate natural language critiques, which help evaluators detect flaws in topic-based summaries, uncovering up to 50% more errors compared to unaided human evaluations. These critiques are effective for both naturally occurring flaws in model-generated content and intentionally misleading summaries. The study reveals that larger models excel at generating critiques, demonstrating an ability to self-assess and refine their own outputs. By integrating critiques as feedback, these models achieve significant improvements in summarization quality, highlighting the potential for self-improvement through internal feedback mechanisms. This approach scales favorably with model size, with larger models maintaining their critique capabilities despite producing more complex, harder-to-assess outputs. The authors also introduce the Generator-Discriminator-Critique (GDC) framework to evaluate gaps between a model's ability to generate, discriminate, and critique outputs. Findings indicate that even state-of-the-art models possess latent knowledge that they fail to articulate effectively as critiques, pointing to opportunities for further development in critique generation.

OVM (Yu et al. (2023)) proposes Outcome-supervised Value Models, expanding on the use of human feedback, which improves reasoning by evaluating incomplete reasoning paths rather than step-level correctness. Unlike traditional approaches that assess step-level correctness, OVM emphasizes evaluating the potential of incomplete reasoning paths, aiming to guide models toward accurate final answers. This methodology aligns with value-based evaluation in reinforcement learning, where the future-oriented perspective enhances decision-making for long-horizon tasks. The OVM framework eliminates the need for step-level correctness annotations, leveraging outcome supervision to train a value model that predicts the overall correctness of a reasoning path. Experiments on multi-step mathematical reasoning datasets, such as GSM8K and Game of 24, demonstrate the efficacy of OVM. Notably, the OVM-7B model achieves state-of-the-art performance with 84.7% accuracy on GSM8K and 78.7% success on Game of 24, surpassing models up to 13B parameters without relying on GPT-4, external datasets, or code execution.

PPO-MCTS is a novel value-guided decoding algorithm that enhances text generation by combining Proximal Policy Optimization (PPO)-trained policies with Monte Carlo Tree Search (Liu et al. (2024b)). As demonstrated in Figure 21, unlike traditional PPO approaches, where the value model is discarded after training, PPO-MCTS retains and utilizes the value network to evaluate partial output sequences during inference. This approach aligns the training and test-time evaluation mechanisms, reducing mismatches and improving the preferability of generated text. The authors introduce critical modifications to MCTS, such as initializing child action values from parent node values , to ensure efficient exploration. Experiments across four text generation tasks, including sentiment steering, toxicity reduction, knowledge introspection, and chatbot generation, demonstrate the efficacy of PPO-MCTS. For instance, on sentiment steering tasks, PPO-MCTS improves success rates by 30% absolute over direct PPO sampling, while human evaluations consistently favor PPO-MCTS outputs. The algorithm also reduces toxicity by 34% relative and generates more useful responses in QA tasks. These results highlight the underutilized potential of PPO value models in inference and establish guided decoding via MCTS as a powerful technique to improve reasoning and alignment in RLHF-trained LLMs.

Qwen-2.5-math proposes enhancements to Process Reward Models (PRMs) for mathematical reasoning by addressing limitations in Monte Carlo estimation and Best-of-N (BoN) evaluation (Zhang et al. (2025)). The authors argue that past MC estimation for generating noisy data undermines step-wise error verification, and identify biases in BoN evaluation, which prioritize outcome-based metrics over process-level correctness. To mitigate these issues, they introduce a consensus filtering mechanism that integrates MC estimation with LLM-as-a-judge for efficient data synthesis and propose combining BoN with step-wise benchmarks like PROCESSBENCH for comprehensive evaluation. Their approach significantly improves error identification and model generalization, setting new benchmarks for process supervision in reasoning tasks.

MATH-SHEPHERD proposed by Wang et al. (2024b), is a process reward model for mathematical reasoning that eliminates the need for human annotations by automatically constructing step-wise supervision data using MCTS. Unlike prior PRM approaches, MATH-SHEPHERD evaluates the correctness of each step based on its potential to deduce the correct final answer, enabling scalable training. It is applied in two scenarios: verification, where it reranks outputs to improve accuracy, and reinforcement learning, where it refines LLMs via step-by-step Proximal Policy Optimization. Experiments on GSM8K and MATH demonstrate substantial gains for open-source LLMs, with Mistral-7B achieving 84.1% accuracy on GSM8K and 33.0% on MATH with PPO, further improving to 89.1% and 43.5% with verification. These results highlight the potential of automated process supervision for advancing reasoning capabilities in LLMs without reliance on manual annotations.

Roadmap to o1 (Zeng et al. (2024)) proposes a comprehensive framework for reproducing the reasoning capabilities of OpenAI's o1 model through reinforcement learning, focusing on policy initialization, reward design, search, and learning. Building on prior work like (Saunders et al., 2022a) and leveraging techniques including as Monte Carlo Tree Search and policy gradient methods, this work outlines how these components collectively enable human-like reasoning, iterative self-correction, and exploration of complex solution spaces. Policy initialization equips models with systematic reasoning behaviors, while reward design provides dense signals to guide learning and search. Search generates high-quality training data during both training and testing phases, and learning integrates these insights to refine policies without the need for costly human annotations. With these techniques, the paper charts a path toward achieving strong reasoning abilities akin to o1, providing a blueprint for future open-source and advanced LLM projects.

5.2.2 Evaluation and Benchmarking for RLHF

As RLHF techniques grow more sophisticated, robust evaluation mechanisms are crucial to measure their effectiveness and identify areas for improvement. This section examines efforts to evaluate RLHF through comprehensive benchmarks and datasets designed to assess models' ability to detect errors, align with human preferences, and generalize to complex reasoning tasks.

PRM800K is an early work conducted by OpenAI (Lightman et al. (2023)), which investigates process supervision versus outcome supervision in training reward models for multi-step reasoning tasks, emphasizing the significance of fine-grained feedback in RLHF. Using the challenging MATH dataset, the authors demonstrate that process supervision, which provides feedback for each reasoning step, significantly outperforms outcome supervision, achieving a 78.2% problem-solving rate on a representative test subset. They further introduce , a large-scale dataset containing 800,000 step-level human feedback labels, enabling detailed evaluation and training of process reward models (PRMs). The study highlights that PRMs trained with process supervision are more reliable, better aligned with human reasoning, and less prone to using flawed intermediate steps to arrive at correct final answers.

ProcessBench (Zheng et al. (2024)) is a benchmark targeting the identification of erroneous steps in mathematical reasoning, particularly in challenging problems like competition-level math. With 3,400 test cases and human-annotated step-level error labels, PROCESSBENCH ensures robust evaluation through its emphasis on problem diversity, large-scale expert annotations, and a straightforward protocol for detecting the first incorrect step. The authors assess two model types: process reward models (PRMs) and critic models, revealing that current PRMs underperform on challenging problems compared to general language models like GPT-4o and QwQ-32B-Preview. ProcessBench highlights the limitations of existing PRMs and demonstrates the potential of general models in automated reasoning assessment, fostering advancements in scalable oversight.

PRMBench (Song et al. (2025)) is a comprehensive benchmark designed to evaluate PRMs in fine-grained error detection across reasoning tasks. Featuring 6,216 problems and 83,456 step-level labels, PRMBench assesses PRM performance across three primary domains—simplicity, soundness, and sensitivity—further divided into nine sub-categories, such as redundancy detection and deception resistance. The dataset emphasizes diverse error types, ensuring robust evaluations. Experiments involving 15 models, including PRMs and state-of-the-art general-purpose LLMs, reveal critical limitations in current PRMs, with even the strongest models marginally outperforming random baselines. This highlights substantial room for improvement, particularly in nuanced error detection and multi-solution consistency. By offering a toolkit for automated evaluation and data generation, PRMBench paves the way for advancing reliable process-level evaluation frameworks, especially in reasoning-intensive RLHF scenarios.

5.3 Direct Preference Optimization (DPO)

Direct Preference Optimization (DPO) (Rafailov et al., 2024; Xiao et al., 2024; Amini et al., 2024) is an emerging training paradigm designed as a simpler alternative to RLHF. Unlike RLHF, which relies on reward modeling and reinforcement learning algorithms like Proximal Policy Optimization (PPO), DPO directly optimizes a language model's outputs to align with human preferences by fine-tuning the model on comparison data. This approach eliminates the complexity of learning a reward function and instead leverages pairwise preference data to improve the quality and alignment of generated outputs, offering an efficient and scalable solution for enhancing reasoning capabilities in LLMs.

CPO proposed by Zhang et al. (2024c), is a novel fine-tuning method that enhances CoT reasoning in LLMs by leveraging preference information from the Tree-of-Thought (ToT) framework. While ToT improves reasoning quality by exploring multiple reasoning paths through tree search, it incurs high inference costs. CPO addresses this limitation by extracting and utilizing the inherent preference data generated during ToT's search process. As demonstrated in Figure 22, At each reasoning step, thoughts included in the best-discovered paths are marked as preferred, while alternative thoughts are labeled as dispreferred. These paired preference data are used to fine-tune LLMs with DPO, aligning CoT reasoning with ToT's strategic depth without increasing inference complexity. Experiments on seven reasoning datasets, including question answering, fact verification, and arithmetic tasks, show that CPO significantly enhances LLM reasoning capabilities, achieving performance comparable to or better than ToT while being over 50 times faster at inference. This work highlights the potential of integrating DPO with CoT to achieve efficient and effective reasoning in LLMs.

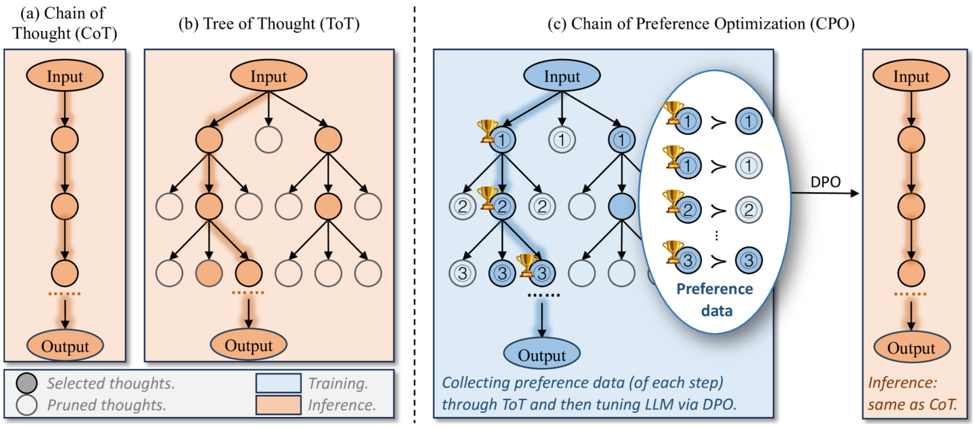


Figure 22: Comparison between the reasoning process of CoT, ToT, and Chain-of-Preference Optimization (CPO). In CoT, the model generates a single reasoning path sequentially, while ToT explores multiple paths at each step, retaining only the top-ranked nodes and pruning the rest. CPO builds on ToT by using its search tree to generate preference data, with preferred nodes marked by trophies. The figure is borrowed from (Zhang et al., 2024c).

SVPO proposed by (Chen et al., 2024c) offers a fine-grained approach to preference learning by focusing on step-level preferences rather than solution-level annotations through *Step-level Value Preference Optimization*. Unlike existing methods that rely heavily on costly, coarse-grained annotations from models like GPT-4, SVPO employs MCTS to autonomously identify step-level preferences during the reasoning process. By analyzing Q-values at each step, SVPO highlights specific reasoning errors and provides tailored signals for improvement. Additionally, SVPO integrates a value model alongside DPO, where the value model aids in navigating effective reasoning paths and refines the learning process. This dual focus on step-level granularity and value integration allows SVPO to bypass labor-intensive annotations while achieving superior mathematical reasoning capabilities, outperforming state-of-the-art models and even rivaling GPT-4 performance on 7B LLMs. The framework demonstrates that step-level preferences provide deeper insights into model reasoning errors and substantially enhance decision-making in multi-step tasks.

PPO-MCTS is a novel approach that enhances Proximal Policy Optimization (PPO) by integrating Monte Carlo Tree Search for value-guided decoding (Liu et al. (2024b)). While previous works like Chain of Preference Optimization leverage preference data during training, PPO+MCTS extends this concept to inference by utilizing the PPO-trained value network to evaluate and guide partial reasoning paths, aligning outputs more closely with human preferences. This method reduces mismatches between training and inference and improves text generation across tasks such as sentiment steering, toxicity reduction, and knowledge introspection. For instance, PPO+MCTS achieves a 30% absolute success rate improvement in sentiment steering and reduces toxicity by 34% relative, significantly outperforming standard PPO. By retaining and utilizing value models during inference, PPO+MCTS demonstrates how preference-driven optimization can further enhance alignment and controllability in RLHF-trained models.

Self-Explore , proposed by Hwang et al. (2024b), is a novel approach to enhance the mathematical reasoning capabilities and overcome the issues of relying on external annotations, especially manual annotation. This approach uses fine-grained rewards derived from self-generated rationales, without relying on external annotations or proprietary models. The method focuses on identifying the "first pit", or the initial incorrect step in a reasoning path, and uses this information to construct granular step-level preference datasets for DPO. By training the model to reduce the likelihood of generating these flawed steps while favoring correct reasoning paths, Self-Explore aligns model outputs with step-by-step improvement. The framework employs the LLM itself as a self-guided explorer to identify pits through iterative sampling and refines its predictions by applying a step-level preference learning objective. Experiments on GSM8K and MATH benchmarks demonstrate significant improvements, with accuracy gains of up to 13.19% on GSM8K and 3.54% on MATH compared to supervised fine-tuning.

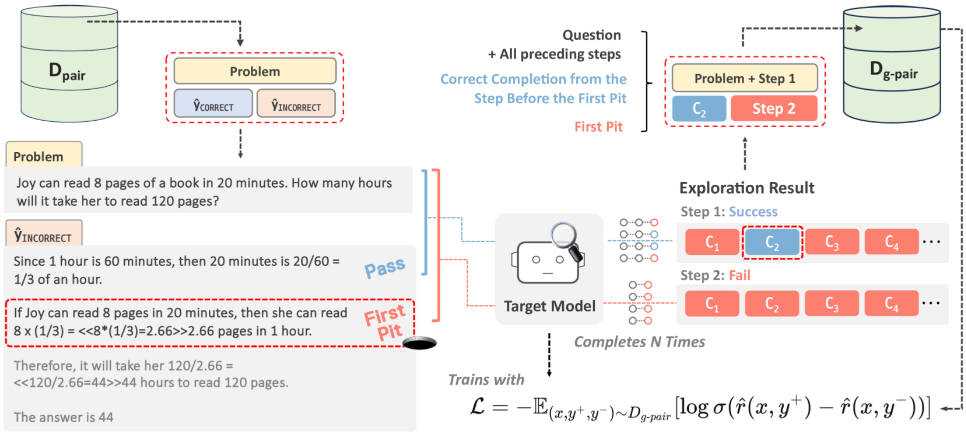


Figure 23: Self-Explore framework overview. Starting from a pairwise dataset built via outcome supervision, the target model generates multiple completions at each step. If no completion reaches the correct answer, the step is identified as the first pit. The dataset is then reorganized into a granular preference dataset enhancing training with finer-grained learning signals. The figure is borrowed from (Hwang et al., 2024b).

AgentQ (Putta et al. (2024)) introduces a novel framework that enhances autonomous AI agents by combining agentic reasoning with DPO to improve performance in dynamic, multi-step environments. The framework addresses challenges such as compounding errors, sparse rewards, and limited exploration inherent in traditional supervised fine-tuning approaches. Agent Q employs Monte Carlo Tree Search to guide exploration, leveraging an LLM as a base model for generating rationales and actions. The novelty of this search process is that it is augmented with AI feedback, which provides process-level rewards to refine decision-making and address credit assignment issues. To further improve, Agent Q utilizes the traces from successful and unsuccessful trajectories to construct preference pairs for offline DPO, enabling the model to learn from both positive and negative outcomes. Experiments on the WebShop benchmark and real-world booking environments demonstrate significant gains: the model's success rate improves from 18.6% to 81.7% in zero-shot booking tasks, further increasing to 95.4% with online search capabilities, even outperforming GPT-4. This work establishes a powerful combination of search and learning, marking a substantial advance in developing reliable, reasoning-capable autonomous agents for real-world tasks.

5.4 Discussion on how training techniques relate to reasoning

Bridging the Pre-training vs. User Objective Gap. Fine-tuning exposes the model to datasets with structured tasks, step-by-step solutions, and explicit rationales (Wang et al., 2022). During fine-tuning, the model learns to produce well-organized, human-like reasoning steps rather than just continuing text. The fine-tuned model more consistently engages in processes that approximate human thought patterns such as identifying salient information, extracting symbolic structures, or performing multi-step inferences.

Integration of Domain-Specific Knowledge and Reasoning. Another advantage of fine-tuning is its capacity for domain adaptation (Peng et al., 2023; Zhou et al., 2023). Supervised fine-tuning allows one to incorporate specialized knowledge into the model's representations (Yang et al., 2023; Sun et al., 2023b). This partial re-optimization ensures that the model retains its broad language proficiency while follows domain constraints, interpret specialized terminology, and handle domain-specific reasoning.

Instruction-based Fine-tuning for Structured Reasoning. Instruction-based fine-tuning could improve chain-of-thought capabilities (Xu et al., 2023). By training on instructions that embody exemplar reasoning processes, LLMs can internalize structured logical flows and reflect them back in their own outputs (Honovich et al., 2022; Kung and Peng, 2023). This mechanism could be useful for tackling complex, multi-step tasks like math derivations, scientific question answering, or code generation, where a "reasoning trace" is central to correctness (?Li et al., 2023).

Policy, Rewards, and Iterative Fine-Tuning. From an RL perspective, an LLM's behavior is captured by its *policy*, which is iteratively updated to produce more desirable outputs. In contrast to supervised approaches, which penalize deviations from a single reference answer, RL-based methods focus on *rewards*: scalar feedback signals indicative of the *quality* of generated text (Schulman et al., 2017; Ouyang et al., 2022). These rewards can be derived from human annotations (*e.g.*, Reinforcement Learning from Human Feedback, RLHF) or automated signals (*e.g.*, Reinforcement Learning from AI Feedback, RLAIF). In either case, the reward model provides an assessment of how relevant, correct, or helpful the LLM's outputs appear. Policy optimization algorithms such as Proximal Policy Optimization (PPO) then adapt the LLM parameters to maximize these rewards. By selectively rewarding coherent reasoning steps or well-articulated justifications, RL enhances the chain-of-thought reasoning of LLMs (Wei et al., 2023; Saeidi et al., 2024).

Strengthening Reasoning Through Exploration and Uncertainty. A key advantage of RL for large language models lies in its inherent *exploration* capabilities (Lee et al., 2023). Instead of forcing an LLM to converge prematurely to a single output, RL-based methods encourage the model to sample multiple plausible solutions and assess their respective merits. By doing so, the model does not merely reproduce the highest-likelihood sequence according to its pre-training distribution; it is guided to probe less obvious or novel reasoning pathways that might yield more insightful, accurate, or creative responses. Over multiple training cycles, the model refines its judgment of how to balance exploration and exploitation: it becomes adept at venturing beyond typical patterns when beneficial, while still capitalizing on well-understood reasoning heuristics (Rafailov et al., 2024).

Addressing Multi-Objective Constraints and Alignment. Reasoning tasks often require LLMs to navigate conflicting objectives: they must be *helpful* while remaining *harmless*, or *accurate* yet *concise*. RL provides a natural way to incorporate multi-objective constraints through carefully designed reward functions that capture multiple dimensions of quality, such as factual correctness, style, safety, and user satisfaction. By decomposing these objectives and assigning corresponding reward signals, an LLM can learn to weigh them appropriately. When combining these multi-faceted objectives, advanced RL strategies (*e.g.*, Safe RLHF or Conditional Online RLHF) help mitigate the risk of "reward hacking," wherein a model might overfit to a single objective at the expense of others (Amodei et al., 2016).

6 Inference

Multi-step reasoning tasks are prone to errors at any step, as small mistakes can cascade into incorrect final answers. To address this, reasoning LLMs often generate multiple reasoning paths for a given input question during the inference stage and choose the answer that aligns best with the most logically consistent and broadly supported reasoning steps.

In this section covers the following three widely used techniques in reasoning LLMs, which we think are crucial in building o1-like reasoning models:

1 **Tree of Thoughts**, which represents the reasoning process as a tree structure and explores various branches to determine the most effective path.

2 **Automated Reasoning Critic**, which employs a trained critic model to evaluate and validate the reasoning steps generated by the LLMs.

3 **Self-Correction**, where the LLM mimics human critical thinking by iteratively reviewing, identifying errors, and refining its reasoning steps to enhance accuracy and logical consistency.

In addition to these three inference techniques, we will also explore "Inference Scaling Laws", which provide insights into how reasoning performance improves as inference time increases, enabling us to balance the trade-off between computational efficiency and reasoning accuracy. An overview of this section is provided in Table 5.

Paper	**Key Innovation**	**Main Techniques**
Sec. 6.1 Tree of Thoughts		
Tree of Thoughts (Yao et al., 2024)	1st tree-structured reasoning framework	BFS/DFS search, Self-evaluation, Backtracking
EURUS (Yuan et al., 2024)	Tree-structured alignment dataset	ULTRAINTERACT dataset, Preference learning
TS-LLM (Feng et al., 2023)	AlphaZero-inspired framework	Markov Decision Process (MDP) formulation, Deep search (64 depth)
MCTSr (Zhang et al., 2024b)	Enhanced MCTS for math	Self-reflection, Dynamic pruning, Upper Confidence Bound (UCB)
ALPHALLM (Tian et al., 2024b)	Self-improvement framework	Option-level MCTS, Adaptive branching, State merging
MCTS-DPO (Xie et al., 2024)	Step-level preference learning w/ MCTS	MCTS guided exploration, DPO updates, Step-level signals
AlphaMath (Chen et al., 2024b)	Self-supervised MCTS reasoning	Step-level value model, Beam search, Self-improvement
ReST-MCTS* (Zhang et al., 2024a)	Process-reward enhanced MCTS	Per-step rewards, Dual optimization, Dynamic exploration
Sec. 6.2 Automated Reasoning Critic		
CriticGPT (McAleese et al., 2024)	LLM-based code critique	Tampered data generation, RLHF, Bugs identifying
AutoMathCritique (Xi et al., 2024)	Two-player math reasoning	Dynamic supervision, Error generation
LLM-ARC (Kalyanpur et al., 2024)	Neuro-symbolic reasoning	LLM + reasoning engine integration, Answer Set Programming (ASP) solver
Sec. 6.3 Self-Correction		
SCoRe (Kumar et al.)	Multi-turn RL framework	Self-generated data, Two-stage training, Reward shaping
CoSC (Gao et al., 2024)	Embedded self-correction	Program generation, execution, and verification, Two-phase fine-tuning
DotaMath (Li et al., 2024)	Integrated mathematical reasoning	Multi-round correction, Python executor, Task decomposition
Sec. 6.4 Inference Scaling Laws		
Scale-Compute (Snell et al., 2024)	Test-time compute analysis	Compute-optimal strategy, Process-based Reward Models (PRMs) search
REBASE (Wu et al., 2024)	Reward balanced search	Dynamic tree optimization, Pruning with a reward model
LLMonkeys (Brown et al., 2024)	Sampling analysis	Repeated sampling, Exponentiated power law, Cost optimization
STILL-2 (Min et al., 2024)	Three-phase training	Imitation, Exploration, Self-improvement, Long-form Thought Dataset
MindStar (Kang et al., 2024)	No-tuning enhancement	PRM-guided search, Dynamic exploration, Levin tree search

Table 5: An overview of LLM Inference Techniques (Sec. 6).

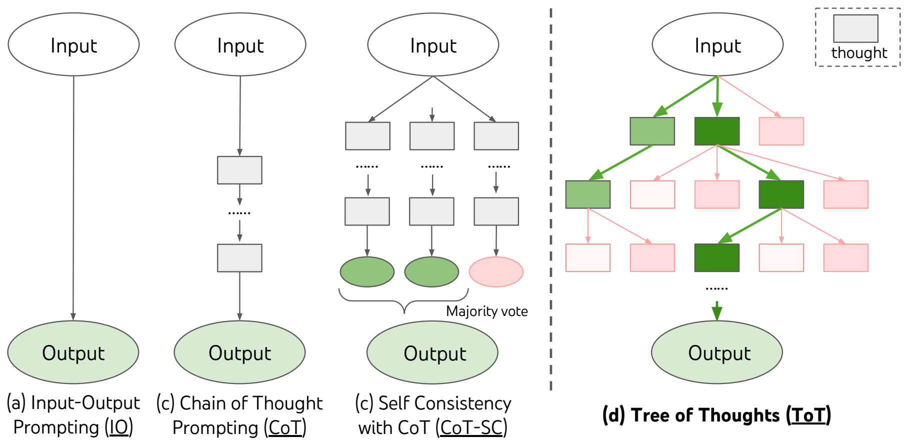


Figure 24: The framework introduced in Tree-of-Thought (ToT) (Yao et al., 2024) conceptualizes problem-solving as a structured tree of intermediate steps. Each "thought" represents a coherent intermediate reasoning step, with ToT distinctively utilizing tree-structured exploration and evaluation. The figure is adapted from (Yao et al., 2024).

6.1 Tree of Thoughts

In complex reasoning tasks, systematically exploring multiple paths of thought is crucial for finding optimal solutions. Tree of Thoughts represents the reasoning process as a tree structure, enabling models to systematically explore and evaluate different solution branches. This approach not only helps models find optimal solutions but also prevents them from getting stuck in local optima. The framework employs various tree search strategies, starting from fundamental methods like Breadth-first and Depth-first Search (Sec. 6.1.1), and advancing to more sophisticated approaches such as Monte Carlo Tree Search (Sec. 6.1.2).

6.1.1 Breadth-first Search & Depth-first Search

Tree of Thoughts (Yao et al., 2024) is introduced to enhance LLMs via enabling deliberate problem-solving through a tree-based reasoning structure. As show in Figure 24, unlike traditional sequential generation, ToT models problem-solving as a search through a tree of potential solutions, allowing exploration of multiple reasoning paths, self-evaluation, and backtracking to achieve globally optimal decisions. By integrating classical heuristic search methods, such as Breadth-first Search (BFS) and Depth-first Search (DFS), with LLMs' generative capabilities, ToT significantly improves performance on tasks requiring complex planning, such as Game of 24 (74% success vs. 4% with CoT), creative writing, and mini crosswords. This modular, general framework does not require additional training, supports various search algorithms, and offers substantial advantages for challenging tasks, though it incurs higher computational costs.

EURUS (Yuan et al., 2024) is a suite of LLMs designed to advance reasoning capabilities through a novel tree-structured alignment dataset called ULTRAINTERACT. While ToT models problem-solving as a search tree of reasoning steps to explore and evaluate multiple paths dynamically, EURUS uses a *predefined preference tree* for training and evaluation, focusing on leveraging high-quality alignment data for systematic reasoning. EURUS achieves state-of-the-art performance among open-source models on tasks like mathematics, logical reasoning, and code generation, surpassing GPT-3.5 Turbo on several benchmarks. The ULTRAINTERACT dataset underpins this success by providing diverse instructions, preference trees for multi-turn interactions, and paired correct/incorrect reasoning paths, enabling both supervised fine-tuning and preference learning.

TS-LLM (Feng et al., 2023) short for Tree-Search-enhanced LLMs is a framework inspired by AlphaZero (Silver et al., 2017), that integrates tree-search algorithms with learned value functions to enhance both the decoding and training of LLMs. TS-LLM formulates language generation as a Markov Decision Process (MDP) and employs tree-search methods to guide reasoning and decision-making. Tree search methods include BFS, DFS, and Monte Carlo Tree Search, which will be introduced in the following

	Monte Carlo Tree Search (MCTS)	Breadth-First Search (BFS)	Depth-First Search (DFS)
Purpose	Optimize decisions by balancing exploration and exploitation	Systematically find shortest paths or explore levels.	Explore deeply for exhaustive search or solution finding
Exploration Strategy	Probabilistic, uses random sampling and simulations	Deterministic, level-by-level traversal	Deterministic, explores one branch deeply before backtracking
Tree Growth	Dynamic, focuses on promising areas based on simulations	Exhaustive, explores all nodes at each level	Exhaustive, explores all nodes in one path before others
Use Case	Decision-making in large, uncertain spaces (*e.g.*, games like Go, chess)	Shortest path, level-order traversal (*e.g.*, social network analysis)	Exhaustive search, backtracking tasks (*e.g.*, mazes, puzzles)

Table 6: Comparison of MCTS, BFS, and DFS.

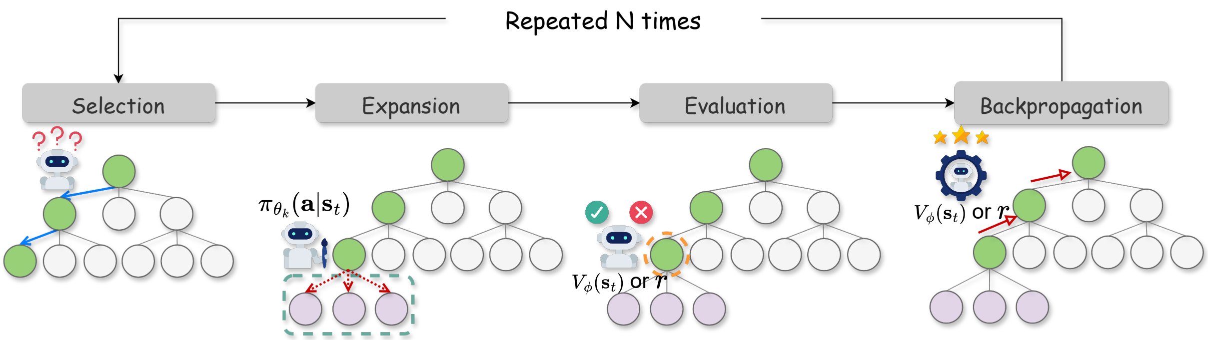


Figure 25: The overview of four key steps of MCTS: selection, expansion, evaluation, and backpropgation. The figure is copied from (Chen et al., 2024b).

part. The TS-LLM framework supports deep searches (up to depth 64) and iterative improvements through a policy-value distillation loop. It is validated on tasks like reasoning, planning, alignment, and decision-making.

6.1.2 Monte Carlo Tree Search

Monte Carlo Tree Search (MCTS) is a decision-making algorithm in environments where the action space is large, and the state space is complex, as in games like Go, Chess, or various planning problems. The core of MCTS lies in building a search tree incrementally by performing simulations (or "rollouts") to estimate the expected outcomes of different actions. The search process involves four key steps: Selection, Expansion, Simulation, and Backpropagation, as shown in Figure 25. Different from that BFS and DFS are deterministic and exhaustive, focusing on systematic exploration, MCTS is adaptive and probabilistic, optimizing for rewards in complex decision-making. More comparison details between BFS, DFS and MCTS can be found in Table 6. Here, we focus on how MCTS can be adapted in the tree-of-thoughts framework.

MCTSr short for **M**onte **C**arlo **T**ree **S**elf-**r**efine algorithm (Zhang et al., 2024b) integrates MCTS with LLMs to tackle complex mathematical reasoning tasks, including Olympiad-level problems. The algorithm incorporates key MCTS processes, while adapting them to the stochastic nature of LLM outputs. It leverages self-reflection and self-evaluation mechanisms, where answers are iteratively refined and scored using model-driven reward sampling. A *dynamic pruning strategy*, combined with an improved Upper Confidence Bound (UCB) formula, balances exploration and exploitation for optimal decision-making. Experiments on datasets like GSM8K, MATH, AIME, and OlympiadBench demonstrate significant improvements, with MCTSr outperforming baseline methods and achieving state-of-the-art results on open-source LLMs. This research highlights the algorithm's potential for advancing LLM applications in complex reasoning and decision-making tasks.

ALPHALLM (Tian et al., 2024b) is a framework designed for the self-improvement of LLMs across various complex reasoning and planning tasks, extending beyond the domain of mathematics. It introduces a flexible *option-level* MCTS, where nodes represent sequences or collections of tokens, effectively reducing search depth in language tasks. The framework further enhances search efficiency and quality through adaptive branching, state merging, and the use of critic models, including a value function, a process reward model, and an outcome reward model. From a data utilization perspective, ALPHALLM actively generates new prompts through its "Imagination" component to mitigate data scarcity, creating a continuously evolving dataset for training and fine-tuning the LLM. By leveraging high-quality trajectories obtained from MCTS-guided searches, ALPHALLM iteratively refines the LLM's policy, driving consistent performance improvements.

MCTS-DPO (Xie et al., 2024) leverages MCTS for *step-level* preference data collection, and transforms instance-level supervision into granular stepwise signals. MCTS utilizes its look-ahead capability and self-evaluation feedback to refine reasoning paths, balancing exploration and exploitation. Using Direct Preference Optimization (DPO), the model policy is iteratively updated, forming a self-improvement loop. Theoretical analysis demonstrates the advantages of online preference learning, and experiments show significant performance improvements on tasks like GSM8K, MATH, and ARC-C, highlighting MCTS's role as a policy improvement operator in aligning LLMs with human-like reasoning. While option-level MCTS helps select the best among several options, offering mid-level granularity for decision-making, step-level MCTS evaluates and refines the reasoning process step by step, providing detailed and precise feedback for complex reasoning tasks.

AlphaMath (Chen et al., 2024b) employs MCTS to enhance mathematical reasoning in LLMs without relying on process-supervised annotations. By combining MCTS with a step-level value model, AlphaMath autonomously generates high-quality reasoning paths and evaluates intermediate steps, facilitating iterative self-improvement. To reduce the computational cost of MCTS inference, the framework introduces a step-level beam search strategy that effectively balances efficiency and reasoning quality. Experimental results on datasets such as GSM8K and MATH verify that AlphaMath achieves superior performance, *even without human or GPT-4 annotated solutions*. This framework underscores the potential of MCTS as a policy optimization strategy for advancing LLM reasoning ability and extends its applicability beyond mathematics to tasks requiring stepwise reasoning and validation.

ReST-MCTS* (Zhang et al., 2024a) integrates per-step process rewards, enabling more granular evaluation and guidance during the reasoning process. These inferred rewards play a dual role: refining the process reward model to improve its evaluation accuracy and identifying high-quality reasoning traces to enhance the policy model through iterative self-training. Compared to AlphaMath, which focuses on self-supervised learning through beam search and prioritizes generating complete reasoning paths from question-answer pairs, ReST-MCTS* adopts a more dynamic and collaborative approach. It employs MCTS* to iteratively explore, evaluate, and refine reasoning steps with the aid of a process reward model. This allows ReST-MCTS* to balance global exploration and local exploitation more effectively, leading to better reasoning paths. Additionally, ReST-MCTS* emphasizes the mutual enhancement of both the policy and reward models, whereas AlphaMath primarily targets policy model improvement. Experimental results on benchmarks such as GSM8K, MATH, and SciBench highlight ReST-MCTS*'s superior accuracy over AlphaMath, especially in tasks involving complex multi-step reasoning.

6.2 Automated Reasoning Critic

In the reasoning process of LLMs, the ability to identify and correct faulty reasoning steps is essential. Automated Reasoning Critic (Barto et al., 1983; Saunders et al., 2022b) introduces dedicated critic models to evaluate the correctness and logical consistency of reasoning steps, thereby improving the reliability of the reasoning process. This approach mirrors how humans validate their thinking process when solving complex problems, providing a systematic way to assess and improve the quality of generated reasoning.

CriticGPT (McAleese et al., 2024) is a novel approach to scalable oversight by leveraging LLM-based critics to assist humans in evaluating outputs generated by LLMs, with a focus on code review tasks. This work is motivated by a critical limitation of reinforcement learning from human feedback (RLHF): as LLMs become more capable, even experts struggle to reliably assess the correctness of their outputs.

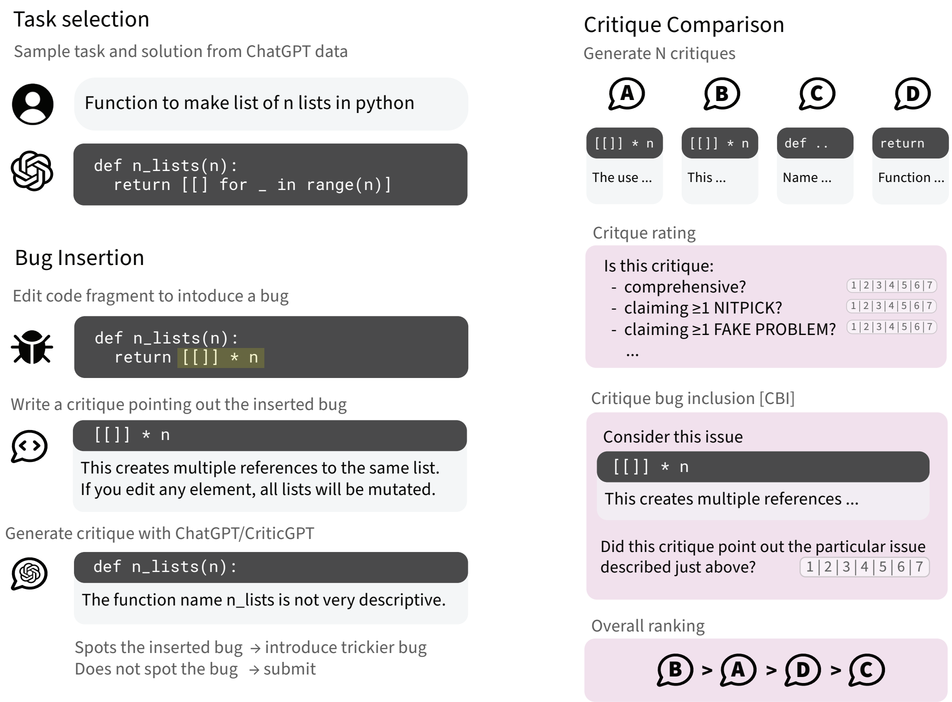


Figure 26: The data generation process proposed in CriticGPT (McAleese et al., 2024) involves contractors introducing subtle bugs into ChatGPT-generated code, documenting the issues, and generating critiques using ChatGPT or CriticGPT. Critiques are evaluated for accuracy and relevance, refining critique quality. The figure is adapted from (McAleese et al., 2024).

CriticGPT is trained using RLHF to generate detailed critiques, identifying flaws in LLM-written code. As shown in Figure 26, a tampering-based data generation approach is proposed to create high-quality training datasets for CriticGPT. Human contractors modify LLM-generated code by deliberately introducing subtle, realistic bugs, ensuring these errors are challenging to detect. Each inserted bug is accompanied by a detailed description, serving as a gold standard for evaluating critiques. The tampered code ensures a diverse range of errors, including logic flaws and unsafe practices, simulating real-world scenarios. The results show that CriticGPT outperforms human reviewers in identifying bugs, with critiques preferred over human-written ones in 63% of cases. CriticGPT also highlights numerous errors (previously labeled as "flawless") in the training data of ChatGPT, demonstrating its efficacy even on tasks outside its training distribution. The paper also introduces *Force Sampling Beam Search*, an innovative inference-time strategy designed to balance comprehensiveness and hallucination rates in critiques.

AutoMathCritique is (Xi et al., 2024) a two-player framework, where a critique model supervises an actor model to enhance LLM reasoning performance, particularly for complex tasks like mathematical problem-solving. In the reasoning phase, AutoMathCritique leverages Automated Reasoning Critic (ARC) by integrating critique models to dynamically supervise and enhance the actor model's reasoning process. As shown in Figure 27, the actor model first generates a reasoning path and a solution, while the critique model analyzes each step, detecting errors and providing detailed feedback in natural language. The actor then refines its reasoning path based on this feedback, iteratively improving the solution. This process continues until no significant errors remain or the desired accuracy is achieved. By identifying logical inconsistencies or calculation errors, it enables the actor to perform precise corrections and achieve better exploration efficiency. This dynamic interaction significantly boosts reasoning performance, particularly for complex queries, as demonstrated on GSM8K and MATH datasets, where the inclusion of critique models led to notable accuracy improvements.

LLM-ARC (Kalyanpur et al., 2024) is a neuro-symbolic framework designed to enhance the logical reasoning capabilities of LLMs by integrating an ARC. It excels at converting complex natural language

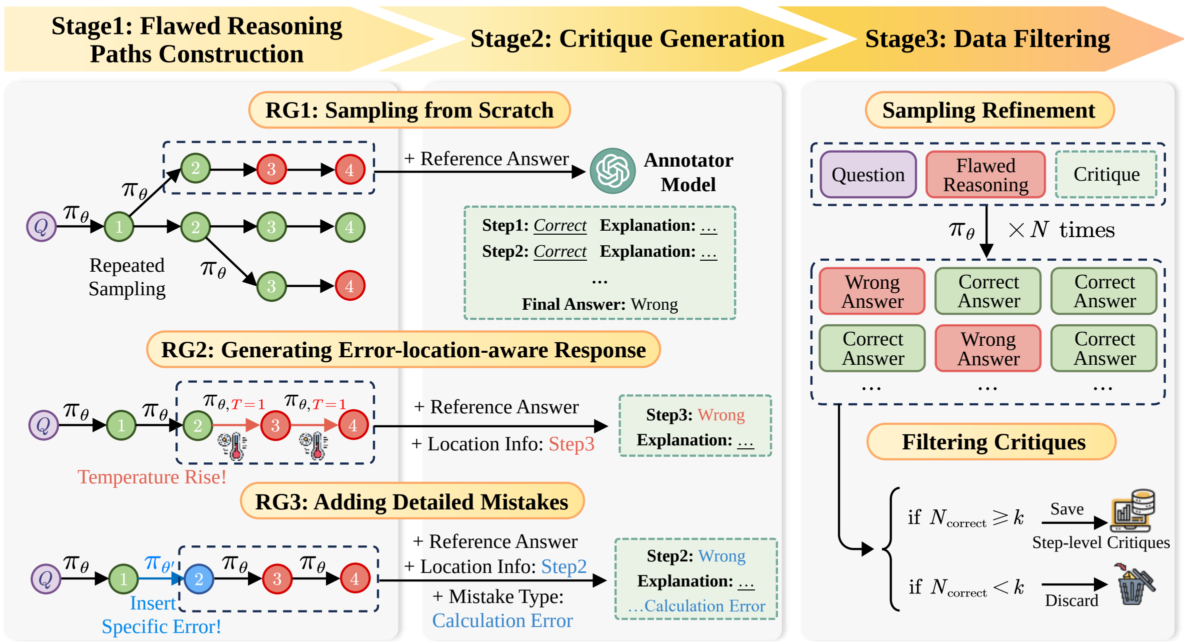


Figure 27: The overview of AutoMathCritique (Xi et al., 2024). The framework has three stages: (1) **Flawed Reasoning Path Construction**, generating erroneous reasoning paths via sampling or error insertion (RG1 ∼ RG3); (2) **Critique Generation**, producing step-by-step critiques using an annotator model; and (3) **Data Filtering**, refining critiques through sampling and retaining only high-quality ones. This ensures accurate step-level critiques for reasoning tasks. RG means Response Generation. The figure is borrowed from (Xi et al., 2024).

descriptions, such as legal clauses or scientific rules, into logic rules that machines can directly understand and verify. The framework combines an LLM with an automated reasoning engine, like an Answer Set Programming (ASP) solver, to form a "self-correcting team". The LLM generates logic code from natural language inputs while simultaneously creating test cases to validate the code's correctness. The reasoning engine then executes the code and tests, providing detailed feedback on any errors or semantic inconsistencies. Through an iterative refinement loop, the two components collaboratively enhance the logic program until it meets all test conditions. This approach enables LLM-ARC to tackle complex logical reasoning tasks, leveraging the reasoning capabilities of engines like ASP to ensure both semantic precision and reliability. On the FOLIO benchmark, LLM-ARC attains a state-of-the-art accuracy of 88.32%, surpassing prior methods by 10%.

6.3 Self-Correction

Errors in reasoning often accumulate progressively, where small mistakes can lead to significant deviations in final results. The Self-Correction mechanism (Huang et al., 2023; Madaan et al., 2024) enables large language models to engage in self-reflection and correction, similar to human critical thinking. Through iteratively reviewing, identifying errors, and refining reasoning steps, this approach improves the accuracy of final answers. This methodology is particularly effective for tasks requiring multi-step reasoning.

SCoRe short for **S**elf-**Co**rrection via **Re**inforcement Learning algorithm (Kumar et al.) is a multi-turn reinforcement learning framework that significantly enhances the intrinsic self-correction capabilities of LLMs using *self-generated data*. Unlike traditional methods that rely on supervised fine-tuning (SFT), prompt engineering, or external supervision, SCoRe addresses key challenges such as distribution shift and behavior collapse. As shown in Figure 28, SCoRe adopts a two-stage approach: the first stage mitigates behavior collapse by initializing training with constrained first-turn responses, while the second stage employs reward shaping to encourage progressive self-correction rather than solely optimizing final correctness. Demonstrating a 15.6% gain in self-correction on mathematical reasoning tasks (MATH) and a 9.1% improvement in coding tasks (HumanEval), SCoRe sets a new benchmark in training LLMs

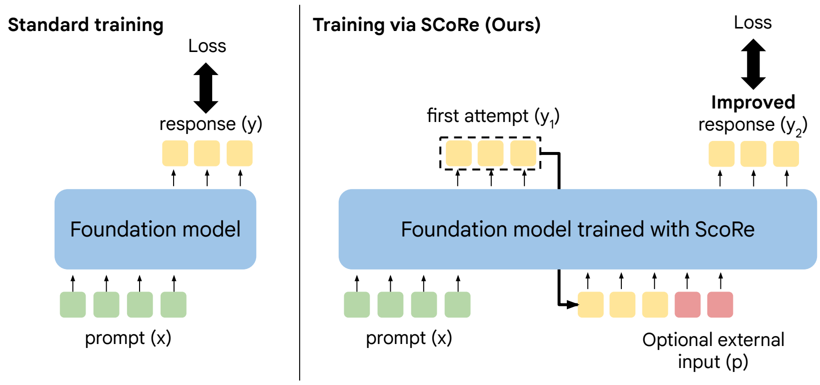


Figure 28: While standard training solely optimizes the final response, SCoRe (Kumar et al.) first trains the model to generate constrained initial responses (y_1) to mitigate behavior collapse. Then, refines responses (y_2) with optional feedback (p) using reward shaping to encourage progressive improvement. The figure is adapted from (Kumar et al.).

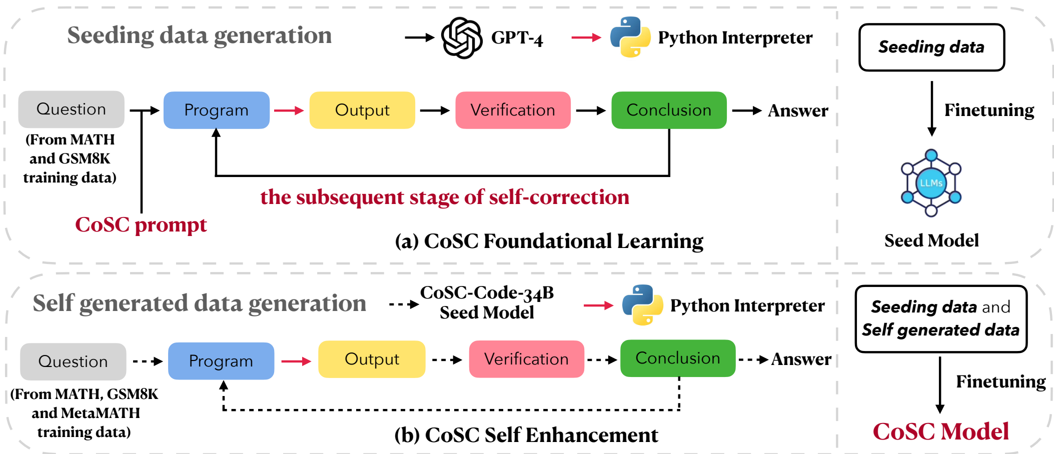


Figure 29: The two-phase training approach proposed in CoSC (Gao et al., 2024). In **Foundational Learning**, a seed model is fine-tuned using GPT-4-generated data from MATH and GSM8K, combining program generation, execution, and verification to establish baseline reasoning proficiency. In **Self-Enhancement**, the seed model generates its own reasoning trajectories from extended datasets (e.g., MetaMATH), enabling large-scale data generation without further GPT-4 intervention. Only correct trajectories are retained to fine-tune the final CoSC model. The figure is adapted from (Gao et al., 2024).

to autonomously refine their outputs without external guidance. Notably, SCoRe modifies the training strategy to enhance the inference ability.

CoSC short for **C**hain **o**f **S**elf-**C**orrection algorithm (Gao et al., 2024) embeds self-correction as an inherent capability in LLMs to enhance their mathematical reasoning abilities. CoSC operates through iterative stages of self-correction, including generating programs, executing code, and verifying outputs to refine reasoning steps progressively. As shown in Figure 29, CoSC employs a two-phase fine-tuning approach during the training stage: foundational learning using seed data generated by GPT-4 and self-enhancement through self-generated data. By modeling mathematical reasoning as a multi-round process, CoSC instills the ability to identify and rectify errors iteratively. During the inference stage, CoSC leverages this training to enable zero-shot self-correction, mimicking human-like "slow thinking" by systematically verifying and refining its outputs. Experiments on mathematical datasets (*e.g.*, MATH and GSM8K) demonstrate that CoSC significantly outperforms state-of-the-art open-source and some proprietary models like GPT-4. This framework establishes a new benchmark for mathematical reasoning tasks and highlights the potential for embedding self-correction as a foundational capability in LLMs.

Problem q	Find the radius of the circle with equation $x^2 + 8x + y^2 - 6y = 0$.
Decomposition1 $d_1 = d_1^1 \oplus d_1^2 \oplus d_1^3$	1. Complete the square for both x and y terms to rewrite the equation in the standard form of a circle's equation, $(x-h)^2+(y-k)^2=r^2$. 2. Extract the value of r^2 from the standard. 3. Compute the radius.
Code1 c_1	from sympy import symbols, Eq, solve, sqrt x, y = symbols('x y') circle_eq = x**2 + 8*x + y**2 - 6*y x_term = (x + 4)**2 – 4**2, y_term = (y - 3)**2 – 3**2 standard_form = x_term + y_term standard_form_eq = Eq(standard_form, 0) r_squared = simplify(-standard_form_eq.rhs) radius = simplify(r_squared**0.5) print(f"Standard form of the circle: (x + 4)^2 + (y - 3)^2 = {r_squared}.") print(f"Squared radius is {r_squared}.") print(f"Radius of the circle is {radius}.")
Result1 $r_1 = r_1^1 \oplus r_1^2 \oplus r_1^3$	Step 1: Standard form of the circle: (x + 4)^2 + (y - 3)^2 = 0. Step 2: Squared radius is 0. Step 3: Radius of the circle is 0.
Error Reason e_1	The error lies in the incorrect simplification of the equation after completing the square, leading to an incorrect radius calculation.
Decomposition2 d_2	1. Complete the square for both x and y terms to rewrite the equation in the standard form of a circle's equation, $(x-h)^2+(y-k)^2=r^2$ 2. Extract the value of r^2 from the standard form and then compute the radius. 3. Calculate the radius of the circle.

Code2 c_2	…
	# Move constants to the right side
	r_squared = 4**2 + 3**2
	nprint(f"Extract the value of r^2 = {r_squared}\")
	r= sqrt(r_squared)

Result2 r_2	Step 1: Complete the square for x and y terms, after completing the square: Eq((x + 4)**2 + (y - 3)**2 - 25, 0). Step 2: Extract the value of r^2 = 25. Step3: The radius of the circle is 5.
Answer a	The radius of the circle is directly obtained from the code as 5. Therefore, the final answer is \boxed{5}.

Figure 30: An example of DotaMath (Li et al., 2024) in action: it decomposes problems into subtasks, generates and executes Python code, and iteratively refines the decomposition and code if results are incorrect. This process repeats until a correct solution is achieved or iterations are exhausted. The figure is adapted from (Li et al., 2024).

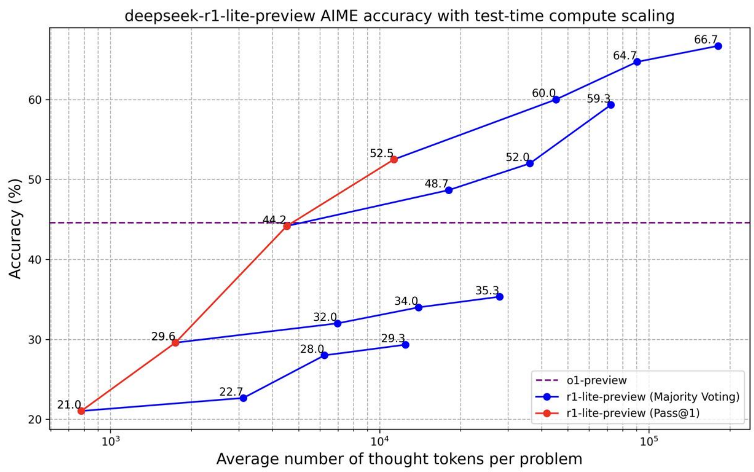


Figure 31: DeepSeek-R1-Lite-Preview (DeepSeek-R1-Lite-Preview, 2024) shows consistent score improvements on AIME as the length of reasoning increases. The figure is adapted from (DeepSeek-R1-Lite-Preview, 2024).

DotaMath (Li et al., 2024) is a novel framework for enhancing mathematical reasoning in LLMs by integrating decomposition of thought, code assistance, and self-correction. The self-correction process in DotaMath is a multi-round iterative procedure. As shown in Figure 30, for a given problem, DotaMath first decomposes it into sub-tasks and generates Python code to solve them. The code is executed using a Python interpreter, producing intermediate results. If the results are incorrect, the model identifies potential errors by verifying the alignment between the problem, code, and outputs. It then refines the task decomposition, adjusts the code, and re-executes the process. This cycle continues until the model either reaches a correct solution or exhausts a preset maximum number of iterations. To support this mechanism, the authors constructed DotaMathQA, a dataset with 574K samples, including single-turn

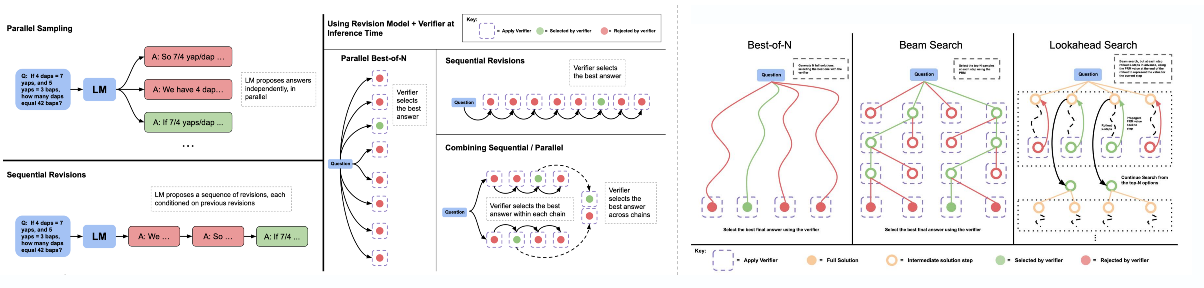


Figure 32: Different approaches used in (Snell et al., 2024) for scaling test-time compute. The figure is adapted from (Snell et al., 2024).

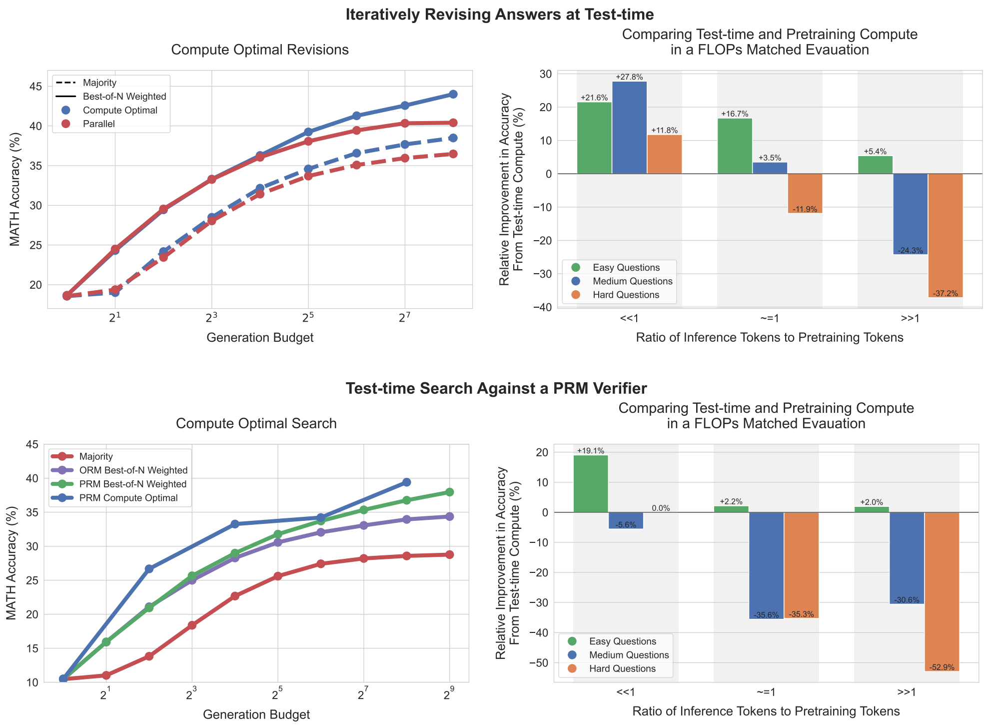


Figure 33: **Left**: A compute-optimal strategy improves test-time computational efficiency. **Right**: For easier questions, augmenting a smaller model with additional test-time computation outperforms using a much larger model. However, for harder questions, training a larger model remains more effective. The figure is adapted from (Snell et al., 2024).

QA (tasks solved in one iteration) and multi-turn QA (tasks requiring iterative self-correction). The dataset includes both automatically generated data and rule-based self-correction data, enabling the model to learn error detection and iterative refinement. Experiments demonstrate that DotaMath significantly outperforms state-of-the-art open-source models across in-domain (*e.g.*, MATH, GSM8K) and out-of-domain benchmarks. By embedding multi-turn self-correction as a core capability, DotaMath enables models to dynamically adapt to complex problem-solving scenarios, setting a new standard for addressing challenging mathematical tasks.

6.4 Inference Scaling Laws

Inference Scaling Laws examine the relationships between inference time, computational resource allocation, and reasoning performance. Research on inference-time compute scaling suggests that spending more computational resources during inference can significantly enhance model performance, a principle applied in o1 models. Very recently, DeepSeek models (DeepSeek-R1-Lite-Preview, 2024) also demonstrate the inference scaling law, as shown in Figure 31. This sub-section explores how understanding

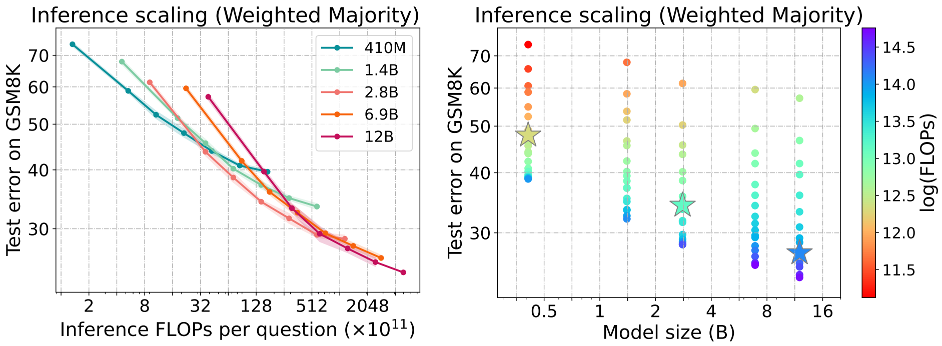


Figure 34: Inference Scaling Laws on GSM8K with Pythia (Biderman et al., 2023) models (Weighted Majority Voting). **Left**: Test error decreases as inference FLOPs increase, demonstrating performance improvements until convergence across various model sizes. **Right**: The optimal model size varies with the compute budget. Smaller models perform better at lower FLOPs (2^{41}, 2^{44}), while larger models excel at higher FLOPs (2^{47}). Both axes are log-scaled. The figure is adapted from (Wu et al., 2024).

these laws can guide the optimal configuration of computational resources, providing theoretical insights for maximizing reasoning capabilities while maintaining efficiency.

Scale-Compute (Snell et al., 2024) provides the first systematic analysis of different approaches (see Figure 32) for scaling test-time compute. Specifically, they investigate how LLMs can enhance their performance by optimizing test-time computation allocation, focusing on efficiency and adaptability. A "compute-optimal" strategy is proposed to dynamically allocate inference resources *based on task difficulty*. Two primary mechanisms are explored: (1) leveraging Process-based Reward Models (PRMs) for iterative search, where PRMs evaluate the correctness of intermediate steps in the reasoning process and guide optimization through tree-based search methods like beam search or lookahead search; and (2) adaptively modifying the output distribution by enabling models to iteratively refine their responses. PRMs are integral to this framework as they enable a more granular evaluation of reasoning steps, improving the ability to explore high-quality solution paths for complex tasks. As shown in Figure 33, experiments show that this compute-optimal strategy significantly outperforms best-of-N baselines, achieving comparable or better results with up to 4× less computation. Moreover, the study explores the trade-off between scaling pretraining versus test-time computation. It finds that additional test-time compute, enhanced by PRM-based verification, can often outperform pretraining on simpler tasks or low inference workloads, while harder tasks benefit more from pretraining larger models.

REBASE short for **RE**ward **BA**lanced **SE**arch is proposed by Wu et al. (2024), who focus on how model size, inference strategies, and compute budgets interact to optimize problem-solving performance in LLMs. They find that performance improves with increased inference compute across model sizes until saturation (Figure 34). Smaller models initially outperform larger ones under low compute budgets, but larger models excel once smaller models' performance saturates. Moreover, sampling and voting-based inference methods face limitations: 1) as accuracy saturates to a fixed upper bound determined by model probabilities, with diminishing returns despite exponential convergence. 2) Without an oracle verifier, simple sampling cannot achieve perfect accuracy. 3) Additionally, MCTS struggles with weighted voting, often producing incomplete solutions that reduce voting effectiveness. To address this, they propose an advanced inference algorithms Reward Balanced Search (REBASE). As shown in Figure 35, REBASE combines reward models with dynamic tree search to optimize intermediate node expansions, achieving Pareto-optimal performance across various compute budgets, outperforming widely-used strategies like MCTS and weighted majority voting.

Large Language Monkeys (Brown et al., 2024) explore the use of *repeated sampling* (see Figure 36) to scale inference compute, revealing its potential to significantly improve task performance by increasing coverage. As shown in Figure 37, across diverse tasks such as mathematical reasoning, coding challenges, and software debugging, the study demonstrates that coverage scales log-linearly with the number of

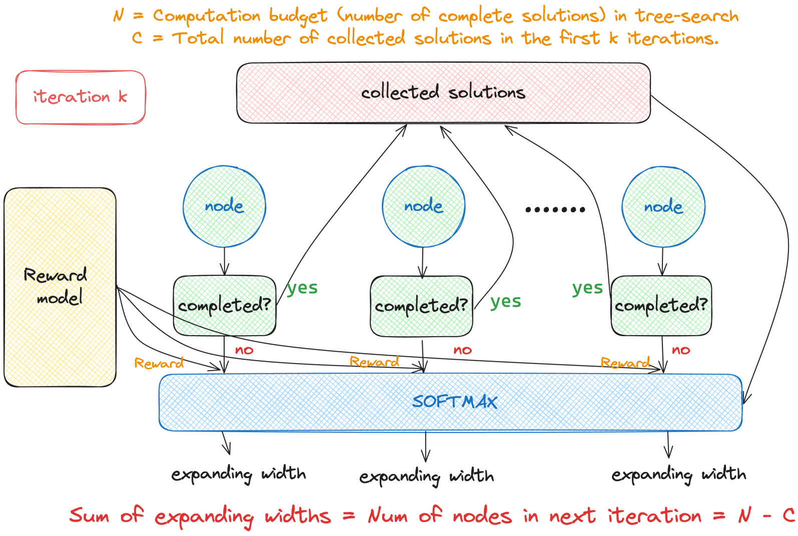


Figure 35: Illustration of Reward Balanced Search (REBASE) (Wu et al., 2024). REBASE integrates tree search's exploitation and pruning with a reward model to estimate the quality of intermediate nodes, eliminating the need for costly rollouts like in MCTS. Nodes are expanded based on softmax-normalized reward scores within a fixed budget, efficiently directing the search depth. The figure is adapted from (Wu et al., 2024).

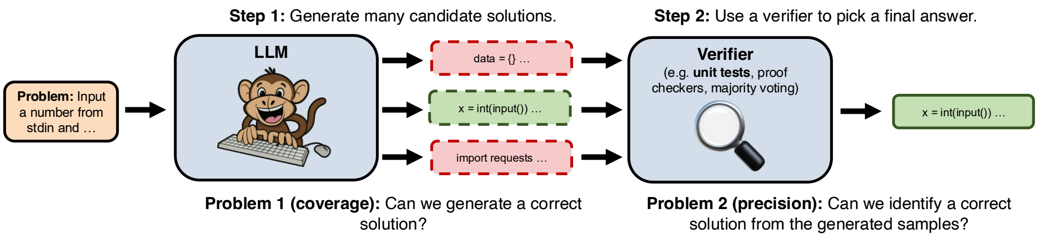


Figure 36: Illustration of a repeated sampling process (Brown et al., 2024): An LLM generates diverse candidate solutions, and a verifier, such as unit tests or majority voting, selects the correct one. This approach tackles challenges in generating and identifying accurate solutions. The figure is adapted from (Brown et al., 2024).

samples, adhering to an exponential power law that defines an inference scaling law. Repeated sampling proves particularly effective in amplifying weaker models, enabling them to outperform stronger models' single-attempt performance while optimizing cost. However, the study also highlights limitations in verification methods like majority voting and reward models, which fail to fully utilize the increased coverage beyond a certain sample threshold. This work deepens the understanding of inference scaling laws, focusing on sample count as a key axis for inference optimization, and provides practical guidelines for balancing computational costs and performance in real-world deployments.

STILL-2 (Min et al., 2024) is a framework for reproducing slow-thinking reasoning systems like OpenAI's o1, using a three-phase training strategy: *Imitate, Explore, and Self-Improve*. As shown in Figure 38, in the imitate phase, the model is fine-tuned with a small dataset of long-form thought demonstrations, collected from open o1-like systems, to generate both detailed reasoning steps and final solutions in a single response. In the explore phase, the model tackles challenging tasks by generating multiple candidate solutions (rollouts) and identifying correct trajectories that lead to the ground truth, thereby expanding its reasoning capacity. In the Self-Improve phase, the model uses these high-quality trajectories to iteratively refine its reasoning abilities, leveraging supervised fine-tuning (SFT) and direct preference optimization (DPO). The experiments, conducted on MATH-OAI, AIME, and GPQA benchmarks, show that STILL-2 approaches the performance of industry-level systems with just 3,900 demonstration examples or 1,100 examples plus exploration. The study emphasizes the significance of long-form thought data, especially for challenging tasks, and demonstrates that reasoning with slow-

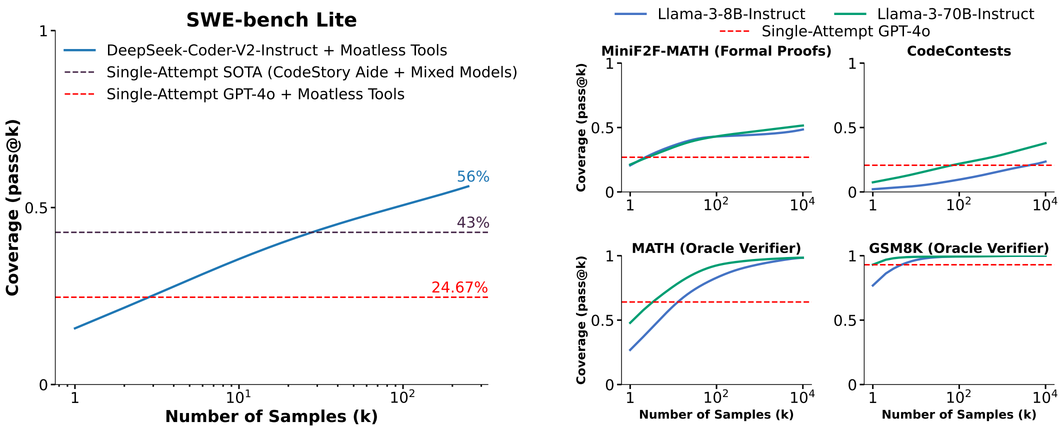


Figure 37: The coverage, defined as the fraction of problems solved by at least one generated sample, increases as the number of samples grows. The figure is adapted from (Brown et al., 2024).

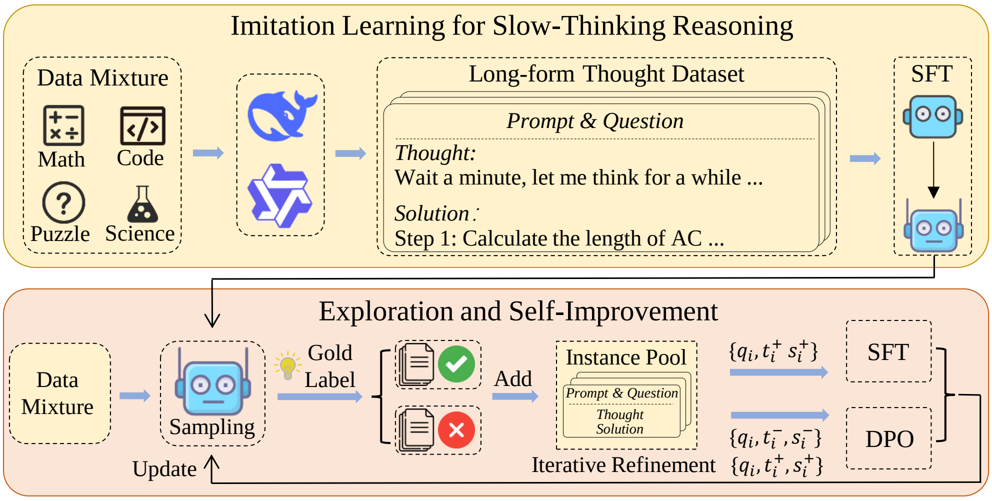


Figure 38: The training pipeline of STILL-2 consists of three phases. In the imitate phase, the model is fine-tuned on a small dataset of long-form reasoning demonstrations, enabling it to generate detailed steps and final solutions in a single response. During the explore phase, the model tackles complex tasks by generating multiple candidate solutions and identifying correct paths to the ground truth, enhancing its reasoning capabilities. Finally, in the self-improve phase, the model iteratively refines its reasoning using high-quality trajectories through supervised fine-tuning (SFT) and direct preference optimization (DPO). The figure is adapted from (Min et al., 2024).

thinking processes can generalize across domains. This aligns with inference scaling laws by showing how high-quality data and iterative training enhance model performance even with limited initial resources.

MindStar (M*) (Kang et al., 2024) is a framework that enhances the reasoning abilities of LLMs at inference time without fine-tuning. By treating reasoning tasks as search problems, M* employs a Process-supervised Reward Model (PRM) to evaluate reasoning steps and integrates tree-based search algorithms, such as Beam Search and Levin Tree Search (LevinTS), for structured exploration and backtracking. Aligning with inference scaling laws, M* shifts computational resources from training to inference, enabling smaller models like Llama-2-13B to achieve GPT-3.5-level performance with 200× lower computational costs. It improves reasoning performance by dynamically scaling the search space, refining intermediate steps, and prioritizing promising paths. Experiments on GSM8K and MATH datasets demonstrate that M* achieves significant accuracy gains by leveraging additional inference-time computation, providing a resource-efficient approach for enhancing complex reasoning capabilities.

7 Evaluation

The o1-like model demonstrates strong reasoning capabilities in many benchmarks. Currently, evaluations tend to focus on math or coding tasks, particularly those involving competitive-level problems due to their high difficulty, as well as some PhD-level questions in the fields of science and engineering. In the future, a set of evaluation metrics specifically designed to assess reasoning abilities may be introduced.

GPQA. The GPQA (Rein et al., 2023) dataset provides a challenging benchmark for evaluating reasoning abilities, particularly in scientific domains such as physics, chemistry, and biology. It consists of graduate-level multiple-choice questions carefully crafted by domain experts to test the limits of human and AI performance. What makes GPQA unique is its difficulty: even experts with PhDs or those pursuing advanced degrees in relevant fields achieve only 65% accuracy, which increases to 74% when accounting for errors identified retrospectively. Highly skilled non-experts, despite having unrestricted access to the internet, achieve a mere 34% accuracy. The dataset is also notably difficult for state-of-the-art AI systems like GPT-4, which achieves only 39% accuracy, significantly above random chance (25%). This makes GPQA an ideal testbed for evaluating large reasoning models. As the AI community continues to explore advanced reasoning capabilities, datasets like GPQA will be crucial in assessing whether AI models can handle tasks that are inherently difficult for both human experts and AI systems alike.

OlympiadBench. OlympiadBench (He et al., 2024a) offers a comprehensive and rigorous benchmark for evaluating reasoning abilities, particularly in mathematics and physics, through a bilingual multi-modal dataset. Comprising 8,476 challenging problems sourced from international Olympiads, Chinese Olympiads, and the Chinese College Entrance Exam (GaoKao), OlympiadBench pushes the boundaries of current AI models. Each problem is annotated with expert-level step-by-step reasoning, ensuring that the dataset captures the full depth of problem-solving processes. Additionally, OlympiadBench addresses a critical gap in existing benchmarks by incorporating multimodal reasoning, as many scientific tasks require not just textual analysis but also an understanding of visual or geometric information. With its rigorous design, OlympiadBench serves as an essential tool for assessing the true reasoning capabilities of state-of-the-art AI models, helping to guide future advancements in artificial general intelligence.

Minerva. Minerva (Lewkowycz et al., 2022) introduces a benchmark specifically focused on testing large language models in quantitative reasoning across various scientific domains, including mathematics, physics, chemistry, and biology. The dataset contains over 200 undergraduate-level problems drawn from MIT's OpenCourseWare (OCW) and other technical sources, providing a broad spectrum of challenges that require step-by-step reasoning and solution generation. Minerva pushes the boundaries of model performance by testing the ability to solve complex, real-world scientific problems without relying on external tools or solvers. The problems in Minerva involve not only natural language processing but also the integration of formal mathematical language, such as equations and diagrams, to model accurate problem-solving procedures. Minerva's diverse and robust set of problems offers a comprehensive platform for assessing how well AI systems can handle multi-step, quantitative reasoning tasks, providing a critical measure for the development of future AI assistants in scientific and engineering fields.

GSM8K. GSM8K (Cobbe et al., 2021a) is a benchmark designed to evaluate the ability of language models to perform multi-step mathematical reasoning at the grade school level. It consists of 8.5K high-quality, linguistically diverse math word problems that cover a wide range of topics. Despite the simplicity of the underlying math concepts, the dataset poses significant challenges due to its high linguistic diversity, requiring models to demonstrate strong reasoning abilities in both interpreting natural language and solving mathematical problems. GSM8K provides a valuable resource for advancing the development of models capable of tackling elementary yet challenging quantitative reasoning tasks, serving as a key tool for testing the reasoning and problem-solving abilities of AI systems.

MATH. The MATH dataset (Hendrycks et al., 2021) presents a challenging benchmark specifically designed to evaluate the mathematical problem-solving abilities of machine learning models. Comprising 12,500 competition-level math problems from high school math competitions, MATH covers a broad range of topics including algebra, geometry, combinatorics, and number theory. Each problem is accompanied by a full step-by-step solution, enabling models to learn both the correct final answer and the reasoning process behind it. The dataset is particularly valuable for testing models' abilities to perform multi-step reasoning and generate coherent explanations. MATH's complexity, even for human experts, combined with its large scale and focus on structured problem-solving, makes it an essential benchmark for pushing the boundaries of AI's reasoning capabilities, particularly in the realm of mathematics.

AIME. The American Invitational Mathematics Examination (AIME) (AI-MO, 2025) serves as a prestigious benchmark for evaluating mathematical reasoning abilities, particularly for high school-level problem-solving. It is originally a selective 15-question, 3-hour exam that is open to students who perform in the top 5% of the AMC 12 exam (or top 2.5% of the AMC 10). The problems tested in the AIME primarily focus on algebra, geometry, trigonometry, number theory, probability, and combinatorics, and often require advanced problem-solving techniques not typically covered in standard high school curricula. For large models, the AIME dataset serves as an important benchmark for evaluating their capabilities in multi-step mathematical reasoning.

Codeforces. Codeforces (Mirzayanov, 2025) is a platform hosts regular programming contests, known as "Codeforces Rounds," which challenge participants to solve algorithmic problems under time pressure. The problems typically span a variety of topics in computer science, including graph theory, dynamic programming, data structures, and number theory, requiring strong analytical and computational reasoning skills. The Codeforces rating system, similar to the Elo system, evaluates contestants based on their performance across these contests. With divisions for different skill levels (Div. 1, Div. 2, Div. 3, and Div. 4), Codeforces offers a wide range of problems suitable for evaluating AI systems at various levels of difficulty. This makes Codeforces an excellent resource for assessing the ability of large models to solve algorithmic and coding problems, particularly those requiring multi-step, logical reasoning and optimization strategies.

8 Analysis of Reasoning LLMs

This section analyzes the reasoning capabilities of large language models (LLMs) through a review of recent studies that investigate this topic from multiple perspectives. It examines factors such as token biases, reasoning step length, and the faithfulness of chain-of-thought (CoT) explanations, providing a comprehensive evaluation of how reasoning impacts model alignment, safety, and generalization.

8.1 Reasoning Enables Safer Language Models

Guan et al. (2024) introduce deliberative alignment, a novel training paradigm that leverages LLMs' reasoning capabilities to improve their safety. This approach trains models to explicitly recall and reason through safety specifications before generating responses. When applied to OpenAI's o-series models, deliberative alignment enables the use of CoT reasoning to analyze user prompts, reference relevant policy guidelines, and produce safer outputs. Experimental results demonstrate that o-series models aligned through deliberative alignment achieve precise compliance with OpenAI's safety policies without relying on human-authored chain-of-thoughts or answers. Additionally, deliberative alignment advances the Pareto frontier by strengthening resistance to jailbreak attempts, lowering overrefusal rates, and enhancing generalization to out-of-distribution contexts. These outcomes underscore that reasoning over clearly defined policies fosters more scalable, reliable, and transparent model alignment.

8.2 Run-Time Strategies for Enhancing o1

Nori et al. (2024) explore the efficacy of various classic run-time prompt strategies applied to OpenAI's o1, the reasoning-native model. Their study focuses on medical challenge problem benchmarks and incorporates MedPrompt (Nori et al., 2023), a structured, multi-step prompting framework designed to enhance the performance of LLMs on these benchmarks. Through systematic experiments, the authors assess the o1-preview model across multiple medical benchmarks, both with and without the use of MedPrompt. The findings are as follows. **First**, o1-preview significantly outperforms the GPT-4 series with MedPrompt, even in the absence of advanced prompting techniques. **Second**, few-shot prompting adversely impacts o1's performance, suggesting that in-context learning may no longer be a viable approach for reasoning-native models. **Third**, ensembling remains a feasible strategy but imposes high resource demands, underscoring the need for careful cost-performance optimization. **Finally**, their analysis identifies a Pareto frontier for run-time strategies: GPT-4o offers a cost-effective solution, while o1-preview achieves state-of-the-art performance at a higher cost. Despite o1-preview's superior performance, GPT-4o retains its value in specific scenarios, especially when combined with strategies like MedPrompt. These findings offer critical guidance for optimizing run-time strategies for reasoning models in medical applications, emphasizing the trade-offs between performance, cost, and resource utilization.

8.3 CoT's Primary Benefits in Mathematics and Symbolic Reasoning

Sprague et al. (2024) investigate the types of tasks where prompt-based CoT reasoning is most effective. Through a quantitative meta-analysis of existing literature and their own experiments across various models, datasets, and prompts, they find that CoT yields significant performance benefits primarily in tasks involving math or logic, with only marginal improvements in other task types (see Figure 39). To further understand CoT's behavior, the authors decouple planning from execution and compare its performance with that of tool-augmented LLMs. Their findings suggest that much of CoT's advantage arises from enhancing symbolic execution, yet it still falls short when compared to dedicated symbolic solvers. These results highlight CoT as a powerful technique, but they also suggest that broader improvements in NLP tasks will require moving beyond prompt-based CoT. Future research directions may include exploring paradigms such as search-based approaches, interacting agent systems, or more effectively fine-tuned models.

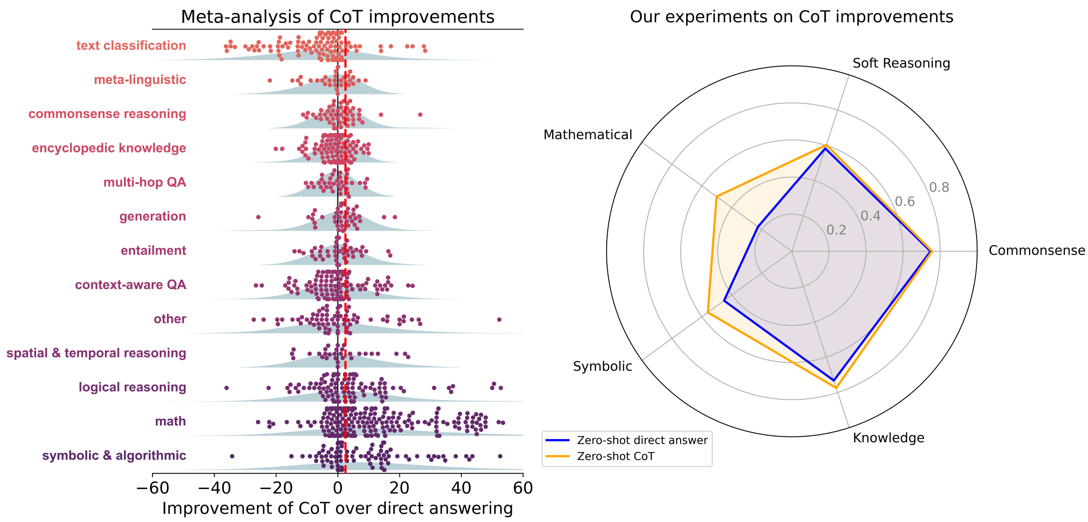


Figure 39: Main results of Sprague et al. (2024). Left: A meta-analysis of Chain-of-Thought (CoT) literature shows the reported performance improvement (delta) of CoT over direct answering for various (LLM, task) pairs. Right: Average performance comparison between zero-shot CoT and direct answer prompts across five reasoning categories, covering 20 datasets and 14 LLMs. Results indicate that math and symbolic reasoning consistently benefit the most from CoT, with the red dotted line representing the average improvement across experiments. The figure is adapted from Sprague et al. (2024).

8.4 Unveiling Token Bias: The Limits of Reasoning in LLMs

Jiang et al. (2024) introduce the concept of token bias: an LLM exhibits token bias in a reasoning task if changes to some or all tokens in the task description (while maintaining the underlying logic) predictably alter the model's output (see Figure 40). To determine whether LLMs are capable of genuine reasoning or if their performance is primarily driven by token biases, the authors propose a hypothesis-testing framework. This framework outlines a set of hypotheses where token biases are readily identifiable, with all null hypotheses assuming the genuine reasoning capabilities of LLMs. The authors conduct experiments with a variety of commercial and open-sourced LLMs, and their statistical analysis suggests that, while LLMs may perform well on classic problems, their success is largely driven by recognizing superficial patterns influenced by strong token bias. This raises concerns about their true reasoning and generalization capabilities. These findings suggest that chain-of-thought prompting and in-context learning may not invoke genuine reasoning in LLMs. Instead, they may lead to semantic shortcuts that superficially mimic the desired behavior. This highlights the need for further investigation into the underlying mechanisms and limitations of LLMs, particularly with respect to their reasoning abilities.

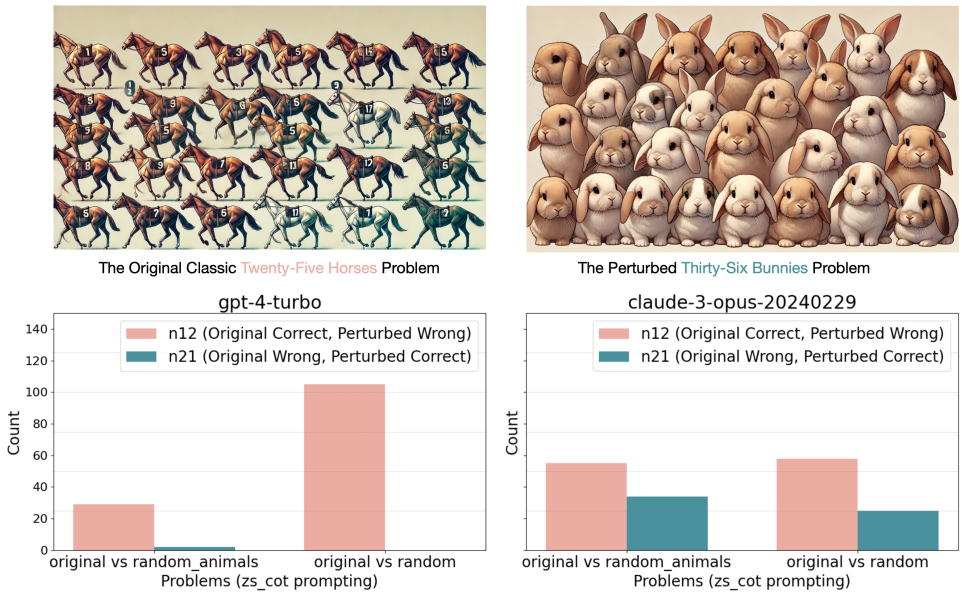


Figure 40: Examples of the concept of token bias proposed in Jiang et al. (2024). The concept of token bias is illustrated using the classic “twenty-five horses” problem in graph theory. The top sub-figures, generated by GPT-4o for illustrative purposes, modify the term “horses” to “bunnies” to demonstrate that such changes do not impact the problem’s underlying logic. The bottom sub-figures present experimental results from GPT-4 and Claude, revealing a significant drop in performance when animal names or numbers are altered. “Original” refers to the unmodified problem, while “random_animals” alters only the animal names, and “random” modifies both names and numbers. Statistical analysis shows that $n12 > n21$, meaning there are more cases where the original problem is solved correctly while the perturbed version is solved incorrectly, compared to the reverse. These findings confirm the presence of token bias in this scenario. The figure is adapted from Jiang et al. (2024).

8.5 Latent Multi-Hop Reasoning in LLMs

Yang et al. (2024b) study the latent multi-hop reasoning abilities of LLMs using complex prompts such as "The mother of the singer of ‘Superstition’ is." Latent multi-hop reasoning refers to an LLM’s ability to implicitly connect multiple pieces of information across reasoning steps. For example, the model might (1) internally identify a bridge entity (e.g., recognizing "the singer of ‘Superstition’" as Stevie Wonder) and (2) leverage related knowledge (e.g., about Stevie Wonder’s mother) to complete the task, without explicitly breaking down these intermediate steps. The authors analyze latent multi-hop reasoning by testing two hops: (1) whether modifying the prompt to indirectly mention the bridge entity improves the LLM’s recall of it, and (2) whether increased recall enhances the LLM’s ability to utilize knowledge about the bridge entity. Their experiments with LLaMA-2 (Touvron et al., 2023) 7B, 13B, and 70B models reveal compelling evidence of latent multi-hop reasoning for certain relation types, with the reasoning pathway utilized in more than 80% of relevant prompts. However, the effectiveness varies by context. Substantial evidence is found for the first hop, but the second hop and full multi-hop traversal show moderate results. Additionally, larger models demonstrate improved performance on the first hop, but not the second. These findings emphasize the need for further research to enhance latent multi-hop reasoning, which is crucial for improving parameter efficiency, generalization, and controllability in LLMs.

8.6 The Impact of Reasoning Step Length

Jin et al. (2024) explore the correlation between the effectiveness of CoT and the length of reasoning steps in prompts. The authors design several empirical experiments that manipulate the length of rationale reasoning steps within CoT demonstrations, either expanding or compressing them, while keeping all other factors constant (see Figure 41). They experiment with models such as GPT-3, GPT-3.5, and GPT-4, and present the following key insights. **First**, lengthening reasoning steps in CoT prompts, even without adding new information, improves LLM reasoning, while shortening them leads to reduced performance. **Second**, even incorrect rationales can enhance outcomes if they maintain a sufficient inference length. This suggests that the length of the reasoning chain is more crucial than its factual accuracy for effective problem-solving. **Third**, the benefits of longer reasoning steps depend on task complexity, with more complex tasks benefiting more from extended inference. These results offer actionable guidance for optimizing CoT strategies, underscoring the importance of reasoning step length in tackling intricate NLP challenges.

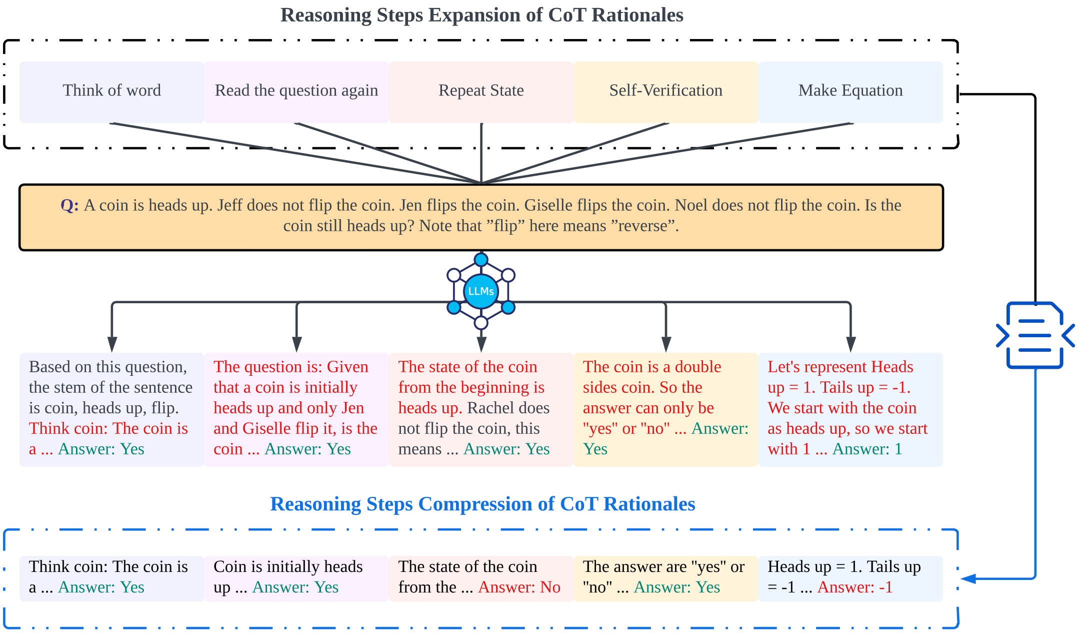


Figure 41: Different methods used to extend or compress the reasoning length in Jin et al. (2024). Use the approach shown in the figure to extend the reasoning chain, and compress it to the greatest extent possible while retaining all information. The figure is adapted from Jin et al. (2024).

8.7 Faithfulness of CoT Reasoning

Lanham et al. (2023) examine whether the reasoning presented in CoT explanations accurately reflects the actual reasoning processes of LLMs. **First**, they evaluate post-hoc reasoning, where reasoning is generated after the conclusion has already been determined, by truncating or introducing errors into the CoT before the final answer. Their findings reveal significant variation in LLMs' reliance on CoT across tasks: some tasks exhibit no dependence on CoT, while others rely on it heavily. Interestingly, post-hoc reasoning tends to worsen with more capable models, indicating that smaller models may be more reliable for tasks requiring faithful reasoning. **Second**, they investigate whether CoT's performance gains stem from increased test-time computation. By replacing CoT with uninformative filler text, they find no accuracy improvements, suggesting that test-time computation alone does not account for CoT's effectiveness. **Third**, they explore whether CoT encodes task-relevant information in ways inaccessible to human interpretation. By substituting CoT with paraphrased versions, they observe no performance degradation, indicating that the specific phrasing of CoT is not crucial to its success. In summary, these findings highlight key challenges in CoT faithfulness and offer valuable methodologies for improving CoT explanations, contributing to the development of systems with more transparent and reliable reasoning processes.

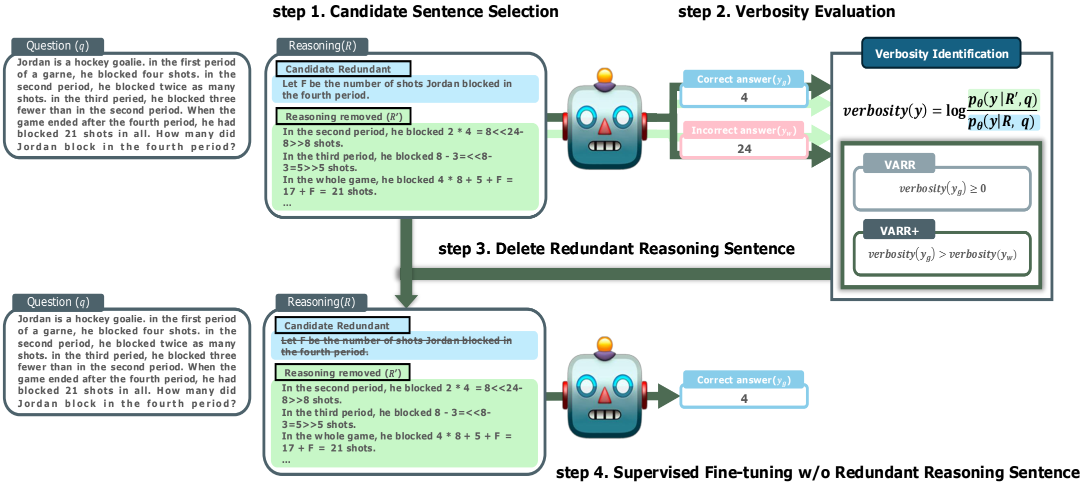


Figure 42: Overview of the VARR/VARR+ framework proposed in Jang et al. (2024). Initially, a candidate sentence is selected from the beginning of the rationale. After removing the candidate sentence, verbosity is evaluated for both $verbosity(y_g)$ and $verbosity(y_w)$. If the candidate sentence meets the verbosity evaluation criteria, it is excluded from subsequent training steps. The model then continues training with the redundant sentence removed from the rationale. The figure is adapted from Jang et al. (2024).

8.8 Controlling Reasoning Length in LLMs

Recent studies have highlighted the need for controlling reasoning length in LLMs, as issues like overthinking, redundant computations, and excessive token usage contribute to inefficient resource allocation and increased costs.

Chen et al. (2024d) tackle the problem of overthinking in reasoning-focused LLMs, where excessive computational resources are allocated to simple tasks without proportional benefits. To address this, the authors introduce efficiency metrics that assess resource usage from both outcome and process perspectives. By leveraging a self-training framework (Zelikman et al., 2022; Ho et al., 2022), where the model generates its own training data, they propose strategies such as length preference optimization and response simplification to optimize reasoning processes. These strategies successfully balance computational demands with performance, demonstrating significant reductions in computational overhead while maintaining accuracy across various benchmarks, including GSM8K, MATH500, GPQA, and AIME. This work offers promising solutions for intelligent resource scaling in reasoning tasks.

Jang et al. (2024) focus on reducing inference costs in LLM reasoning by eliminating redundant intermediate reasoning sentences. The authors propose VARR, a sentence-level rationale reduction framework based on a principled verbosity criterion (see Figure 42). VARR utilizes a likelihood-based approach to identify and remove unnecessary reasoning sentences during training, retaining only essential steps. This framework maintains reasoning integrity while minimizing the likelihood of generating incorrect answers, effectively balancing efficiency and accuracy. In contrast to token-level methods, VARR significantly reduces generation costs while preserving reasoning quality. Experimental results show a substantial average cost reduction of 17.15% without sacrificing performance, highlighting the importance of sentence-level rationale reduction for efficient reasoning.

Han et al. (2024) address inefficiencies in CoT reasoning caused by excessive token usage, which increases computational costs. They propose TALE, a token-budget-aware reasoning framework that dynamically estimates token budgets based on task complexity. This approach guides the reasoning process, effectively compressing unnecessary reasoning length while preserving performance. Experimental results demonstrate that TALE reduces token redundancy by an average of 68.9%, with only a 5% reduction in accuracy, offering a better balance between efficiency and performance across a range of LLMs.

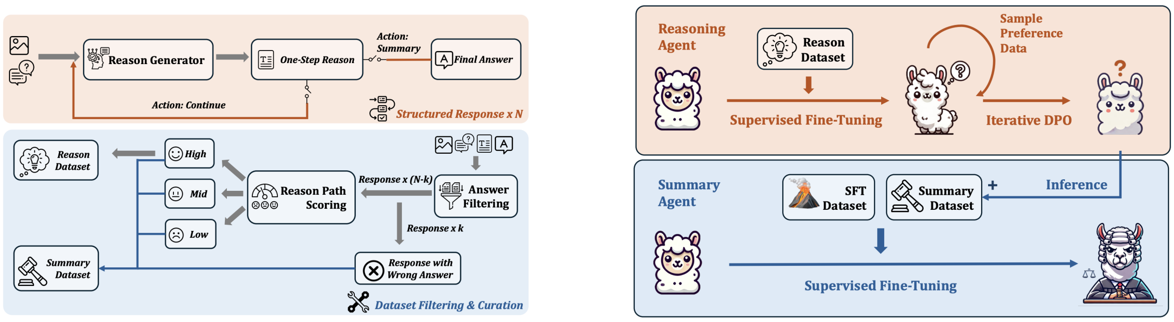
9 Multi-modal Reasoning LLMs

Model or Framework	Base Model	Input Modality	Pretraining Data Scale	Fine-tuning Data Scale	Open-source
Insight-V (Dong et al., 2024)	Qwen-2.5-7B	Text/Image	558K	4M images	✓[1]
LLaVA-CoT-11B (Xu et al., 2024)	Llama-3.2-11B-Vision-Instruct	Text/Image	-	99K	✓[2]
Sketchpad (Hu et al., 2024)	GPT-4o	Text/Image	-	-	✓[3]
ChartPaLI-5B (Carbune et al., 2024)	PaLI-3	Text/Image(chart)	2.37M	544.9K	✗
SpatialVLM (Chen et al., 2024a)	PaLM 2-E	Text/Image(3d)	-	-	✓[4]
Chain-of-Table (Wang et al., 2024f)	PaLM 2-S, Llama-2-17B-chat	Text(table)	-	-	✓[5]
QVQ-72B-Preview (Team, 2024a)	Qwen2-VL-72B	Text/Image	-	-	✓[6]

[1] https://github.com/dongyh20/Insight-V
[2] https://github.com/PKU-YuanGroup/LLaVA-CoT
[3] https://github.com/Yushi-Hu/VisualSketchpad
[4] https://github.com/remyxai/VQASynth
[5] https://github.com/google-research/chain-of-table
[6] https://huggingface.co/Qwen/QVQ-72B-Preview

Table 7: An overview of emerging LLMs designed for multi-modal reasoning.

While current O1 models do not yet support multi-modal functions, future iterations hold significant potential for integrating them. Therefore, in this section, we explore the emerging class of LLMs designed for multi-modal reasoning, which can process and integrate information from various data modalities, such as text, images, and audio. These models extend the capabilities of traditional LLMs by bridging the gap between language understanding and sensory perception, enabling more advanced reasoning across different forms of data. By incorporating the ability to process sensory inputs like images and audio alongside text, these models could greatly enhance holistic reasoning, offering users a richer and more comprehensive experience. An overview of these models is shown in Table 7.



(a) The data generation pipeline of Insight-V (Dong et al., 2024). The reasoning data is created progressively using a reasoning generator and is subsequently evaluated by a multi-level assessment system to guarantee high-quality reasoning data. The figure is borrowed from Dong et al. (2024).

(b) The multi-agent system of Insight-V (Dong et al., 2024). The multi-agent system is based on the same underlying model. Through task decomposition and personalized training, the reasoning process becomes better and the answers become more rational. The figure is borrowed from Dong et al. (2024).

Figure 43: The data generation pipeline and multi-agent system of Insight-V (Dong et al., 2024).

9.1 Insight-V

Insight-V (Dong et al., 2024) is a framework designed to enhance the multi-step visual reasoning capabilities of MLLMs by constructing reliable multi-step reasoning data and developing a refined training process. The authors propose the following techniques: (1) a flexible strategy for generating multi-step reasoning data for complex multi-modal tasks, (2) a multi-agent system that divides task handling processes into reasoning and summarization parts to enhance response quality, and (3) a two-stage training process to better cultivate agents' abilities. As shown in Figure 43a, the data generation pipeline utilizes a progressive method to create formatted multi-step reasoning data with various reasoning paths and a multi-level assessment system to evaluate the quality of the generated reasoning data and divide them into different datasets. As illustrated in Figure 43b, the multi-agent system employs a reasoning agent to generate detailed reasoning steps and a summarization agent to extract core logic and generate concise response. The two-stage training pipeline involves SFT of a base MLLM to develop the reasoning and summarization agents, followed by iterative DPO to align the reasoning agent with human preferences. The proposed techniques lead to significantly improved performance on complex multi-modal visual reasoning benchmarks and effortlessly retains or elevates its performance on multi-modal perception tasks.

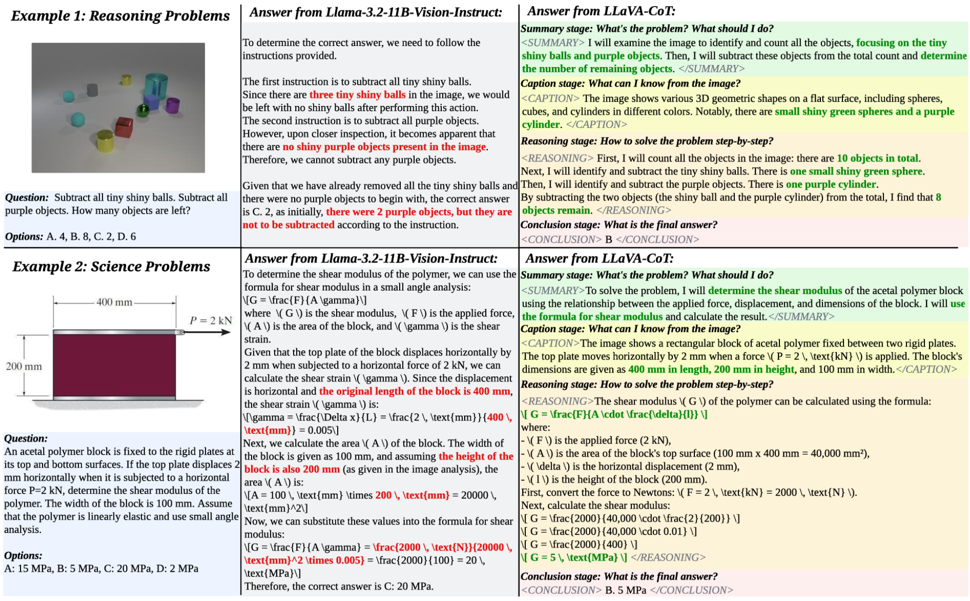


Figure 44: The optimized inference process of LLaVA-CoT (Xu et al., 2024). The base model Llama-3.2-11B-Vision-Instruct produces multiple errors during inference. In contrast, LLaVA-CoT ensures a strong logical chain through a clearly defined, progressive inference process. The figure and caption are borrowed from Xu et al. (2024).

9.2 LLaVA-CoT-11B

LLaVA-CoT-11B (Xu et al., 2024) is a visual language model (VLM) designed for rigorous and accurate visual reasoning. This work addresses the reasoning challenges faced by current VLMs, particularly when dealing with complex visual question-answering tasks. The authors enhance progressive reasoning in the model through SFT, enabling the generation of four distinct reasoning stages: summarization, interpretation, reasoning, and conclusion. This enables VLMs to significantly improve precision in tasks requiring rigorous reasoning. As shown in Figure 44, each stage is tagged (*e.g.*, <SUMMARY>) to ensure clarity and structure, promoting organized thinking as opposed to traditional freeform CoT reasoning. The authors construct the LLaVA-CoT-100k dataset, which includes visual question-answering data from various sources, with structured inference tags. Furthermore, they introduce inference-time stage-level beam search, as shown in Figure 45. This method optimizes reasoning quality and scalability by selecting the best candidate at each stage, facilitating efficient time scaling during inference. It is noteworthy that with only 100k training samples and a straightforward but powerful method for inference scaling, LLaVA-CoT-11B achieves a 7.4% performance improvement over its base model across a range of multi-modal reasoning benchmarks. It also outperforms larger models, including closed-source ones like Gemini-1.5-pro (Team et al., 2024a), and Llama-3.2-90B-Vision-Instruct (Grattafiori et al., 2024).

9.3 Sketchpad

Sketchpad (Hu et al., 2024) is a framework designed to enhance the multi-step multi-modality reasoning process by inserting image processing behaviors in the inference phase. The authors introduce this technique to remedy the shortcomings of current CoT and tool-use paradigms, which rely solely on text during intermediate reasoning stages. Unlike prior works where language models (LMs) generate

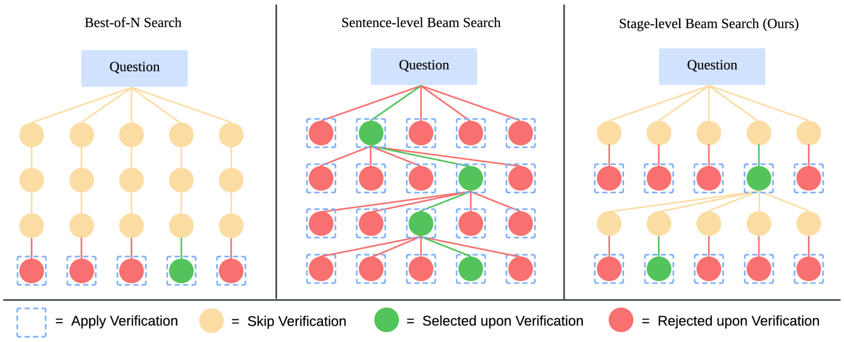


Figure 45: Stage-level beam search method in LLaVA-CoT (Xu et al., 2024). The Best-of-N search is applied at the response level, while the Sentence-level Beam Search operates at the sentence level. The newly proposed Stage-level Beam Search is executed at each reasoning phase, such as summary and reasoning, choosing a more rational granularity and achieving the best performance. The figure is borrowed from Xu et al. (2024).

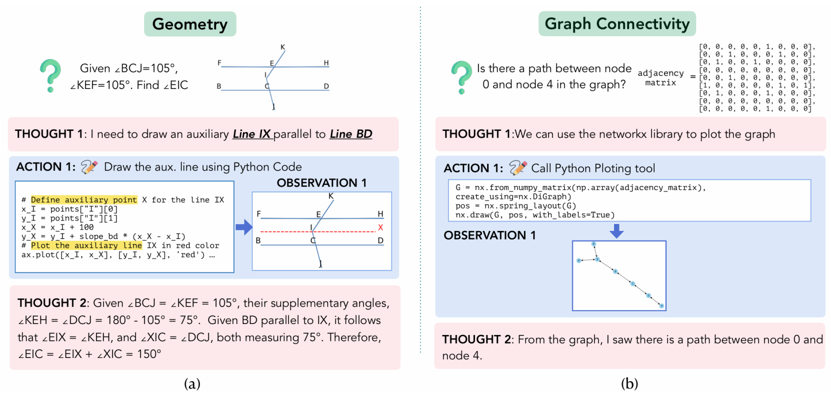


Figure 46: Example of applying Sketchpad (Hu et al., 2024). The agent generates a sketching plan based on a multi-modal query (*Thought*), and then activates a program to generate intermediate sketches (*Action*). Through the analysis of the sketches (*Observation*), the model crafts a comprehensive and final response to the query. The figure is borrowed from Hu et al. (2024).

images via text-to-image models, the authors equip LMs with the ability to draw lines, boxes, marks, etc., mimicking human sketching and thus improving the reasoning process. An example of applying Sketchpad is shown in Figure 46. Additionally, to improve visual perception and reasoning, Sketchpad leverages specialized vision models to optimize its sketching process (*e.g.*, using object detection models to draw bounding boxes and segmentation models to create masks). Evaluation experiments for this work were conducted on several kinds of benchmark datasets, covering topics such as geometry, functions, graphs, chess, and challenging visual reasoning tasks. Compared to powerful baseline models without applying proposed technique, Sketchpad significantly boosts performance across all tasks. Specifically, it improves average performance on math tasks by 12.7% and visual tasks by 8.6%. Using the proposed technique, GPT-4o achieves the best performance across all benchmarks, such as V*Bench (Wu and Xie, 2023) with a score of 80.3%, and visual correspondence at 80.8%.

9.4 ChartPaLI-5B

ChartPaLI-5B (Carbune et al., 2024) is a MLLM based on PaLI3-5B (Chen et al., 2023b) designed to improve the chart-related reasoning abilities of VLMs. To narrow the reasoning ability gap between smaller VLMs and LLMs, the authors propose a method to transfer knowledge from LLMs. First, they adopt the improved chart-to-table conversion (Liu et al., 2023) and use this refined chart representation to undergo pre-training. Then, they construct a dataset that is 20 times larger than the original training set. Following that, the authors design reasoning steps with table representations of charts to strengthen both reasoning and numerical capabilities. Finally, they fine-tune the model using a multitask loss (Hsieh et al., 2023) on the constructed datasets. These datasets contains reasoning steps generated by more powerful

LLMs, enabling the transfer of reasoning abilities. ChartPaLI-5B achieves state-of-the-art performance on ChartQA and significantly improves performance on PlotQA and FigureQA. Moreover, even without an upstream OCR system, ChartPaLI-5B surpasses much larger models like PaLIX-55B while maintaining similar inference times as its base model PaLI3-5B. Additionally, by adopting a straightforward program-of-thought prompt (Chen et al., 2023a) to refine the logic chain, ChartPaLI-5B even outperforms the recently released Gemini Ultra and GPT-4V.

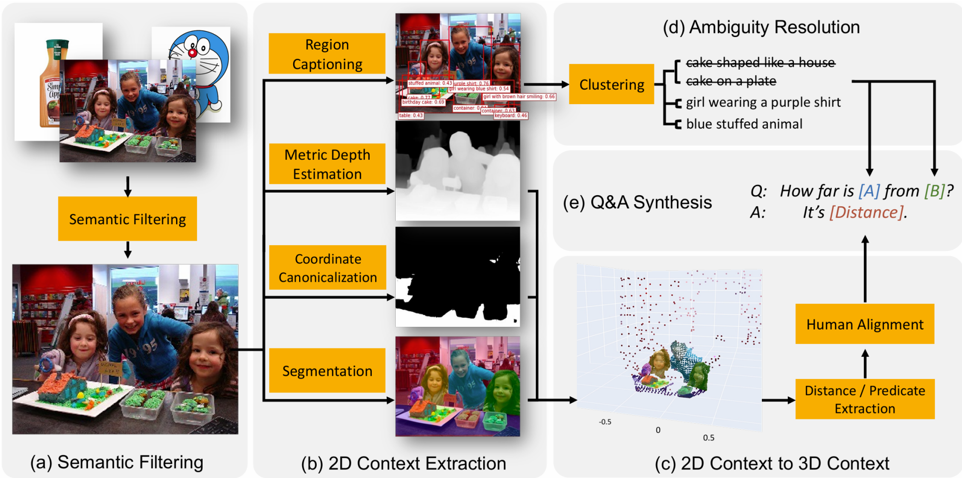


Figure 47: The data synthesis pipeline in SpatialVLM (Chen et al., 2024a). The authors leverage CLIP (Radford et al., 2021) to eliminate irrelevant images, keeping only scene-level photos. Pre-trained expert models are applied to large-scale internet images for object-centric segmentation, depth estimation, and caption generation. They convert 2D images into 3D point clouds, enabling the extraction of properties like 3D bounding boxes through shape analysis. To reduce ambiguity, object captions are clustered based on CLIP similarity scores. Finally, millions of spatial queries and their corresponding answers are generated by combining object descriptions with extracted spatial attributes. The figure is borrowed from Chen et al. (2024a).

9.5 SpatialVLM

SpatialVLM (Chen et al., 2024a) is a framework designed to enhance the spatial understanding and reasoning capabilities of VLMs by leveraging out-of-the-box vision models to generate spatial annotations on the training data. This work tackles the difficulties encountered by VLMs in spatial comprehension and reasoning, particularly in tasks involving the interpretation of numerical relationships between physical entities, such as variations in size and spatial distance. The authors suggest that this limitation arises from the lack of annotation of spatial information in the training data. The proposed solution is to enhance VLMs by training them on a large-scale spatial reasoning dataset. First, they develop an automated framework for generating visual question answering (VQA) data with rich spatial information annotations. As shown in Figure 47, by integrating techniques such as region captioning and segmentation, this framework annotates real-world data at scale and formats it for training VLMs on diverse tasks. With this framework and 10 million real-world images, they finally gain 2 billion VQA examples. Next, they explore several key factors in the training process, such as model architecture and data quality, trying to develop an optimized training mechanism. The natural language interface of a powerful VLM using SpatialVLM can support complex spatial reasoning by facilitating a CoT process, making it efficient for tackling sophisticated spatial problems. It also enables the model to serve as an open-vocabulary reward annotator for tasks involving rearrangement. Training a VLM on the dataset created using the proposed techniques improves the model's qualitative and quantitative spatial understanding and reasoning capabilities, enabling it to achieve significant performance improvements on related tasks. VLMs applying this technique can further carry out more complex spatial perception applications , thanks to their abilities to make quantitative estimations.

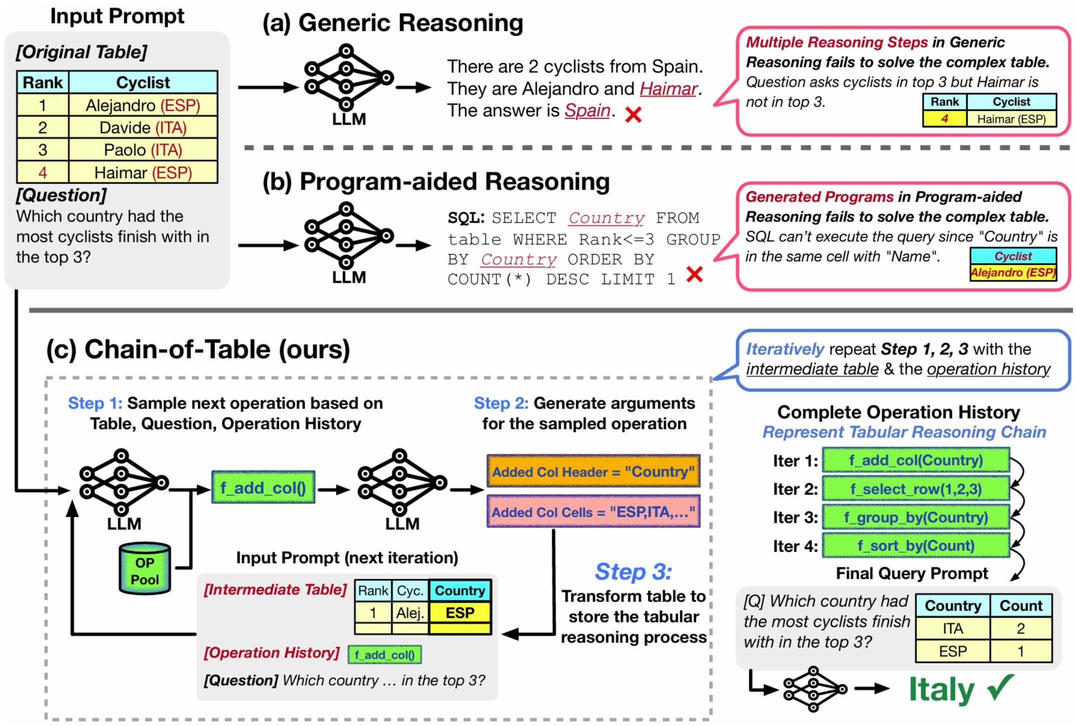


Figure 48: Overview of Chain-of-Table (Wang et al., 2024f). When faced with a complex table where a cyclist's name and nationality share a single cell, (a) fails to provide the correct answer due to complexity, while (b) uses programmatic approaches like SQL queries but struggles to accurately parse the information. In contrast, (c) employs a step-by-step process to transform the table into a simplified, question-specific format, enabling the LLM to deduce the correct answer effectively. The figure is borrowed from Wang et al. (2024f).

9.6 Chain-of-Table

Chain-of-Table (Wang et al., 2024f) is a framework designed to improve the reasoning abilities of LLMs when working with table-based data. While CoT and similar methods integrate reasoning processes as textual context, effectively incorporating tabular data into this reasoning chain remains a challenge. Table-based reasoning involves extracting semantics from unstructured questions and partially structured tabular information, which differs from conventional reasoning tasks. The authors propose a method that directly utilizes tabular data in the intermediate steps of the reasoning chain, carrying out progressive reasoning through tabular operations, thereby forming a chain of intermediate tables. Figure 48 illustrates the difference between Chain-of-Table and traditional reasoning methods. The authors employ in-context learning to teach the model to use table operations (*e.g.*, adding columns, filtering rows, or grouping) step by step to refine or simplify the table. This enables LLMs to dynamically plan each subsequent action based on the intermediary tables in the operation history. Such a process better utilizes the semantics of the table that is continuously optimized during reasoning. Chain-of-Table sets a new benchmark in performance on the WikiTQ (Pasupat and Liang, 2015), FeTaQA (Nan et al., 2022), and TabFact (Chen et al., 2020) datasets.

9.7 QVQ-72B-Preview

QVQ-72B-Preview (Team, 2024a) is a MLLM built upon Qwen2-VL-72B (Wang et al., 2024c), designed to enhance visual reasoning capabilities through step-by-step reasoning. It aims to improve LLMs' cognitive abilities by incorporating visual understanding. However, few technical details are currently available. The team mainly presents evaluation results and discusses the model's limitations. QVQ-72B-Preview has achieved impressive results across several benchmarks, including an outstanding 70.3% on the MMMU benchmark, demonstrating QVQ's strong ability in multi-domain reasoning and comprehension. The model's substantial improvements on MathVision (Wang et al., 2024a) highlight its advancements in mathematical problem-solving. OlympiadBench (He et al., 2024b) further showcases its enhanced capability to address complex challenges. Despite these achievements, the model has several limitations. For instance, it may mix languages or enter recursive reasoning loops, affecting response clarity and conciseness. Although it has made advancements in visual reasoning, it struggles with multi-step reasoning, occasionally hallucinating or losing focus, and does not outperform Qwen2-VL-72B in basic recognition tasks. Additionally, the model is limited to single-round dialogues and image outputs, with no support for video inputs.

10 Conclusion

In this survey, we presented a view of reasoning LLMs by focusing on dataset construction, supervised fine-tuning, reinforcement learning, and advanced inference strategies (chain-of-thought and automated critiques) through the lens of OpenAI's o1 model. Despite the progress, several challenges exist. Formal verification and robust error detection are necessary to improve the interpretability and trustworthiness of reasoning trace. Reliance on purely text-based logic necessitates neuro-symbolic frameworks that combine continuous embeddings with external symbolic manipulators for advanced mathematics, proofs, or legal argumentation. Beyond the targeted fine-tuning in math or coding, real-world applications demand broader domain adaptation and multi-modal reasoning, integrating signals from text, vision, audio, and beyond. The transition of LLMs from mere next-token predictors to structured reasoners is under way, and while o1 showcases the promise of today's solutions, forging robust, trustworthy, and multi-modal reasoning engines necessitates substantial future works.

References

Yasin Abbasi-Yadkori, Peter Bartlett, Kush Bhatia, Nevena Lazic, Csaba Szepesvari, and Gellért Weisz. 2019. Politex: Regret bounds for policy iteration using expert prediction. In *International Conference on Machine Learning*, pages 3692–3702. PMLR.

Janice Ahn, Rishu Verma, Renze Lou, Di Liu, Rui Zhang, and Wenpeng Yin. 2024. Large language models for mathematical reasoning: Progresses and challenges. *arXiv preprint arXiv:2402.00157*.

AI-MO. 2025. Aimo validation dataset. Accessed: 2025-01-12.

Afra Amini, Tim Vieira, and Ryan Cotterell. 2024. Direct preference optimization with an offset.

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. 2016. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Hritik Bansal, Arian Hosseini, Rishabh Agarwal, Vinh Q. Tran, and Mehran Kazemi. 2024. Smaller, weaker, yet better: Training llm reasoners via compute-optimal sampling.

Andrew G Barto, Richard S Sutton, and Charles W Anderson. 1983. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE transactions on systems, man, and cybernetics*, (5):834–846.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR.

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V Le, Christopher Ré, and Azalia Mirhoseini. 2024. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*.

Victor Carbune, Hassan Mansoor, Fangyu Liu, Rahul Aralikatte, Gilles Baechler, Jindong Chen, and Abhanshu Sharma. 2024. Chart-based reasoning: Transferring capabilities from llms to vlms.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. 2024. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 15(3):1–45.

Boyuan Chen, Zhuo Xu, Sean Kirmani, Brain Ichter, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. 2024a. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14455–14465.

Guoxin Chen, Minpeng Liao, Chengxi Li, and Kai Fan. 2024b. Alphamath almost zero: process supervision without process. *arXiv preprint arXiv:2405.03553*.

Guoxin Chen, Minpeng Liao, Chengxi Li, and Kai Fan. 2024c. Step-level value preference optimization for mathematical reasoning.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W. Cohen. 2023a. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks.

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. 2020. Tabfact : A large-scale dataset for table-based fact verification. In *International Conference on Learning Representations (ICLR)*, Addis Ababa, Ethiopia.

Xi Chen, Xiao Wang, Lucas Beyer, Alexander Kolesnikov, Jialin Wu, Paul Voigtlaender, Basil Mustafa, Sebastian Goodman, Ibrahim Alabdulmohsin, Piotr Padlewski, Daniel Salz, Xi Xiong, Daniel Vlasic, Filip Pavetic, Keran Rong, Tianli Yu, Daniel Keysers, Xiaohua Zhai, and Radu Soricut. 2023b. Pali-3 vision language models: Smaller, faster, stronger.

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

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021a. Training verifiers to solve math word problems.

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

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runxin Xu, Ruoyu Zhang, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Shuting Pan, T. Wang, Tao Yun, Tian Pei, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wanjia Zhao, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaokang Zhang, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xingkai Yu, Xinnan Song, Xinxia Shan, Xinyi Zhou, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. X. Zhu, Yang Zhang, Yanhong Xu, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Li, Yaohui Wang, Yi Yu, Yi Zheng, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Ying Tang, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yu Wu, Yuan Ou, Yuchen Zhu, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yukun Zha, Yunfan Xiong, Yunxian Ma, Yuting Yan, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Z. F. Wu, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao, Zhipeng Xu, Zhiyu Wu, Zhongyu Zhang, Zhuoshu Li, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Ziyi Gao, and Zizheng Pan. 2024. Deepseek-v3 technical report.

DeepSeek-R1-Lite-Preview. 2024. Deepseek-r1-lite-preview.

Yuhao Dong, Zuyan Liu, Hai-Long Sun, Jingkang Yang, Winston Hu, Yongming Rao, and Ziwei Liu. 2024. Insight-v: Exploring long-chain visual reasoning with multimodal large language models. *arXiv preprint arXiv:2411.14432*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Xidong Feng, Ziyu Wan, Muning Wen, Stephen Marcus McAleer, Ying Wen, Weinan Zhang, and Jun Wang. 2023. Alphazero-like tree-search can guide large language model decoding and training. *arXiv preprint arXiv:2309.17179*.

Jiahui Gao, Renjie Pi, Jipeng Zhang, Jiacheng Ye, Wanjun Zhong, Yufei Wang, Lanqing Hong, Jianhua Han, Hang Xu, Zhenguo Li, and Lingpeng Kong. 2023. G-llava: Solving geometric problem with multi-modal large language model.

Kuofeng Gao, Huanqia Cai, Qingyao Shuai, Dihong Gong, and Zhifeng Li. 2024. Embedding self-correction as an inherent ability in large language models for enhanced mathematical reasoning. *arXiv preprint arXiv:2410.10735*.

Team GLM, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Diego Rojas, Guanyu Feng, Hanlin Zhao, Hanyu Lai, et al. 2024. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *arXiv preprint arXiv:2406.12793*.

Google AI. Thinking Mode - Gemini API Documentation. `https://ai.google.dev/gemini-api/docs/thinking-mode`. Accessed: 2025-01-16.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Minlie Huang, Nan Duan, and Weizhu Chen. 2024. Tora: A tool-integrated reasoning agent for mathematical problem solving.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. 2024. The llama 3 herd of models.

Melody Y Guan, Manas Joglekar, Eric Wallace, Saachi Jain, Boaz Barak, Alec Heylar, Rachel Dias, Andrea Vallone, Hongyu Ren, Jason Wei, et al. 2024. Deliberative alignment: Reasoning enables safer language models. *arXiv preprint arXiv:2412.16339*.

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

Tingxu Han, Chunrong Fang, Shiyu Zhao, Shiqing Ma, Zhenyu Chen, and Zhenting Wang. 2024. Token-budget-aware llm reasoning. *arXiv preprint arXiv:2412.18547*.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. 2024a. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. 2024b. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems.

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

Namgyu Ho, Laura Schmid, and Se-Young Yun. 2022. Large language models are reasoning teachers. *arXiv preprint arXiv:2212.10071*.

Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. 2022. Unnatural instructions: Tuning language models with (almost) no human labor. *arXiv preprint arXiv:2212.09689*.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. 2023. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes.

Yushi Hu, Weijia Shi, Xingyu Fu, Dan Roth, Mari Ostendorf, Luke Zettlemoyer, Noah A Smith, and Ranjay Krishna. 2024. Visual sketchpad: Sketching as a visual chain of thought for multimodal language models. *arXiv preprint arXiv:2406.09403*.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. 2023. Large language models cannot self-correct reasoning yet. *arXiv preprint arXiv:2310.01798*.

Hyeonbin Hwang, Doyoung Kim, Seungone Kim, Seonghyeon Ye, and Minjoon Seo. 2024a. Self-explore: Enhancing mathematical reasoning in language models with fine-grained rewards.

Hyeonbin Hwang, Doyoung Kim, Seungone Kim, Seonghyeon Ye, and Minjoon Seo. 2024b. Self-explore: Enhancing mathematical reasoning in language models with fine-grained rewards.

Joonwon Jang, Jaehee Kim, Wonbin Kweon, and Hwanjo Yu. 2024. Verbosity-aware rationale reduction: Effective reduction of redundant rationale via principled criteria. *arXiv preprint arXiv:2412.21006*.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Bowen Jiang, Yangxinyu Xie, Zhuoqun Hao, Xiaomeng Wang, Tanwi Mallick, Weijie J Su, Camillo J Taylor, and Dan Roth. 2024. A peek into token bias: Large language models are not yet genuine reasoners. *arXiv preprint arXiv:2406.11050*.

Mingyu Jin, Qinkai Yu, Dong Shu, Haiyan Zhao, Wenyue Hua, Yanda Meng, Yongfeng Zhang, and Mengnan Du. 2024. The impact of reasoning step length on large language models. *arXiv preprint arXiv:2401.04925*.

Aditya Kalyanpur, Kailash Karthik Saravanakumar, Victor Barres, Jennifer Chu-Carroll, David Melville, and David Ferrucci. 2024. Llm-arc: Enhancing llms with an automated reasoning critic. *arXiv preprint arXiv:2406.17663*.

Jikun Kang, Xin Zhe Li, Xi Chen, Amirreza Kazemi, Qianyi Sun, Boxing Chen, Dong Li, Xu He, Quan He, Feng Wen, et al. 2024. Mindstar: Enhancing math reasoning in pre-trained llms at inference time. *arXiv preprint arXiv:2405.16265*.

Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, John D Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, et al. Training language models to self-correct via reinforcement learning, 2024. *URL https://arxiv. org/abs/2409.12917*.

Po-Nien Kung and Nanyun Peng. 2023. Do models really learn to follow instructions? an empirical study of instruction tuning. *ArXiv*, abs/2305.11383.

Tamera Lanham, Anna Chen, Ansh Radhakrishnan, Benoit Steiner, Carson Denison, Danny Hernandez, Dustin Li, Esin Durmus, Evan Hubinger, Jackson Kernion, et al. 2023. Measuring faithfulness in chain-of-thought reasoning. *arXiv preprint arXiv:2307.13702*.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Kellie Lu, Thomas Mesnard, Colton Bishop, Victor Carbune, and Abhinav Rastogi. 2023. Rlaif: Scaling reinforcement learning from human feedback with ai feedback. *arXiv preprint arXiv:2309.00267*.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. 2022. Solving quantitative reasoning problems with language models.

Chengpeng Li, Guanting Dong, Mingfeng Xue, Ru Peng, Xiang Wang, and Dayiheng Liu. 2024. Dotamath: Decomposition of thought with code assistance and self-correction for mathematical reasoning. *arXiv preprint arXiv:2407.04078*.

Yuanzhi Li, Sébastien Bubeck, Ronen Eldan, Allie Del Giorno, Suriya Gunasekar, and Yin Tat Lee. 2023. Textbooks are all you need ii: phi-1.5 technical report. *arXiv preprint arXiv:2309.05463*.

Minpeng Liao, Chengxi Li, Wei Luo, Wu Jing, and Kai Fan. 2024a. MARIO: MAth reasoning with code interpreter output - a reproducible pipeline. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 905–924, Bangkok, Thailand. Association for Computational Linguistics.

Minpeng Liao, Wei Luo, Chengxi Li, Jing Wu, and Kai Fan. 2024b. Mario: Math reasoning with code interpreter output – a reproducible pipeline.

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

Aixin Liu, Bei Feng, Bin Wang, Bingxuan Wang, Bo Liu, Chenggang Zhao, Chengqi Dengr, Chong Ruan, Damai Dai, Daya Guo, et al. 2024a. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434*.

Fangyu Liu, Julian Martin Eisenschlos, Francesco Piccinno, Syrine Krichene, Chenxi Pang, Kenton Lee, Mandar Joshi, Wenhu Chen, Nigel Collier, and Yasemin Altun. 2023. Deplot: One-shot visual language reasoning by plot-to-table translation.

Jiacheng Liu, Andrew Cohen, Ramakanth Pasunuru, Yejin Choi, Hannaneh Hajishirzi, and Asli Celikyilmaz. 2024b. Don't throw away your value model! generating more preferable text with value-guided monte-carlo tree search decoding.

Shengcai Liu, Caishun Chen, Xinghua Qu, Ke Tang, and Yew-Soon Ong. 2024c. Large language models as evolutionary optimizers. pages 1–8.

Zimu Lu, Aojun Zhou, Houxing Ren, Ke Wang, Weikang Shi, Junting Pan, Mingjie Zhan, and Hongsheng Li. 2024. Mathgenie: Generating synthetic data with question back-translation for enhancing mathematical reasoning of llms.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. 2024. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36.

Nat McAleese, Rai Michael Pokorny, Juan Felipe Ceron Uribe, Evgenia Nitishinskaya, Maja Trebacz, and Jan Leike. 2024. Llm critics help catch llm bugs. *arXiv preprint arXiv:2407.00215*.

Jincheng Mei, Chenjun Xiao, Ruitong Huang, Dale Schuurmans, and Martin Müller. 2019. On principled entropy exploration in policy optimization. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 3130–3136.

Yingqian Min, Zhipeng Chen, Jinhao Jiang, Jie Chen, Jia Deng, Yiwen Hu, Yiru Tang, Jiapeng Wang, Xiaoxue Cheng, Huatong Song, et al. 2024. Imitate, explore, and self-improve: A reproduction report on slow-thinking reasoning systems. *arXiv preprint arXiv:2412.09413*.

Mike Mirzayanov. 2025. Codeforces. Accessed: 2025-01-12.

Mistral AI. 2024. Mixtral-8x22b-v0.1. https://huggingface.co/mistralai/Mixtral-8x22B-v0.1.

Linyong Nan, Chiachun Hsieh, Ziming Mao, Xi Victoria Lin, Neha Verma, Rui Zhang, Wojciech Kryściński, Hailey Schoelkopf, Riley Kong, Xiangru Tang, Mutethia Mutuma, Ben Rosand, Isabel Trindade, Renusree Bandaru, Jacob Cunningham, Caiming Xiong, and Dragomir Radev. 2022. Fetaqa: Free-form table question answering. *Transactions of the Association for Computational Linguistics*, 10:35–49.

Harsha Nori, Yin Tat Lee, Sheng Zhang, Dean Carignan, Richard Edgar, Nicolo Fusi, Nicholas King, Jonathan Larson, Yuanzhi Li, Weishung Liu, et al. 2023. Can generalist foundation models outcompete special-purpose tuning? case study in medicine. *arXiv preprint arXiv:2311.16452*.

Harsha Nori, Naoto Usuyama, Nicholas King, Scott Mayer McKinney, Xavier Fernandes, Sheng Zhang, and Eric Horvitz. 2024. From medprompt to o1: Exploration of run-time strategies for medical challenge problems and beyond. *arXiv preprint arXiv:2411.03590*.

Open Source O1. 2025. Open-o1. Accessed: 2025-01-17.

OpenAI o1 Contributors. 2024. Learning to reason with llms. `https://openai.com/index/learning-to-reason-with-llms/`.

Skywork o1 Team. 2024. Skywork-o1 open series. `https://huggingface.co/Skywork`.

OpenAI. 2023. Gpt-4 technical report.

OpenAI. 2024a. Hello, GPT-4o. https://openai.com/index/hello-gpt-4o/.

OpenAI. 2024b. O-1: Optimization for language models with continuous integration. https://openai.com/o1/.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Panupong Pasupat and Percy Liang. 2015. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1470–1480, Beijing, China. Association for Computational Linguistics.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*.

Pranav Putta, Edmund Mills, Naman Garg, Sumeet Motwani, Chelsea Finn, Divyansh Garg, and Rafael Rafailov. 2024. Agent q: Advanced reasoning and learning for autonomous ai agents.

Yiwei Qin, Xuefeng Li, Haoyang Zou, Yixiu Liu, Shijie Xia, Zhen Huang, Yixin Ye, Weizhe Yuan, Hector Liu, Yuanzhi Li, and Pengfei Liu. 2024. O1 replication journey: A strategic progress report – part 1.

QwenLM, QVQ. QVQ-72B Preview - QwenLM Blog. `https://qwenlm.github.io/blog/qvq-72b-preview`. Accessed: 2025-01-16.

QwenLM, QwQ. QWQ-32B Preview - QwenLM Blog. `https://qwenlm.github.io/blog/qwq-32b-preview`. Accessed: 2025-01-16.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2023. Gpqa: A graduate-level google-proof q&a benchmark.

Amir Saeidi, Shivanshu Verma, and Chitta Baral. 2024. Insights into alignment: Evaluating dpo and its variants across multiple tasks, 2024. https://api.semanticscholar.org/CorpusID269303161.

William Saunders, Catherine Yeh, Jeff Wu, Steven Bills, Long Ouyang, Jonathan Ward, and Jan Leike. 2022a. Self-critiquing models for assisting human evaluators.

William Saunders, Catherine Yeh, Jeff Wu, Steven Bills, Long Ouyang, Jonathan Ward, and Jan Leike. 2022b. Self-critiquing models for assisting human evaluators. *arXiv preprint arXiv:2206.05802*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Paulo Shakarian, Abhinav Koyyalamudi, Noel Ngu, and Lakshmivihari Mareedu. 2023. An independent evaluation of chatgpt on mathematical word problems (mwp). *arXiv preprint arXiv:2302.13814*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024a. Deepseekmath: Pushing the limits of mathematical reasoning in open language models.

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

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H Chi, Nathanael Schärli, and Denny Zhou. 2023. Large language models can be easily distracted by irrelevant context. In *International Conference on Machine Learning*, pages 31210–31227. PMLR.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. 2017. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*.

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

Mingyang Song, Zhaochen Su, Xiaoye Qu, Jiawei Zhou, and Yu Cheng. 2025. Prmbench: A fine-grained and challenging benchmark for process-level reward models.

Zayne Sprague, Fangcong Yin, Juan Diego Rodriguez, Dongwei Jiang, Manya Wadhwa, Prasann Singhal, Xinyu Zhao, Xi Ye, Kyle Mahowald, and Greg Durrett. 2024. To cot or not to cot? chain-of-thought helps mainly on math and symbolic reasoning. *arXiv preprint arXiv:2409.12183*.

Hao Sun, Alihan Hüyük, and Mihaela van der Schaar. 2023a. Query-dependent prompt evaluation and optimization with offline inverse rl.

Xiaofei Sun, Linfeng Dong, Xiaoya Li, Zhen Wan, Shuhe Wang, Tianwei Zhang, Jiwei Li, Fei Cheng, Lingjuan Lyu, Fei Wu, and Guoyin Wang. 2023b. Pushing the limits of chatgpt on nlp tasks.

Xiaofei Sun, Xiaoya Li, Jiwei Li, Fei Wu, Shangwei Guo, Tianwei Zhang, and Guoyin Wang. 2023c. Text classification via large language models. *arXiv preprint arXiv:2305.08377*.

Xiaofei Sun, Xiaoya Li, Shengyu Zhang, Shuhe Wang, Fei Wu, Jiwei Li, Tianwei Zhang, and Guoyin Wang. 2023d. Sentiment analysis through llm negotiations. *arXiv preprint arXiv:2311.01876*.

Gokul Swamy, Christoph Dann, Rahul Kidambi, Zhiwei Steven Wu, and Alekh Agarwal. 2024. A minimaximalist approach to reinforcement learning from human feedback. *arXiv preprint arXiv:2401.04056*.

Zhengyang Tang, Xingxing Zhang, Benyou Wang, and Furu Wei. 2024. Mathscale: Scaling instruction tuning for mathematical reasoning.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, Soroosh Mariooryad, Yifan Ding, Xinyang Geng, Fred Alcober, Roy Frostig, Mark Omernick, Lexi Walker, Cosmin Paduraru, Christina Sorokin, Andrea Tacchetti, Colin Gaffney, Samira Daruki, Olcan Sercinoglu, Zach Gleicher, Juliette Love, Paul Voigtlaender, Rohan Jain, Gabriela Surita, Kareem Mohamed, Rory Blevins, Junwhan Ahn, Tao Zhu, Kornraphop Kawintiranon, Orhan Firat, Yiming Gu, Yujing Zhang, Matthew Rahtz, Manaal Faruqui, Natalie Clay, Justin Gilmer, JD Co-Reyes, Ivo Penchev, Rui Zhu, Nobuyuki Morioka, Kevin Hui, Krishna Haridasan, Victor Campos, Mahdis Mahdieh, Mandy Guo, Samer Hassan, Kevin Kilgour, Arpi Vezer, Heng-Tze Cheng, Raoul de Liedekerke, Siddharth Goyal, Paul Barham, DJ Strouse, Seb Noury, Jonas Adler, Mukund Sundararajan, Sharad Vikram, Dmitry Lepikhin, Michela Paganini, Xavier Garcia, Fan Yang, Dasha Valter, Maja Trebacz, Kiran Vodrahalli, Chulayuth Asawaroengchai,

Roman Ring, Norbert Kalb, Livio Baldini Soares, Siddhartha Brahma, David Steiner, Tianhe Yu, Fabian Mentzer, Antoine He, Lucas Gonzalez, Bibo Xu, Raphael Lopez Kaufman, Laurent El Shafey, Junhyuk Oh, Tom Hennigan, George van den Driessche, Seth Odoom, Mario Lucic, Becca Roelofs, Sid Lall, Amit Marathe, Betty Chan, Santiago Ontanon, Luheng He, Denis Teplyashin, Jonathan Lai, Phil Crone, Bogdan Damoc, Lewis Ho, Sebastian Riedel, Karel Lenc, Chih-Kuan Yeh, Aakanksha Chowdhery, Yang Xu, Mehran Kazemi, Ehsan Amid, Anastasia Petrushkina, Kevin Swersky, Ali Khodaei, Gowoon Chen, Chris Larkin, Mario Pinto, Geng Yan, Adria Puigdomenech Badia, Piyush Patil, Steven Hansen, Dave Orr, Sebastien M. R. Arnold, Jordan Grimstad, Andrew Dai, Sholto Douglas, Rishika Sinha, Vikas Yadav, Xi Chen, Elena Gribovskaya, Jacob Austin, Jeffrey Zhao, Kaushal Patel, Paul Komarek, Sophia Austin, Sebastian Borgeaud, Linda Friso, Abhimanyu Goyal, Ben Caine, Kris Cao, Da-Woon Chung, Matthew Lamm, Gabe Barth-Maron, Thais Kagohara, Kate Olszewska, Mia Chen, Kaushik Shivakumar, Rishabh Agarwal, Harshal Godhia, Ravi Rajwar, Javier Snaider, Xerxes Dotiwalla, Yuan Liu, Aditya Barua, Victor Ungureanu, Yuan Zhang, Bat-Orgil Batsaikhan, Mateo Wirth, James Qin, Ivo Danihelka, Tulsee Doshi, Martin Chadwick, Jilin Chen, Sanil Jain, Quoc Le, Arjun Kar, Madhu Gurumurthy, Cheng Li, Ruoxin Sang, Fangyu Liu, Lampros Lamprou, Rich Munoz, Nathan Lintz, Harsh Mehta, Heidi Howard, Malcolm Reynolds, Lora Aroyo, Quan Wang, Lorenzo Blanco, Albin Cassirer, Jordan Griffith, Dipanjan Das, Stephan Lee, Jakub Sygnowski, Zach Fisher, James Besley, Richard Powell, Zafarali Ahmed, Dominik Paulus, David Reitter, Zalan Borsos, Rishabh Joshi, Aedan Pope, Steven Hand, Vittorio Selo, Vihan Jain, Nikhil Sethi, Megha Goel, Takaki Makino, Rhys May, Zhen Yang, Johan Schalkwyk, Christina Butterfield, Anja Hauth, Alex Goldin, Will Hawkins, Evan Senter, Sergey Brin, Oliver Woodman, Marvin Ritter, Eric Noland, Minh Giang, Vijay Bolina, Lisa Lee, Tim Blyth, Ian Mackinnon, Machel Reid, Obaid Sarvana, David Silver, Alexander Chen, Lily Wang, Loren Maggiore, Oscar Chang, Nithya Attaluri, Gregory Thornton, Chung-Cheng Chiu, Oskar Bunyan, Nir Levine, Timothy Chung, Evgenii Eltyshev, Xiance Si, Timothy Lillicrap, Demetra Brady, Vaibhav Aggarwal, Boxi Wu, Yuanzhong Xu, Ross McIlroy, Kartikeya Badola, Paramjit Sandhu, Erica Moreira, Wojciech Stokowiec, Ross Hemsley, Dong Li, Alex Tudor, Pranav Shyam, Elahe Rahimtoroghi, Salem Haykal, Pablo Sprechmann, Xiang Zhou, Diana Mincu, Yujia Li, Ravi Addanki, Kalpesh Krishna, Xiao Wu, Alexandre Frechette, Matan Eyal, Allan Dafoe, Dave Lacey, Jay Whang, Thi Avrahami, Ye Zhang, Emanuel Taropa, Hanzhao Lin, Daniel Toyama, Eliza Rutherford, Motoki Sano, HyunJeong Choe, Alex Tomala, Chalence Safranek-Shrader, Nora Kassner, Mantas Pajarskas, Matt Harvey, Sean Sechrist, Meire Fortunato, Christina Lyu, Gamaleldin Elsayed, Chenkai Kuang, James Lottes, Eric Chu, Chao Jia, Chih-Wei Chen, Peter Humphreys, Kate Baumli, Connie Tao, Rajkumar Samuel, Cicero Nogueira dos Santos, Anders Andreassen, Nemanja Rakićević, Dominik Grewe, Aviral Kumar, Stephanie Winkler, Jonathan Caton, Andrew Brock, Sid Dalmia, Hannah Sheahan, Iain Barr, Yingjie Miao, Paul Natsev, Jacob Devlin, Feryal Behbahani, Flavien Prost, Yanhua Sun, Artiom Myaskovsky, Thanumalayan Sankaranarayana Pillai, Dan Hurt, Angeliki Lazaridou, Xi Xiong, Ce Zheng, Fabio Pardo, Xiaowei Li, Dan Horgan, Joe Stanton, Moran Ambar, Fei Xia, Alejandro Lince, Mingqiu Wang, Basil Mustafa, Albert Webson, Hyo Lee, Rohan Anil, Martin Wicke, Timothy Dozat, Abhishek Sinha, Enrique Piqueras, Elahe Dabir, Shyam Upadhyay, Anudhyan Boral, Lisa Anne Hendricks, Corey Fry, Josip Djolonga, Yi Su, Jake Walker, Jane Labanowski, Ronny Huang, Vedant Misra, Jeremy Chen, RJ Skerry-Ryan, Avi Singh, Shruti Rijhwani, Dian Yu, Alex Castro-Ros, Beer Changpinyo, Romina Datta, Sumit Bagri, Arnar Mar Hrafnkelsson, Marcello Maggioni, Daniel Zheng, Yury Sulsky, Shaobo Hou, Tom Le Paine, Antoine Yang, Jason Riesa, Dominika Rogozinska, Dror Marcus, Dalia El Badawy, Qiao Zhang, Luyu Wang, Helen Miller, Jeremy Greer, Lars Lowe Sjos, Azade Nova, Heiga Zen, Rahma Chaabouni, Mihaela Rosca, Jiepu Jiang, Charlie Chen, Ruibo Liu, Tara Sainath, Maxim Krikun, Alex Polozov, Jean-Baptiste Lespiau, Josh Newlan, Zeyncep Cankara, Soo Kwak, Yunhan Xu, Phil Chen, Andy Coenen, Clemens Meyer, Katerina Tsihlas, Ada Ma, Juraj Gottweis, Jinwei Xing, Chenjie Gu, Jin Miao, Christian Frank, Zeynep Cankara, Sanjay Ganapathy, Ishita Dasgupta, Steph Hughes-Fitt, Heng Chen, David Reid, Keran Rong, Hongmin Fan, Joost van Amersfoort, Vincent Zhuang, Aaron Cohen, Shixiang Shane Gu, Anhad Mohananey, Anastasija Ilic, Taylor Tobin, John Wieting, Anna Bortsova, Phoebe Thacker, Emma Wang, Emily Caveness, Justin Chiu, Eren Sezener, Alex Kaskasoli, Steven Baker, Katie Millican, Mohamed Elhawaty, Kostas Aisopos, Carl Lebsack, Nathan Byrd, Hanjun Dai, Wenhao Jia, Matthew Wiethoff, Elnaz Davoodi, Albert Weston, Lakshman Yagati, Arun Ahuja, Isabel Gao, Golan Pundak, Susan Zhang, Michael Azzam, Khe Chai Sim, Sergi Caelles, James Keeling, Abhanshu Sharma, Andy Swing, YaGuang Li, Chenxi Liu, Carrie Grimes Bostock, Yamini Bansal, Zachary Nado, Ankesh Anand, Josh Lipschultz, Abhijit Karmarkar, Lev Proleev, Abe Ittycheriah, Soheil Hassas Yeganeh, George Polovets, Aleksandra Faust, Jiao Sun, Alban Rrustemi, Pen Li, Rakesh Shivanna, Jeremiah Liu, Chris Welty, Federico Lebron, Anirudh Baddepudi, Sebastian Krause, Emilio Parisotto, Radu Soricut, Zheng Xu, Dawn Bloxwich, Melvin Johnson, Behnam Neyshabur, Justin Mao-Jones, Renshen Wang, Vinay Ramasesh, Zaheer Abbas, Arthur Guez, Constant Segal, Duc Dung Nguyen, James Svensson, Le Hou, Sarah York, Kieran Milan, Sophie Bridgers, Wiktor Gworek, Marco Tagliasacchi, James Lee-Thorp, Michael Chang, Alexey Guseynov, Ale Jakse Hartman, Michael Kwong, Ruizhe Zhao, Sheleem Kashem, Elizabeth Cole, Antoine Miech, Richard Tanburn, Mary Phuong, Filip Pavetic, Sebastien Cevey, Ramona Comanescu, Richard Ives, Sherry Yang, Cosmo Du, Bo Li, Zizhao Zhang, Mariko Iinuma, Clara Huiyi Hu, Aurko Roy, Shaan Bijwadia, Zhenkai Zhu, Danilo Martins, Rachel Saputro, Anita Gergely, Steven Zheng, Dawei Jia, Ioannis Antonoglou, Adam Sadovsky, Shane Gu, Yingying Bi, Alek Andreev, Sina Samangooei, Mina Khan, Tomas Kocisky, Angelos Filos, Chintu Kumar, Colton Bishop, Adams Yu, Sarah Hodkinson, Sid Mittal, Premal Shah, Alexandre Moufarek, Yong Cheng, Adam Bloniarz, Jaehoon Lee, Pedram Pejman, Paul Michel, Stephen Spencer, Vladimir Feinberg, Xuehan Xiong, Nikolay Savinov, Charlotte Smith,

Siamak Shakeri, Dustin Tran, Mary Chesus, Bernd Bohnet, George Tucker, Tamara von Glehn, Carrie Muir, Yiran Mao, Hideto Kazawa, Ambrose Slone, Kedar Soparkar, Disha Shrivastava, James Cobon-Kerr, Michael Sharman, Jay Pavagadhi, Carlos Araya, Karolis Misiunas, Nimesh Ghelani, Michael Laskin, David Barker, Qiujia Li, Anton Briukhov, Neil Houlsby, Mia Glaese, Balaji Lakshminarayanan, Nathan Schucher, Yunhao Tang, Eli Collins, Hyeontaek Lim, Fangxiaoyu Feng, Adria Recasens, Guangda Lai, Alberto Magni, Nicola De Cao, Aditya Siddhant, Zoe Ashwood, Jordi Orbay, Mostafa Dehghani, Jenny Brennan, Yifan He, Kelvin Xu, Yang Gao, Carl Saroufim, James Molloy, Xinyi Wu, Seb Arnold, Solomon Chang, Julian Schrittwieser, Elena Buchatskaya, Soroush Radpour, Martin Polacek, Skye Giordano, Ankur Bapna, Simon Tokumine, Vincent Hellendoorn, Thibault Sottiaux, Sarah Cogan, Aliaksei Severyn, Mohammad Saleh, Shantanu Thakoor, Laurent Shefey, Siyuan Qiao, Meenu Gaba, Shuo yiin Chang, Craig Swanson, Biao Zhang, Benjamin Lee, Paul Kishan Rubenstein, Gan Song, Tom Kwiatkowski, Anna Koop, Ajay Kannan, David Kao, Parker Schuh, Axel Stjerngren, Golnaz Ghiasi, Gena Gibson, Luke Vilnis, Ye Yuan, Felipe Tiengo Ferreira, Aishwarya Kamath, Ted Klimenko, Ken Franko, Kefan Xiao, Indro Bhattacharya, Miteyan Patel, Rui Wang, Alex Morris, Robin Strudel, Vivek Sharma, Peter Choy, Sayed Hadi Hashemi, Jessica Landon, Mara Finkelstein, Priya Jhakra, Justin Frye, Megan Barnes, Matthew Mauger, Dennis Daun, Khuslen Baatarsukh, Matthew Tung, Wael Farhan, Henryk Michalewski, Fabio Viola, Felix de Chaumont Quitry, Charline Le Lan, Tom Hudson, Qingze Wang, Felix Fischer, Ivy Zheng, Elspeth White, Anca Dragan, Jean baptiste Alayrac, Eric Ni, Alexander Pritzel, Adam Iwanicki, Michael Isard, Anna Bulanova, Lukas Zilka, Ethan Dyer, Devendra Sachan, Srivatsan Srinivasan, Hannah Muckenhirn, Honglong Cai, Amol Mandhane, Mukarram Tariq, Jack W. Rae, Gary Wang, Kareem Ayoub, Nicholas FitzGerald, Yao Zhao, Woohyun Han, Chris Alberti, Dan Garrette, Kashyap Krishnakumar, Mai Gimenez, Anselm Levskaya, Daniel Sohn, Josip Matak, Inaki Iturrate, Michael B. Chang, Jackie Xiang, Yuan Cao, Nishant Ranka, Geoff Brown, Adrian Hutter, Vahab Mirrokni, Nanxin Chen, Kaisheng Yao, Zoltan Egyed, Francois Galilee, Tyler Liechty, Praveen Kallakuri, Evan Palmer, Sanjay Ghemawat, Jasmine Liu, David Tao, Chloe Thornton, Tim Green, Mimi Jasarevic, Sharon Lin, Victor Cotruta, Yi-Xuan Tan, Noah Fiedel, Hongkun Yu, Ed Chi, Alexander Neitz, Jens Heitkaemper, Anu Sinha, Denny Zhou, Yi Sun, Charbel Kaed, Brice Hulse, Swaroop Mishra, Maria Georgaki, Sneha Kudugunta, Clement Farabet, Izhak Shafran, Daniel Vlasic, Anton Tsitsulin, Rajagopal Ananthanarayanan, Alen Carin, Guolong Su, Pei Sun, Shashank V, Gabriel Carvajal, Josef Broder, Iulia Comsa, Alena Repina, William Wong, Warren Weilun Chen, Peter Hawkins, Egor Filonov, Lucia Loher, Christoph Hirnschall, Weiyi Wang, Jingchen Ye, Andrea Burns, Hardie Cate, Diana Gage Wright, Federico Piccinini, Lei Zhang, Chu-Cheng Lin, Ionel Gog, Yana Kulizhskaya, Ashwin Sreevatsa, Shuang Song, Luis C. Cobo, Anand Iyer, Chetan Tekur, Guillermo Garrido, Zhuyun Xiao, Rupert Kemp, Huaixiu Steven Zheng, Hui Li, Ananth Agarwal, Christel Ngani, Kati Goshvadi, Rebeca Santamaria-Fernandez, Wojciech Fica, Xinyun Chen, Chris Gorgolewski, Sean Sun, Roopal Garg, Xinyu Ye, S. M. Ali Eslami, Nan Hua, Jon Simon, Pratik Joshi, Yelin Kim, Ian Tenney, Sahitya Potluri, Lam Nguyen Thiet, Quan Yuan, Florian Luisier, Alexandra Chronopoulou, Salvatore Scellato, Praveen Srinivasan, Minmin Chen, Vinod Koverkathu, Valentin Dalibard, Yaming Xu, Brennan Saeta, Keith Anderson, Thibault Sellam, Nick Fernando, Fantine Huot, Junehyuk Jung, Mani Varadarajan, Michael Quinn, Amit Raul, Maigo Le, Ruslan Habalov, Jon Clark, Komal Jalan, Kalesha Bullard, Achintya Singhal, Thang Luong, Boyu Wang, Sujeevan Rajayogam, Julian Eisenschlos, Johnson Jia, Daniel Finchelstein, Alex Yakubovich, Daniel Balle, Michael Fink, Sameer Agarwal, Jing Li, Dj Dvijotham, Shalini Pal, Kai Kang, Jaclyn Konzelmann, Jennifer Beattie, Olivier Dousse, Diane Wu, Remi Crocker, Chen Elkind, Siddhartha Reddy Jonnalagadda, Jong Lee, Dan Holtmann-Rice, Krystal Kallarackal, Rosanne Liu, Denis Vnukov, Neera Vats, Luca Invernizzi, Mohsen Jafari, Huanjie Zhou, Lilly Taylor, Jennifer Prendki, Marcus Wu, Tom Eccles, Tianqi Liu, Kavya Kopparapu, Francoise Beaufays, Christof Angermueller, Andreea Marzoca, Shourya Sarcar, Hilal Dib, Jeff Stanway, Frank Perbet, Nejc Trdin, Rachel Sterneck, Andrey Khorlin, Dinghua Li, Xihui Wu, Sonam Goenka, David Madras, Sasha Goldshtein, Willi Gierke, Tong Zhou, Yaxin Liu, Yannie Liang, Anais White, Yunjie Li, Shreya Singh, Sanaz Bahargam, Mark Epstein, Sujoy Basu, Li Lao, Adnan Ozturel, Carl Crous, Alex Zhai, Han Lu, Zora Tung, Neeraj Gaur, Alanna Walton, Lucas Dixon, Ming Zhang, Amir Globerson, Grant Uy, Andrew Bolt, Olivia Wiles, Milad Nasr, Ilia Shumailov, Marco Selvi, Francesco Piccinno, Ricardo Aguilar, Sara McCarthy, Misha Khalman, Mrinal Shukla, Vlado Galic, John Carpenter, Kevin Villela, Haibin Zhang, Harry Richardson, James Martens, Matko Bosnjak, Shreyas Rammohan Belle, Jeff Seibert, Mahmoud Alnahlawi, Brian McWilliams, Sankalp Singh, Annie Louis, Wen Ding, Dan Popovici, Lenin Simicich, Laura Knight, Pulkit Mehta, Nishesh Gupta, Chongyang Shi, Saaber Fatehi, Jovana Mitrovic, Alex Grills, Joseph Pagadora, Tsendsuren Munkhdalai, Dessie Petrova, Danielle Eisenbud, Zhishuai Zhang, Damion Yates, Bhavishya Mittal, Nilesh Tripuraneni, Yannis Assael, Thomas Brovelli, Prateek Jain, Mihajlo Velimirovic, Canfer Akbulut, Jiaqi Mu, Wolfgang Macherey, Ravin Kumar, Jun Xu, Haroon Qureshi, Gheorghe Comanici, Jeremy Wiesner, Zhitao Gong, Anton Ruddock, Matthias Bauer, Nick Felt, Anirudh GP, Anurag Arnab, Dustin Zelle, Jonas Rothfuss, Bill Rosgen, Ashish Shenoy, Bryan Seybold, Xinjian Li, Jayaram Mudigonda, Goker Erdogan, Jiawei Xia, Jiri Simsa, Andrea Michi, Yi Yao, Christopher Yew, Steven Kan, Isaac Caswell, Carey Radebaugh, Andre Elisseeff, Pedro Valenzuela, Kay McKinney, Kim Paterson, Albert Cui, Eri Latorre-Chimoto, Solomon Kim, William Zeng, Ken Durden, Priya Ponnapalli, Tiberiu Sosea, Christopher A. Choquette-Choo, James Manyika, Brona Robenek, Harsha Vashisht, Sebastien Pereira, Hoi Lam, Marko Velic, Denese Owusu-Afriyie, Katherine Lee, Tolga Bolukbasi, Alicia Parrish, Shawn Lu, Jane Park, Balaji Venkatraman, Alice Talbert, Lambert Rosique, Yuchung Cheng, Andrei Sozanschi, Adam Paszke, Praveen Kumar, Jessica Austin, Lu Li, Khalid Salama, Bartek Perz, Wooyeol Kim, Nandita Dukkipati, Anthony Baryshnikov, Christos Kaplanis, XiangHai Sheng, Yuri Chervonyi, Caglar Unlu, Diego de Las Casas,

Harry Askham, Kathryn Tunyasuvunakool, Felix Gimeno, Siim Poder, Chester Kwak, Matt Miecnikowski, Vahab Mirrokni, Alek Dimitriev, Aaron Parisi, Dangyi Liu, Tomy Tsai, Toby Shevlane, Christina Kouridi, Drew Garmon, Adrian Goedeckemeyer, Adam R. Brown, Anitha Vijayakumar, Ali Elqursh, Sadegh Jazayeri, Jin Huang, Sara Mc Carthy, Jay Hoover, Lucy Kim, Sandeep Kumar, Wei Chen, Courtney Biles, Garrett Bingham, Evan Rosen, Lisa Wang, Qijun Tan, David Engel, Francesco Pongetti, Dario de Cesare, Dongseong Hwang, Lily Yu, Jennifer Pullman, Srini Narayanan, Kyle Levin, Siddharth Gopal, Megan Li, Asaf Aharoni, Trieu Trinh, Jessica Lo, Norman Casagrande, Roopali Vij, Loic Matthey, Bramandia Ramadhana, Austin Matthews, CJ Carey, Matthew Johnson, Kremena Goranova, Rohin Shah, Shereen Ashraf, Kingshuk Dasgupta, Rasmus Larsen, Yicheng Wang, Manish Reddy Vuyyuru, Chong Jiang, Joana Ijazi, Kazuki Osawa, Celine Smith, Ramya Sree Boppana, Taylan Bilal, Yuma Koizumi, Ying Xu, Yasemin Altun, Nir Shabat, Ben Bariach, Alex Korchemniy, Kiam Choo, Olaf Ronneberger, Chimezie Iwuanyanwu, Shubin Zhao, David Soergel, Cho-Jui Hsieh, Irene Cai, Shariq Iqbal, Martin Sundermeyer, Zhe Chen, Elie Bursztein, Chaitanya Malaviya, Fadi Biadsy, Prakash Shroff, Inderjit Dhillon, Tejasi Latkar, Chris Dyer, Hannah Forbes, Massimo Nicosia, Vitaly Nikolaev, Somer Greene, Marin Georgiev, Pidong Wang, Nina Martin, Hanie Sedghi, John Zhang, Praseem Banzal, Doug Fritz, Vikram Rao, Xuezhi Wang, Jiageng Zhang, Viorica Patraucean, Dayou Du, Igor Mordatch, Ivan Jurin, Lewis Liu, Ayush Dubey, Abhi Mohan, Janek Nowakowski, Vlad-Doru Ion, Nan Wei, Reiko Tojo, Maria Abi Raad, Drew A. Hudson, Vaishakh Keshava, Shubham Agrawal, Kevin Ramirez, Zhichun Wu, Hoang Nguyen, Ji Liu, Madhavi Sewak, Bryce Petrini, DongHyun Choi, Ivan Philips, Ziyue Wang, Ioana Bica, Ankush Garg, Jarek Wilkiewicz, Priyanka Agrawal, Xiaowei Li, Danhao Guo, Emily Xue, Naseer Shaik, Andrew Leach, Sadh MNM Khan, Julia Wiesinger, Sammy Jerome, Abhishek Chakladar, Alek Wenjiao Wang, Tina Ornduff, Folake Abu, Alireza Ghaffarkhah, Marcus Wainwright, Mario Cortes, Frederick Liu, Joshua Maynez, Andreas Terzis, Pouya Samangouei, Riham Mansour, Tomasz Kępa, François-Xavier Aubet, Anton Algymr, Dan Banica, Agoston Weisz, Andras Orban, Alexandre Senges, Ewa Andrejczuk, Mark Geller, Niccolo Dal Santo, Valentin Anklin, Majd Al Merey, Martin Baeuml, Trevor Strohman, Junwen Bai, Slav Petrov, Yonghui Wu, Demis Hassabis, Koray Kavukcuoglu, Jeff Dean, and Oriol Vinyals. 2024a. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. 2024b. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*.

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, Chuning Tang, Congcong Wang, Dehao Zhang, Enming Yuan, Enzhe Lu, Fengxiang Tang, Flood Sung, Guangda Wei, Guokun Lai, Haiqing Guo, Han Zhu, Hao Ding, Hao Hu, Hao Yang, Hao Zhang, Haotian Yao, Haotian Zhao, Haoyu Lu, Haoze Li, Haozhen Yu, Hongcheng Gao, Huabin Zheng, Huan Yuan, Jia Chen, Jianhang Guo, Jianlin Su, Jianzhou Wang, Jie Zhao, Jin Zhang, Jingyuan Liu, Junjie Yan, Junyan Wu, Lidong Shi, Ling Ye, Longhui Yu, Mengnan Dong, Neo Zhang, Ningchen Ma, Qiwei Pan, Qucheng Gong, Shaowei Liu, Shengling Ma, Shupeng Wei, Sihan Cao, Siying Huang, Tao Jiang, Weihao Gao, Weimin Xiong, Weiran He, Weixiao Huang, Wenhao Wu, Wenyang He, Xianghui Wei, Xianqing Jia, Xingzhe Wu, Xinran Xu, Xinxing Zu, Xinyu Zhou, Xuehai Pan, Y. Charles, Yang Li, Yangyang Hu, Yangyang Liu, Yanru Chen, Yejie Wang, Yibo Liu, Yidao Qin, Yifeng Liu, Ying Yang, Yiping Bao, Yulun Du, Yuxin Wu, Yuzhi Wang, Zaida Zhou, Zhaoji Wang, Zhaowei Li, Zhen Zhu, Zheng Zhang, Zhexu Wang, Zhilin Yang, Zhiqi Huang, Zihao Huang, Ziyao Xu, and Zonghan Yang. 2025. Kimi k1.5: Scaling reinforcement learning with llms.

Qwen Team. 2024a. Qvq: To see the world with wisdom.

Qwen Team. 2024b. Qwq: Reflect deeply on the boundaries of the unknown.

Ye Tian, Baolin Peng, Linfeng Song, Lifeng Jin, Dian Yu, Haitao Mi, and Dong Yu. 2024a. Toward self-improvement of llms via imagination, searching, and criticizing.

Ye Tian, Baolin Peng, Linfeng Song, Lifeng Jin, Dian Yu, Haitao Mi, and Dong Yu. 2024b. Toward self-improvement of llms via imagination, searching, and criticizing. *arXiv preprint arXiv:2404.12253*.

Manan Tomar, Lior Shani, Yonathan Efroni, and Mohammad Ghavamzadeh. 2020. Mirror descent policy optimization. *arXiv preprint arXiv:2005.09814*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

Alan Wake, Albert Wang, Bei Chen, CX Lv, Chao Li, Chengen Huang, Chenglin Cai, Chujie Zheng, Daniel Cooper, Ethan Dai, et al. 2024. Yi-lightning technical report. *arXiv preprint arXiv:2412.01253*.

Zhen Wan, Fei Cheng, Zhuoyuan Mao, Qianying Liu, Haiyue Song, Jiwei Li, and Sadao Kurohashi. 2023. Gpt-re: In-context learning for relation extraction using large language models. *arXiv preprint arXiv:2305.02105*.

Ke Wang, Junting Pan, Weikang Shi, Zimu Lu, Mingjie Zhan, and Hongsheng Li. 2024a. Measuring multimodal mathematical reasoning with math-vision dataset.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2024b. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9426–9439, Bangkok, Thailand. Association for Computational Linguistics.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. 2024c. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*.

Qineng Wang, Zihao Wang, Ying Su, Hanghang Tong, and Yangqiu Song. 2024d. Rethinking the bounds of llm reasoning: Are multi-agent discussions the key? *arXiv preprint arXiv:2402.18272*.

Shuhe Wang, Beiming Cao, Shengyu Zhang, Xiaoya Li, Jiwei Li, Fei Wu, Guoyin Wang, and Eduard Hovy. 2023a. Sim-gpt: Text similarity via gpt annotated data. *arXiv preprint arXiv:2312.05603*.

Shuhe Wang, Xiaofei Sun, Xiaoya Li, Rongbin Ouyang, Fei Wu, Tianwei Zhang, Jiwei Li, and Guoyin Wang. 2023b. Gpt-ner: Named entity recognition via large language models. *arXiv preprint arXiv:2304.10428*.

Shuhe Wang, Shengyu Zhang, Jie Zhang, Runyi Hu, Xiaoya Li, Tianwei Zhang, Jiwei Li, Fei Wu, Guoyin Wang, and Eduard Hovy. 2024e. Reinforcement learning enhanced llms: A survey. *arXiv preprint arXiv:2412.10400*.

Tianduo Wang and Wei Lu. 2023. Learning multi-step reasoning by solving arithmetic tasks. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1229–1238.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022. Self-instruct: Aligning language models with self-generated instructions. *arXiv preprint arXiv:2212.10560*.

Zilong Wang, Hao Zhang, Chun-Liang Li, Julian Martin Eisenschlos, Vincent Perot, Zifeng Wang, Lesly Miculicich, Yasuhisa Fujii, Jingbo Shang, Chen-Yu Lee, and Tomas Pfister. 2024f. Chain-of-table: Evolving tables in the reasoning chain for table understanding. *ICLR*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models.

Penghao Wu and Saining Xie. 2023. V*: Guided visual search as a core mechanism in multimodal llms. *arXiv preprint arXiv:2312.14135*.

Yangzhen Wu, Zhiqing Sun, Shanda Li, Sean Welleck, and Yiming Yang. 2024. Inference scaling laws: An empirical analysis of compute-optimal inference for problem-solving with language models. *arXiv preprint arXiv:2408.00724*.

Zhiheng Xi, Dingwen Yang, Jixuan Huang, Jiafu Tang, Guanyu Li, Yiwen Ding, Wei He, Boyang Hong, Shihan Do, Wenyu Zhan, et al. 2024. Enhancing llm reasoning via critique models with test-time and training-time supervision. *arXiv preprint arXiv:2411.16579*.

Wenyi Xiao, Zechuan Wang, Leilei Gan, Shuai Zhao, Wanggui He, Luu Anh Tuan, Long Chen, Hao Jiang, Zhou Zhao, and Fei Wu. 2024. A comprehensive survey of direct preference optimization: Datasets, theories, variants, and applications.

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

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. 2023. Wizardlm: Empowering large language models to follow complex instructions. *arXiv preprint arXiv:2304.12244*.

Guowei Xu, Peng Jin, Hao Li, Yibing Song, Lichao Sun, and Li Yuan. 2024. Llava-cot: Let vision language models reason step-by-step.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. 2024a. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Jingfeng Yang, Hongye Jin, Ruixiang Tang, Xiaotian Han, Qizhang Feng, Haoming Jiang, Bing Yin, and Xia Hu. 2023. Harnessing the power of llms in practice: A survey on chatgpt and beyond. *arXiv preprint arXiv:2304.13712*.

Sohee Yang, Elena Gribovskaya, Nora Kassner, Mor Geva, and Sebastian Riedel. 2024b. Do large language models latently perform multi-hop reasoning? *arXiv preprint arXiv:2402.16837*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2024. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36.

Fei Yu, Anningzhe Gao, and Benyou Wang. 2023. Outcome-supervised verifiers for planning in mathematical reasoning. *arXiv preprint arXiv:2311.09724*.

Lifan Yuan, Ganqu Cui, Hanbin Wang, Ning Ding, Xingyao Wang, Jia Deng, Boji Shan, Huimin Chen, Ruobing Xie, Yankai Lin, et al. 2024. Advancing llm reasoning generalists with preference trees. *arXiv preprint arXiv:2404.02078*.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. 2022. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35:15476–15488.

Zhiyuan Zeng, Qinyuan Cheng, Zhangyue Yin, Bo Wang, Shimin Li, Yunhua Zhou, Qipeng Guo, Xuanjing Huang, and Xipeng Qiu. 2024. Scaling of search and learning: A roadmap to reproduce o1 from reinforcement learning perspective.

Dan Zhang, Sining Zhoubian, Ziniu Hu, Yisong Yue, Yuxiao Dong, and Jie Tang. 2024a. Rest-mcts*: Llm self-training via process reward guided tree search. *arXiv preprint arXiv:2406.03816*.

Di Zhang, Xiaoshui Huang, Dongzhan Zhou, Yuqiang Li, and Wanli Ouyang. 2024b. Accessing gpt-4 level mathematical olympiad solutions via monte carlo tree self-refine with llama-3 8b. *arXiv preprint arXiv:2406.07394*.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, et al. 2023. Instruction tuning for large language models: A survey. *arXiv preprint arXiv:2308.10792*.

Xuan Zhang, Chao Du, Tianyu Pang, Qian Liu, Wei Gao, and Min Lin. 2024c. Chain of preference optimization: Improving chain-of-thought reasoning in llms.

Yuxiang Zhang, Shangxi Wu, Yuqi Yang, Jiangming Shu, Jinlin Xiao, Chao Kong, and Jitao Sang. 2024d. o1-coder: an o1 replication for coding. *arXiv preprint arXiv:2412.00154*.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025. The lessons of developing process reward models in mathematical reasoning.

Yu Zhao, Huifeng Yin, Bo Zeng, Hao Wang, Tianqi Shi, Chenyang Lyu, Longyue Wang, Weihua Luo, and Kaifu Zhang. 2024a. Marco-o1: Towards open reasoning models for open-ended solutions. *arXiv preprint arXiv:2411.14405*.

Yu Zhao, Huifeng Yin, Bo Zeng, Hao Wang, Tianqi Shi, Chenyang Lyu, Longyue Wang, Weihua Luo, and Kaifu Zhang. 2024b. Marco-o1: Towards open reasoning models for open-ended solutions.

Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2024. Processbench: Identifying process errors in mathematical reasoning.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, L. Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023. Lima: Less is more for alignment. *ArXiv*, abs/2305.11206.