Large reasoning models & Test-time Scaling (cont'd)

CS 4804: Introduction to AI

Fall 2025

https://tuvllms.github.io/ai-fall-2025/

Tu Vu

VIRGINIA TECH

Logistics

- Final project proposal feedback: next week
- Final presentations: 12/4 & 12/9
- Teaching & learning evaluation: 11/4
- Quiz 2 & HW 2: postponed to after fall break

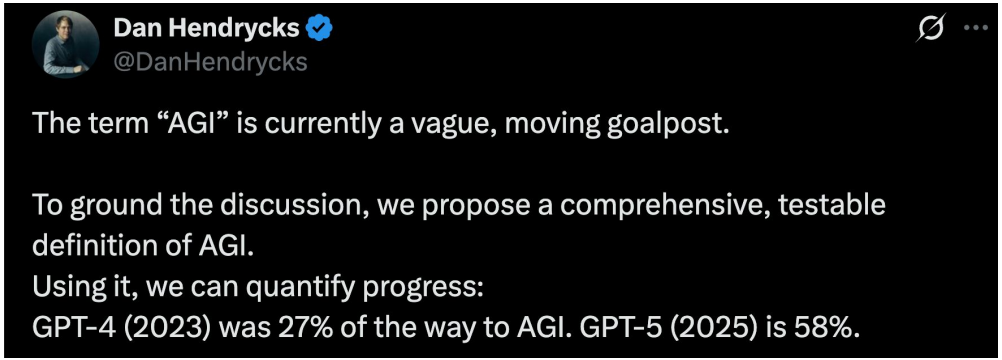
A few hours ago

A Definition of AGI

Dan Hendrycks[1], Dawn Song[2], Christian Szegedy[3], Honglak Lee[4], Yarin Gal[5], Sharon Li[6], Andy Zou[1,7,8], Lionel Levine[9], Bo Han[10], Jie Fu[11], Ziwei Liu[12], Jinwoo Shin[13], Kimin Lee[13], Mantas Mazeika[1], Long Phan[1], George Ingebretsen[1], Adam Khoja[1], Cihang Xie[14], Olawale Salaudeen[15], Matthias Hein[16], Kevin Zhao[17], Alex Pan[2], David Duvenaud[18,19], Bo Li[20], Steve Omohundro[21], Gabriel Alfour[22], Max Tegmark[15], Kevin McGrew[23], Gary Marcus[24], Jaan Tallinn[25], Eric Schmidt[15], Yoshua Bengio[26,27]

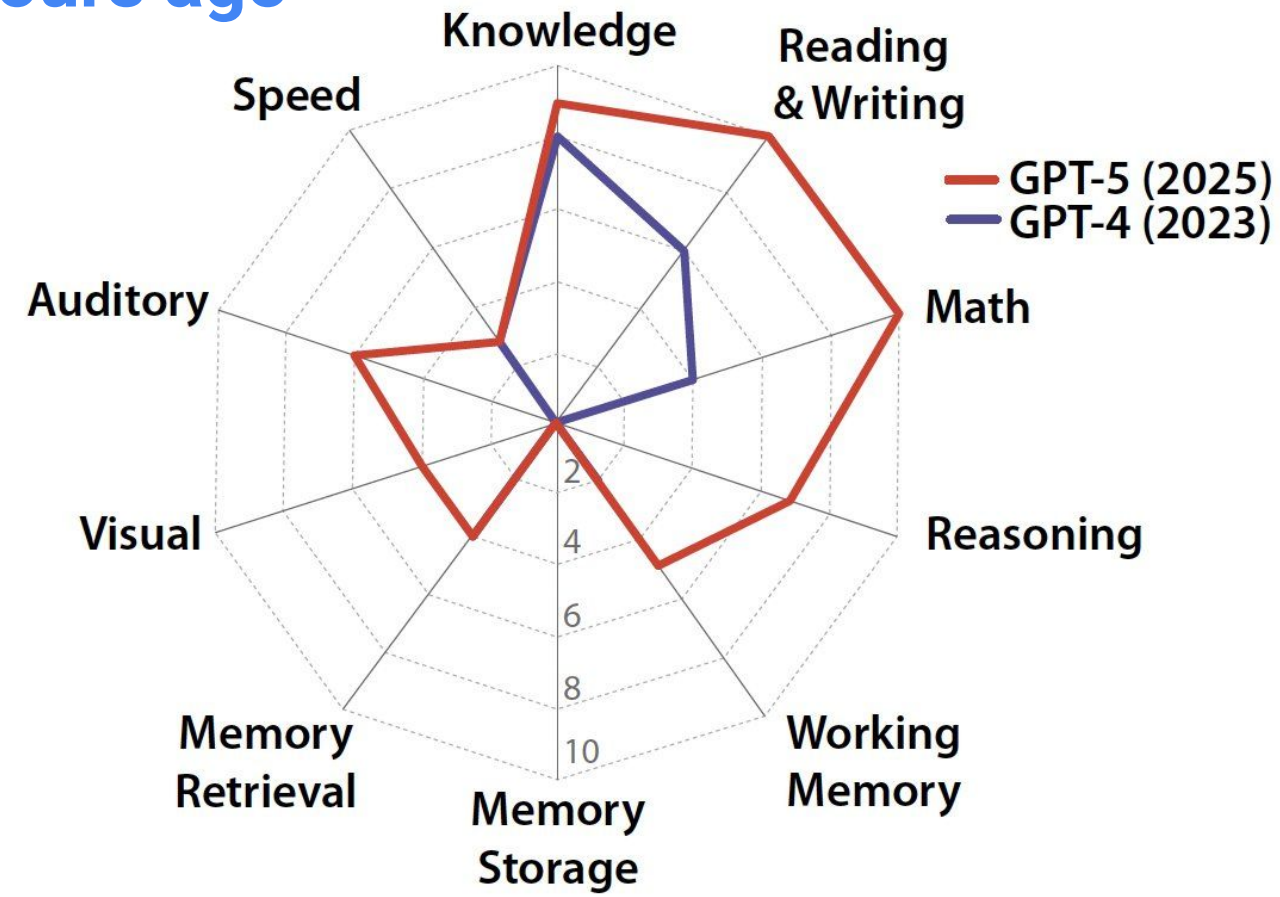
[1]Center for AI Safety [2]University of California, Berkeley [3]Morph Labs
[4]University of Michigan [5]University of Oxford [6]University of Wisconsin–Madison
[7]Gray Swan AI [8]Carnegie Mellon University [9]Cornell University
[10]Hong Kong Baptist University [11]HKUST [12]Nanyang Technological University [13]KAIST
[14]University of California, Santa Cruz [15]Massachusetts Institute of Technology
[16]University of Tübingen [17]University of Washington [18]University of Toronto
[19]Vector Institute [20]University of Chicago [21]Beneficial AI Research
[22]Conjecture [23]Institute for Applied Psychometrics [24]New York University
[25]CSER [26]Université de Montréal [27]LawZero

A few hours ago



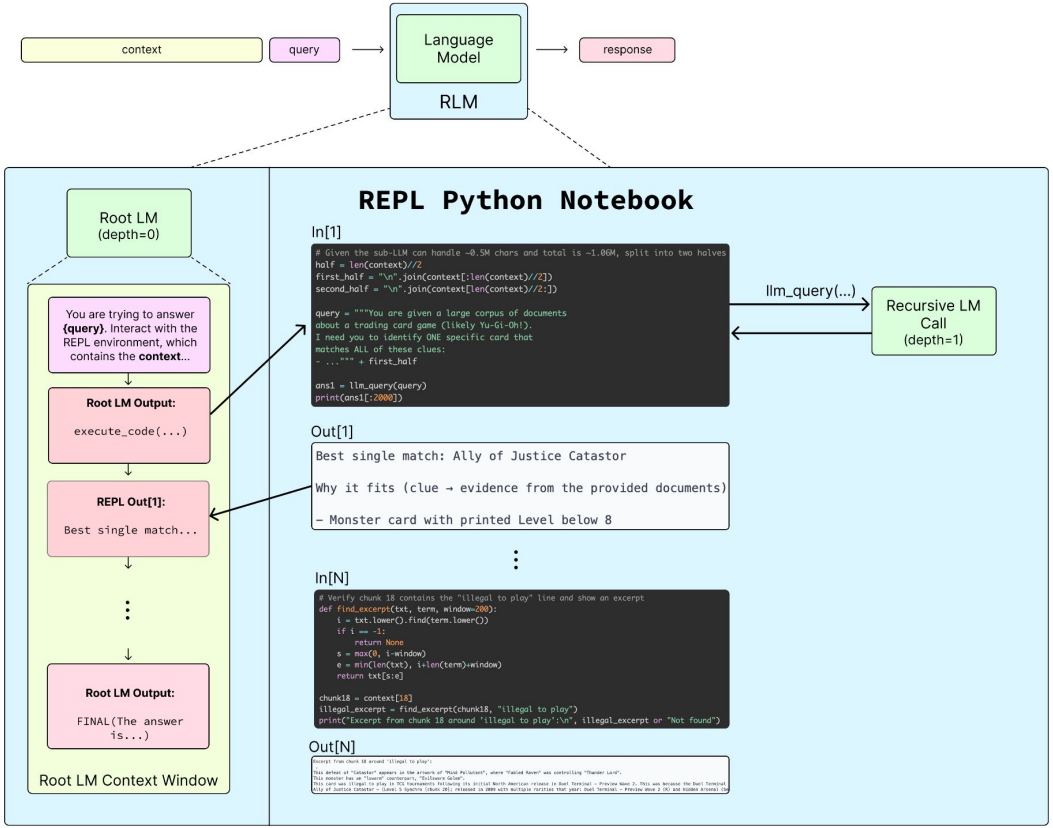
The lack of a concrete definition for Artificial General Intelligence (AGI) obscures the gap between today’s specialized AI and human-level cognition. This paper introduces a quantifiable framework to address this, defining AGI as matching the cognitive versatility and proficiency of a well-educated adult. To operationalize this, we ground our methodology in Cattell-Horn-Carroll theory, the most empirically validated model of human cognition. The framework dissects general intelligence into ten core cognitive domains—including reasoning, memory, and perception—and adapts established human psychometric batteries to evaluate AI systems. Application of this framework reveals a highly “jagged” cognitive profile in contemporary models. While proficient in knowledge-intensive domains, current AI systems have critical deficits in foundational cognitive machinery, particularly long-term memory storage. The resulting AGI scores (e.g., GPT-4 at 27%, GPT-5 at 58%) concretely quantify both rapid progress and the substantial gap remaining before AGI.

A few hours ago



Recursive Language Models

RLM with a REPL environment:



https://alexzhang13.github.io/blog/2025/rlm/

Recursive Language Models (cont'd)

Input: a long report of 10,000 words about a company's 5-year strategy
Query: What risks did the report identify in year 3?

AI: High-level scan / peek at the report's table of contents

```
# find the header line number
hdr=$(grep -inE '^(table of contents|contents|toc)\s*$' doc.txt | head -1 | cut -d: -f1)
# print the next 100 lines that look like TOC entries
tail -n +"$hdr" doc.txt | head -n 100 | grep -nE '(^\s*\d+(\.\d+)*\s+.+)|(^\s*.+\s+\.{2,}\s*\d+\s*$)'
```

Select sub-chunks
Based on that peek, it might decide: "Chapter 4 (pages 40–60) and Chapter 6 (pages 90–110) are likely to mention Year 3 risks."
Recursive call on chunks It then issues sub-queries:
"From pages 40–60, what risks are mentioned for year 3?"
"From pages 90–110, what risks are mentioned for year 3?"
Aggregate / refine
"In year 3 the report warned of supply chain disruption, regulatory changes, and declining demand in Asia."

Recursive Language Models (cont'd)

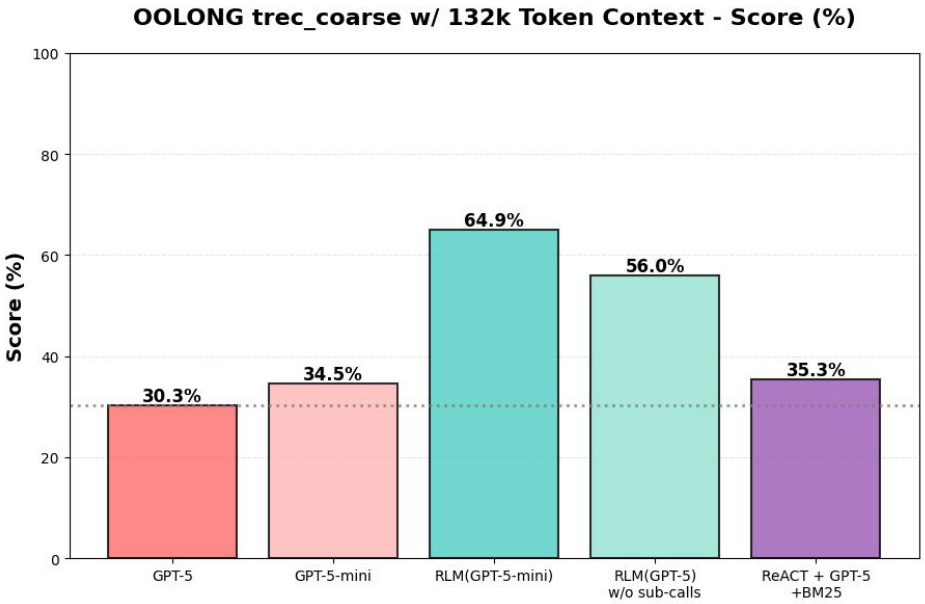
Input: a long report of 10,000 words about a company's 5-year strategy
Query: What risks did the report identify in year 3?

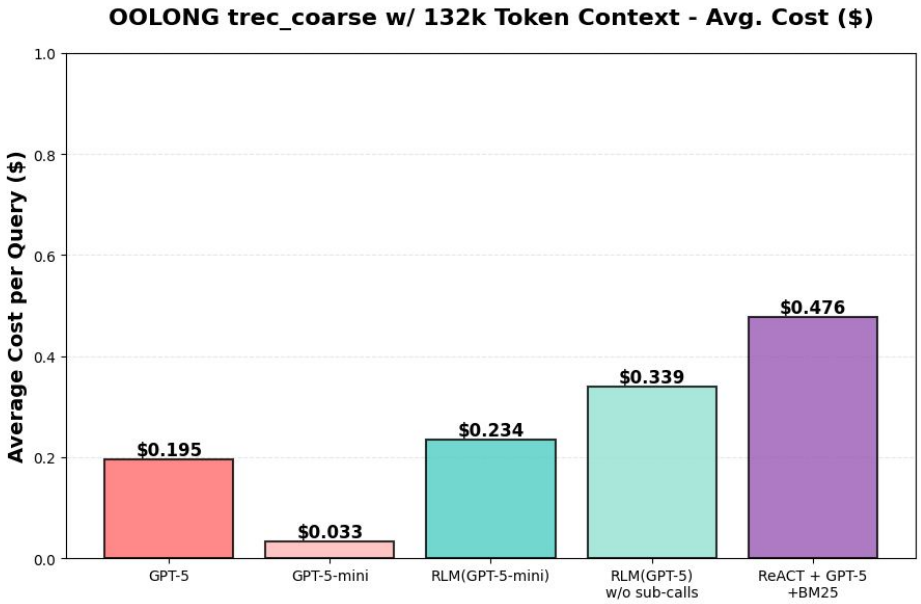
AI: High-level scan / peek at the report's table of contents

```
# find the header line number
hdr=$(grep -inE '^(table of contents|contents|toc)\s*$' doc.txt | head -1 | cut -d: -f1)
# print the next 100 lines that look like TOC entries
tail -n +"$hdr" doc.txt | head -n 100 | grep -nE '(^\s*\d+(\.\d+)*\s+.+)|(^\s*.+\s+\.{2,}\s*\d+\s*$)'
```

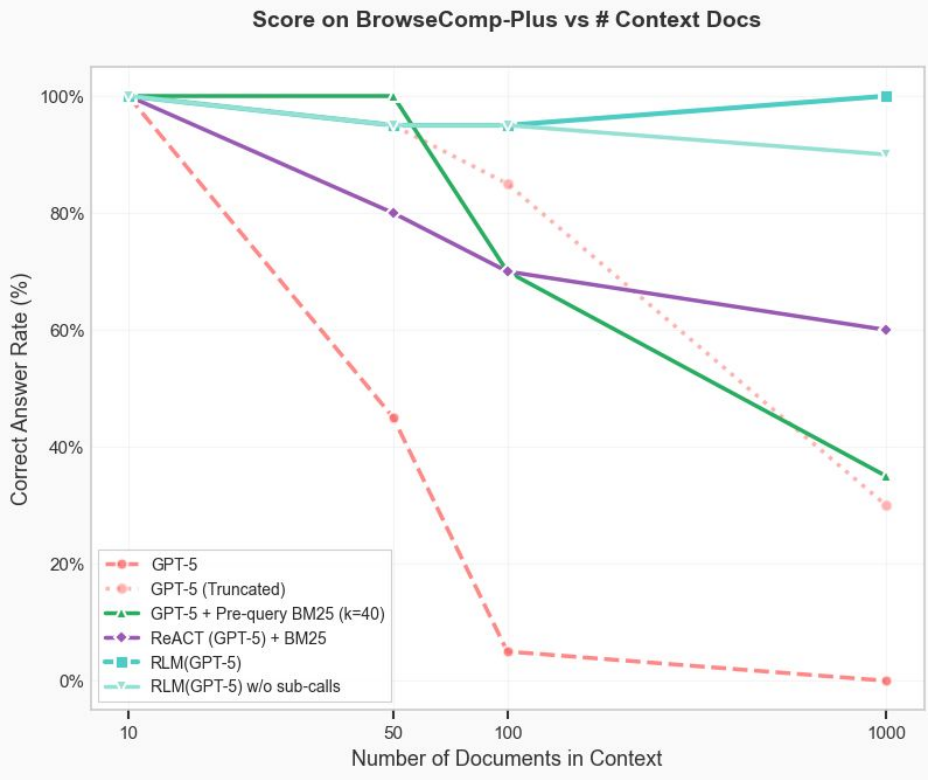
Select sub-chunks
Based on that peek, it might decide: "Chapter 4 (pages 40–60) and Chapter 6 (pages 90–110) are likely to mention Year 3 risks."
Recursive call on chunks It then issues sub-queries:
"From pages 40–60, what risks are mentioned for year 3?"
"From pages 90–110, what risks are mentioned for year 3?"
Aggregate / refine
"In year 3 the report warned of supply chain disruption, regulatory changes, and declining demand in Asia."

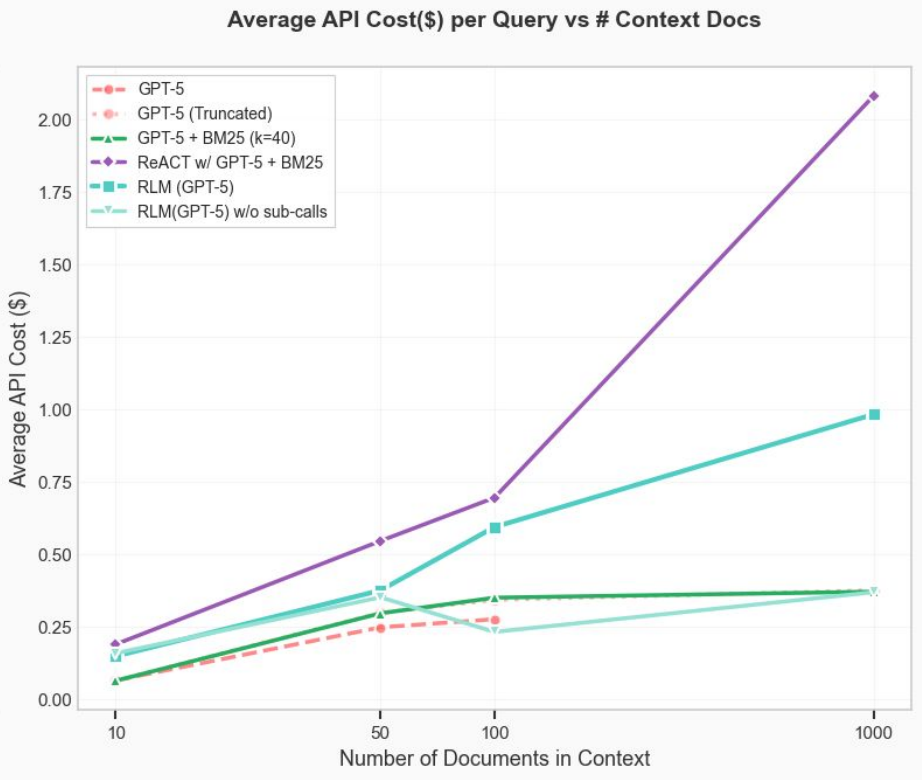
Recursive Language Models (cont'd)





Recursive Language Models (cont'd)



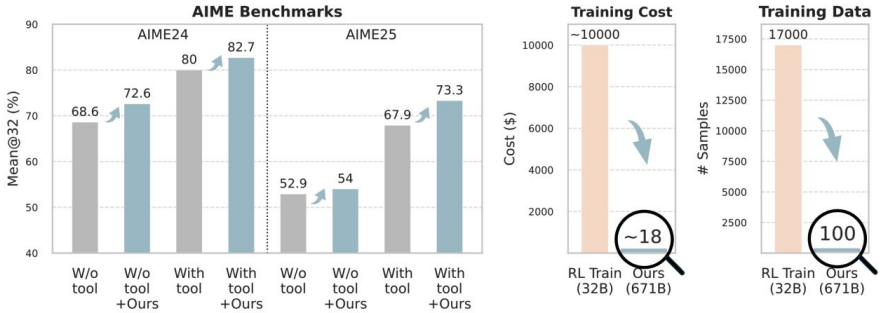


Training-Free Group Relative Policy Optimization

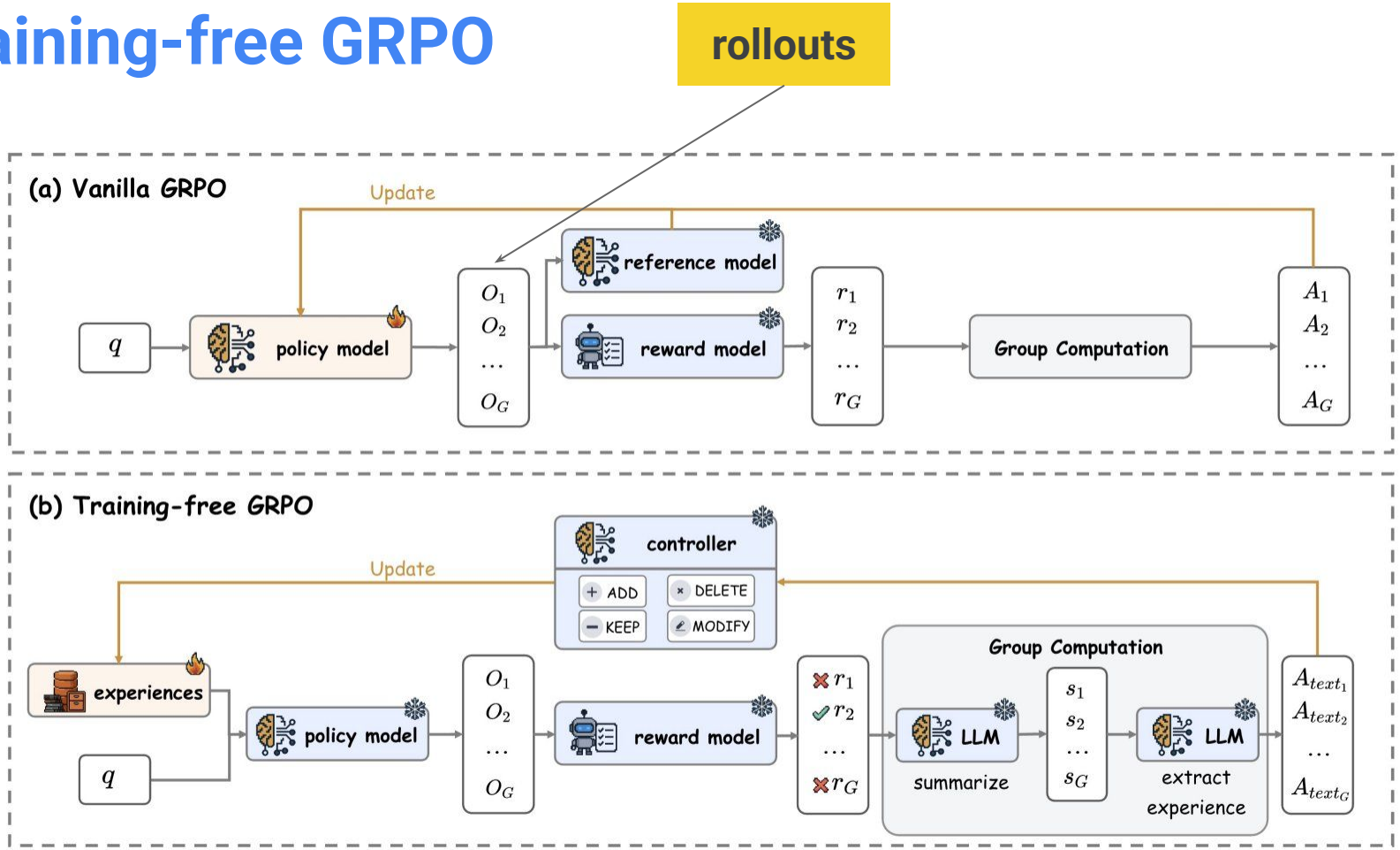
*Youtu-Agent Team**

Recent advances in Large Language Model (LLM) agents have demonstrated their promising general capabilities. However, their performance in specialized real-world domains often degrades due to challenges in effectively integrating external tools and specific prompting strategies. While methods like agentic reinforcement learning have been proposed to address this, they typically rely on costly parameter updates, for example, through a process that uses Supervised Fine-Tuning (SFT) followed by a Reinforcement Learning (RL) phase with Group Relative Policy Optimization (GRPO) to alter the output distribution. However, we argue that LLMs can achieve a similar effect on the output distribution by learning experiential knowledge as a token prior, which is a far more lightweight approach that not only addresses practical data scarcity but also avoids the common issue of overfitting. To this end, we propose Training-Free Group Relative Policy Optimization (Training-Free GRPO), a cost-effective solution that enhances LLM agent performance without any parameter updates. Our method leverages the group relative semantic advantage instead of numerical ones within each group of rollouts, iteratively distilling high-quality experiential knowledge during multi-epoch learning on a minimal ground-truth data. Such knowledge serves as the learned token prior, which is seamlessly integrated during LLM API calls to guide model behavior. Experiments on mathematical reasoning and web searching tasks demonstrate that Training-Free GRPO, when applied to DeepSeek-V3.1-Terminus, significantly improves out-of-domain performance. With just a few dozen training samples, Training-Free GRPO outperforms fine-tuned small LLMs with marginal training data and cost.

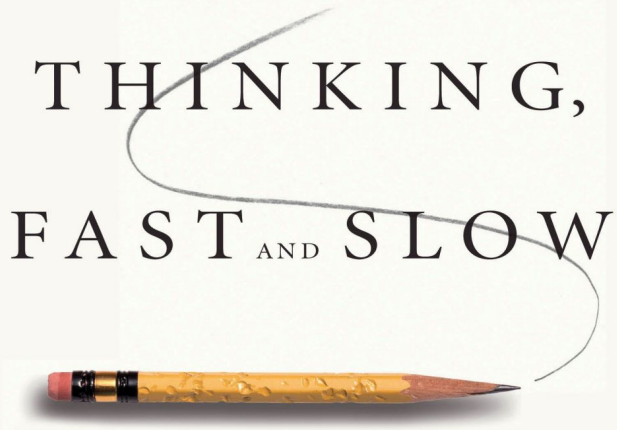
Date: October 9, 2025
Correspondence: tristanli@tencent.com
Code: https://github.com/TencentCloudADP/youtu-agent/tree/training_free_GRPO



Training-free GRPO



THINKING, FAST AND SLOW

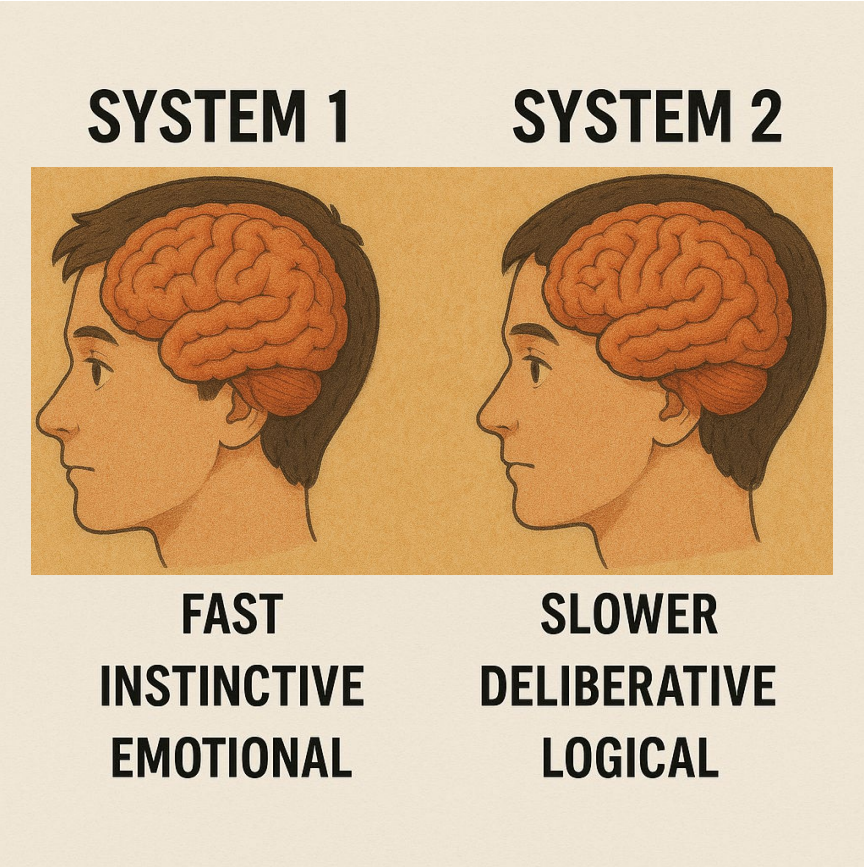


DANIEL KAHNEMAN

WINNER OF THE NOBEL PRIZE IN ECONOMICS

System 1 & System 2

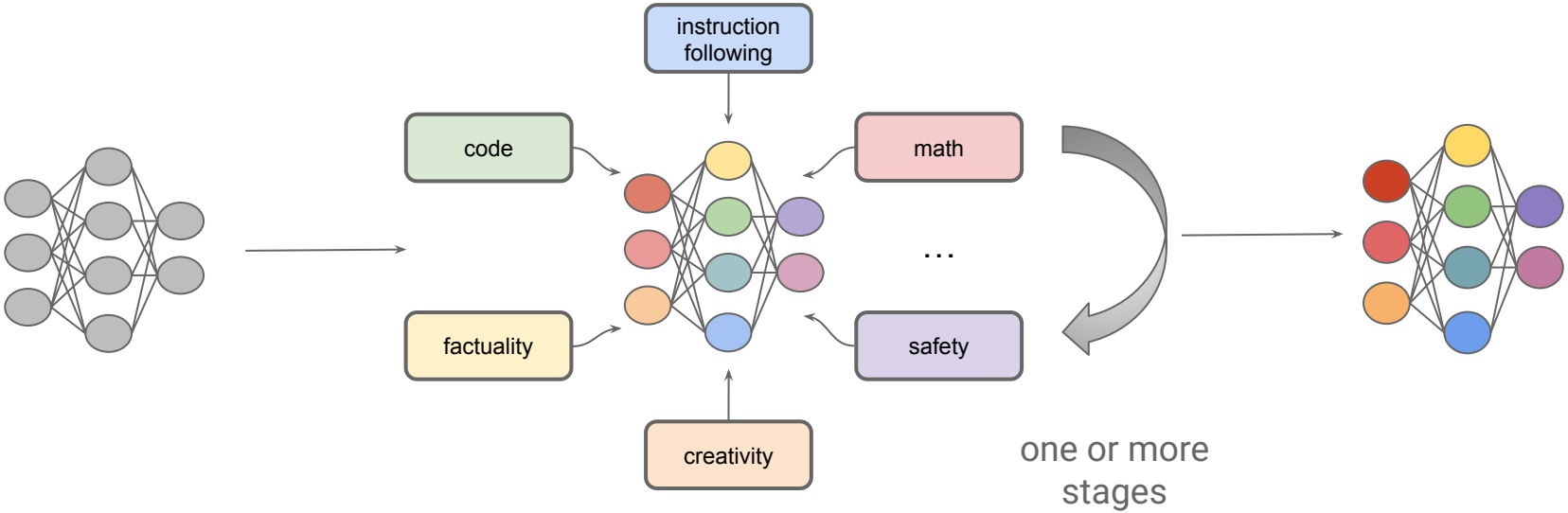
Which is better?



ChatGPT image

From System 1 to System 2

RL vs. SFT: RL allows the model to encode the subtle nuances of human preferences



System 1

Supervised Fine-tuning and/or Reinforcement Learning on long Chain-of-Thought data

System 2

Training reasoning models via reinforcement learning

Similar to how a human may think for a long time before responding to a difficult question, o1 uses a chain of thought when attempting to solve a problem. Through reinforcement learning, o1 learns to hone its chain of thought and refine the strategies it uses. It learns to recognize and correct its mistakes. It learns to break down tricky steps into simpler ones. It learns to try a different approach when the current one isn't working. This process dramatically improves the model's ability to reason.

DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning

DeepSeek-AI

research@deepseek.com

Abstract

We introduce our first-generation reasoning models, DeepSeek-R1-Zero and DeepSeek-R1. DeepSeek-R1-Zero, a model trained via large-scale reinforcement learning (RL) without supervised fine-tuning (SFT) as a preliminary step, demonstrates remarkable reasoning capabilities. Through RL, DeepSeek-R1-Zero naturally emerges with numerous powerful and intriguing reasoning behaviors. However, it encounters challenges such as poor readability, and language mixing. To address these issues and further enhance reasoning performance, we introduce DeepSeek-R1, which incorporates multi-stage training and cold-start data before RL. DeepSeek-R1 achieves performance comparable to OpenAI-o1-1217 on reasoning tasks. To support the research community, we open-source DeepSeek-R1-Zero, DeepSeek-R1, and six dense models (1.5B, 7B, 8B, 14B, 32B, 70B) distilled from DeepSeek-R1 based on Qwen and Llama.

Reinforcement Learning from Verifiable Rewards (RLVR)

The reward is the source of the training signal, which decides the optimization direction of RL. To train DeepSeek-R1-Zero, we adopt a rule-based reward system that mainly consists of two types of rewards:

- **Accuracy rewards**: The accuracy reward model evaluates whether the response is correct. For example, in the case of math problems with deterministic results, the model is required to provide the final answer in a specified format (e.g., within a box), enabling reliable rule-based verification of correctness. Similarly, for LeetCode problems, a compiler can be used to generate feedback based on predefined test cases.
- **Format rewards**: In addition to the accuracy reward model, we employ a format reward model that enforces the model to put its thinking process between '<think>' and '</think>' tags.

Guided Chain-of-Thought (CoT) template

A conversation between User and Assistant. The user asks a question, and the Assistant solves it. The assistant first thinks about the reasoning process in the mind and then provides the user with the answer. The reasoning process and answer are enclosed within <think> </think> and <answer> </answer> tags, respectively, i.e., <think> reasoning process here </think> <answer> answer here </answer>. User: prompt. Assistant:

Group Relative Policy Optimization (GRPO) (cont'd)

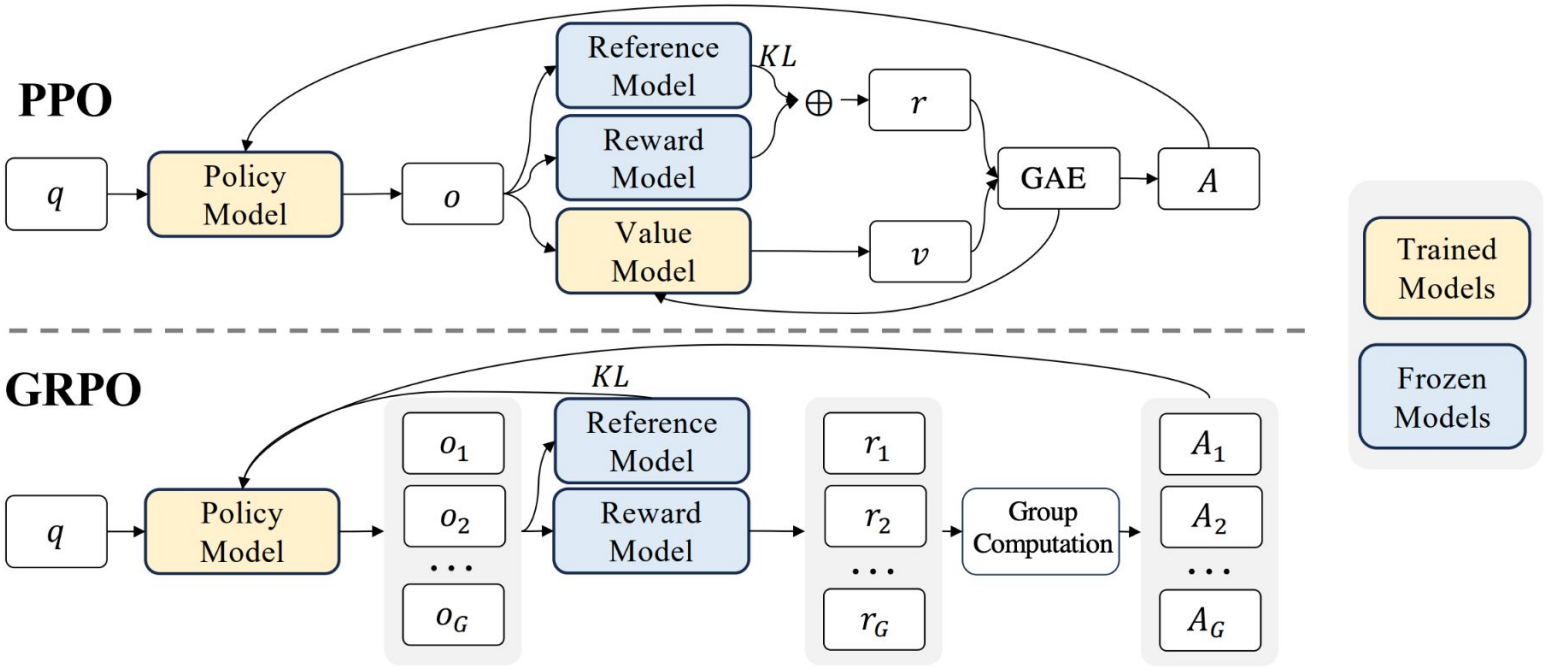


Figure 4 | Demonstration of PPO and our GRPO. GRPO foregoes the value model, instead estimating the baseline from group scores, significantly reducing training resources.

An interesting “aha moment” of DeepSeek-R1-Zero

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:

$(a - x^2)^2 = a + x \implies a^2 - 2ax^2 + (x^2)^2 = a + x \implies x^4 - 2ax^2 - x + (a^2 - a) = 0$

$\cdots$

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:

$\sqrt{a - \sqrt{a + x}} = x$

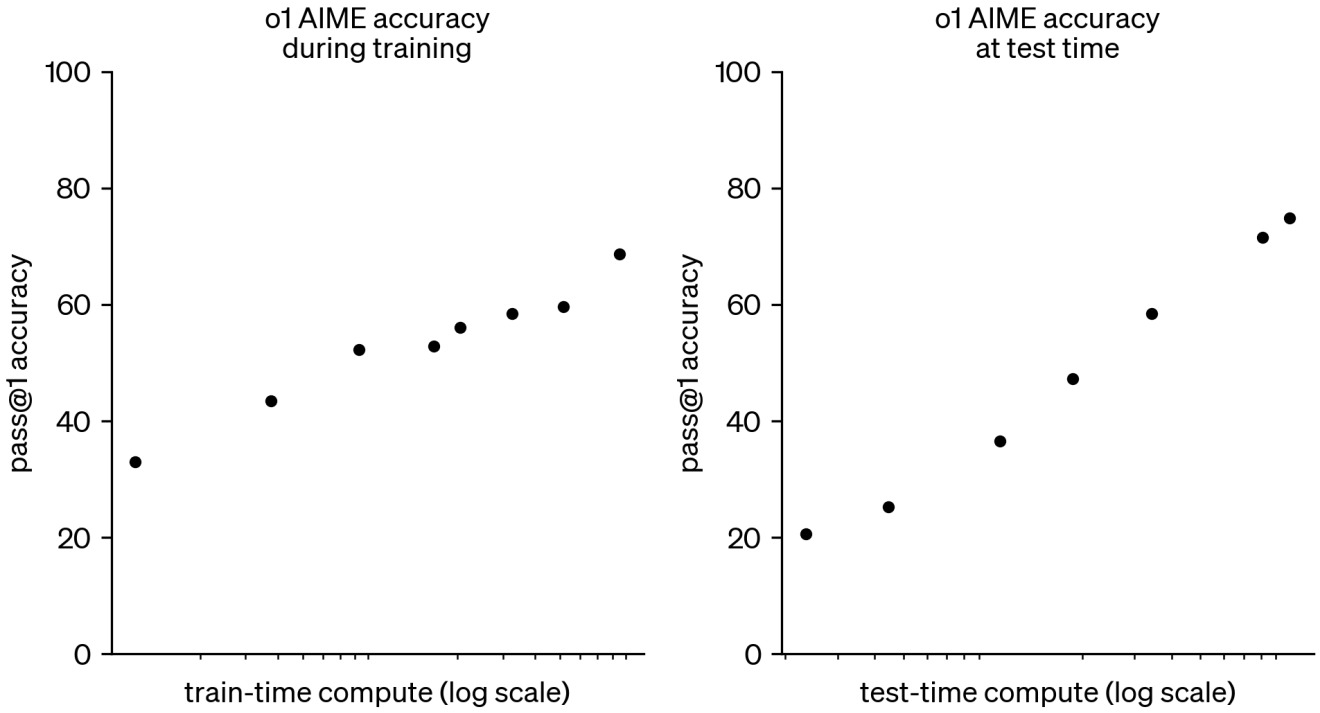
First, let's square both sides:

$a - \sqrt{a + x} = x^2 \implies \sqrt{a + x} = a - x^2$

Next, I could square both sides again, treating the equation: $\cdots$

$\cdots$

o1 performance smoothly improves with both train-time and test-time compute



https://openai.com/index/learning-to-reason-with-llms/

Test-time scaling

- Uses extra test-time compute to improve performance

Discussion: pros and cons of test-time scaling

-

Test-time scaling methods

- **Parallel (repeated sampling)**
  - multiple solution attempts (run independently)
  - chooses the most frequent or the best response
  - Brown et al. (2024); Irvine et al. (2023); Levi (2024)
- **Sequential**
  - later computations depend on earlier ones (e.g., a long reasoning trace)
  - allows it to refine each attempt based on previous outcomes
  - Muennighoff et al.(2025); Snell et al. (2024); Hou et al. (2025); Lee et al. (2025)

Large Language Monkeys: Scaling Inference Compute with Repeated Sampling

Bradley Brown*†‡, Jordan Juravsky*†, Ryan Ehrlich*†, Ronald Clark‡, Quoc V. Le§, Christopher Ré†, and Azalia Mirhoseini†§

†Department of Computer Science, Stanford University
‡University of Oxford
§Google DeepMind

bradley.brown@cs.ox.ac.uk, jbj@stanford.edu, ryanehrlich@cs.stanford.edu, ronald.clark@cs.ox.ac.uk, qvl@google.com, chrismre@stanford.edu, azalia@stanford.edu

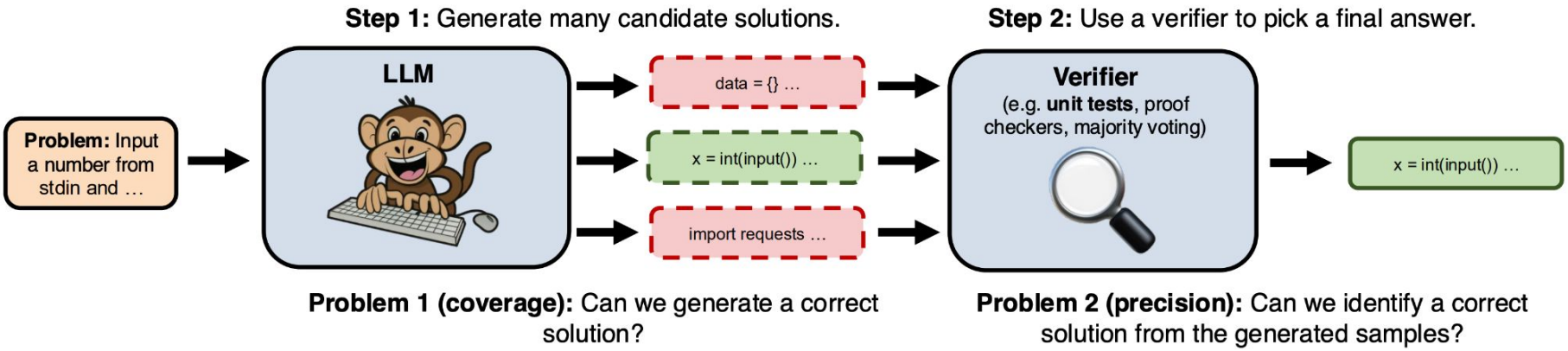


Figure 1: The repeated sampling procedure that we follow in this paper. 1) We generate many independent candidate solutions for a given problem by sampling from an LLM with a positive temperature. 2) We use a domain-specific verifier (ex. unit tests for code) to select a final answer from the generated samples.

Coverage increases as we scale the number of samples

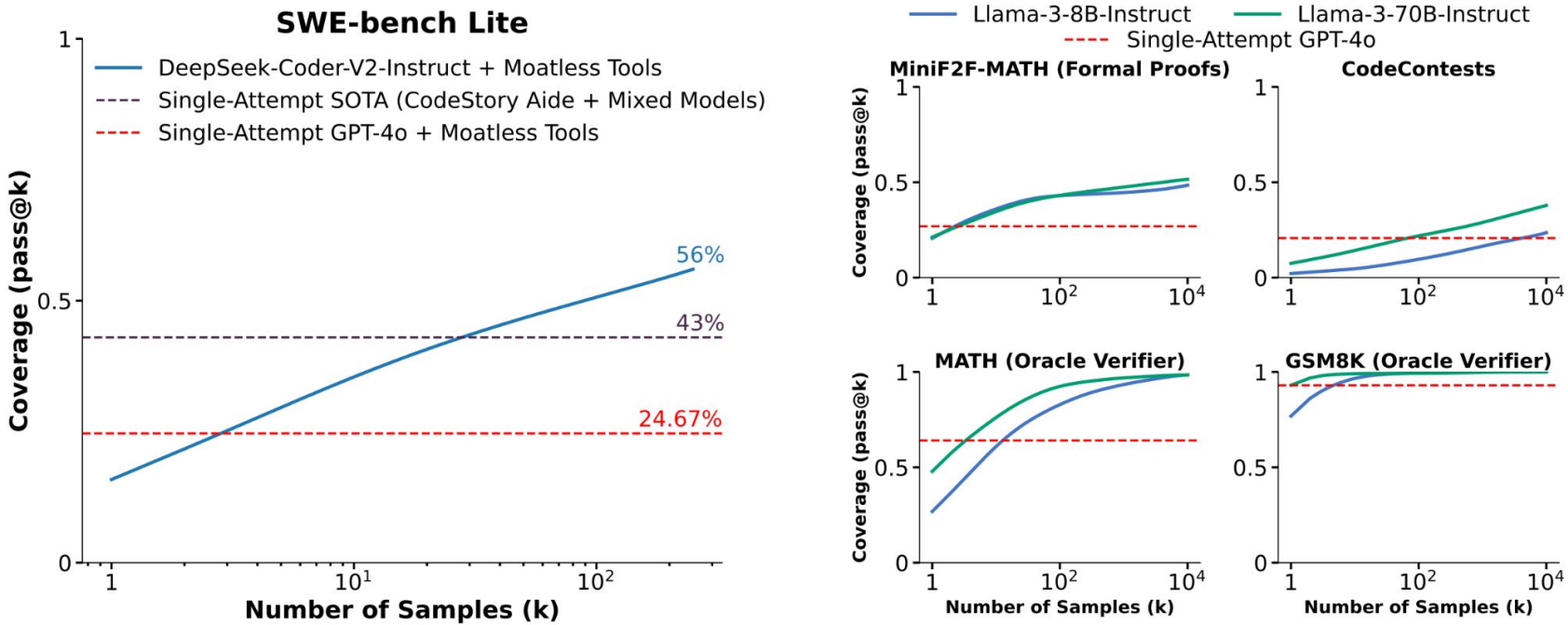


Figure 2: Across five tasks, we find that coverage (the fraction of problems solved by at least one generated sample) increases as we scale the number of samples. Notably, using repeated sampling, we are able to increase the solve rate of an open-source method from 15.9% to 56% on SWE-bench Lite.

Scaling inference time compute via repeated sampling leads to consistent coverage gains

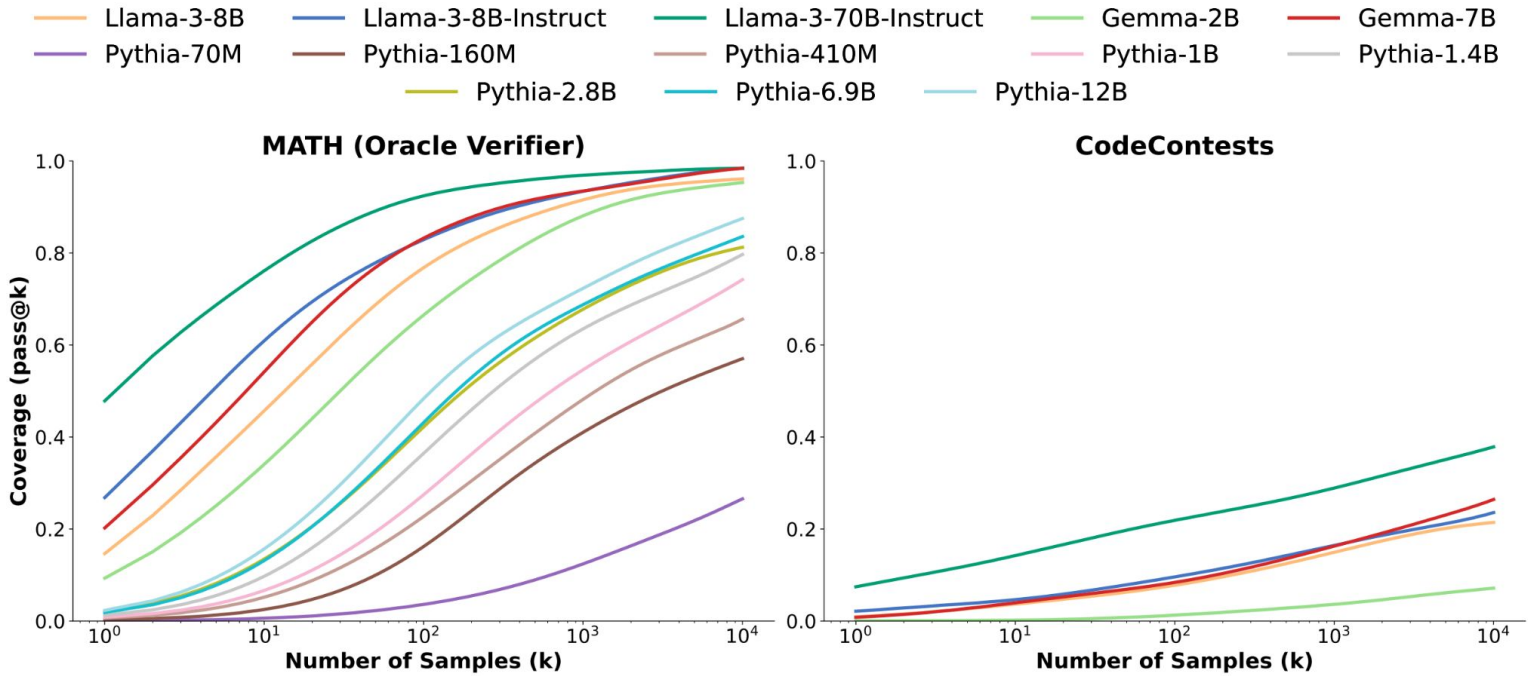


Figure 3: Scaling inference time compute via repeated sampling leads to consistent coverage gains across a variety of model sizes (70M-70B), families (Llama, Gemma and Pythia) and levels of post-training (Base and Instruct models).

API cost

Model	Cost per attempt (USD)	Number of attempts	Issues solved (%)	Total cost (USD)	Relative total cost
DeepSeek-Coder-V2-Instruct	0.0072	5	29.62	10.8	1x
GPT-4o	0.13	1	24.00	39	3.6x
Claude 3.5 Sonnet	0.17	1	26.70	51	4.7x

Table 1: Comparing API cost (in US dollars) and performance for various models on the SWE-bench Lite dataset using the Moatless Tools agent framework. When sampled more, the open-source DeepSeek-Coder-V2-Instruct model can achieve the same issue solve-rate as closed-source frontier models for under a third of the price.

s1: Simple test-time scaling

Niklas Muennighoff [*1 3 4] **Zitong Yang** [*1] **Weijia Shi** [*2 3] **Xiang Lisa Li** [*1] **Li Fei-Fei** [1] **Hannaneh Hajishirzi** [2 3]
Luke Zettlemoyer [2] **Percy Liang** [1] **Emmanuel Candès** [1] **Tatsunori Hashimoto** [1]

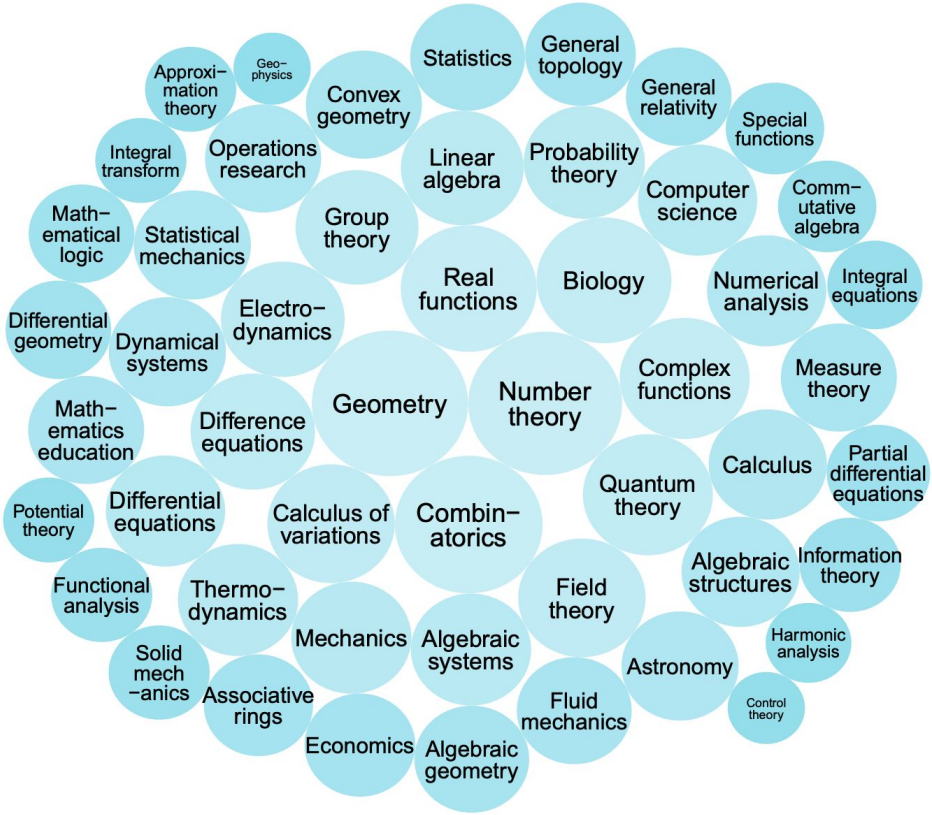
s1

We seek the simplest approach to achieve test-time scaling and strong reasoning performance

- First, we curate a small dataset s1K of 1,000 questions paired with reasoning traces relying on three criteria we validate through ablations: difficulty, diversity, and quality.

- Second, we develop budget forcing to control test-time compute by forcefully terminating the model's thinking process or lengthening it by appending "Wait" multiple times to the model's generation when it tries to end. This can lead the model to double check its answer, often fixing incorrect reasoning steps.

s1K is a dataset of 1,000 high-quality, diverse, and difficult questions with reasoning traces.



Test-time scaling with s1-32B

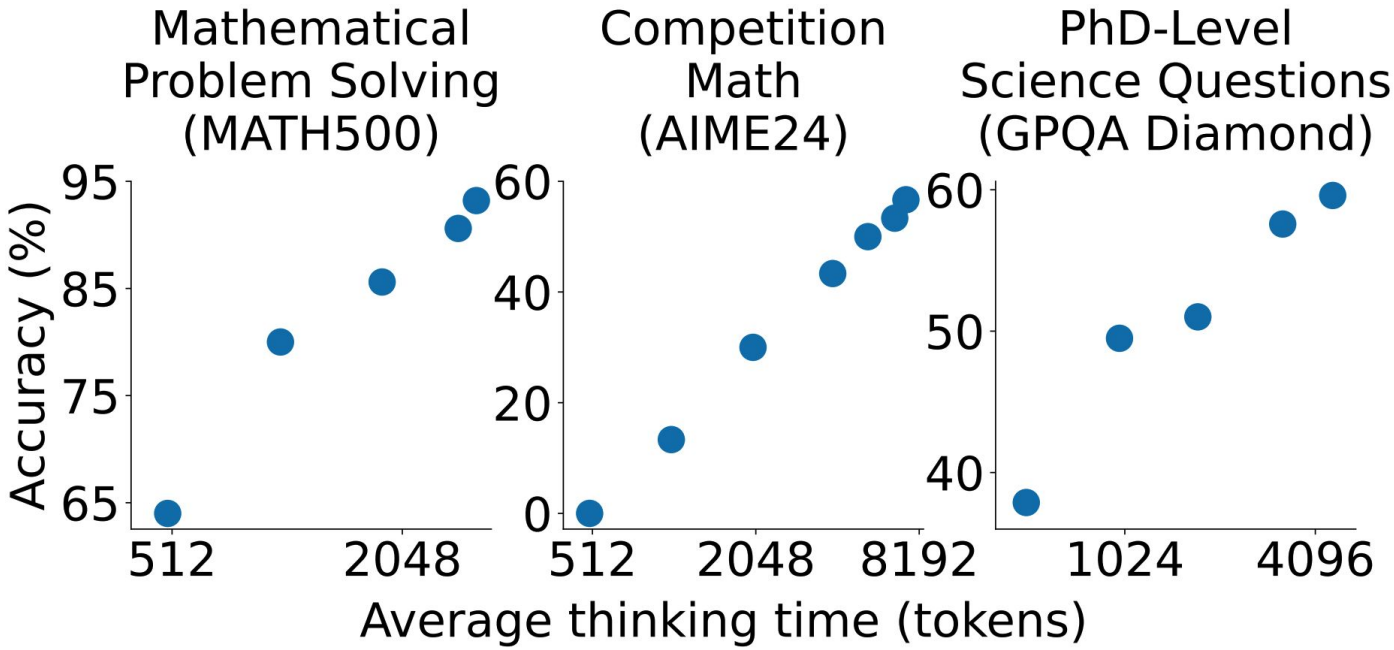
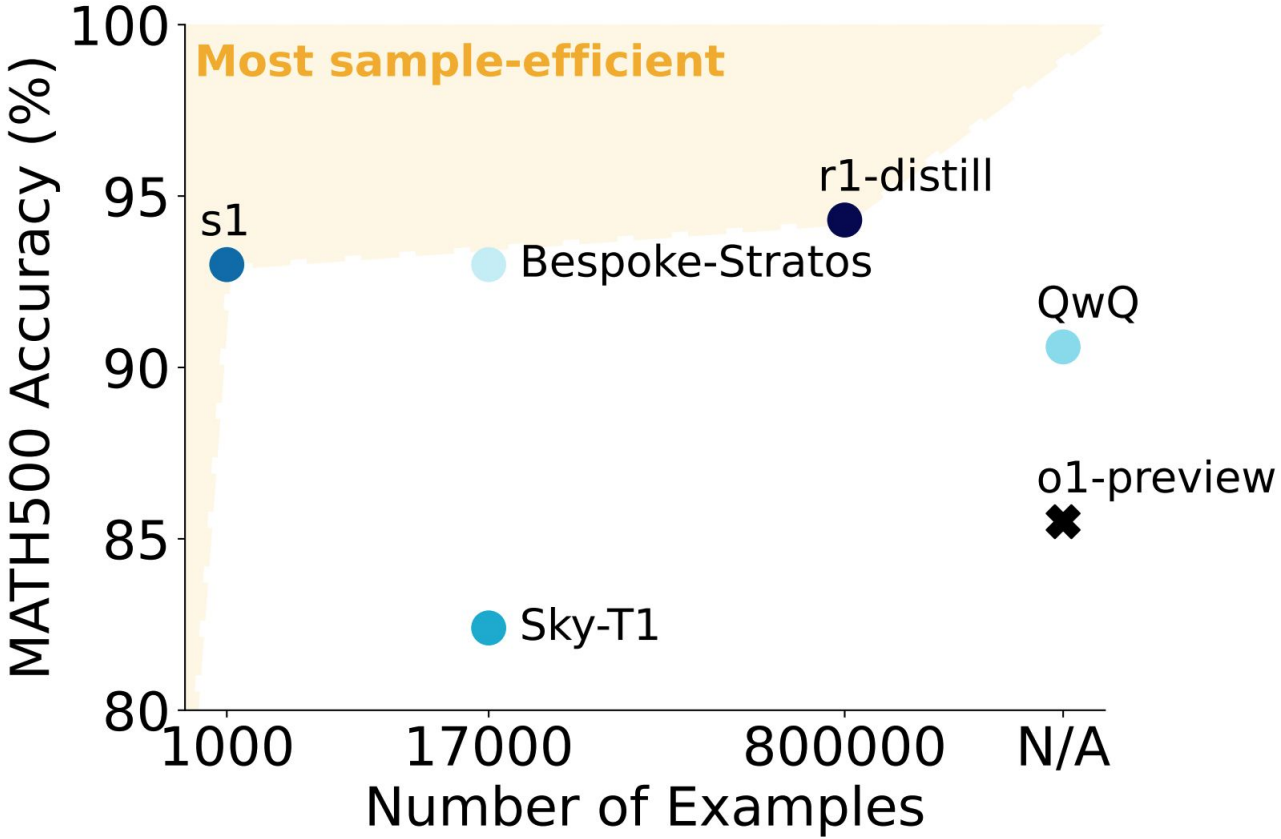


Figure 1. **Test-time scaling with s1-32B.** We benchmark **s1-32B** on reasoning-intensive tasks and vary test-time compute.

s1-32B is on the sample-efficiency frontier



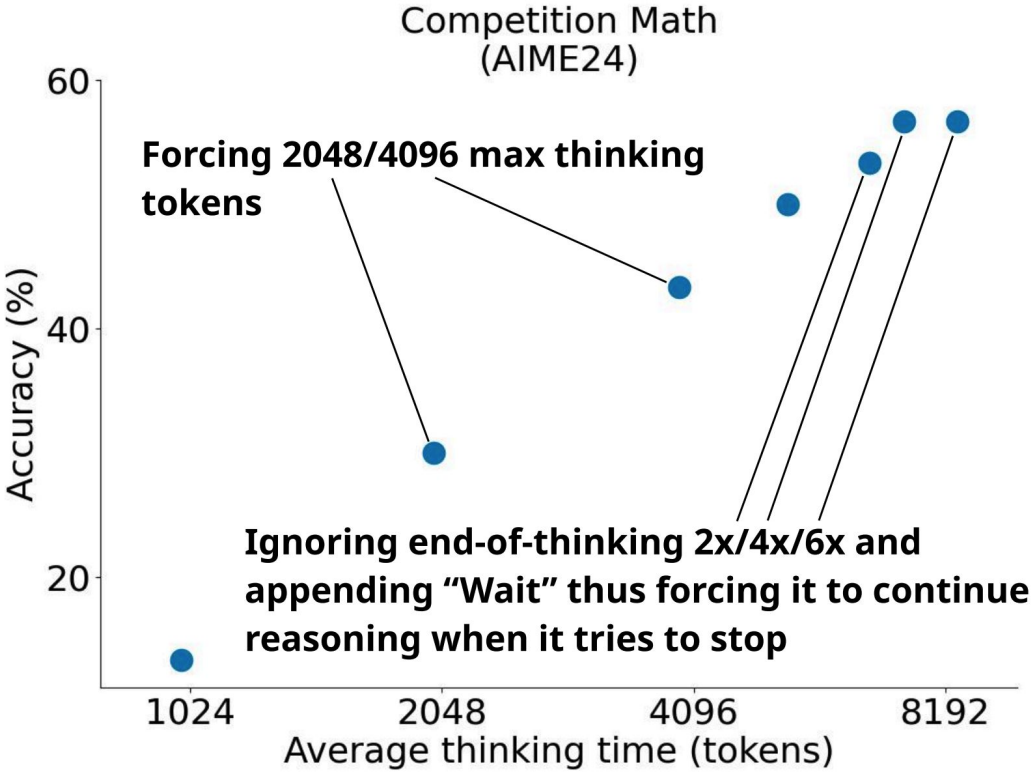
Budget forcing

How many r in raspberry? **Question**

Let's break down the process of counting the letter 'r' in the word "raspberry" ...
* First letter: 'r' - This is an 'r', count = 1.
* Second letter: 'a' - Not an 'r', count remains 1 ...
* Sixth letter: 'e' - Not an 'r', count remains 1.
* Seventh letter: 'r' - This is an 'r', count = 2.
* Eighth letter: 'y' - Not an 'r', count remains 2 ...
The number of 'r's in "raspberry" is 2.
Wait, let's re-read the question carefully. It asks "How many r in raspberry?" ... * r - a - s - p - b - e - r - r - y ... * First 'r' ... * Second 'r' ... * Third 'r' ... Count = 3 ... **Reasoning trace**

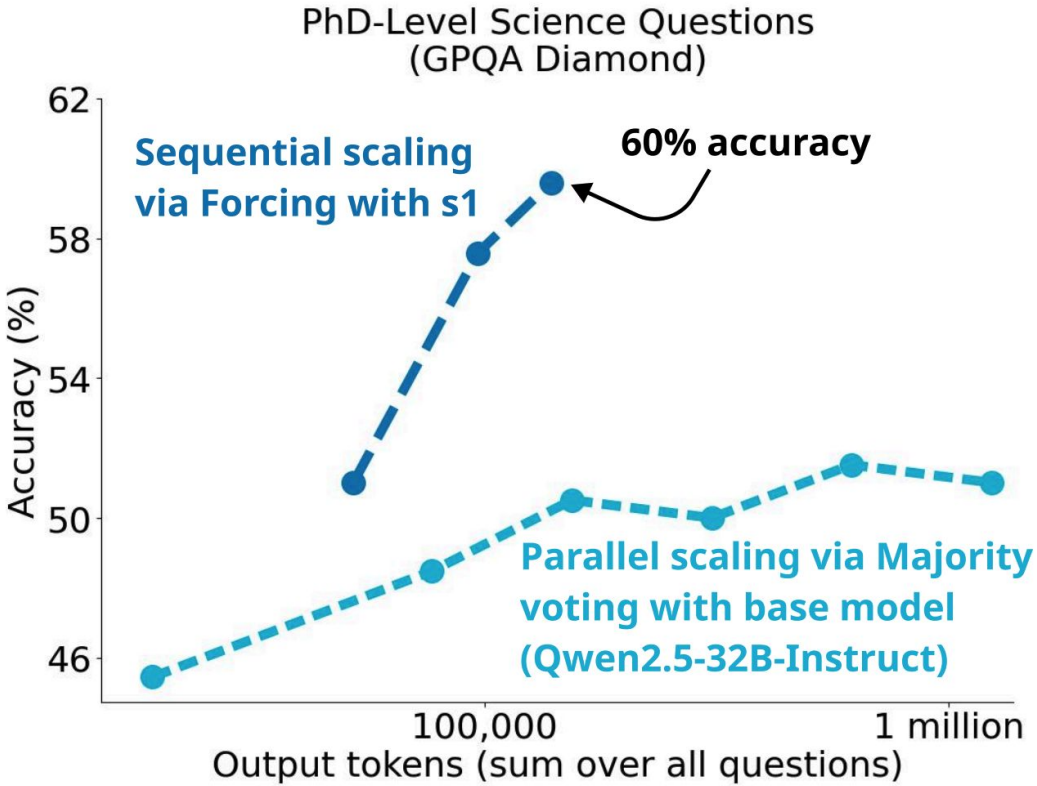
My initial answer of 2 was incorrect due to a quick reading of the word. **Final Answer:** The final answer is $\boxed{3}$ **Response**

Budget forcing shows clear scaling trends and extrapolates to some extent



(a) Sequential scaling via budget forcing

Parallel scaling via majority voting



(b) Parallel scaling via majority voting

s1-32B is a strong open reasoning model

Model	# ex.	AIME 2024	MATH 500	GPQA Diamond
API only				
o1-preview	N.A.	44.6	85.5	73.3
o1-mini	N.A.	70.0	90.0	60.0
o1	N.A.	**74.4**	**94.8**	**77.3**
Gemini 2.0 Flash Think.	N.A.	60.0	N.A.	N.A.
Open Weights				
Qwen2.5-32B-Instruct	N.A.	26.7	84.0	49.0
QwQ-32B	N.A.	50.0	90.6	54.5
r1	≫800K	**79.8**	**97.3**	**71.5**
r1-distill	800K	72.6	94.3	62.1
Open Weights and Open Data				
Sky-T1	17K	43.3	82.4	56.8
Bespoke-32B	17K	**63.3**	93.0	58.1
s1 w/o BF	**1K**	50.0	92.6	56.6
s1-32B	**1K**	56.7	**93.0**	**59.6**

s1K data ablations

Model	AIME 2024	MATH 500	GPQA Diamond
1K-random	36.7 [-26.7%, -3.3%]	90.6 [-4.8%, 0.0%]	52.0 [-12.6%, 2.5%]
1K-diverse	26.7 [-40.0%, -10.0%]	91.2 [-4.0%, 0.2%]	54.6 [-10.1%, 5.1%]
1K-longest	33.3 [-36.7%, 0.0%]	90.4 [-5.0%, -0.2%]	59.6 [-5.1%, 10.1%]
59K-full	53.3 [-13.3%, 20.0%]	92.8 [-2.6%, 2.2%]	58.1 [-6.6%, 8.6%]
s1K	50.0	93.0	57.6

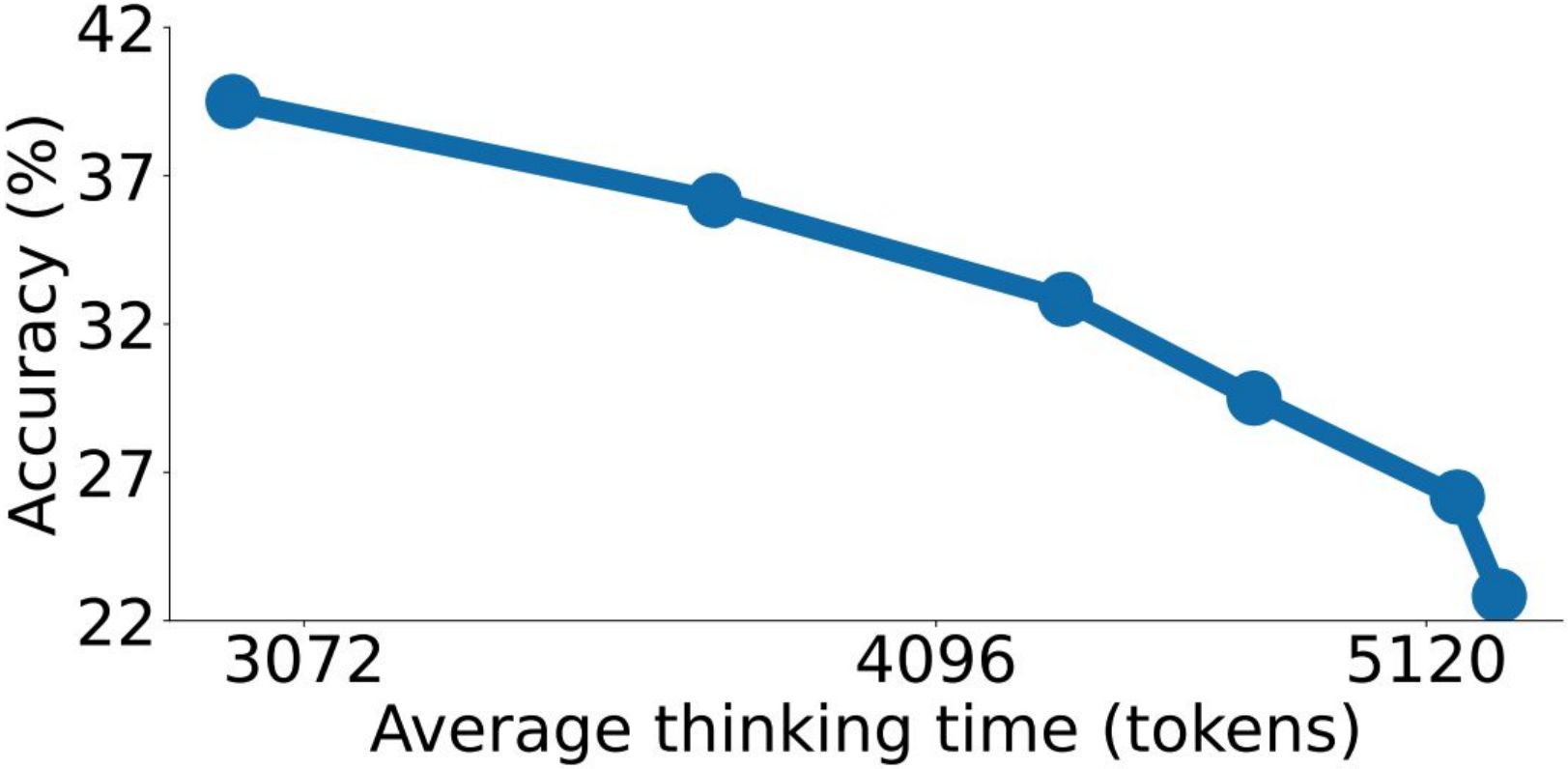
Ablations on methods to scale test-time compute

Method	Control	Scaling	Performance	$\lvert\mathcal{A}\rvert$
BF	**100%**	15	**56.7**	5
TCC	40%	-24	40.0	5
TCC + BF	**100%**	13	40.0	5
SCC	60%	3	36.7	5
SCC + BF	**100%**	6	36.7	5
CCC	50%	**25**	36.7	2
RS	**100%**	-35	40.0	5

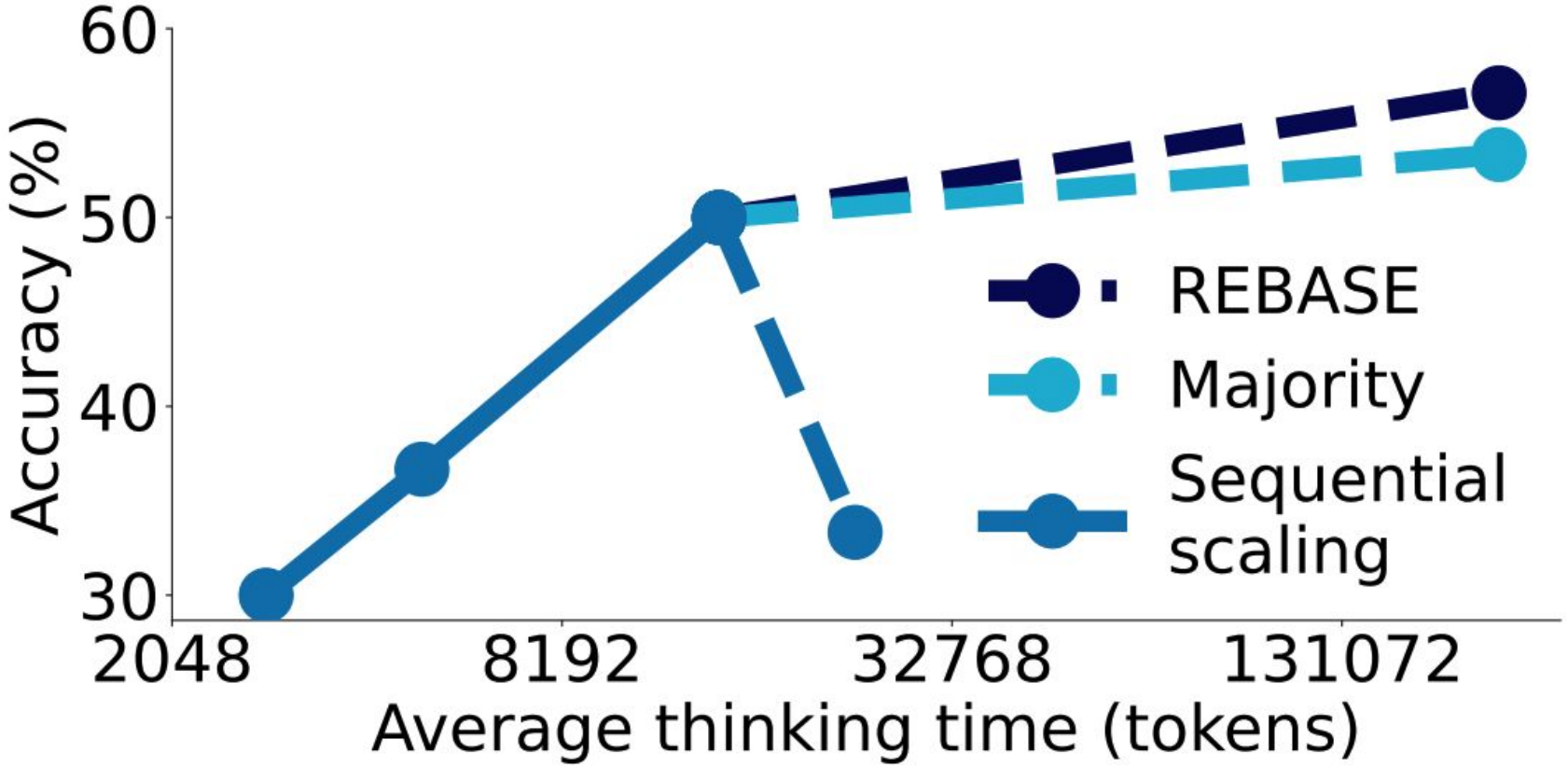
Budget forcing extrapolation ablations

Model	AIME 2024	MATH 500	GPQA Diamond
No extrapolation	50.0	**93.0**	57.6
2x without string	50.0	90.2	55.1
2x “Alternatively”	50.0	92.2	**59.6**
2x “Hmm”	50.0	**93.0**	**59.6**
2x “Wait”	**53.3**	**93.0**	**59.6**

Rejection sampling



Augmenting s1 with REBASE (process reward model)



Why does supervised fine-tuning on just 1,000 samples lead to such performance gains?

- We hypothesize that the model is already exposed to large amounts of reasoning data during pretraining which spans trillions of tokens.
- Thus, the ability to perform reasoning is already present in our model.
- Our sample-efficient fine-tuning stage just activates it and we scale it further at test time with budget forcing.

Superficial Alignment Hypothesis

- LIMA: Less is more for alignment (Zhou et al., 2023)
  - 1,000 examples can be sufficient to align a model to adhere to user preferences

LIMO: Less is More for Reasoning

Yixin Ye* Zhen Huang* Yang Xiao Ethan Chern Shijie Xia Pengfei Liu†

SJTU, SII, GAIR

Superficial Alignment Hypothesis

- LIMA: Less is more for alignment (Zhou et al., 2023)
  - 1,000 examples can be sufficient

- LIMO: even competition-level complex reasoning abilities can be effectively elicited through minimal but curated training samples

- LIMO: a promising technical pathway toward AGI - any sophisticated reasoning capability, no matter how complex, could potentially be activated with minimal samples given two key conditions:

  - (1) sufficient domain knowledge embedded during pre-training
  - (2) optimal cognitive reasoning chains for activation

Categorizing the reasoning chains into five

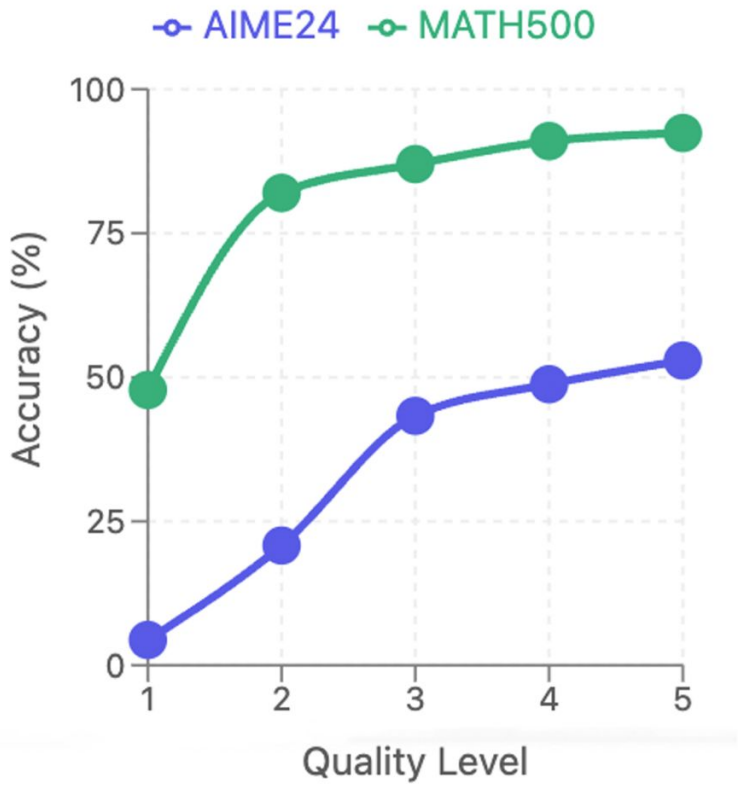
How well the reasoning steps were organized, whether important logical transitions were properly explained, and if the solution included self-verification steps

- **L5:** excellent organization with clear, well-explained steps and thorough self-verification
- **L4:** well-structured but perhaps with slightly less rigorous checking
- **L3:** decent organization but sometimes skipped over explaining crucial logical leaps
- **L2:** often provided abbreviated reasoning without much explanation
- **L1:** just listed basic steps with minimal elaboration and rarely included any verification

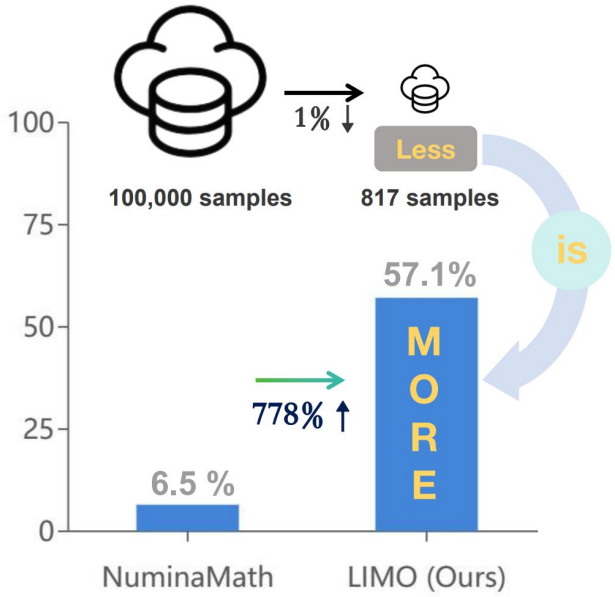
Statistical analysis of different quality levels

Data Quality Level	Avg. Tokens per response	Avg. Lines per response	Top 10 Frequently Occurring Keywords (in order)
Level 1	230	9.21	since, however, number, let, thus, which, get, two, triangle, theta
Level 2	444.88	50.68	number, need, times, which, find, list, thus, since, triangle, sum
Level 3	4956.11	375.60	**perhaps**, **alternatively**, **consider**, number, **wait**, which, sides, need, equal, seems
Level 4	4726.97	354.87	**wait**, which, number, **perhaps**, therefore, let, since, **maybe**, sides, two
Level 5	5290.26	239.29	**wait**, therefore, which, number, since, lets, two, sides, let, **maybe**

Comparison of models trained on reasoning chains of different quality levels



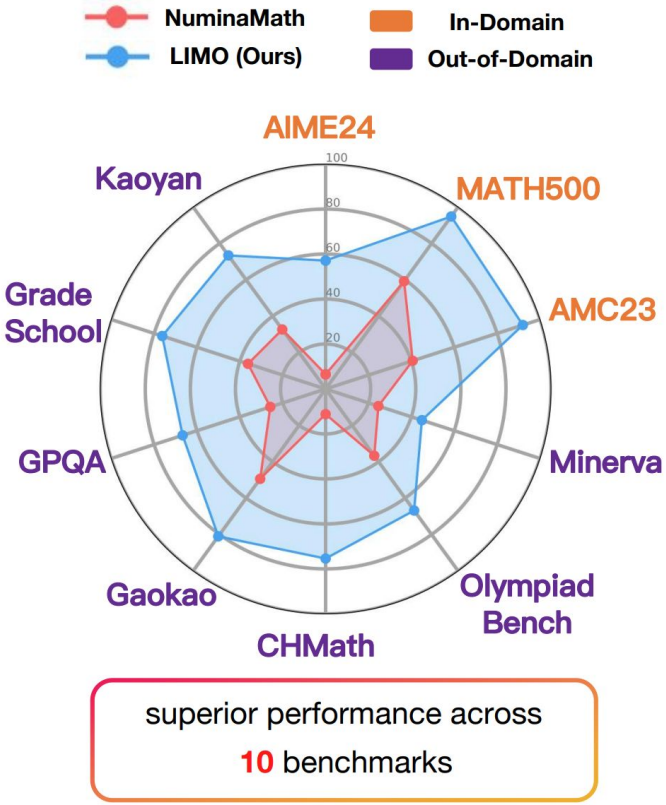
LIMO achieves substantial improvement over NuminaMath with fewer samples



completely **same** backbone

1% data → **778%** gain on AIME24 (pass@1)

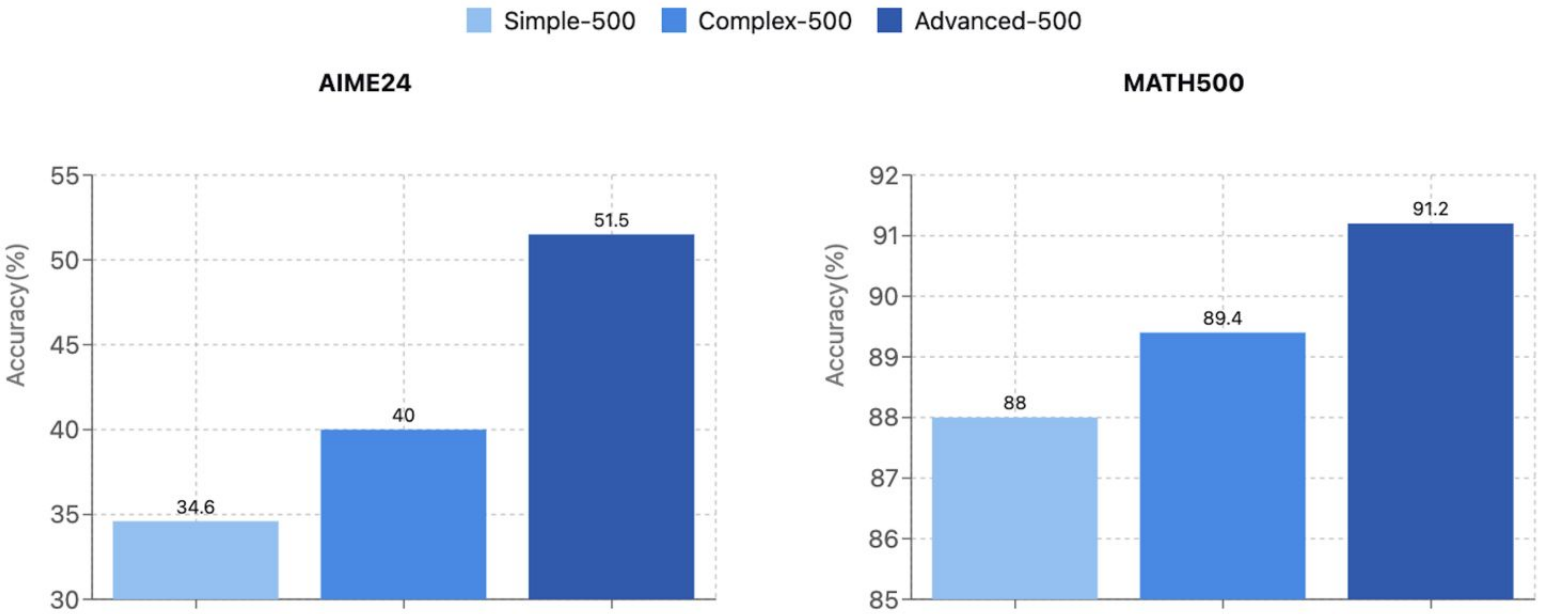
… while excelling across diverse mathematical and multi-discipline benchmarks



LIMO achieves superior performance despite using significantly fewer training examples

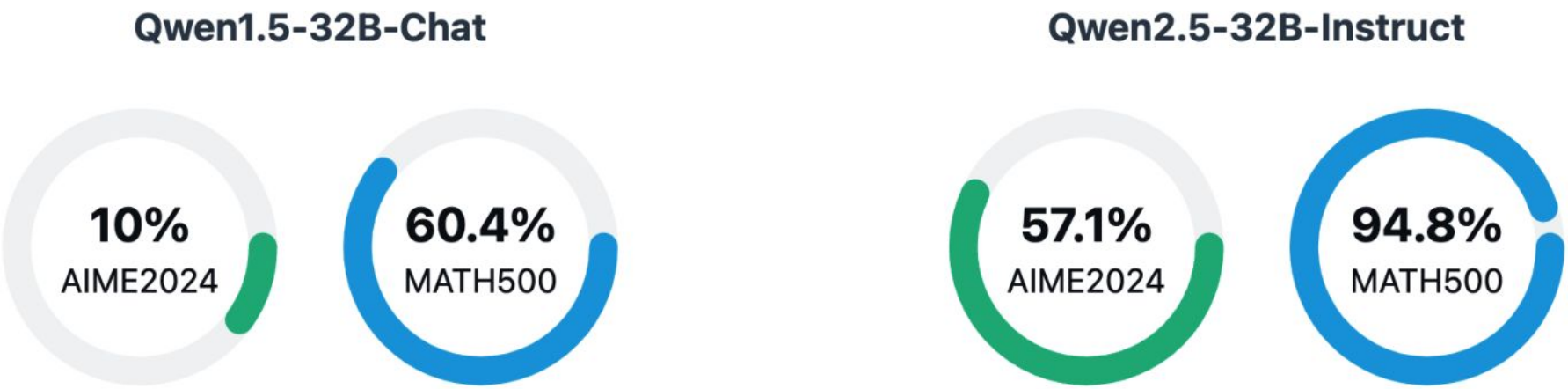
Datasets	OpenAI-o1 -preview	Qwen2.5-32B -Instruct	QwQ-32B-preview	OpenThoughts (114k)	NuminaMath (100k)	LIMO ours(817)
In Domain						
AIME24	44.6	16.5	50.0	50.2	6.5	**57.1**
MATH500	85.5	79.4	89.8	80.6	59.2	**94.8**
AMC23	81.8	64.0	83.6	80.5	40.6	**92.0**
Out of Domain						
OlympiadBench	52.1	45.3	58.5	56.3	36.7	**66.8**
CHMath	50.0	27.3	68.5	74.1	11.2	**75.4**
Gaokao	62.1	72.1	80.1	63.2	49.4	**81.0**
Kaoyan	51.5	48.2	70.3	54.7	32.7	**73.4**
GradeSchool	62.8	56.7	63.8	39.0	36.2	**76.2**
Minerva	**47.1**	41.2	39.0	41.1	24.6	44.9
GPQA	**73.3**	48.0	65.1	42.9	25.8	66.7
AVG.	61.1	49.9	66.9	58.3	32.3	**72.8**

Models trained on different question quality



- **Simple-500**: 500 simple problems randomly selected problems from MATH levels 1 and 2.
- **Complex-500**: 500 complex problems randomly selected problems from MATH levels 3, 4, and 5.
- **Advanced-500**: 500 advanced problems randomly selected problems from past AIME tests.

Impact of pretrained model choice



Example model outputs

Problem: Every morning Aya goes for a 9 km long walk and stops at a coffee shop afterwards. When she walks at a constant speed of s kilometers per hour, the walk takes her 4 hours, including t minutes spent in the coffee shop. When she walks $s+2$ kilometers per hour, the walk takes her 2 hours and 24 minutes, including t minutes spent in the coffee shop. Suppose Aya walks at $s+\\frac{1}{2}$ kilometers per hour. Find the number of minutes the walk takes her, including the t minutes spent in the coffee shop.

Qwen2.5-32B-Instruct

To solve the problem, we need to determine the time Aya spends walking [omitted for brevity…]
The time spent walking is the distance divided by the speed, so the time spent walking is (9/s) hours. Therefore, the equation for the total time is: **9/s + t = 240,** where t is the time spent in the coffee shop in minutes.
Next, we know that [omitted for brevity…]. The time spent walking is 9/(s+2) hours. Therefore, the equation for the total time is: **9/(s+2) + t = 144**.
We now have two equations:
1. 9/s + t = 240
2. 9/(s+2) + t = 144
[omitted for brevity…]
So, s = 3 / 16 km/h. Now we substitute s = 3 / 16 km/h back into equation (1) to find t:
9 / (3 / 16) + t = 240, 48 + t = 240, t = 192
[omitted for brevity…]
So, the number of minutes the walk takes her, including the t minutes spent in the coffee shop, is 205.

unable to correct false statements above

Did not cross verify equation 2

DeepSeek-R1

Okay, let's try to figure out this problem step by step. So, Aya goes for a 9-kilometer walk every morning and then stops at a coffee shop. [omitted for brevity…]
First, let's parse the problem. Let's denote the walking [omitted for brevity…]. Our goal is to find the total time when she walks at s+1/2 km/h. **Hmm**, so we need to figure out the values of s and t first.
Let me write down the equations based on the given information. [omitted for brevity…]
Now we have two equations:
1) 9/s + t/60 = 4, 2) 9/(s+2) + t/60 = 2.4
We can subtract the second equation from the first to eliminate [omitted for brevity…]
So the left side becomes: [omitted for brevity…]
Multiply both sides by: [omitted for brevity…]
Now, solving this quadratic equation: [omitted for brevity…]. So, Aya's original speed is 2.5 km/h.
Now, let's find t [omitted for brevity…]. **Wait, let me confirm**. **But wait**, the question says [omitted for brevity…]. Therefore, the answer should be 204 minutes.
But let me just check again to make sure I didn't make any mistakes in the quadratic equation solving. Let's verify that s = 2.5 is correct. [omitted for brevity…]. Thus, the answer should be 204.

think with anthropomorphic tone

detailed steps in calculation

self-reflection

LIMO

Okay, let's see. So Aya goes for a 9-km walk every morning and then stops at a coffee shop. The time she takes for the walk, including the time in the coffee shop, depends on her walking speed. [omitted for brevity…]. When she walks a bit faster, at s plus 2 kilometers per hour, the total time is 2 hours and 24 minutes, which is 2 hours and 24 minutes, so that's 2.4 hours?
Wait, 24 minutes is 0.4 hours? Wait, no. Wait, 60 minutes is 1 hour, so 24 minutes is 24/60, which is 0.4 hours. So 2 hours and 24 minutes is 2.4 hours. [omitted for brevity…]
So, let's set up equations. Let's denote the walking time as T1 when she walks at s km/h, and T2 when she walks at s+2 km/h. Then, T1 + t = 4 hours, and T2 + t = 2.4 hours. **But t is in minutes, so we need to convert that to hours or convert the total time to minutes. Maybe it's better to keep everything in hours.** So t minutes is t/60 hours. [omitted for brevity…]
For the first case: walking time T1 = 9/s hours, then total time T1 + t/60 = 4 hours.
For the second case: walking time T2 = 9/(s+2) hours, then total time T2 + t/60 = 2.4 hours. So now we have two equations: 1) 9/s + t/60 = 4, 2) 9/(s+2) + t/60 = 2.4
Let me write that as:
9*(1/s - 1/(s+2)) = 1.6. So, 1/s - 1/(s+2) = 1.6/9
Let me compute 1.6 divided by 9. 1.6 divided by 9 is 0.1777... or 16/90, which simplifies to 8/45. So 1.6/9 = 8/45 ≈ 0.1777… So, 1/s - 1/(s+2) = 8/45
Now, let's compute the left side: [omitted for brevity…]. Multiply both sides by 2: [omitted for brevity…]. Let me write that as a quadratic equation: [omitted for brevity…]. Now, let's solve this quadratic equation: [omitted for brevity…]. So, s is 2.5 km/h.
Let's verify that. If s is 2.5 km/h, then walking 9 km would take 9/2.5 = 3.6 hour [omitted for brevity…]
But let me check again. Maybe I made a mistake in calculations. First, s = 2.5 km/h. Then, walking time at s is 9/2.5 = 3.6 hours…[omitted for brevity…] **Alternatively, we could have converted everything to minutes from the start**. Let's see: First walk: 9/(s) hours + t minutes = 4 hours. 4 hours is 240 minutes. [omitted for brevity…] I think that's solid. So the answer is 204.

Thank you!