

Test Time Compute

NLP: Fall 2025

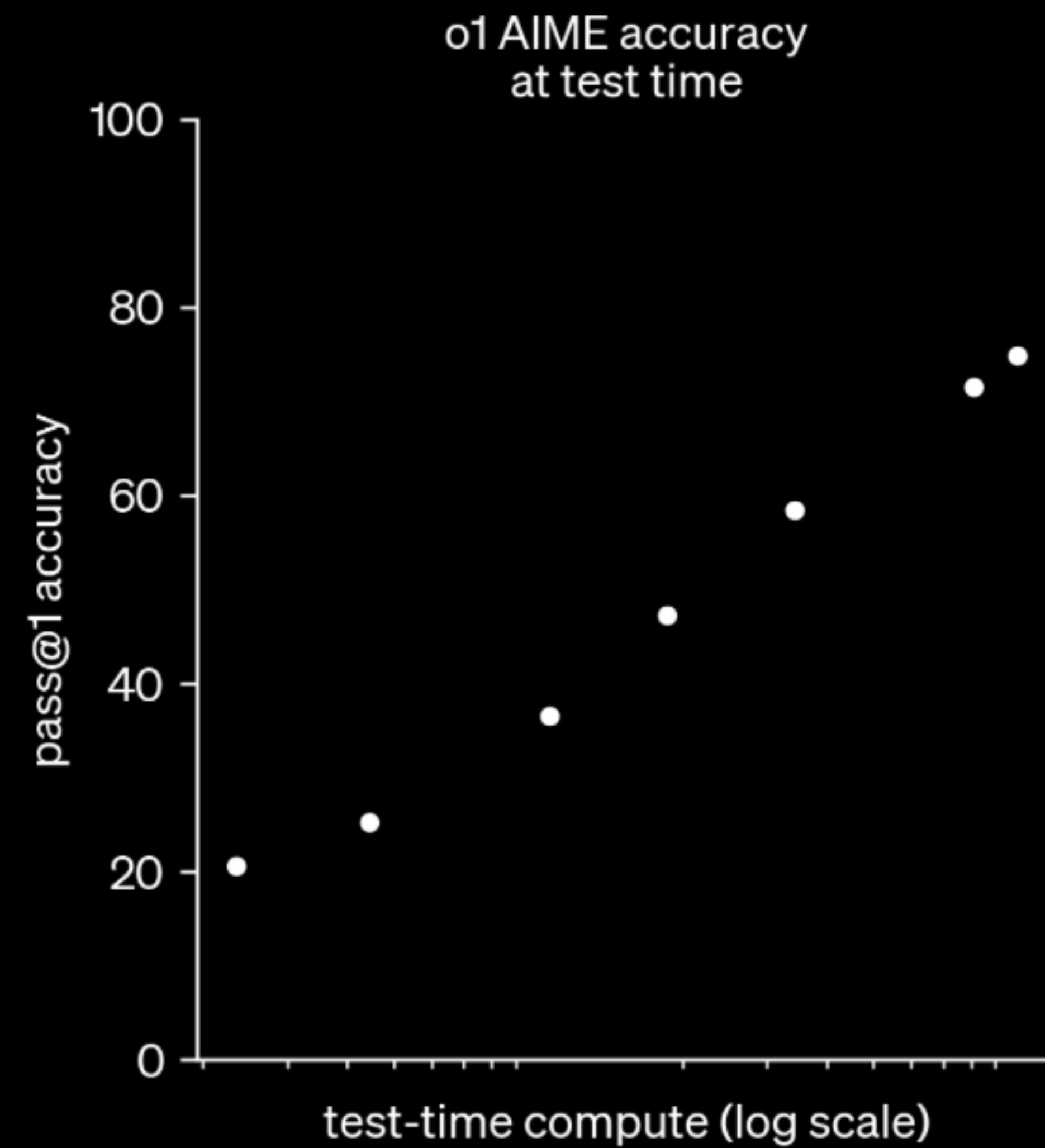
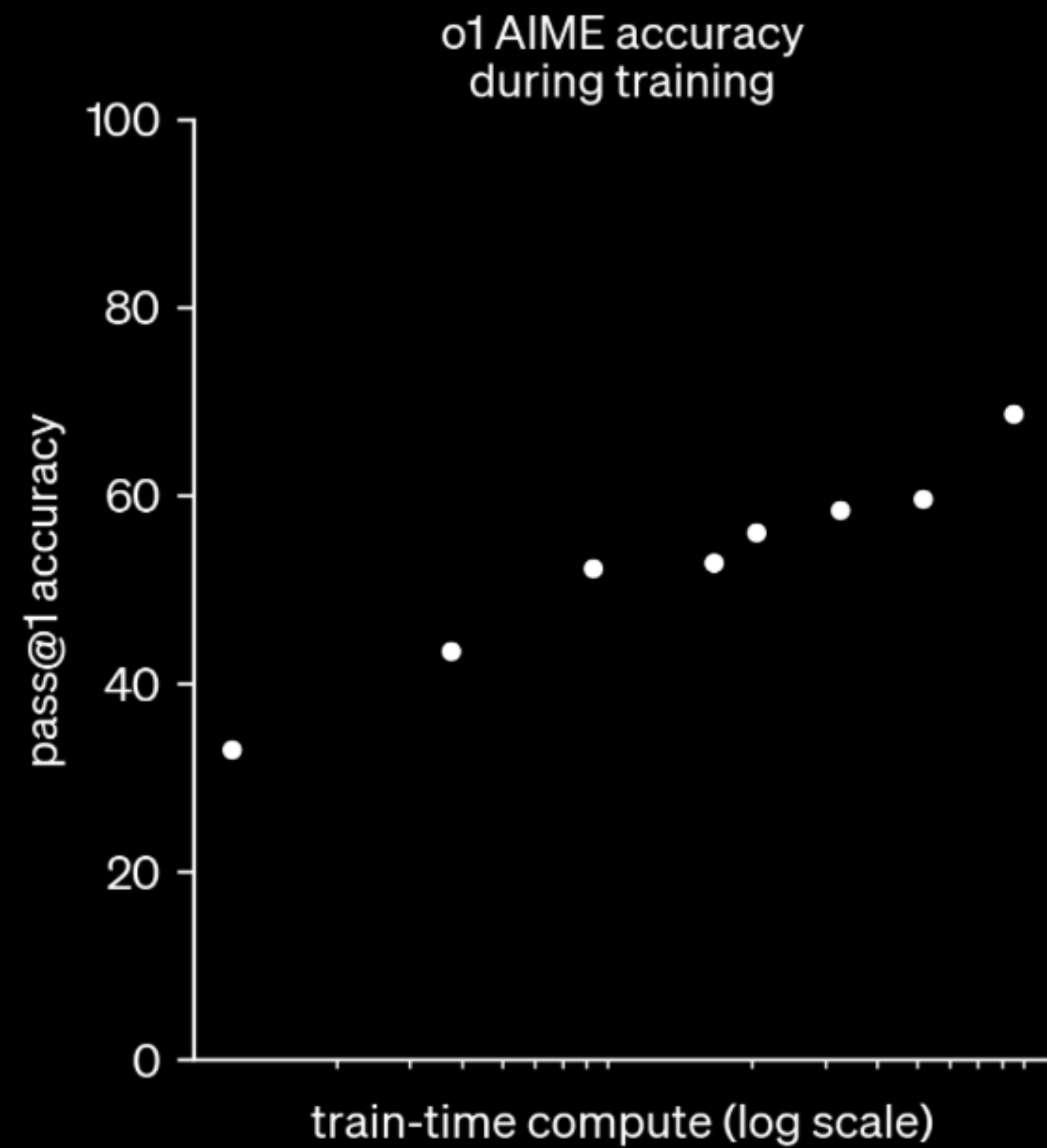
Anoop Sarkar

Test time compute

also known as reasoning models

- Sept 12, 2024 OpenAI released their first reasoning model called o1
- Produces chain of thought tokens in the output
- Model was trained using reinforcement learning to produce "thinking" tokens and then use those tokens to follow the prompt
 - Learns to recognize and correct its mistakes.
 - Learns to break down tricky steps into simpler ones.
 - Learns to try a different approach when the current one isn't working

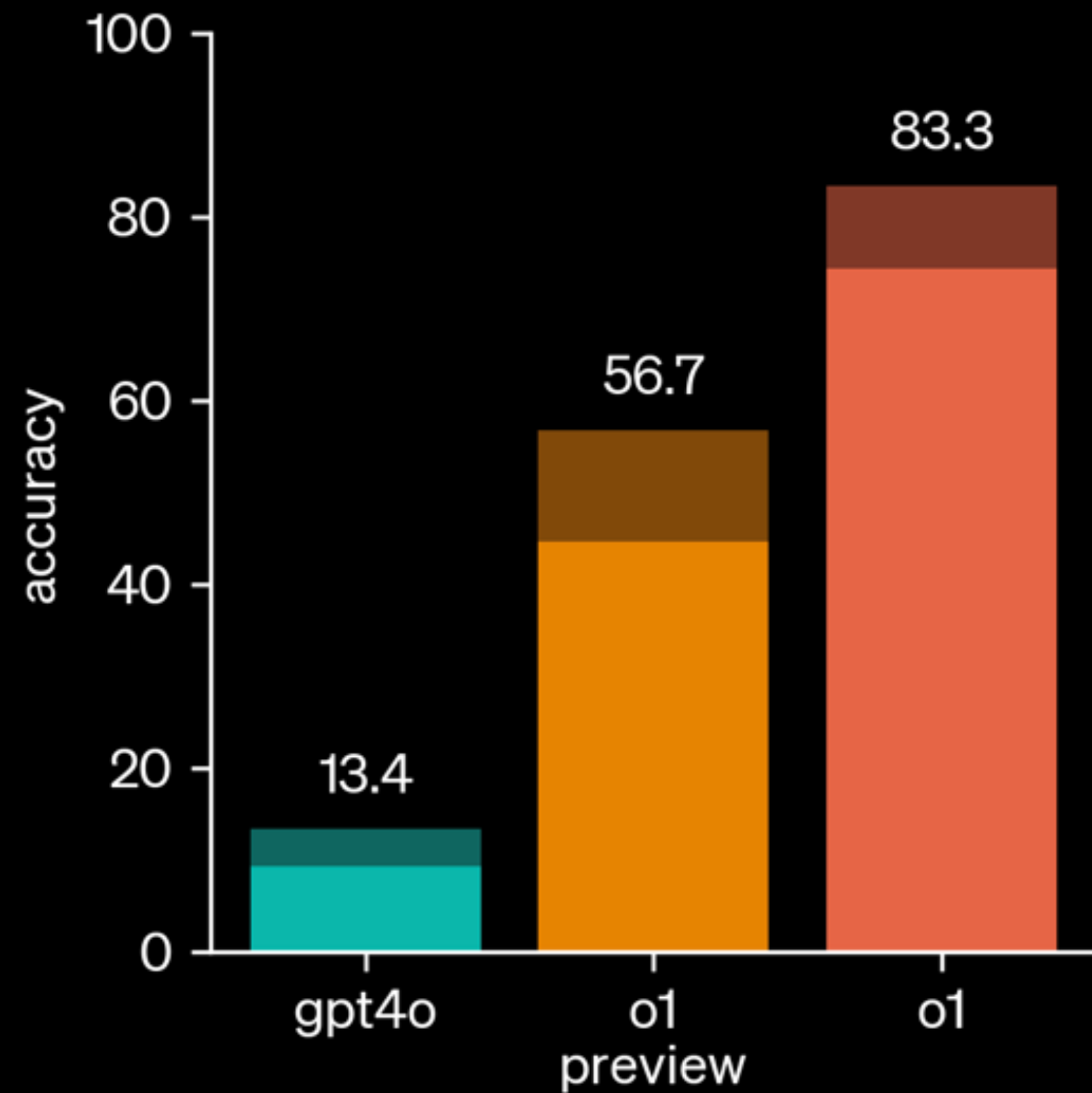
<https://openai.com/index/learning-to-reason-with-llms/>



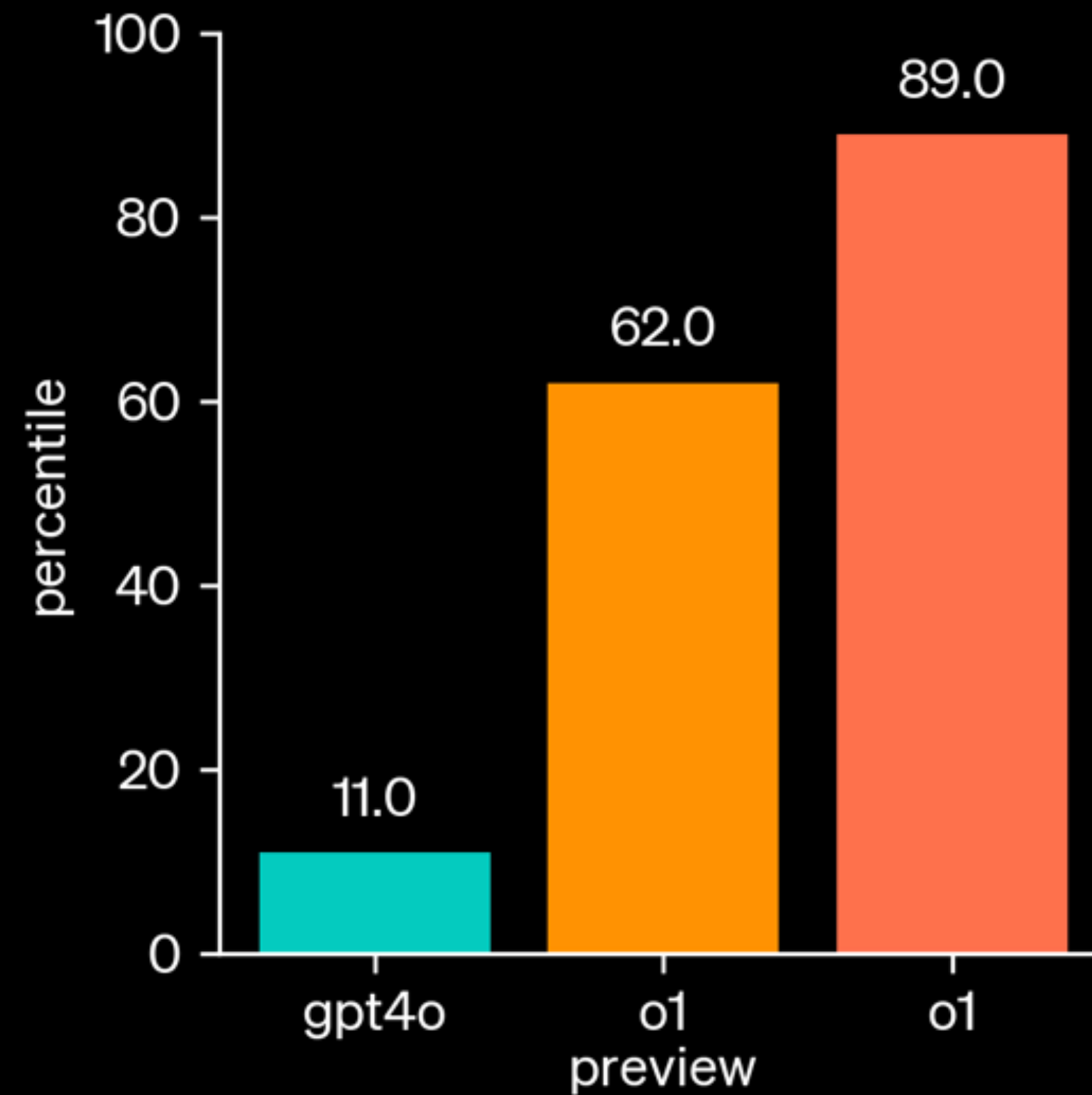
o1 performance smoothly improves with both train-time and test-time compute

AIME is a challenging benchmark of Math Olympiad problems

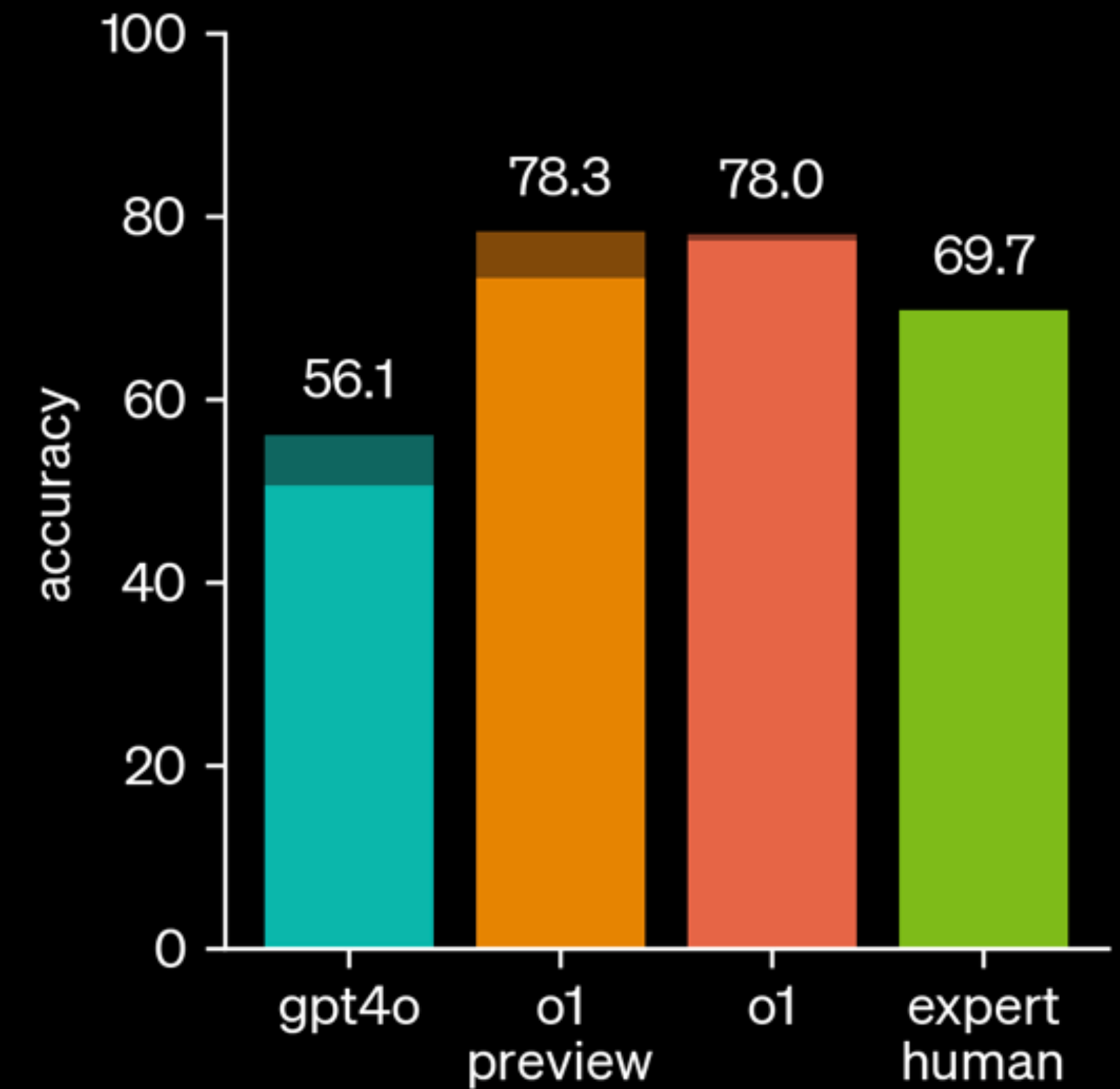
Competition Math
(AIME 2024)



Competition Code
(Codeforces)



PhD-Level Science Questions
(GPQA Diamond)



o1 greatly improves over GPT-4o on challenging reasoning benchmarks. Solid bars show pass@1 accuracy and the shaded region shows the performance of majority vote (consensus) with 64 samples.

<https://openai.com/index/learning-to-reason-with-llms/#:~:text=Cipher>

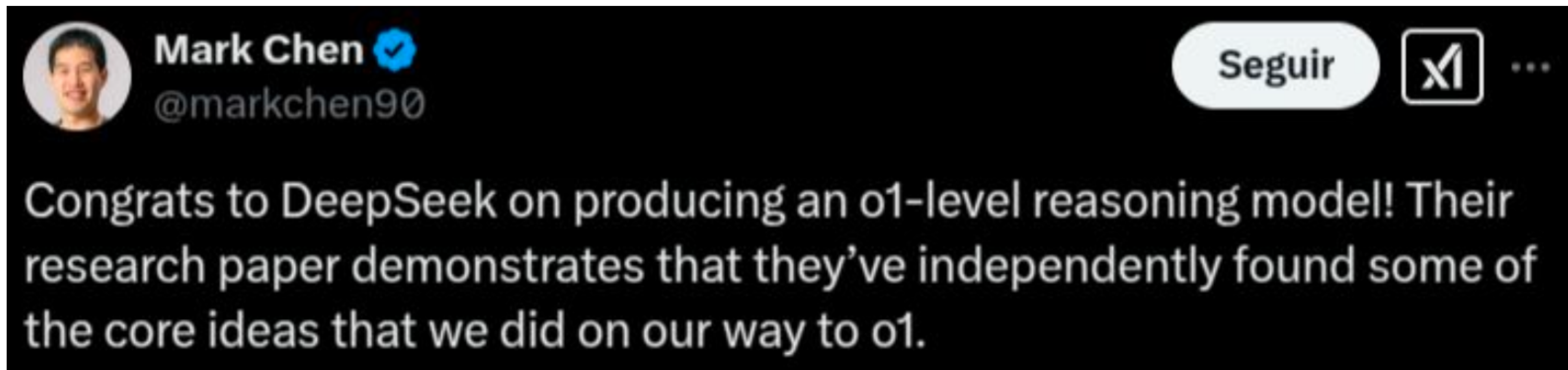
o1 release

- OpenAI did not release any details on the training of o1
- Even hid the reasoning tokens sent back to the client
- This led to more structured output from the LLM servers: OpenAI's Harmony

```
<|start|>user<|message|>What is the weather in SF?<|end|><|start|>assistant<|channel|>analysis<|message|>User asks: "What is the weather in SF?" We need to use lookup_weather tool.<|end|><|start|>assistant to=functions.lookup_weather<|channel|>commentary <|constrain|>json<|message|>{"location": "San Francisco"}<|end|><|start|>assistant<|channel|>analysis<|message|>The weather in San Francisco is sunny. I can answer<|end|><|start|>assistant<|channel|>final<|message|>It's sunny in SF<|end|>
```

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

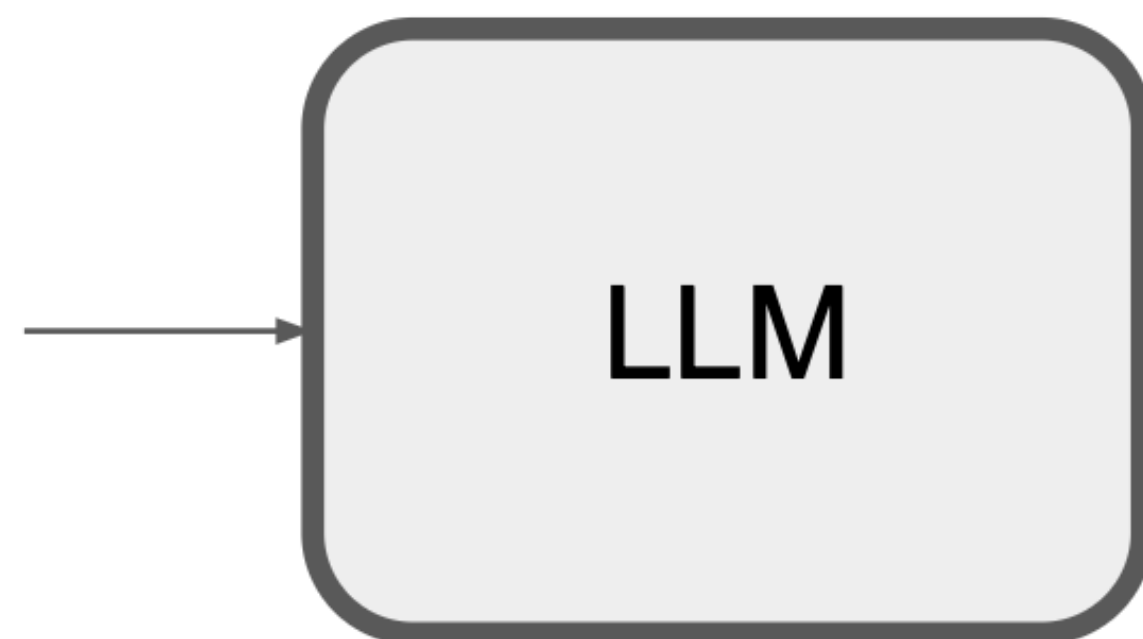
DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning



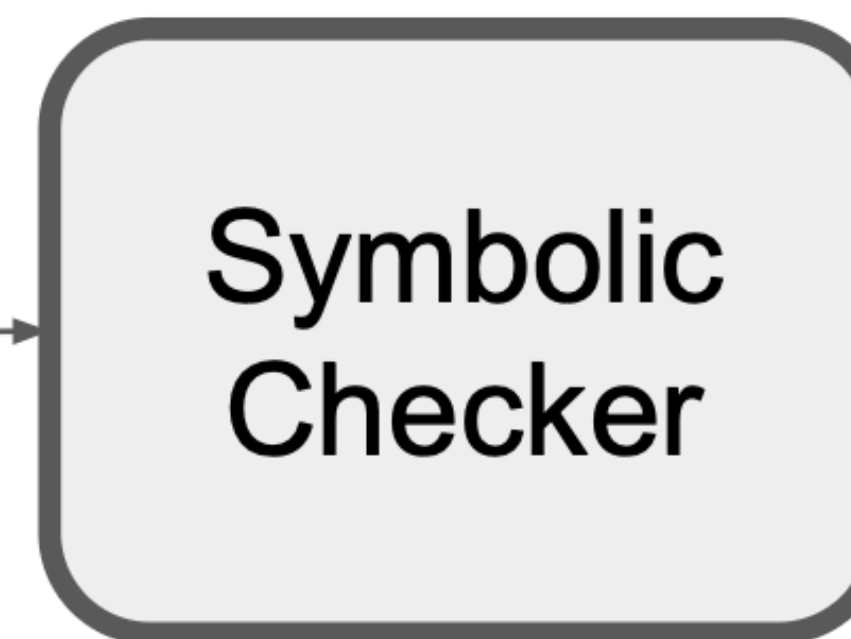
Mark Chen: Chief Research Officer at OpenAI

Deepseek R1-Zero

Thinking
Prompt
+ Problem
Definition



Reasoning
+ answer



REINFORCE
(GRPO)

Rule based reward system

Reinforcement learning needs a reward model

- **Accuracy reward model**
 - Evaluate if the response sampled from the instruct tuned LLM is correct or not
 - For example, in the case of math problems with deterministic results, the model is required to provide the final answer in a specified format
- **Format reward model**
 - A format reward model enforces the model to put its thinking process between <think> and </think> tags

Training 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:

Table 1 | Template for DeepSeek-R1-Zero. **prompt** will be replaced with the specific reasoning question during training.

Group Relative Policy Optimization

GRPO

- **Train the model to produce reasoning tokens**
 - Sample multiple sequences that attempt to solve the input problem
 - Keep all the attempts in group and use entire group
 - Contrast with pairwise comparisons: Direct Preference Optimization (DPO)
- **Group relative advantage**
 - $\text{Advantage} = (\text{reward} - \text{mean}(\text{group_rewards})) / \text{std}(\text{group_rewards})$
- **Stable training**
 - Stable training using KL divergence from model from previous iteration

GRPO

Input:

- initial_policy: Starting *model* to be trained
- reward_function: Function that evaluates outputs
- training_prompts: *Set* of *training examples*
- group_size: Number of outputs per prompt (typically 4-16)

Algorithm *GRPO*:

1. For *each training iteration*:

a. *Set* *reference_policy* = *initial_policy* (snapshot current policy)

b. For *each prompt in batch*:

i. Generate *group_size* different outputs using *initial_policy*

ii. Compute *rewards* for each output using *reward_function*

iii. Normalize *rewards* within group:

$$\text{normalized_advantage} = (\text{reward} - \text{mean}(\text{rewards})) / \text{std}(\text{rewards})$$

iv. Update policy by *maximizing* the clipped ratio:

$$\min(\text{prob_ratio} * \text{normalized_advantage},$$

$$\text{clip}(\text{prob_ratio}, 1-\text{epsilon}, 1+\text{epsilon}) * \text{normalized_advantage})$$

- $\text{kl_weight} * \text{KL}(\text{initial_policy} || \text{reference_policy})$

where *prob_ratio* is *current_prob* / *reference_prob*

Output: Optimized *policy model*

[https://
huggingface.
co/learn/llm-
course/
chapter12/3](https://huggingface.co/learn/llm-course/chapter12/3)

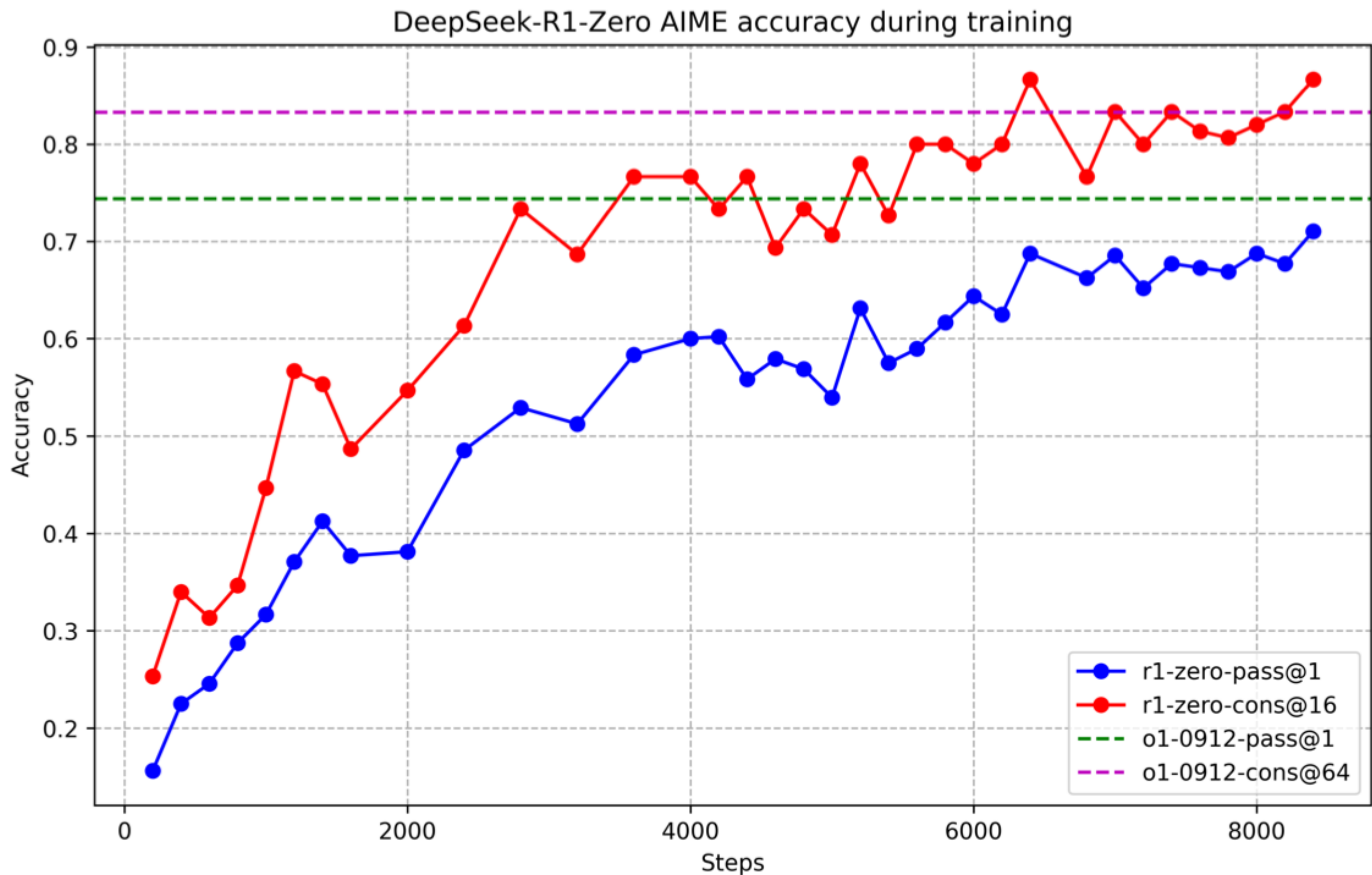


Figure 2 | AIME accuracy of DeepSeek-R1-Zero during training. For each question, we sample 16 responses and calculate the overall average accuracy to ensure a stable evaluation.

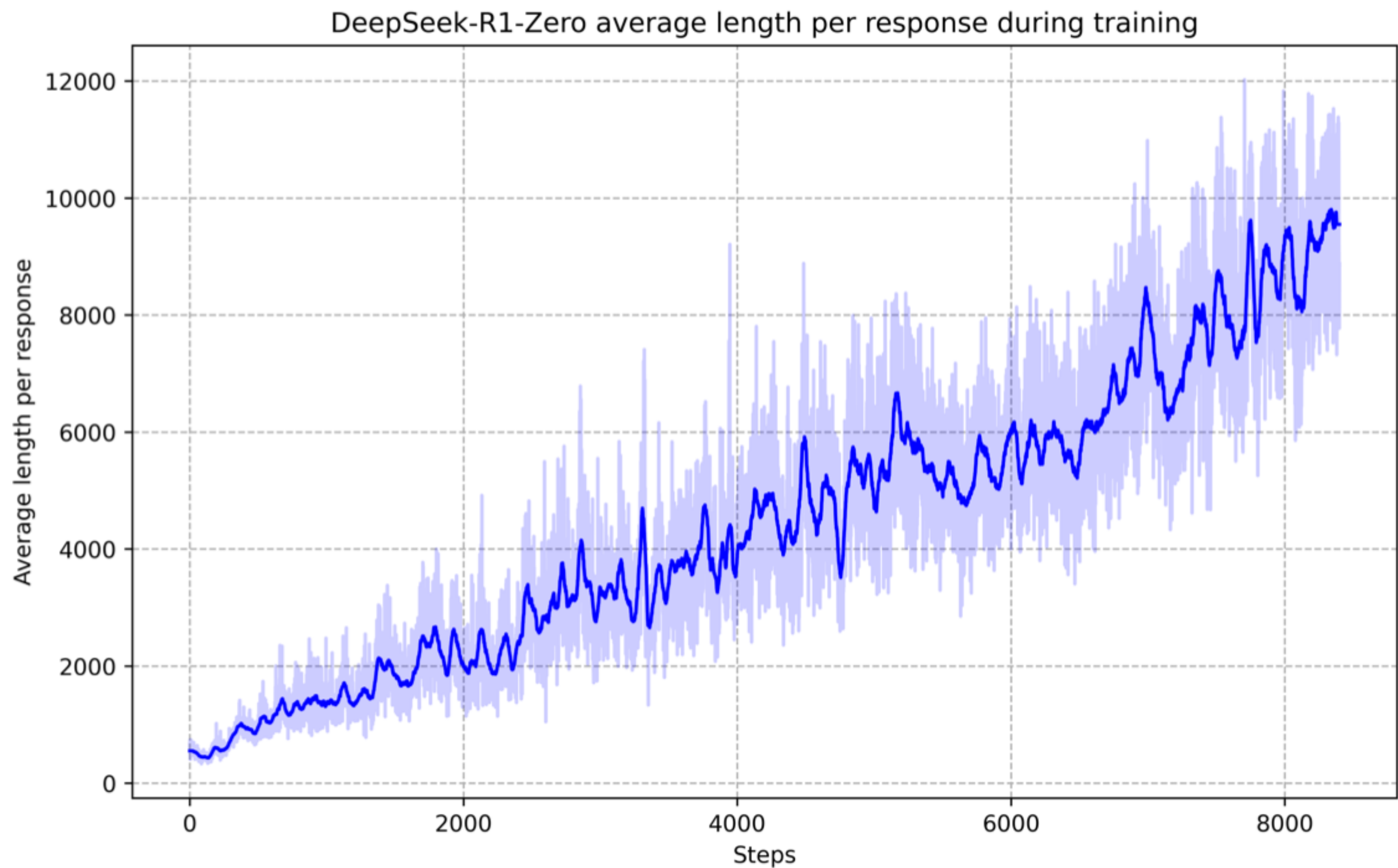


Figure 3 | The average response length of DeepSeek-R1-Zero on the training set during the RL process. DeepSeek-R1-Zero naturally learns to solve reasoning tasks with more thinking time.

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 $\dots$

$$\left(\sqrt{a - \sqrt{a + x}}\right)^2 = x^2 \implies a - \sqrt{a + x} = x^2.$$

Rearrange to isolate the inner square root term:

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

$\dots$

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 $\dots$

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: $\dots$

$\dots$

Challenges

- **Cold start**
 - Use human written reasoning responses to initialize the model
 - Prevents unstable start phase of RL training from base LLM
- **Code switching in reasoning**
 - Add an additional format reward to prefer monolingual reasoning tokens
- **Non-reasoning data**
 - Avoid reasoning for some input prompts
 - e.g. for creative writing, factual QA, self-cognition and translation use
 - Add base LLM output to training data for such cases

Challenges

- **Generation Cost:** Generating multiple completions (4-16) for each prompt increases compute compared to methods that generate only one or two completions.
- **Batch Size Constraints:** The need to process groups of completions together can limit effective batch sizes slowing down training.
- **Reward Function Design:** The quality of training heavily depends on well-designed reward functions. Poorly designed rewards can lead to unintended behaviors or optimization for the wrong objectives.
- **Group Size Tradeoffs:** Choosing the optimal group size involves balancing diversity of solutions against computational cost. Too few samples may not provide enough diversity, while too many increase training time and resource requirements.
- **KL Divergence Tuning:** Finding the right balance for the KL divergence penalty requires careful tuning - too high and the model won't learn effectively, too low and it may diverge too far from its initial capabilities.