



香港科技大學  
THE HONG KONG  
UNIVERSITY OF SCIENCE  
AND TECHNOLOGY

COMP 4901B  
Large Language Models

# Deep Reasoning Models

Junxian He

Oct 31, 2025

# Recap: Chain-of-Thought Reasoning

## Standard Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The answer is 27. X

## Chain-of-Thought Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

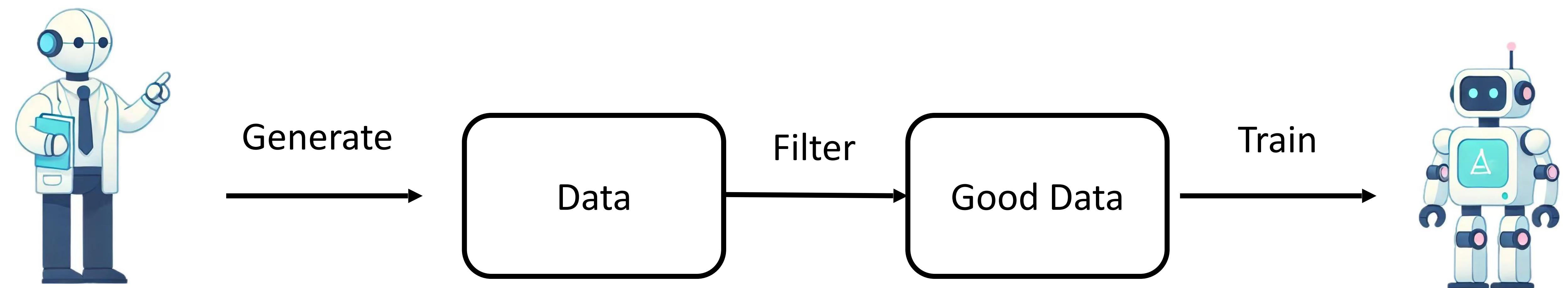
A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls.  $5 + 6 = 11$ . The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

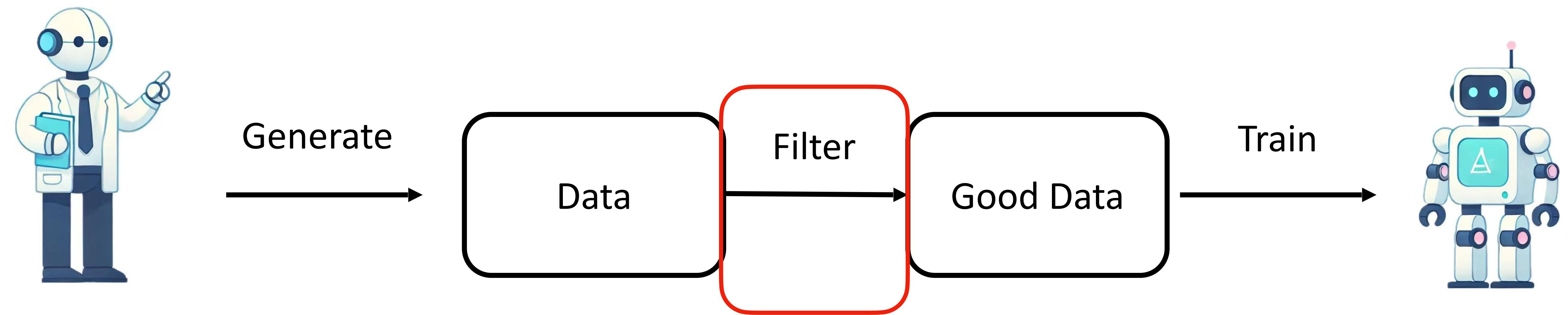
Model Output

A: The cafeteria had 23 apples originally. They used 20 to make lunch. So they had  $23 - 20 = 3$ . They bought 6 more apples, so they have  $3 + 6 = 9$ . The answer is 9. ✓

# Recap: Distillation from a Strong Teacher Model



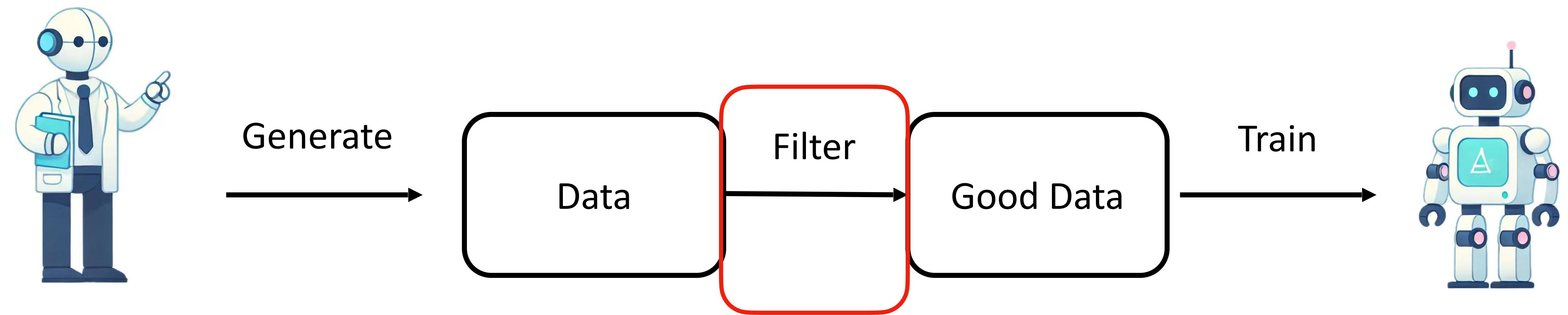
# Recap: Distillation from a Strong Teacher Model



Filter is optional. In mathematical reasoning for example, we filter with final answer correctness. For code, we filter with whether to pass the unit test passing

final answer  
number

# Recap: Distillation from a Strong Teacher Model

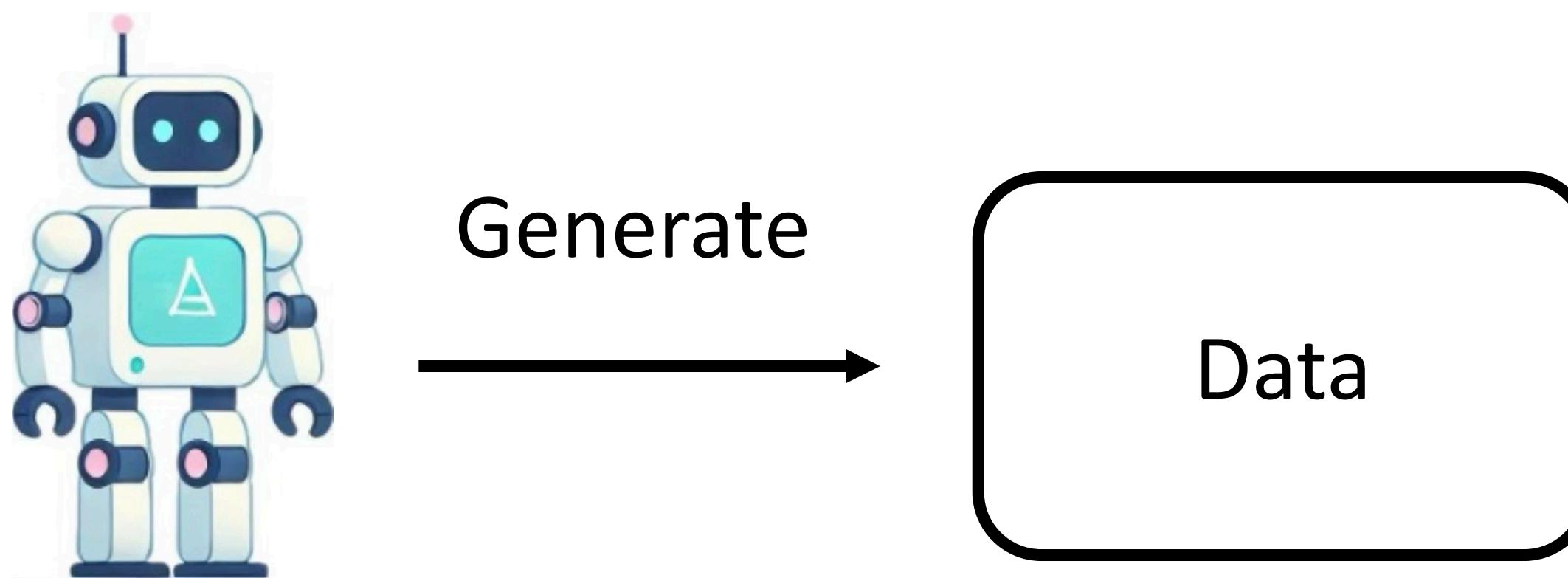


Filter is optional. In mathematical reasoning for example, we filter with final answer correctness. For code, we filter with whether to pass the unit test passing

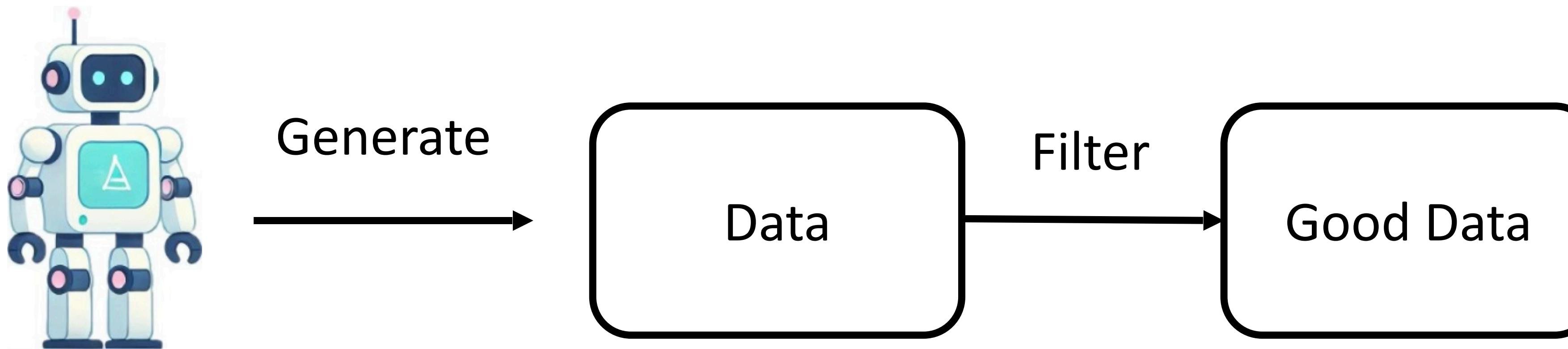
If you still remember, when we talked about evaluation, we mentioned final answer correctness does not entail reasoning correctness

# Recap: Self-Improving

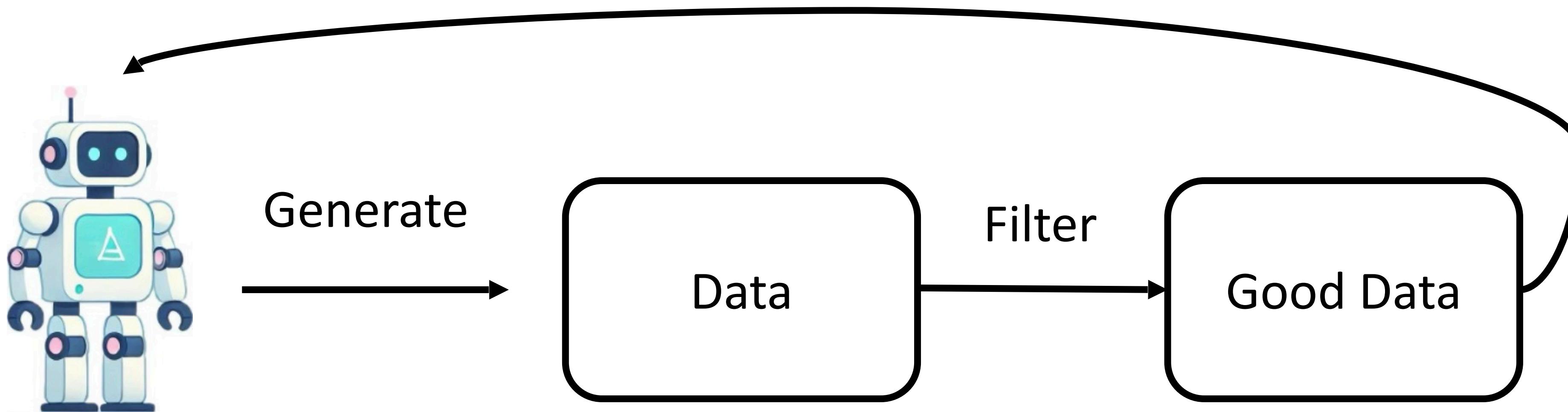
# Recap: Self-Improving



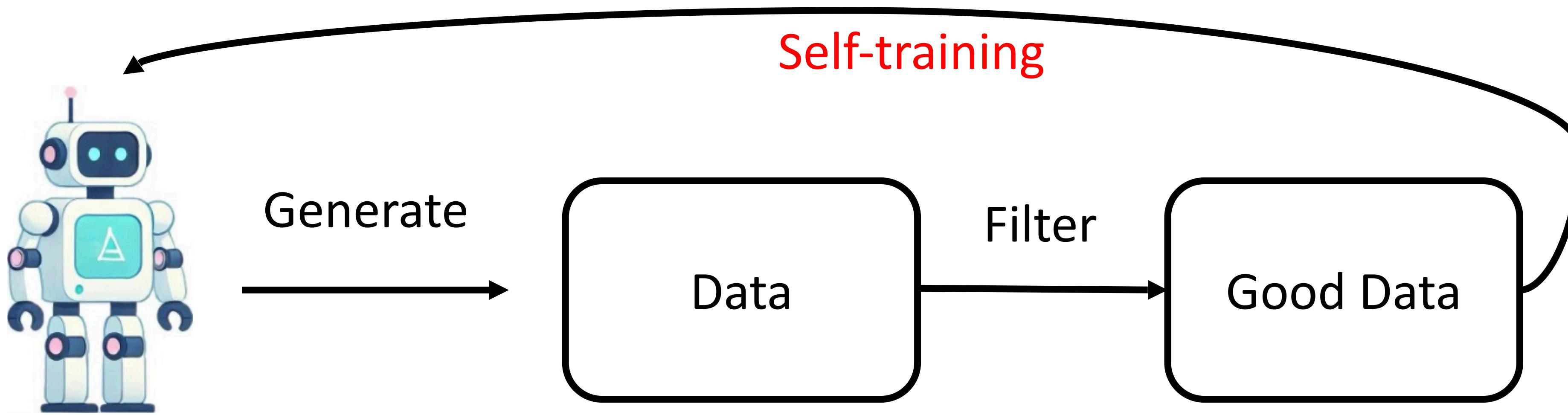
# Recap: Self-Improving



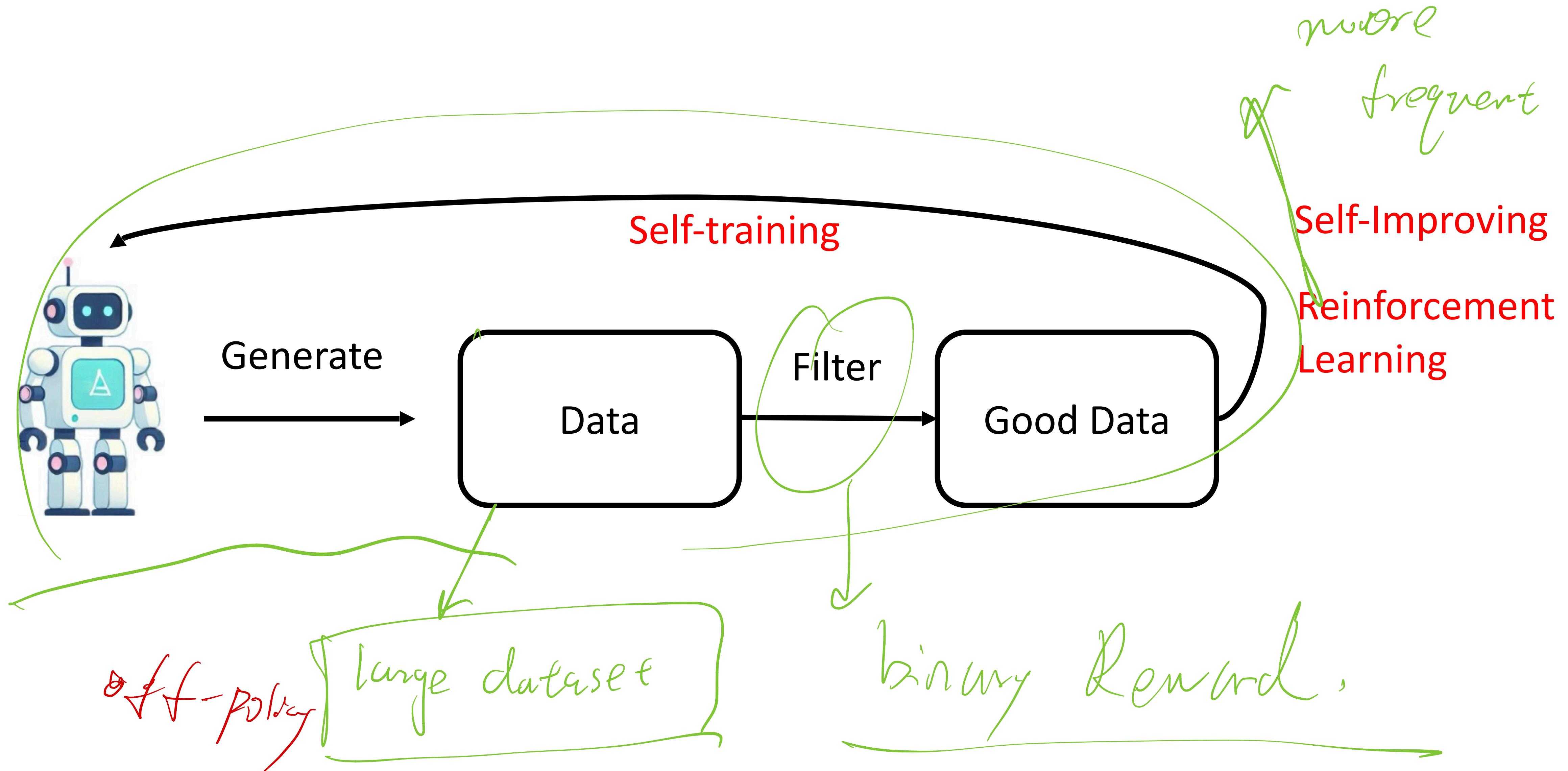
# Recap: Self-Improving



# Recap: Self-Improving



# Recap: Self-Improving



# Test-Time Scaling

## The Bitter Lesson

Rich Sutton

March 13, 2019

The biggest lesson that can be read from 70 years of AI research is that general methods that leverage computation are ultimately the most effective, and by a large margin. The ultimate reason for this is Moore's law, or rather its generalization of continued exponentially falling cost per unit of computation. Most AI research has been conducted as if the computation available to the agent were constant (in which case leveraging human knowledge would be one of the only ways to improve performance) but, over a slightly longer time than a typical research project, massively more computation inevitably becomes available. Seeking an improvement that makes a difference in the shorter term, researchers seek to leverage their human knowledge of the domain, but the only thing that matters in the long run is the leveraging of computation. These two need not run counter to each other, but in practice they tend to. Time spent on one is time not spent on the other. There are psychological commitments to investment in one approach or the other. And the human-knowledge approach tends to complicate methods in ways that make them less suited to taking advantage of general methods leveraging computation. There were many examples of AI researchers' belated learning of this bitter lesson, and it is instructive to review some of the most prominent.

transformer

short answer

no way to spend more compute

# Test-Time Scaling

## The Bitter Lesson

Rich Sutton

March 13, 2019

The biggest lesson that can be read from 70 years of AI research is that general methods that leverage computation are ultimately the most effective, and by a large margin. The ultimate reason for this is Moore's law, or rather its generalization of continued exponentially falling cost per unit of computation. Most AI research has been conducted as if the computation available to the agent were constant (in which case leveraging human knowledge would be one of the only ways to improve performance) but, over a slightly longer time than a typical research project, massively more computation inevitably becomes available. Seeking an improvement that makes a difference in the shorter term, researchers seek to leverage their human knowledge of the domain, but the only thing that matters in the long run is the leveraging of computation. These two need not run counter to each other, but in practice they tend to. Time spent on one is time not spent on the other. There are psychological commitments to investment in one approach or the other. And the human-knowledge approach tends to complicate methods in ways that make them less suited to taking advantage of general methods leveraging computation. There were many examples of AI researchers' belated learning of this bitter lesson, and it is instructive to review some of the most prominent.

# Test-Time Scaling

The most important part about CoT, is that itself can scale, providing a new scaling dimension.

By reasoning longer and longer, the performance gets better and better, this is one way of test-time scaling

This is very natural, because humans think longer when dealing with more complex problems. Before CoT, transformers didn't have such a mechanism!

# Test-Time Scaling

Long CoT opens a new era and paradigm shift (e.g., o1, o3, GPT-5, DeepSeek-R1)

# The Evolution of “LLMs Reasoning”

---



- Wei et al. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. 2022.
- Yu et al. MetaMath: Bootstrap Your Own Mathematical Questions for Large Language Models. 2023
- Yue et al. MAMmoTH: Building Math Generalist Models through Hybrid Instruction Tuning. 2023
- Zelikman et al. STaR: Bootstrapping Reasoning With Reasoning. 2022
- Singh et al. Beyond Human Data: Scaling Self-Training for Problem-Solving with Language Models. 2023
- Lightman et al. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. 2023
- Wang et al. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. 2023
- Shao et al. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. 2024
- OpenAI. Introducing OpenAI o1. 2024
- DeepSeek-AI. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. 2025
- Kimi Team. Kimi k1.5: Scaling Reinforcement Learning with LLMs. 2025

# The Evolution of “LLMs Reasoning”

Prompting CoT  
Reasoning

CoT [Wei et al. 2022]



- Wei et al. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. 2022.  
Yu et al. MetaMath: Bootstrap Your Own Mathematical Questions for Large Language Models. 2023  
Yue et al. MAMmoTH: Building Math Generalist Models through Hybrid Instruction Tuning. 2023  
Zelikman et al. STaR: Bootstrapping Reasoning With Reasoning. 2022  
Singh et al. Beyond Human Data: Scaling Self-Training for Problem-Solving with Language Models. 2023  
Lightman et al. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. 2023  
Wang et al. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. 2023  
Shao et al. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. 2024  
OpenAI. Introducing OpenAI o1. 2024  
DeepSeek-AI. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. 2025  
Kimi Team. Kimi k1.5: Scaling Reinforcement Learning with LLMs. 2025

# The Evolution of “LLMs Reasoning”

Prompting CoT  
Reasoning

Distilling from  
(Stronger )Teachers

CoT [Wei et al. 2022]

MetaMath [Yu et al. 2023]

MAmmoTH [Yue et al. 2023]



Wei et al. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. 2022.

Yu et al. MetaMath: Bootstrap Your Own Mathematical Questions for Large Language Models. 2023

Yue et al. MAmmoTH: Building Math Generalist Models through Hybrid Instruction Tuning. 2023

Zelikman et al. STaR: Bootstrapping Reasoning With Reasoning. 2022

Singh et al. Beyond Human Data: Scaling Self-Training for Problem-Solving with Language Models. 2023

Lightman et al. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. 2023

Wang et al. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. 2023

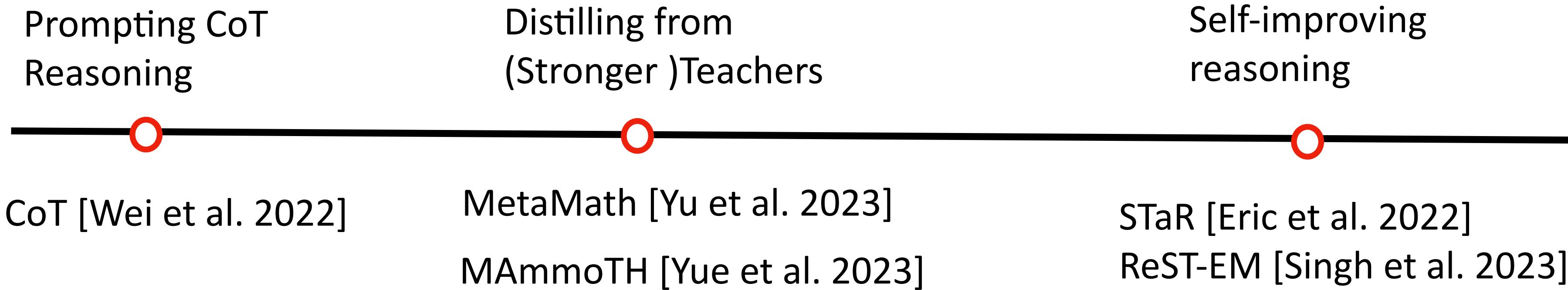
Shao et al. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. 2024

OpenAI. Introducing OpenAI o1. 2024

DeepSeek-AI. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. 2025

Kimi Team. Kimi k1.5: Scaling Reinforcement Learning with LLMs. 2025

# The Evolution of “LLMs Reasoning”



- Wei et al. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. 2022.  
Yu et al. MetaMath: Bootstrap Your Own Mathematical Questions for Large Language Models. 2023  
Yue et al. MAmmoTH: Building Math Generalist Models through Hybrid Instruction Tuning. 2023  
Zelikman et al. STaR: Bootstrapping Reasoning With Reasoning. 2022  
Singh et al. Beyond Human Data: Scaling Self-Training for Problem-Solving with Language Models. 2023  
Lightman et al. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. 2023  
Wang et al. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. 2023  
Shao et al. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. 2024  
OpenAI. Introducing OpenAI o1. 2024  
DeepSeek-AI. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. 2025  
Kimi Team. Kimi k1.5: Scaling Reinforcement Learning with LLMs. 2025

# The Evolution of “LLMs Reasoning”

RLHF

Prompting CoT  
Reasoning

Distilling from  
(Stronger )Teachers

Self-improving  
reasoning

(RL with)  
Model-based  
Process Reward

CoT [Wei et al. 2022]

MetaMath [Yu et al. 2023]

MAmmoTH [Yue et al. 2023]

STaR [Eric et al. 2022]

ReST-EM [Singh et al. 2023]

Verify-step [Lightman et al. 2023]  
Math-Shepherd [Singh et al. 2023]  
DeepSeek-Math [Shao et al. 2023]



Wei et al. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. 2022.

Yu et al. MetaMath: Bootstrap Your Own Mathematical Questions for Large Language Models. 2023

Yue et al. MAmmoTH: Building Math Generalist Models through Hybrid Instruction Tuning. 2023

Zelikman et al. STaR: Bootstrapping Reasoning With Reasoning. 2022

Singh et al. Beyond Human Data: Scaling Self-Training for Problem-Solving with Language Models. 2023

Lightman et al. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. 2023

Wang et al. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. 2023

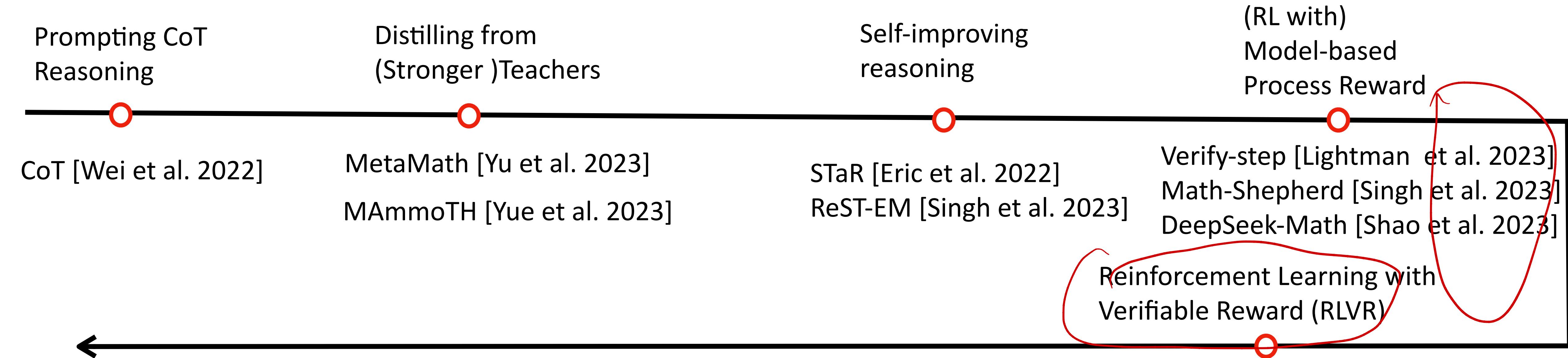
Shao et al. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. 2024

OpenAI. Introducing OpenAI o1. 2024

DeepSeek-AI. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. 2025

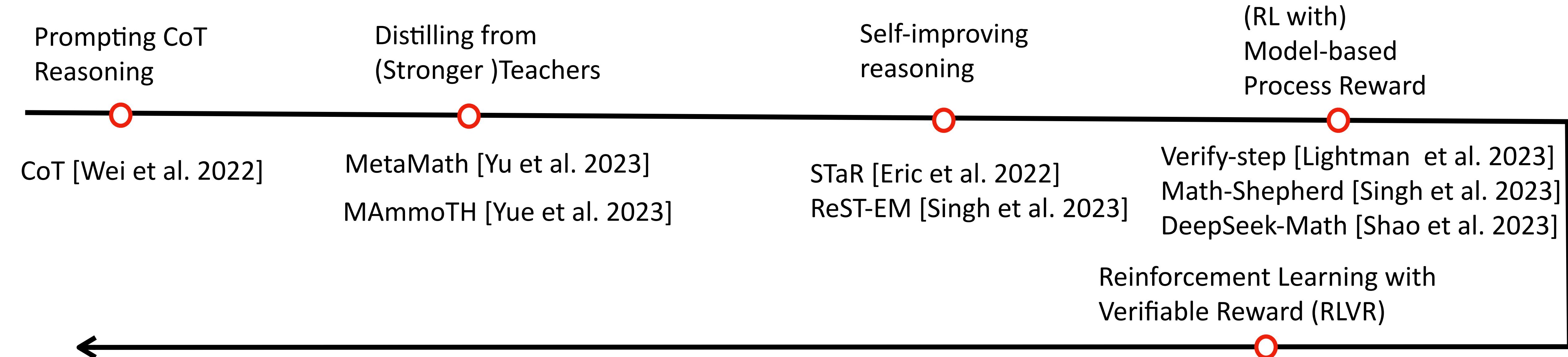
Kimi Team. Kimi k1.5: Scaling Reinforcement Learning with LLMs. 2025

# The Evolution of “LLMs Reasoning”



- Wei et al. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. 2022.  
Yu et al. MetaMath: Bootstrap Your Own Mathematical Questions for Large Language Models. 2023  
Yue et al. MAmmoTH: Building Math Generalist Models through Hybrid Instruction Tuning. 2023  
Zelikman et al. STaR: Bootstrapping Reasoning With Reasoning. 2022  
Singh et al. Beyond Human Data: Scaling Self-Training for Problem-Solving with Language Models. 2023  
Lightman et al. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. 2023  
Wang et al. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. 2023  
Shao et al. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. 2024  
OpenAI. Introducing OpenAI o1. 2024  
DeepSeek-AI. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. 2025  
Kimi Team. Kimi k1.5: Scaling Reinforcement Learning with LLMs. 2025

# The Evolution of “LLMs Reasoning”



- Wei et al. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. 2022.  
Yu et al. MetaMath: Bootstrap Your Own Mathematical Questions for Large Language Models. 2023  
Yue et al. MAmmoTH: Building Math Generalist Models through Hybrid Instruction Tuning. 2023  
Zelikman et al. STaR: Bootstrapping Reasoning With Reasoning. 2022  
Singh et al. Beyond Human Data: Scaling Self-Training for Problem-Solving with Language Models. 2023  
Lightman et al. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. 2023  
Wang et al. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. 2023  
Shao et al. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. 2024  
OpenAI. Introducing OpenAI o1. 2024  
DeepSeek-AI. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. 2025  
Kimi Team. Kimi k1.5: Scaling Reinforcement Learning with LLMs. 2025

# Long Chain of Thought

## O1 Solution

10/2024

Okay, let's see. So Aya goes for a 9-kilometer walk every morning and stops at a coffee shop. The problem gives two scenarios with different speeds and total times, and we need to find the total time when she walks at a different speed. Hmm.

First, I need to parse the information given. The key points are:

1. The distance is always 9 km.
2. The total time includes both walking time and t minutes at the coffee shop.
3. Two different speeds are given, each resulting in different total times, but the coffee shop time t is the same in both cases.

**Wait, let me make sure.** The problem states when she walks at  $s$  km/h, the total time is 4 hours (which includes  $t$  minutes). When she walks at  $s + 2$  km/h, the total time is 2 hours and 24 minutes (which also includes  $t$  minutes). So both cases have the same  $t$ , but different walking times leading to different total times.

So, the total time is walking time plus  $t$ . Since  $t$  is in minutes, but the total time is given in hours and minutes, I need to convert everything to the same units. Maybe convert all times to minutes to avoid confusion.

.....  
Since speed can't be negative,  $s = (5)/2 = 2.5$  km/h

Wait,  $\sqrt{49}=7$ , so  $(-2+7)/2=5/2=2.5$ , yes. So  $s=2.5$  km/h.

**Now check if that makes sense.**

.....  
Third scenario: speed is  $s + 0.5 = 3$  km/h. Walking time =  $9/3 = 3$  hours = 180 minutes. Add  $t = 24$ , total = 204.

**Yes, seems correct.**

# Long Chain of Thought

## O1 Solution

Okay, let's see. So Aya goes for a 9-kilometer walk every morning and stops at a coffee shop. The problem gives two scenarios with different speeds and total times, and we need to find the total time when she walks at a different speed. Hmm.

First, I need to parse the information given. The key points are:

1. The distance is always 9 km.
2. The total time includes both walking time and t minutes at the coffee shop.
3. Two different speeds are given, each resulting in different total times, but the coffee shop time t is the same in both cases.

**Wait, let me make sure.** The problem states when she walks at  $s$  km/h, the total time is 4 hours (which includes  $t$  minutes). When she walks at  $s + 2$  km/h, the total time is 2 hours and 24 minutes (which also includes  $t$  minutes). So both cases have the same  $t$ , but different walking times leading to different total times.

So, the total time is walking time plus  $t$ . Since  $t$  is in minutes, but the total time is given in hours and minutes, I need to convert everything to the same units. Maybe convert all times to minutes to avoid confusion.

.....  
Since speed can't be negative,  $s = (5)/2 = 2.5$  km/h

Wait,  $\sqrt{49}=7$ , so  $(-2+7)/2=5/2=2.5$ , yes. So  $s=2.5$  km/h.

**Now check if that makes sense.**

.....  
Third scenario: speed is  $s + 0.5=3$  km/h. Walking time= $9/3=3$  hours= $180$  minutes. Add  $t=24$ , total= $204$ .

**Yes, seems correct.**

## Long Chain of Thought (CoT)

+ certain cognitive behaviors  
(e.g., Self-Reflection)

# How to Train Models to Reason Longer?

# How to Train Models to Reason Longer?

- Collect Human Data?

# How to Train Models to Reason Longer?

- Collect Human Data? Human does not write down the inherent thinking

# How to Train Models to Reason Longer?

- Collect Human Data? Human does not write down the inherent thinking
- Synthesize/Distill Data from another model?

# How to Train Models to Reason Longer?

- Collect Human Data? Human does not write down the inherent thinking
- Synthesize/Distill Data from another model?  
At that time, only o1 has this ability, but o1 does not reveal its thinking process

# The Limitation of Human Reasoning

## Data in the Wild

Okay, let's see. So Aya goes for a 9-kilometer walk every morning and stops at a coffee shop. The problem gives two scenarios with different speeds and total times, and we need to find the total time when she walks at a different speed. Hmm.

First, I need to parse the information given. The key points are:

1. The distance is always 9 km.
2. The total time includes both walking time and  $t$  minutes at the coffee shop.
3. Two different speeds are given, each resulting in different total times, but the coffee shop time  $t$  is the same in both cases.

**Wait, let me make sure.** The problem states when she walks at  $s$  km/h, the total time is 4 hours (which includes  $t$  minutes). When she walks at  $s + 2$  km/h, the total time is 2 hours and 24 minutes (which also includes  $t$  minutes). So both cases have the same  $t$ , but different walking times leading to different total times.

So, the total time is walking time plus  $t$ . Since  $t$  is in minutes, but the total time is given in hours and minutes, I need to convert everything to the same units. Maybe convert all times to minutes to avoid confusion.

.....  
Since speed can't be negative,  $s = (5)/2 = 2.5$  km/h

Wait,  $\sqrt{49}=7$ , so  $(-2+7)/2=5/2=2.5$ , yes. So  $s=2.5$  km/h.

**Now check if that makes sense.**

.....  
Third scenario: speed is  $s + 0.5 = 3$  km/h. Walking time =  $9/3 = 3$  hours = 180 minutes. Add  $t = 24$ , total = 204.

**Yes, seems correct.**

# The Limitation of Human Reasoning

## Data in the Wild

Okay, let's see. So Aya goes for a 9-kilometer walk every morning and stops at a coffee shop. The problem gives two scenarios with different speeds and total times, and we need to find the total time when she walks at a different speed. Hmm.

First, I need to parse the information given. The key points are:

1. The distance is always 9 km.
2. The total time includes both walking time and t minutes at the coffee shop.
3. Two different speeds are given, each resulting in different total times, but the coffee shop time t is the same in both cases.

**Wait, let me make sure.** The problem states when she walks at  $s$  km/h, the total time is 4 hours (which includes  $t$  minutes). When she walks at  $s + 2$  km/h, the total time is 2 hours and 24 minutes (which also includes  $t$  minutes). So both cases have the same  $t$ , but different walking times leading to different total times.

So, the total time is walking time plus  $t$ . Since  $t$  is in minutes, but the total time is given in hours and minutes, I need to convert everything to the same units. Maybe convert all times to minutes to avoid confusion.

.....

Since speed can't be negative,  $s = (5)/2 = 2.5$  km/h

Wait,  $\sqrt{49}=7$ , so  $(-2+7)/2=5/2=2.5$ , yes. So  $s=2.5$  km/h.

Now check if that makes sense.

.....

Third scenario: speed is  $s + 0.5=3$  km/h. Walking time= $9/3=3$  hours=180 minutes. Add  $t=24$ , total=204.

Yes, seems correct.

Humans may confirm, reflect, experience many incorrect trials to reach the final solution, but humans often only write down the final, correct solution without the intermediate draft

# <think> 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:

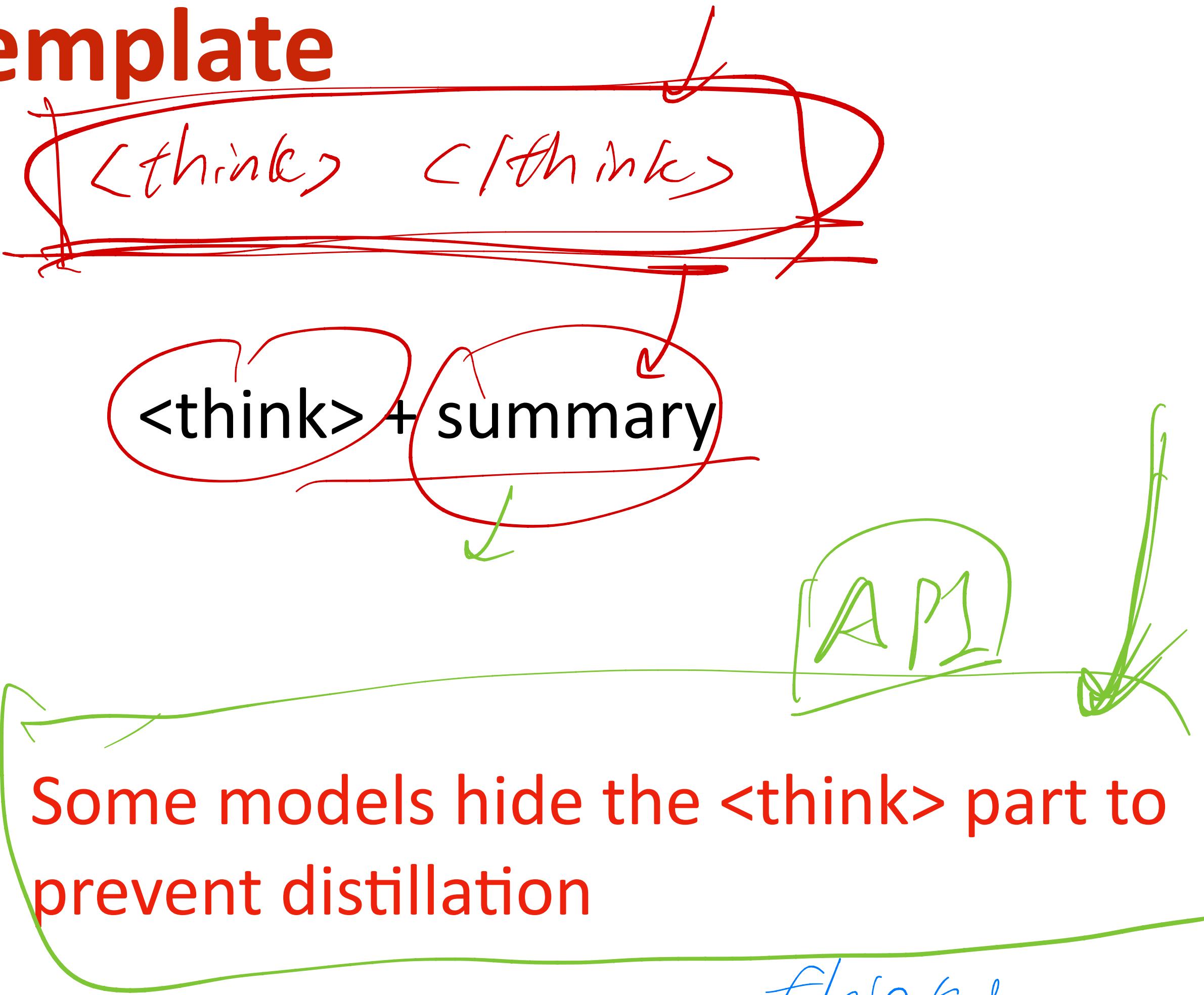
System (global) Prompt

`<think>`

`</think>`

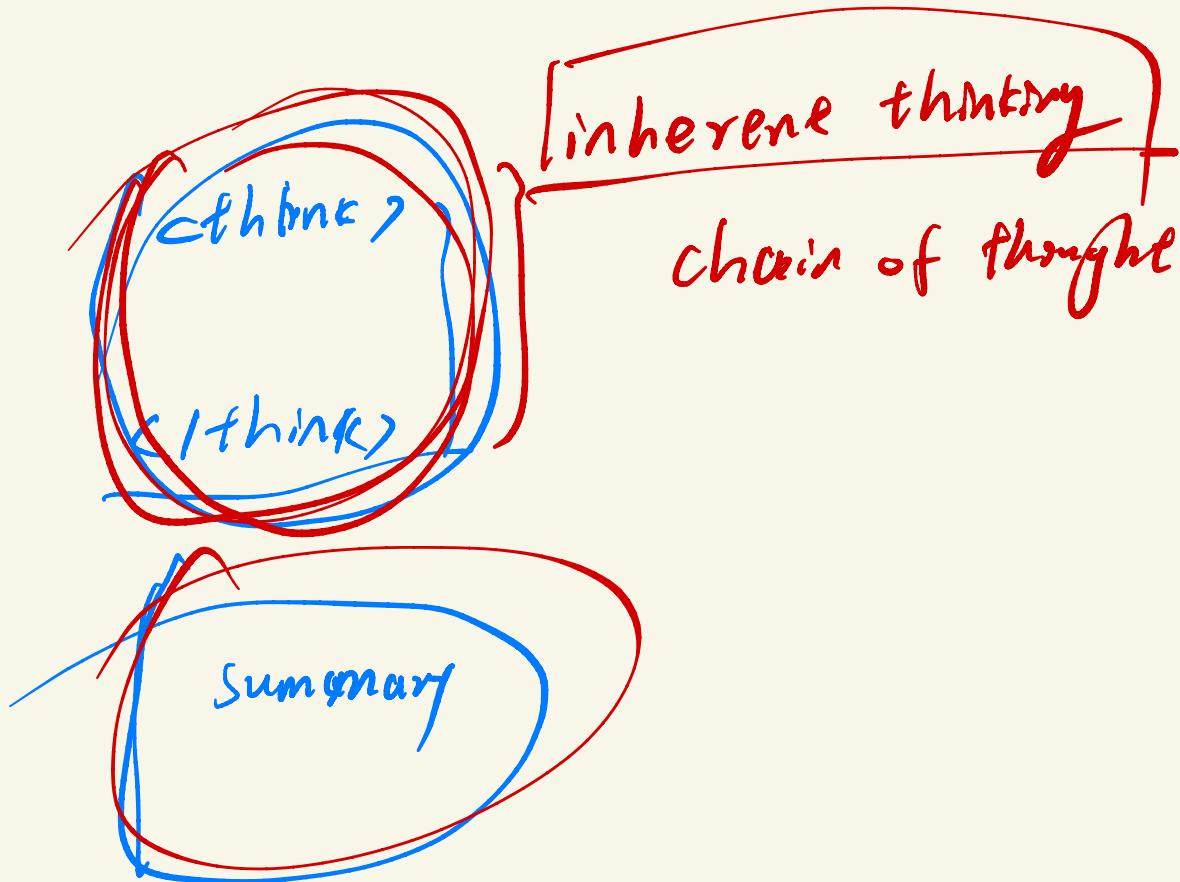
# **<think> Template**

```
>>> Solve  $4x + 3 < 6x + 7$ 
<think>
I start by isolating the variable x. To do this, I subtract  $4x$  from both sides of the
inequality to get  $3 < 2x + 7$ .
Next, I need to isolate  $2x$  on its own. I'll subtract 7 from both sides, resulting in  $-4 < 2x$ .
Finally, I divide both sides by 2 to solve for x, which gives me  $-2 < x$ .
</think>
To solve the inequality  $(4x + 3 < 6x + 7)$ , follow these steps:
1. **Subtract  $(4x)$  from both sides** to begin isolating the variable term:
\[
4x + 3 - 4x < 6x + 7 - 4x
\]
Simplifying this gives:
\[
3 < 2x + 7
\]
2. **Subtract  $(7)$  from both sides** to further isolate the term with  $(x)$ :
\[
3 - 7 < 2x + 7 - 7
\]
Simplifying this gives:
\[
-4 < 2x
\]
3. **Divide both sides by  $(2)$ ** to solve for  $(x)$ :
\[
\frac{-4}{2} < \frac{2x}{2}
\]
Simplifying this gives:
\[
-2 < x
\]
Alternatively, since  $(x > -2)$  is the same as  $(-2 < x)$ , you can express it as:
\[
x > -2
\]
**Final Answer:**
\boxed{x > -2}
```



Some models hide the **<think>** part to prevent distillation

*thinking*  
**<think>** -- - - - -  
**<think>** response



# Recap: Policy Gradient

# Recap: Policy Gradient

$$\text{Objective} = \sum_{i=1}^n \frac{1}{n} R(x^{(i)}) \log p_\theta(x^{(i)})$$

reward model

from  $p_\theta(x)$

$$x^{(1)}, \dots, x^{(n)} \sim p_\theta(x)$$

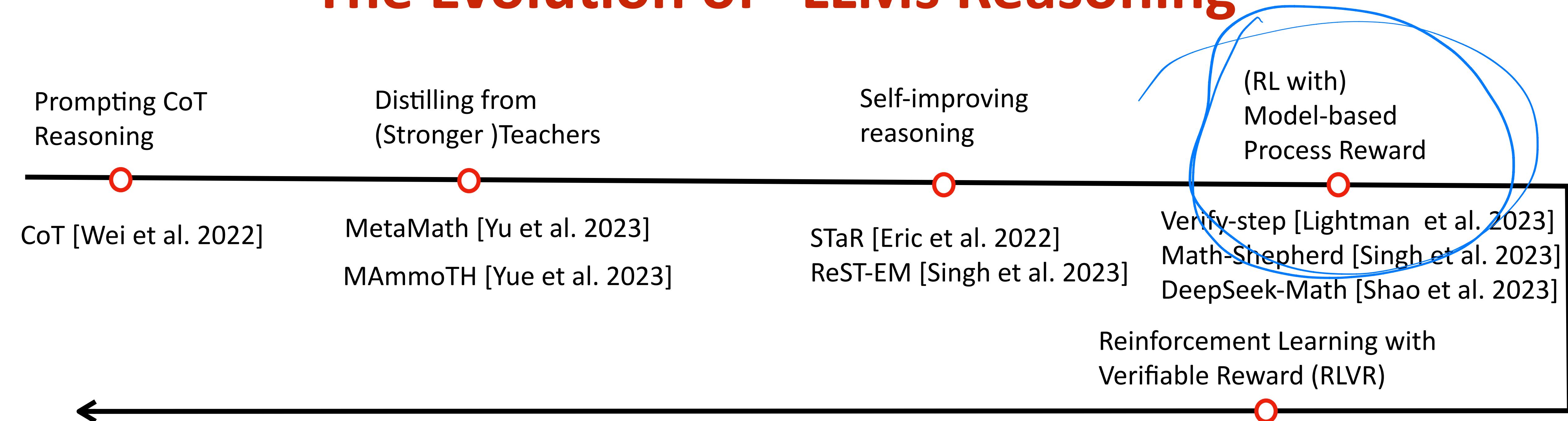
# Recap: Policy Gradient

$$\text{Objective} = \sum_{i=1}^n \frac{1}{n} R(x^{(i)}) \log p_\theta(x^{(i)}) \quad x^{(1)}, \dots, x^{(n)} \sim p_\theta(x)$$

$R()$  is a reward model in RLHF

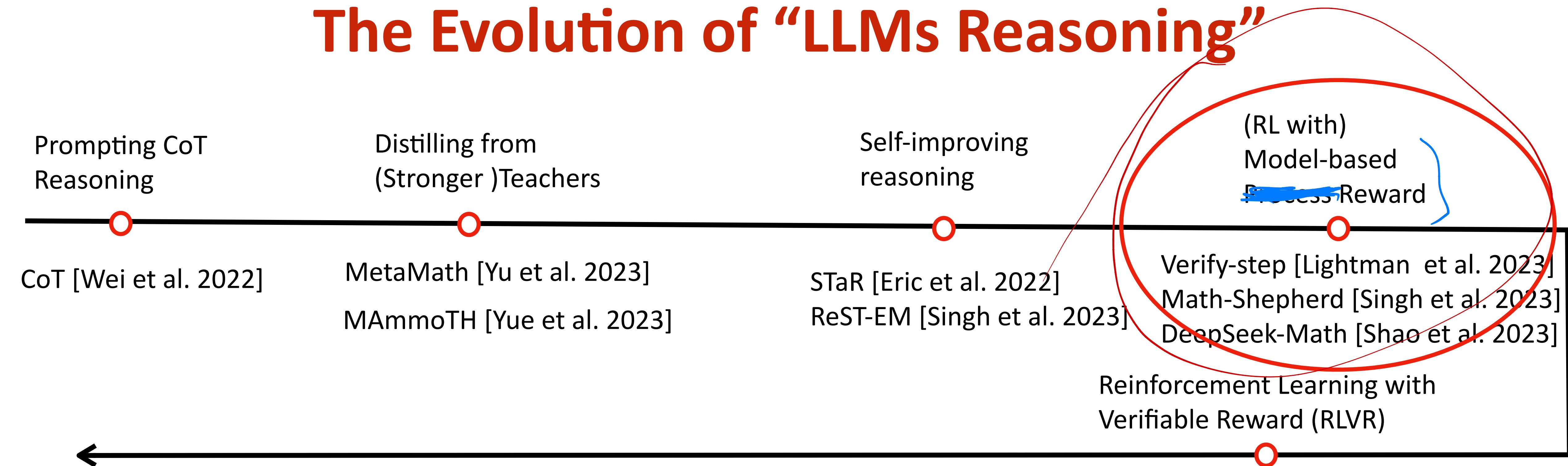
$R_C$

# The Evolution of “LLMs Reasoning”



- Wei et al. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. 2022.  
Yu et al. MetaMath: Bootstrap Your Own Mathematical Questions for Large Language Models. 2023  
Yue et al. MAmmoTH: Building Math Generalist Models through Hybrid Instruction Tuning. 2023  
Zelikman et al. STaR: Bootstrapping Reasoning With Reasoning. 2022  
Singh et al. Beyond Human Data: Scaling Self-Training for Problem-Solving with Language Models. 2023  
Lightman et al. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. 2023  
Wang et al. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. 2023  
Shao et al. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. 2024  
OpenAI. Introducing OpenAI o1. 2024  
DeepSeek-AI. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. 2025  
Kimi Team. Kimi k1.5: Scaling Reinforcement Learning with LLMs. 2025

# The Evolution of “LLMs Reasoning”



Wei et al. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. 2022.

Yu et al. MetaMath: Bootstrap Your Own Mathematical Questions for Large Language Models. 2023

Yue et al. MAmmoTH: Building Math Generalist Models through Hybrid Instruction Tuning. 2023

Zelikman et al. STaR: Bootstrapping Reasoning With Reasoning. 2022

Singh et al. Beyond Human Data: Scaling Self-Training for Problem-Solving with Language Models. 2023

Lightman et al. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. 2023

Wang et al. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. 2023

Shao et al. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. 2024

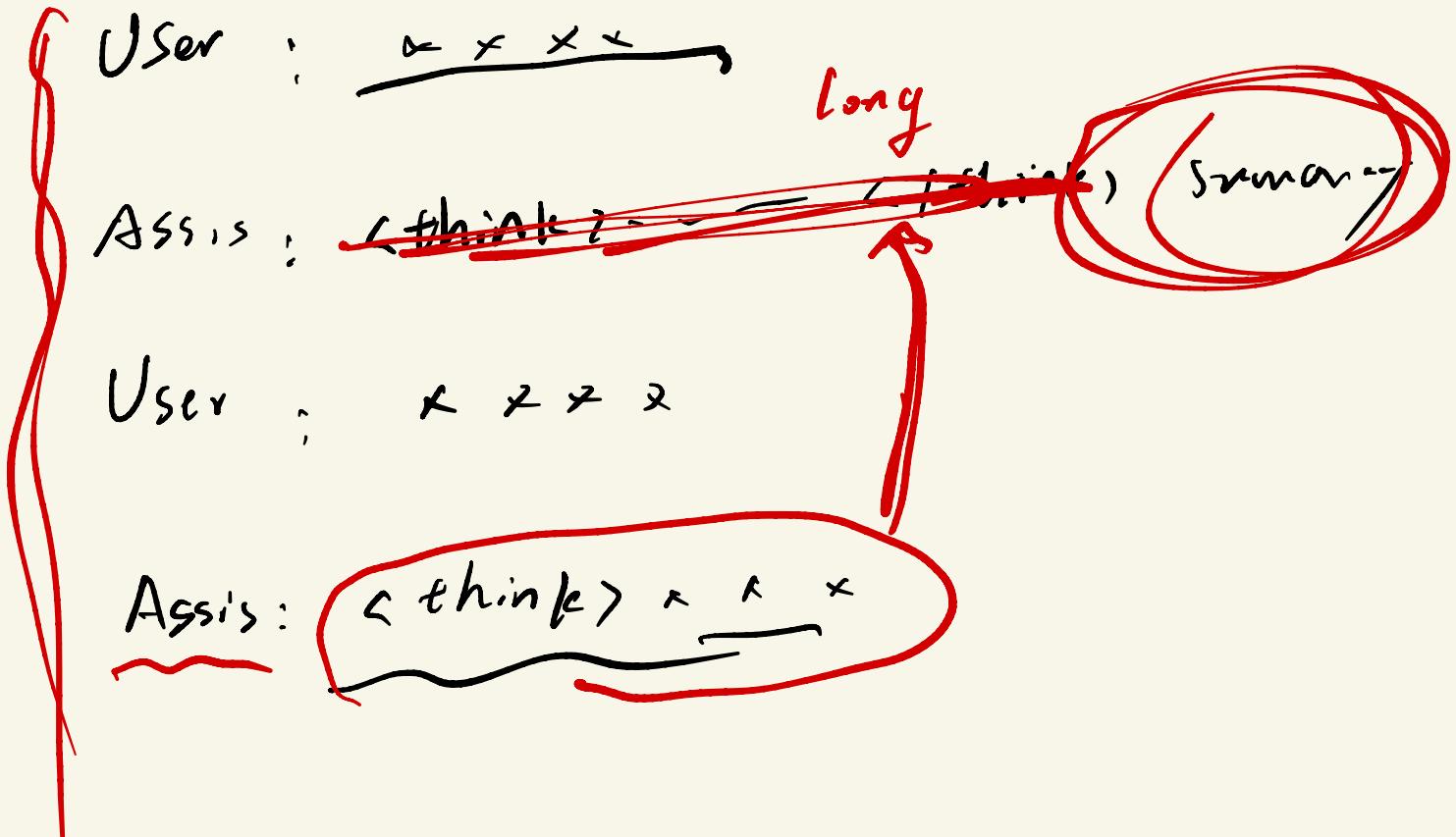
OpenAI. Introducing OpenAI o1. 2024

DeepSeek-AI. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. 2025

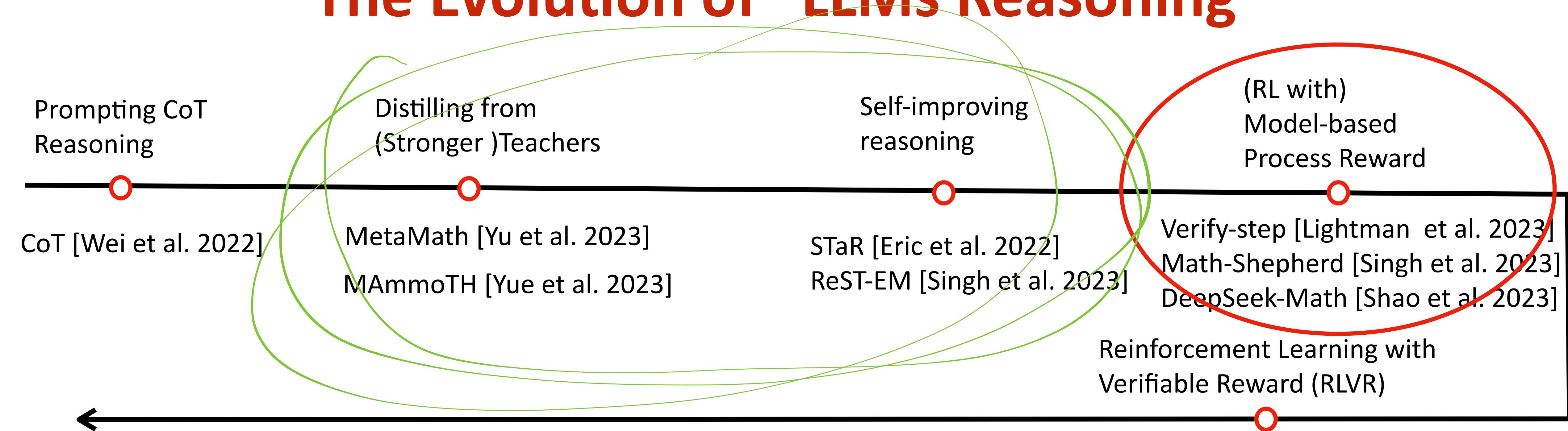
Kimi Team. Kimi k1.5: Scaling Reinforcement Learning with LLMs. 2025

# Multi - turn Conv

$O(n^2)$



# The Evolution of “LLMs Reasoning”



Wei et al. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. 2022.

Yu et al. MetaMath: Bootstrap Your Own Mathematical Questions for Large Language Models. 2023

Yue et al. MAmmoTH: Building Math Generalist Models through Hybrid Instruction Tuning. 2023

Zelikman et al. STaR: Bootstrapping Reasoning With Reasoning. 2022

Singh et al. Beyond Human Data: Scaling Self-Training for Problem-Solving with Language Models. 2023

Lightman et al. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. 2023

Wang et al. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. 2023

Shao et al. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. 2024

OpenAI. Introducing OpenAI o1. 2024

DeepSeek-AI. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. 2025

Kimi Team. Kimi k1.5: Scaling Reinforcement Learning with LLMs. 2025

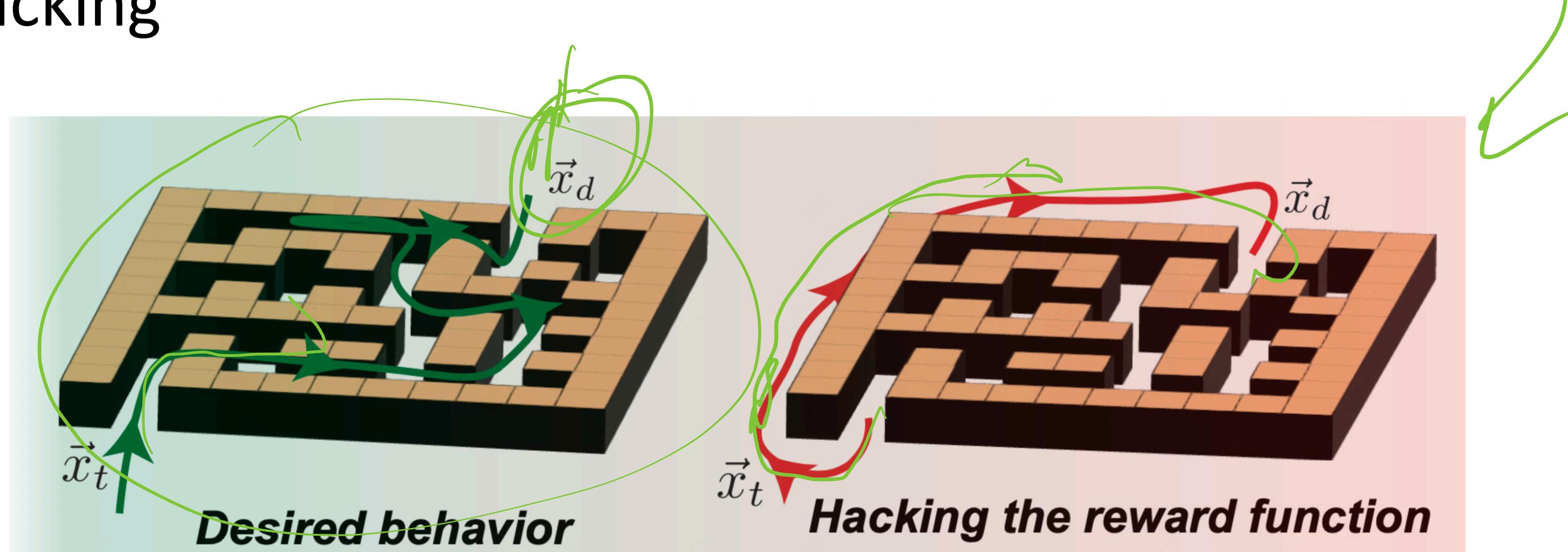
**Reward model did not work well in reasoning**

# Why Reward Models Fail?

- Reward model can produce inaccurate judgement, making the feedback noisy (judging correctness of mathematical reasoning is hard than judging simple responses)

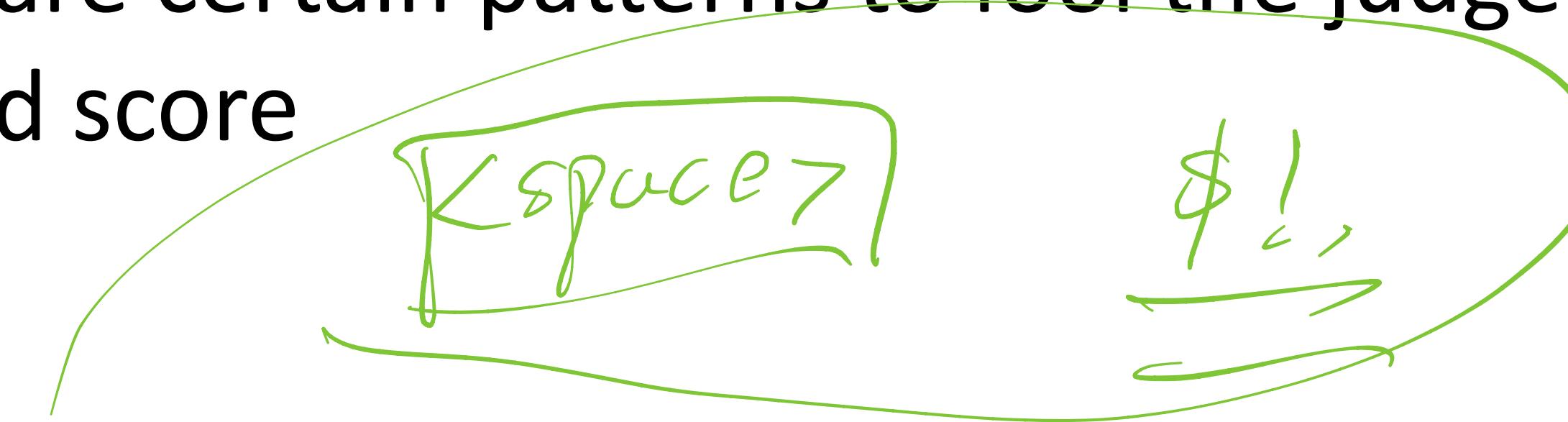
# Why Reward Models Fail?

- Reward model can produce inaccurate judgement, making the feedback noisy (judging correctness of mathematical reasoning is hard than judging simple responses)
- Reward hacking



# Reward Hacking

- In LLMs, maybe there are certain patterns to fool the judge model to make it always output a good score



# Reward Hacking

- In LLMs, maybe there are certain patterns to fool the judge model to make it always output a good score
- If such patterns/weaknesses exist, it is very likely to be learned by the policy model during RL

*max Reward* → *policy* → *[\$, !]*

# Reward Hacking

- In LLMs, maybe there are certain patterns to fool the judge model to make it always output a good score
- If such patterns/weaknesses exist, it is very likely to be learned by the policy model during RL

Reward mode is not robust

# Reinforcement Learning with Verifiable Reward (RLVR)

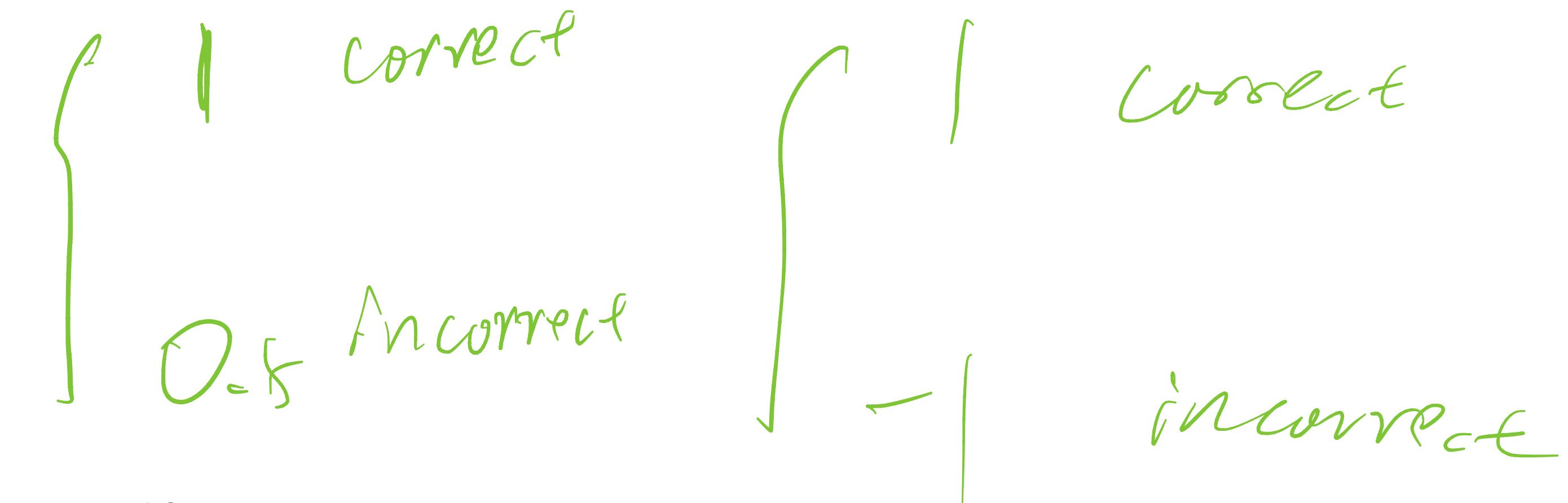
$$\text{Objective} = \sum_{i=1}^n \frac{1}{n} R(x^{(i)}) \log p_\theta(x^{(i)}) \quad x^{(1)}, \dots, x^{(n)} \sim p_\theta(x)$$

Make  $R()$  as a rule-based reward, so that it can never be hacked

# Reinforcement Learning with Verifiable Reward (RLVR)

$$\text{Objective} = \sum_{i=1}^n \frac{1}{n} R(x^{(i)}) \log p_\theta(x^{(i)}) \quad x^{(1)}, \dots, x^{(n)} \sim p_\theta(x)$$

Make  $R()$  as a rule-based reward, so that it can never be hacked



# What is Rule-based Reward?

```
>>> Solve 4x + 3 < 6x + 7
<think>
I start by isolating the variable x. To do this, I subtract 4x from both sides of the
inequality to get 3 < 2x + 7.

Next, I need to isolate 2x on its own. I'll subtract 7 from both sides, resulting in -4
< 2x.

Finally, I divide both sides by 2 to solve for x, which gives me -2 < x.
</think>

To solve the inequality  $(4x + 3 < 6x + 7)$ , follow these steps:

1. **Subtract  $(4x)$  from both sides** to begin isolating the variable term:

$$4x + 3 - 4x < 6x + 7 - 4x$$

Simplifying this gives:

$$3 < 2x + 7$$


2. **Subtract  $(7)$  from both sides** to further isolate the term with  $(x)$ :

$$3 - 7 < 2x + 7 - 7$$

Simplifying this gives:

$$-4 < 2x$$


3. **Divide both sides by  $(2)$ ** to solve for  $(x)$ :

$$\frac{-4}{2} < \frac{2x}{2}$$

Simplifying this gives:

$$-2 < x$$

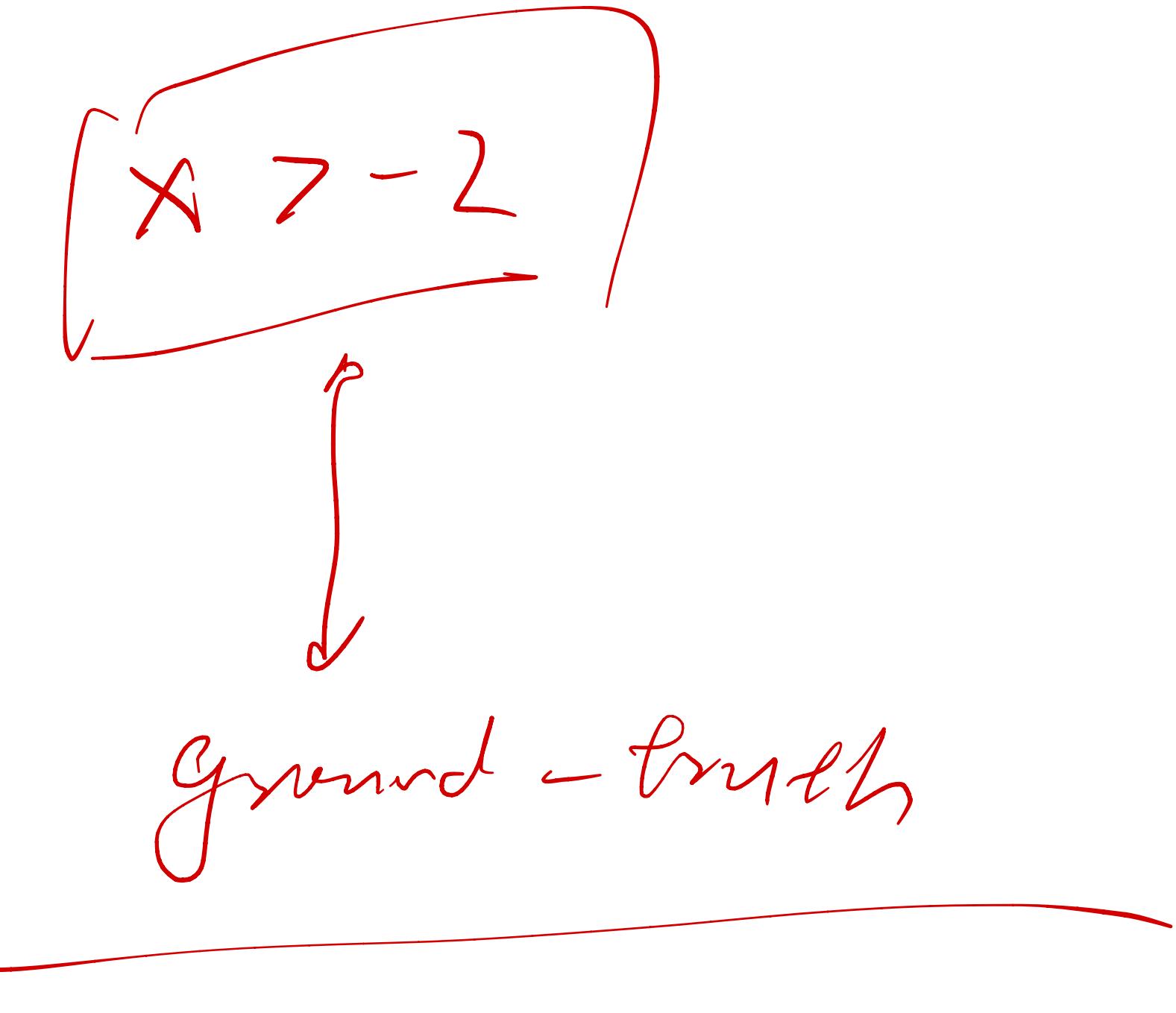

Alternatively, since  $(x > -2)$  is the same as  $(-2 < x)$ , you can express it as:

$$x > -2$$


**Final Answer:**  


$$\boxed{x > -2}$$

```



# What is Rule-based Reward?

```
>>> Solve 4x + 3 < 6x + 7
<think>
I start by isolating the variable x. To do this, I subtract 4x from both sides of the
inequality to get 3 < 2x + 7.

Next, I need to isolate 2x on its own. I'll subtract 7 from both sides, resulting in -4
< 2x.

Finally, I divide both sides by 2 to solve for x, which gives me -2 < x.
</think>

To solve the inequality  $(4x + 3 < 6x + 7)$ , follow these steps:

1. **Subtract  $(4x)$  from both sides** to begin isolating the variable term:
\[
4x + 3 - 4x < 6x + 7 - 4x
\]
Simplifying this gives:
\[
3 < 2x + 7
\]

2. **Subtract  $(7)$  from both sides** to further isolate the term with  $(x)$ :
\[
3 - 7 < 2x + 7 - 7
\]
Simplifying this gives:
\[
-4 < 2x
\]

3. **Divide both sides by  $(2)$ ** to solve for  $(x)$ :
\[
\frac{-4}{2} < \frac{2x}{2}
\]
Simplifying this gives:
\[
-2 < x
\]

Alternatively, since  $(x > -2)$  is the same as  $(-2 < x)$ , you can express it as:
\[
x > -2
\]

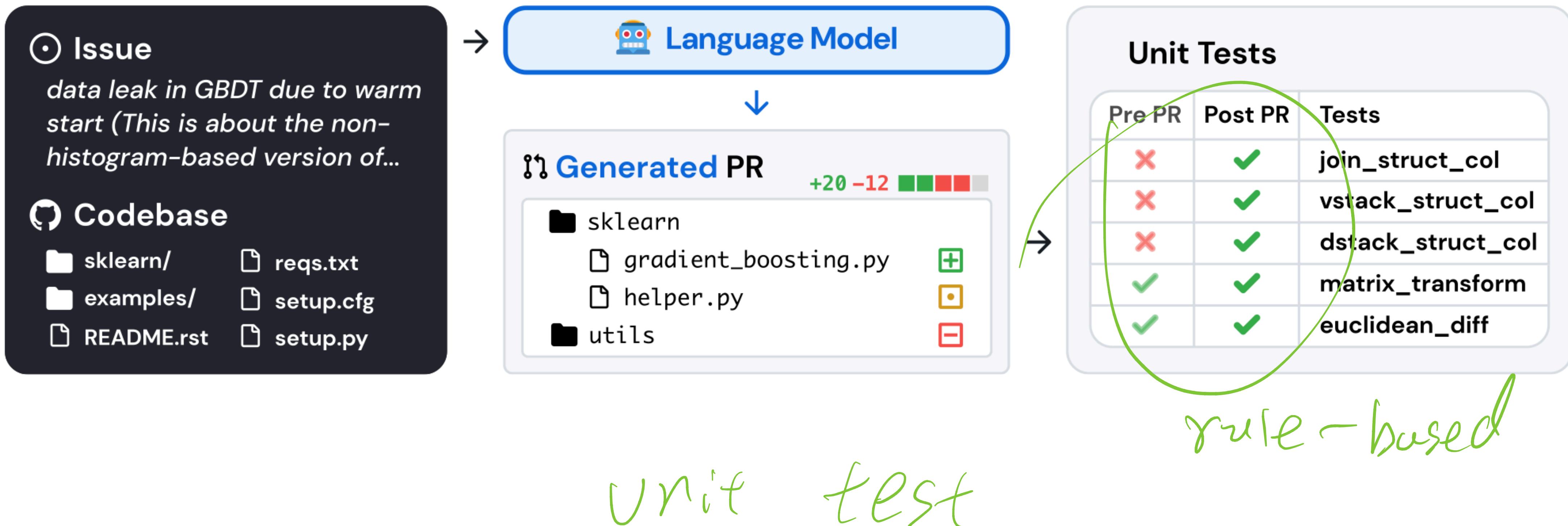
**Final Answer:**
\boxed{x > -2}
```

require knowing

ground-truth

For mathematical reasoning, extract the final answer and compare with the ground-truth

# What is Rule-based Reward?



mathematical Reasoning  
code generation

Verifiable tasks

essay

long answer

RLQR

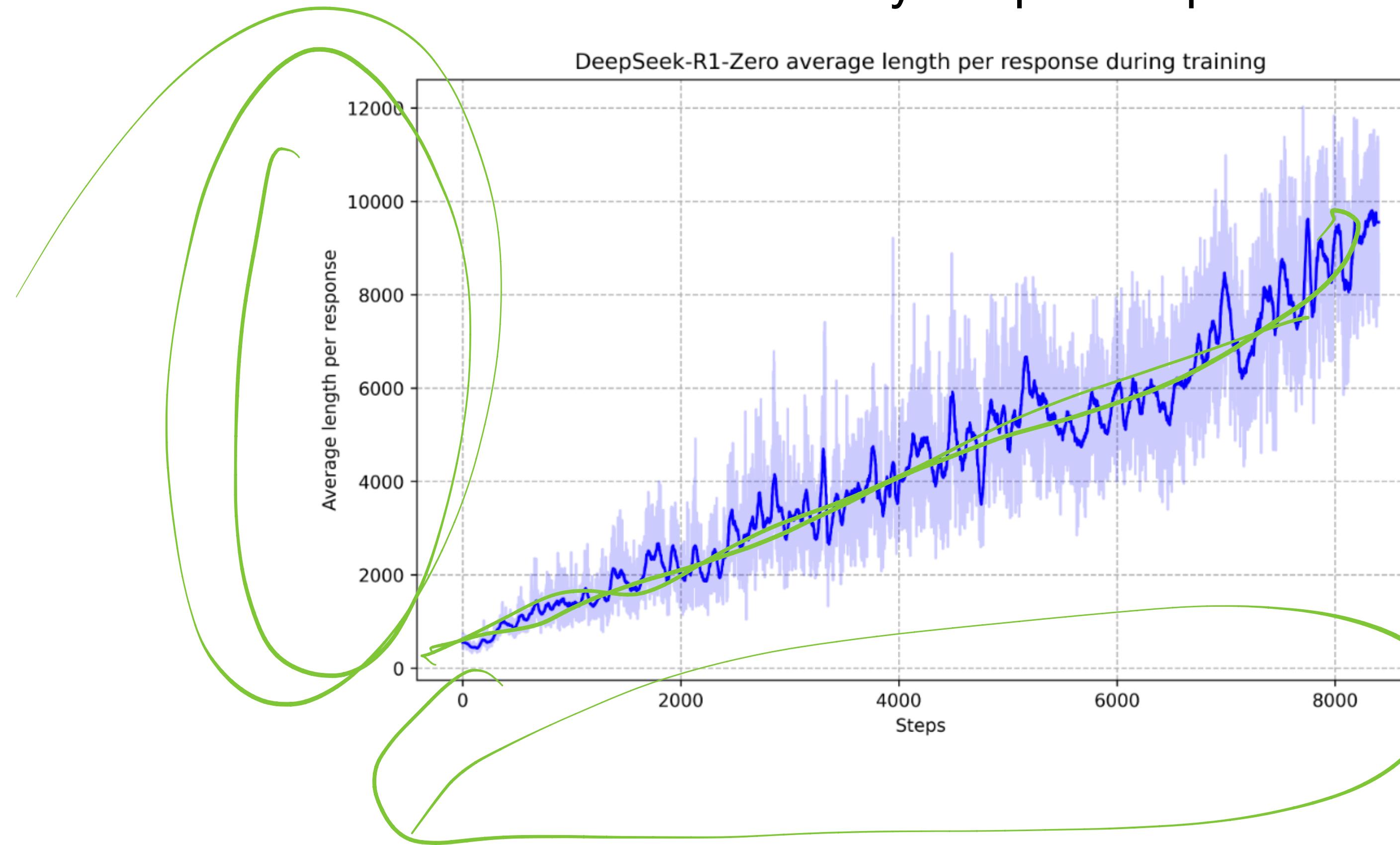
suggestion/investment

# Long CoT Patterns Emerging in RLVR Training

DeepSeek-R1 and Kimi-k1.5 choose the extremely simple recipe – Reinforcement Learning

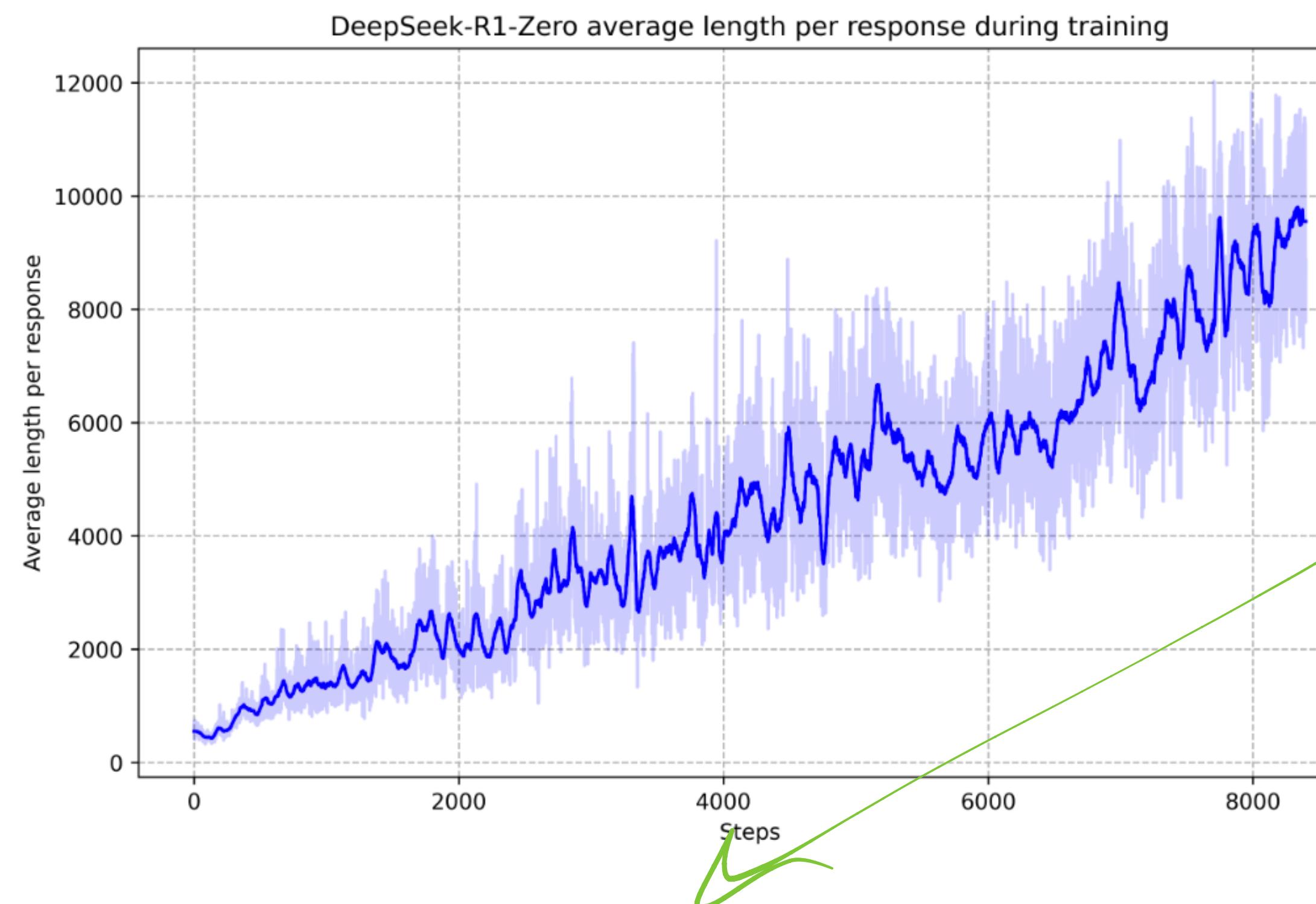
# Long CoT Patterns Emerging in RLVR Training

DeepSeek-R1 and Kimi-k1.5 choose the extremely simple recipe – Reinforcement Learning



# Long CoT Patterns Emerging in RLVR Training

DeepSeek-R1 and Kimi-k1.5 choose the extremely simple recipe – Reinforcement Learning



Model naturally develops long CoT during RL

# Self-Reflection Emerging in RL Training

Example:

---

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 ...

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

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 ...

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: ...

...

skylane RL > SFT

# Self-Reflection Emerging in RL Training

Example:

---

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 ...

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

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 ...

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: ...

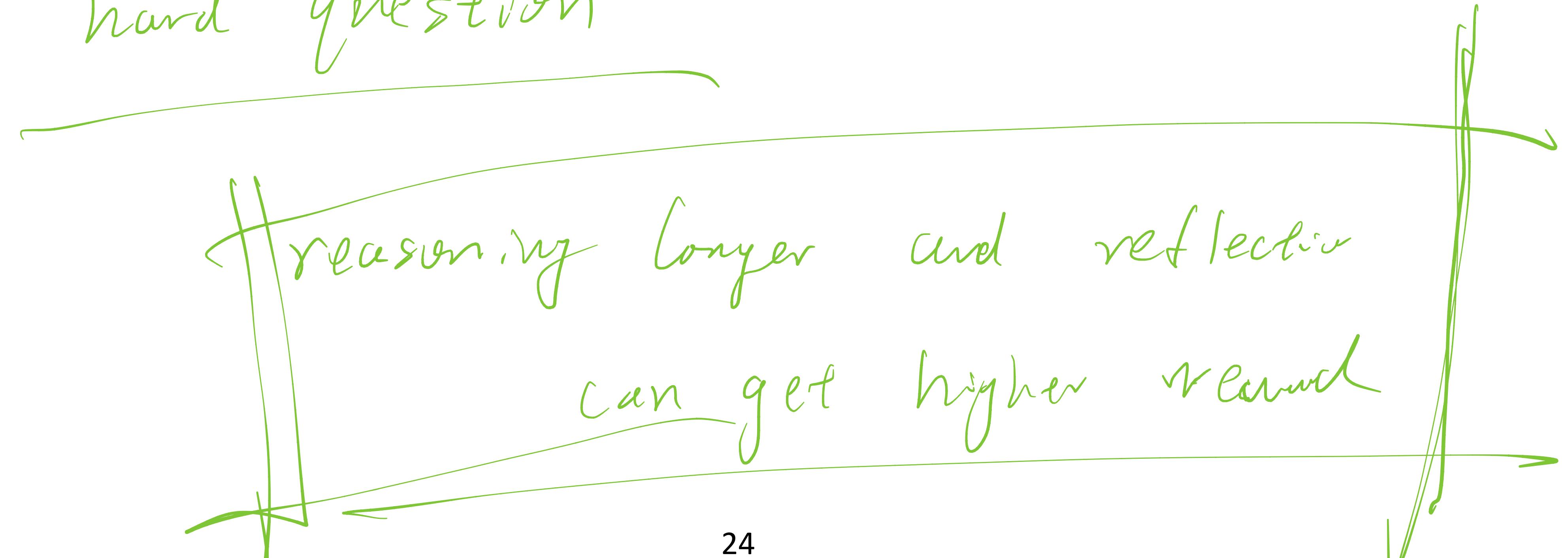
...

---

Without reward model, long CoT and self-reflection just naturally emerge.



hard question



**RL Training is not a new thing**

**RL Training is not a new thing**

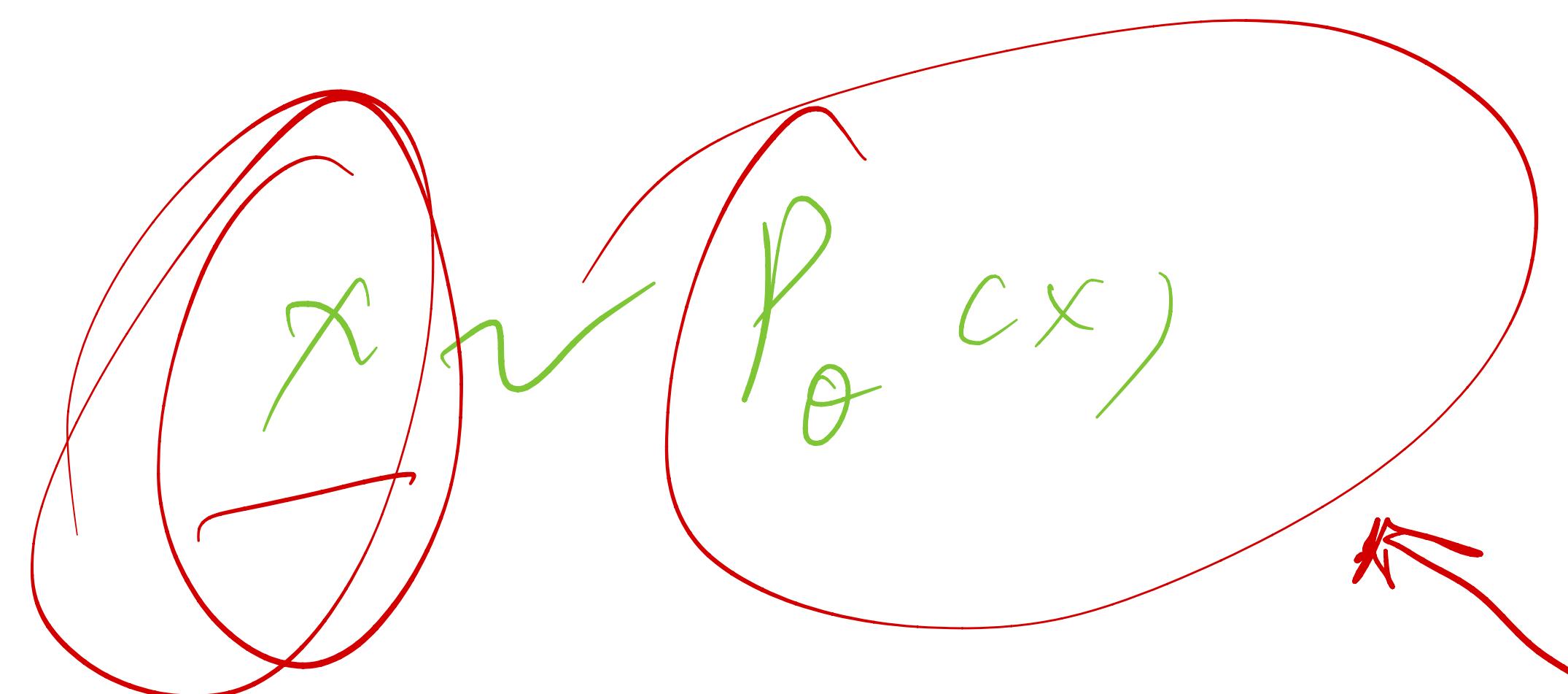
**But the emergence of  
Long CoT plus Self Reflection is new**

NN

Turing Award

**RL finally works, because it has a  
strong prior (LLMs)**

Raven



protruding

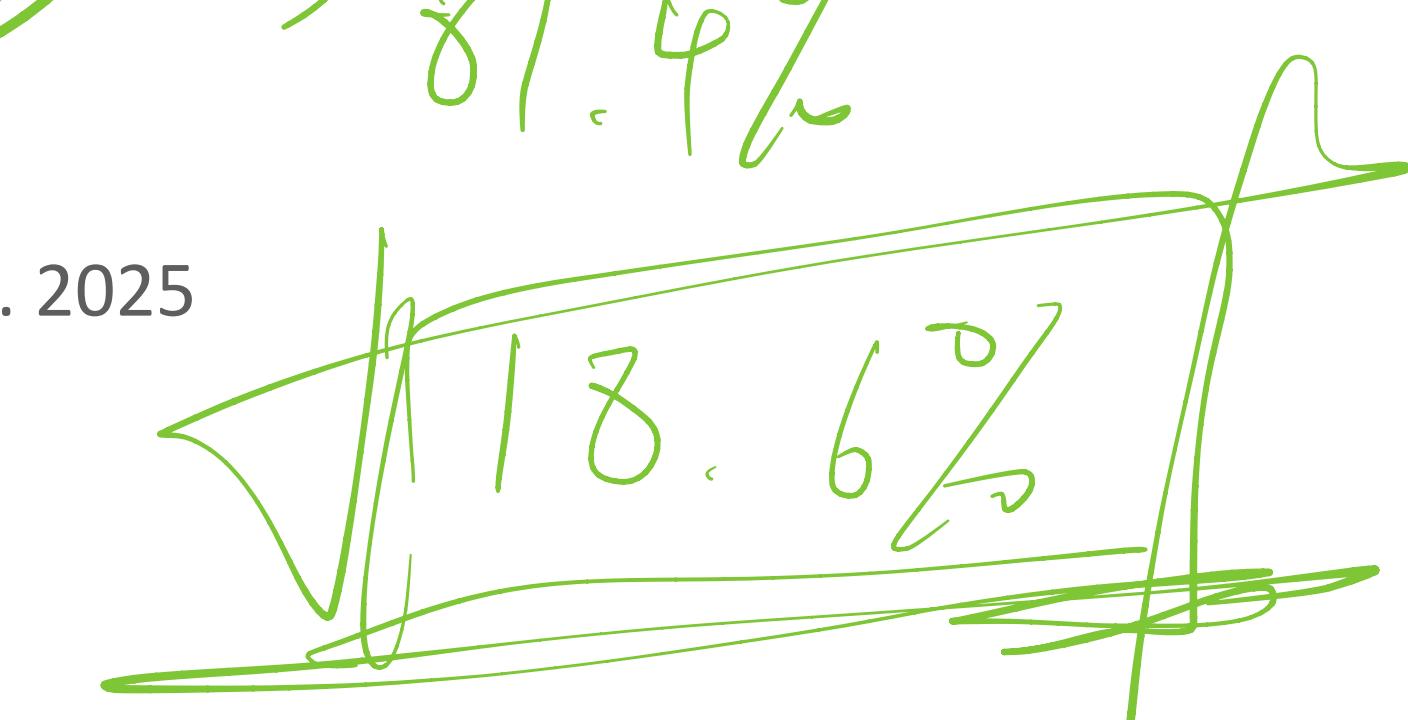
# RL Training only Updates a Sparse Network

Algo.	Init Model	RL Model	Update Sparsity	On-Policy	KL	Online
DPO	Llama-3.1-Tulu-3-8B-SFT	Llama-3.1-Tulu-3-8B-DPO	81.4	x	✓	x
	Llama-3.1-Tulu-3-70B-SFT	Llama-3.1-Tulu-3-70B-DPO	95.2	x	✓	x
GRPO	deepseek-math-7b-instruct	deepseek-math-7b-rl	68.5	✓	✓	✓
	DeepSeek v3 base	DeepSeek-R1-Zero	86.0	✓	✓	✓
ORPO	mistral-7B-v0.1	mistral-orpo-beta	76.9	x	x	x
KTO	Eurus-7b-sft	Eurus-7b-kto	96.0	x	✓	x
	Llama-3-Base-8B-SFT	Llama-3-Base-8B-SFT-KTO	81.2	x	✓	x
PPO	mistral-7b-sft	math-shepherd-mistral-7b-rl	80.8	✓	✓	✓
SimPO	Meta-Llama-3-8B-Instruct	Llama-3-Instruct-8B-SimPO	86.5	x	x	x
PRIME	Eurus-2-7b-sft	Eurus-2-7B-PRIME	77.0	✓	x	✓

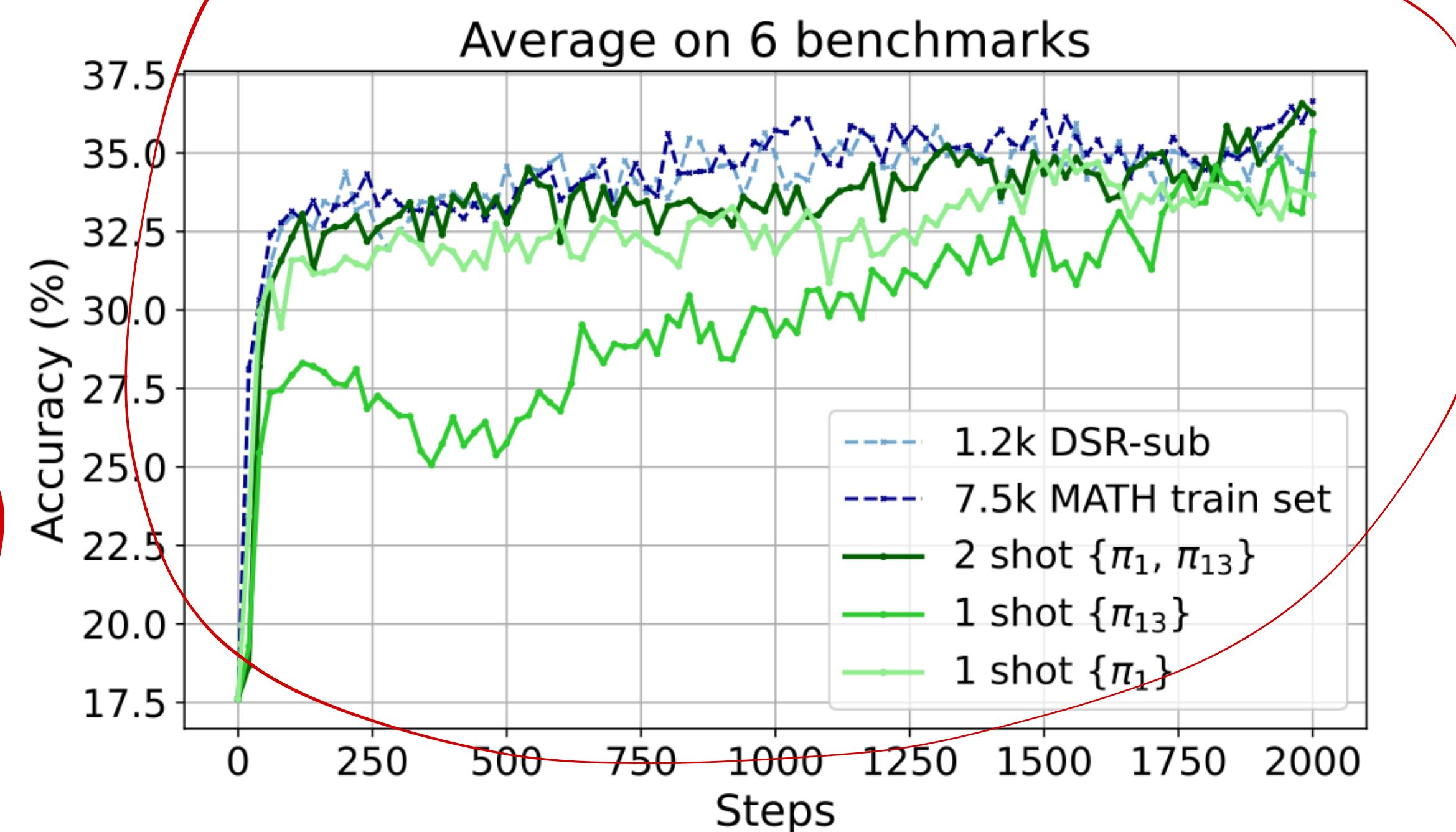
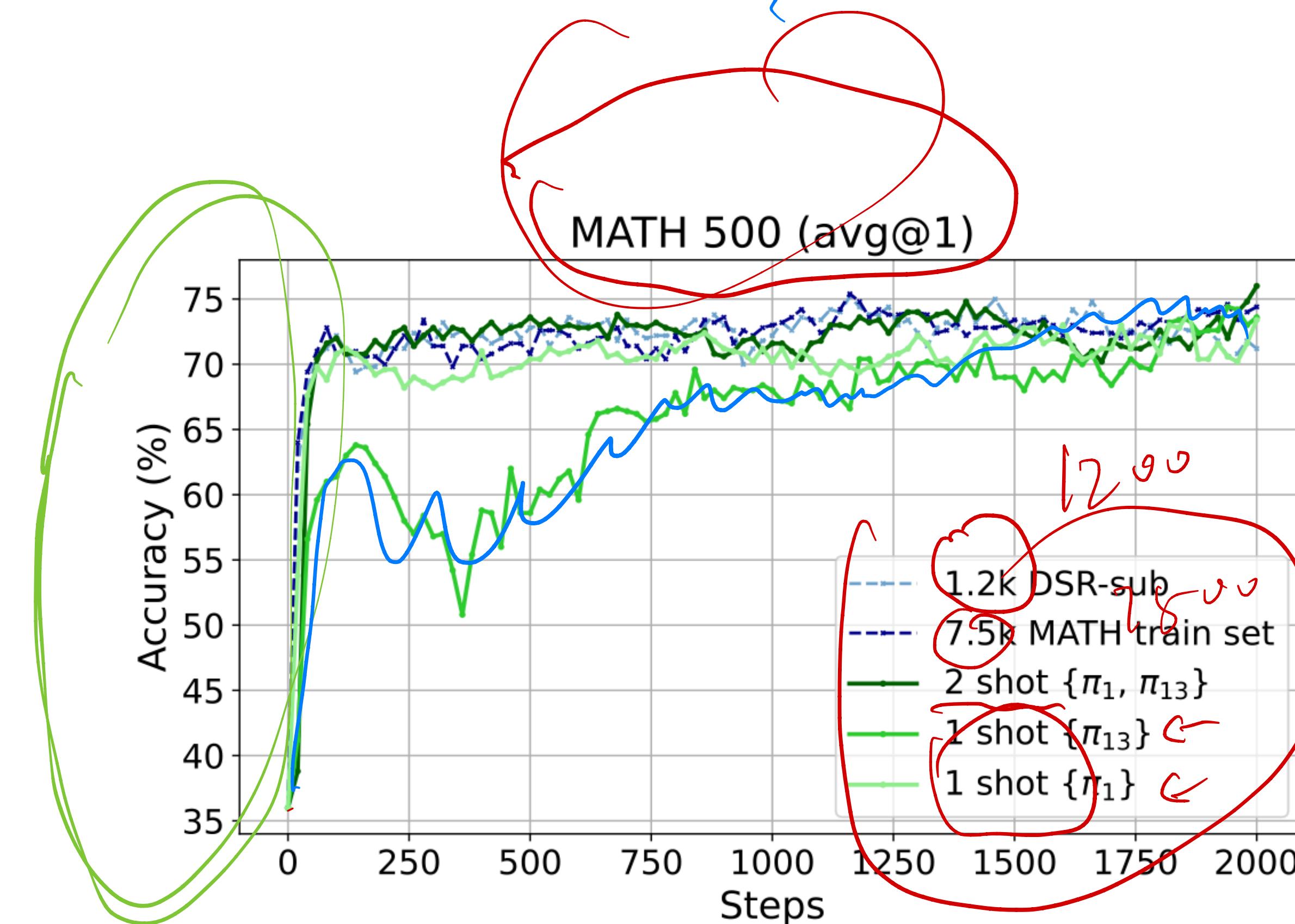
LLMs do not need to

81.4%

change much

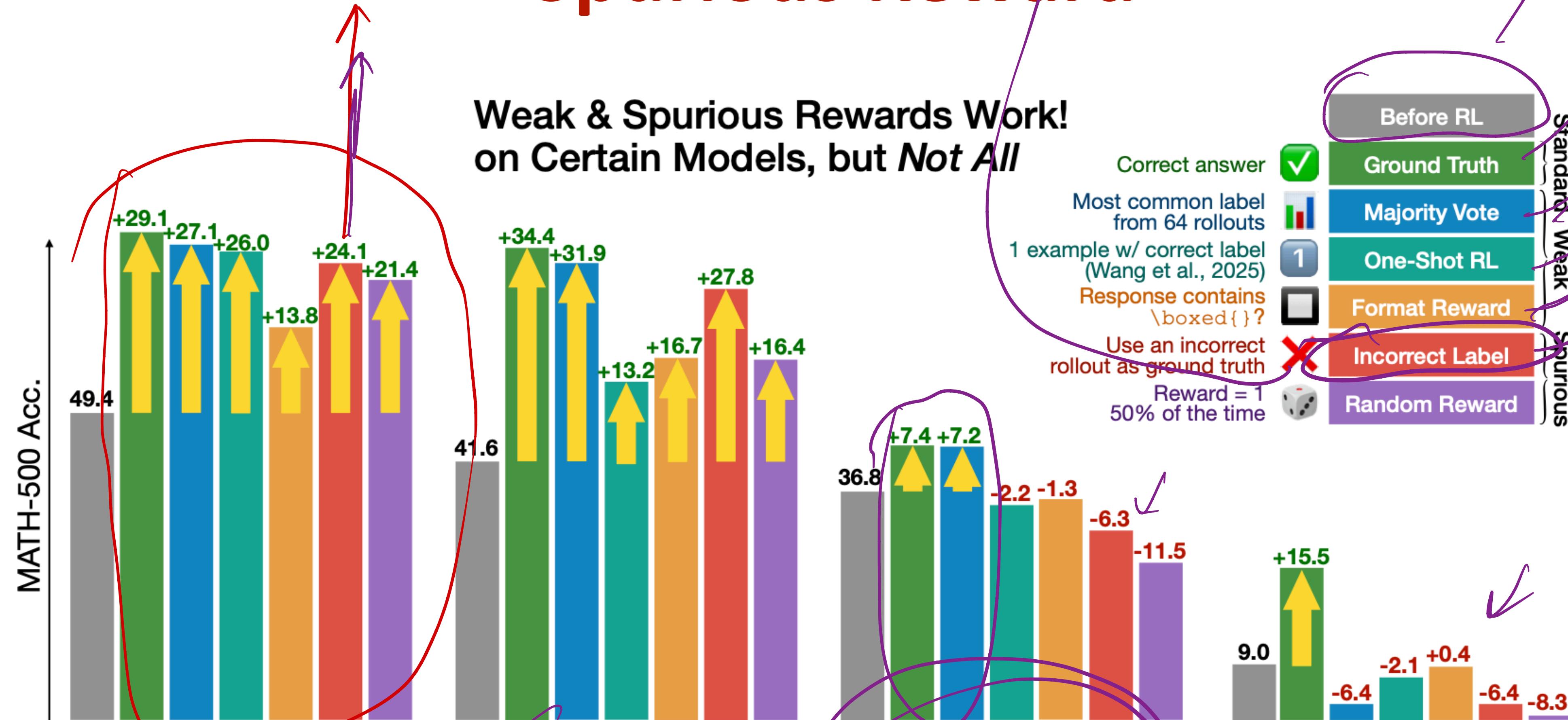


# Owen One-Shot RL!



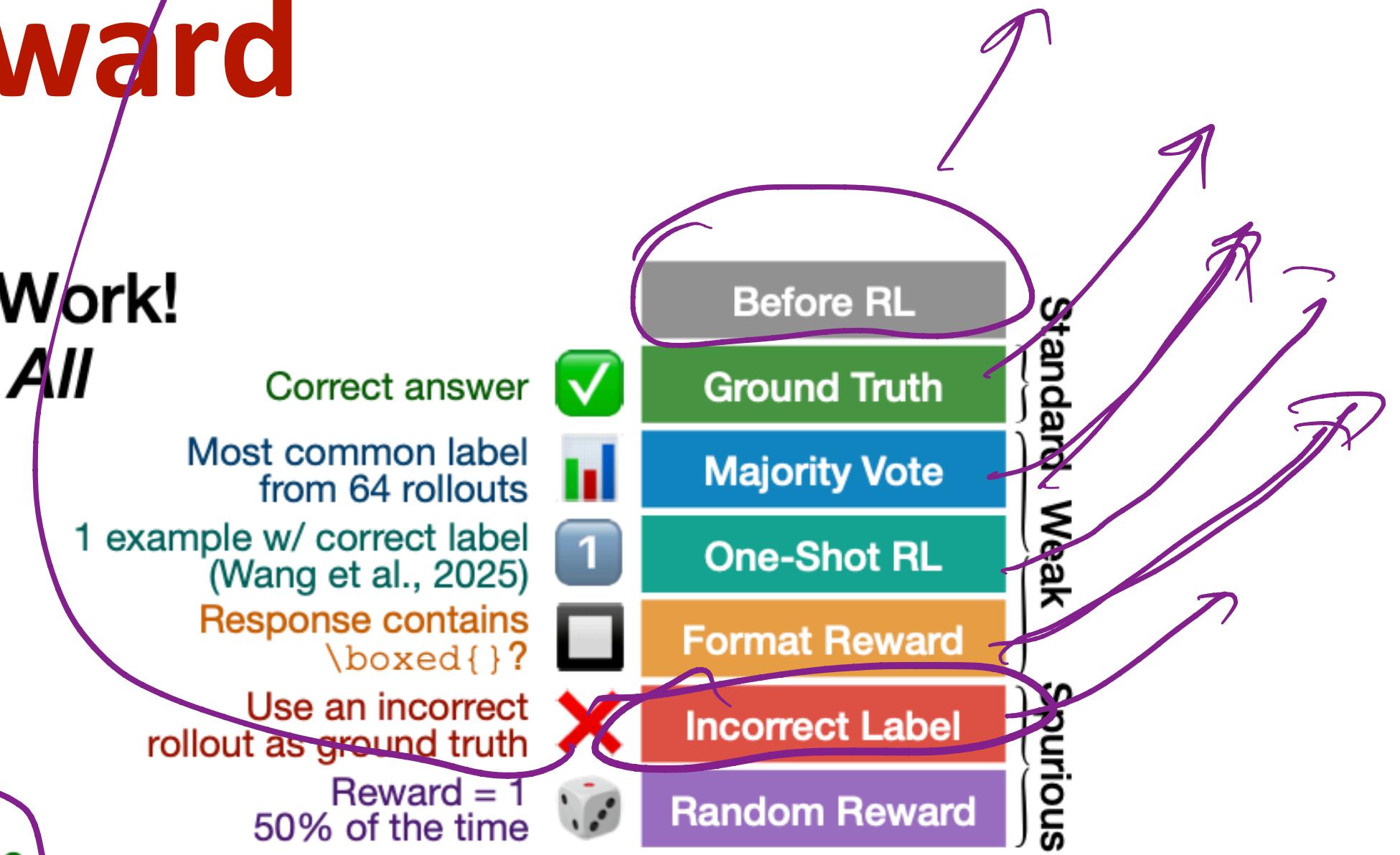
RLVR one training example

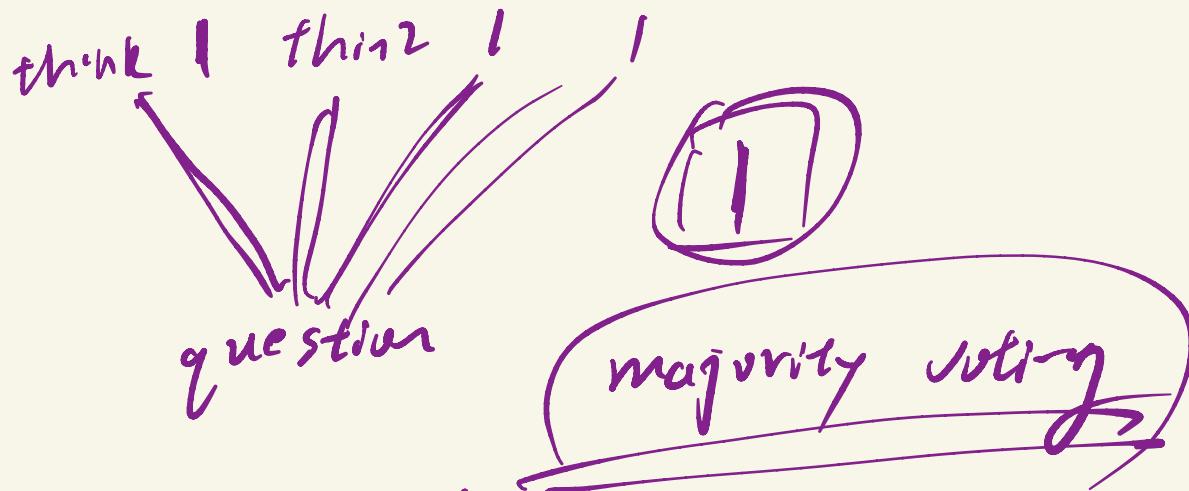
# Spurious Reward



Shao et al. Spurious Rewards: Rethinking Training Signals in RLVR. 2025

↑ working Reward LVR





format Reward:

$$\begin{array}{c}
 \overbrace{\langle \text{think} \rangle \times \times \langle \text{think} \rangle}^{\text{---}} - \text{---} \\
 \times \times \times \langle \text{think} \rangle \quad 0
 \end{array}$$

# Thank You!