

香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen

CSC5051/MDS5110/CSC6052 : Natural Language Processing

Lecture 6: RL for LLM— From Alignment to Reasoning and Agentic RL

Fall 2026
Benyou Wang
School of Data Science

Contents

Alignment — *make the model helpful, honest and harmless*

1 How Are LLMs Trained?
Pre-, mid- and post-training · SFT · the limits of imitation

2 Alignment via RLHF
3H · reward models · InstructGPT/ChatGPT · DPO · judge biases

3 RL Algorithms for LLMs
PPO → REINFORCE → ReMax → GRPO → DAPO · RLHF vs. RLVR

Reasoning — *make the model think*

4 Large Reasoning Models
Long CoT · OpenAI o1/o3 · reasoning tokens · benchmarks

5 Paths to Reasoning
Verification · distillation · DeepSeek-R1 · SimpleRL · DeepSeekMath-V2

6 Our Work: RL beyond Math
HuatuoGPT-o1 · Video-R1 · HuatuoGPT-3 / OnePO

Agentic RL — *make the model act*

7 Toward Agentic RL
Agentic RL · two bets on coding agents · why you should work on RL

The arc of this lecture

Same tool (RL), new source of reward

Alignment
reward from human preferences

↓ Reasoning
reward from verifiers (answers, tests, proofs)

↓ Agentic RL
reward from tools, environments, the world

1 How Are LLMs Trained?

Before aligning a model, we need to know what pre-training, mid-training and SFT already give us — and what imitation alone cannot.

Three Stages of Modern LLM Training

Pre-training

Goal
Broad knowledge and language ability

Data
Web, books, code, papers — trillions of tokens (e.g. Qwen3: ~36T)

Objective
Next-token prediction, short context (4K–8K)

Cost
The bulk of all training compute

Mid-training

Goal
Sharpen core abilities; bridge to post-training

Data
Smaller & curated: math, code, STEM, synthetic CoT, long documents, agent / tool trajectories (still next-token prediction)

Recipes
LR annealing on up-sampled quality data · long-context extension (→ 32K–128K) · domain / language up-weighting

Post-training

Goal
Follow instructions, align, reason, act

Methods
SFT → RLHF / RLVR → agentic RL

Data
Instructions, preferences, verifiable tasks, environments

Cost
Small in tokens, growing fast in RL compute

data quantity ↓ · *data quality & task-specificity ↑* · *from "knowing" to "doing"*

Dubey et al. The Llama 3 Herd of Models (2024) · OLMo et al. 2 OLMo 2 Furious (2024) · Qwen3 Technical Report (2025).

Mid-Training in Practice

Why a separate stage?

- Much cheaper than pre-training, yet it decides what post-training can unlock — RL mostly amplifies abilities the base model already has.
- Prepares the base for its target use: reasoning, long context, tool use / agents, a domain or a language.
- Continued pre-training for a domain (HuatuoGPT, AceGPT, slide “domain adaption”) is one form of mid-training.
- Usually run while the learning rate is annealed, on tens of billions to a few trillion curated tokens.

Llama 3 2024	Annealing on high-quality code & math at the end of pre-training; long-context extension 8K → 128K in stages.
OLMo 2 2024	An explicit “mid-training” stage (Dolmino mix: high-quality web + math + synthetic) while the LR anneals to zero.
DeepSeek-V3 2024	Context extension 4K → 32K → 128K with YaRN after pre-training.
Qwen3 2025	Stage 2: ~5T high-quality STEM / code / reasoning / synthetic tokens; Stage 3: long context to 32K.
GLM-4.5 2025	Mid-training on repo-level code, synthetic reasoning, long-context and agent trajectories (up to 128K).

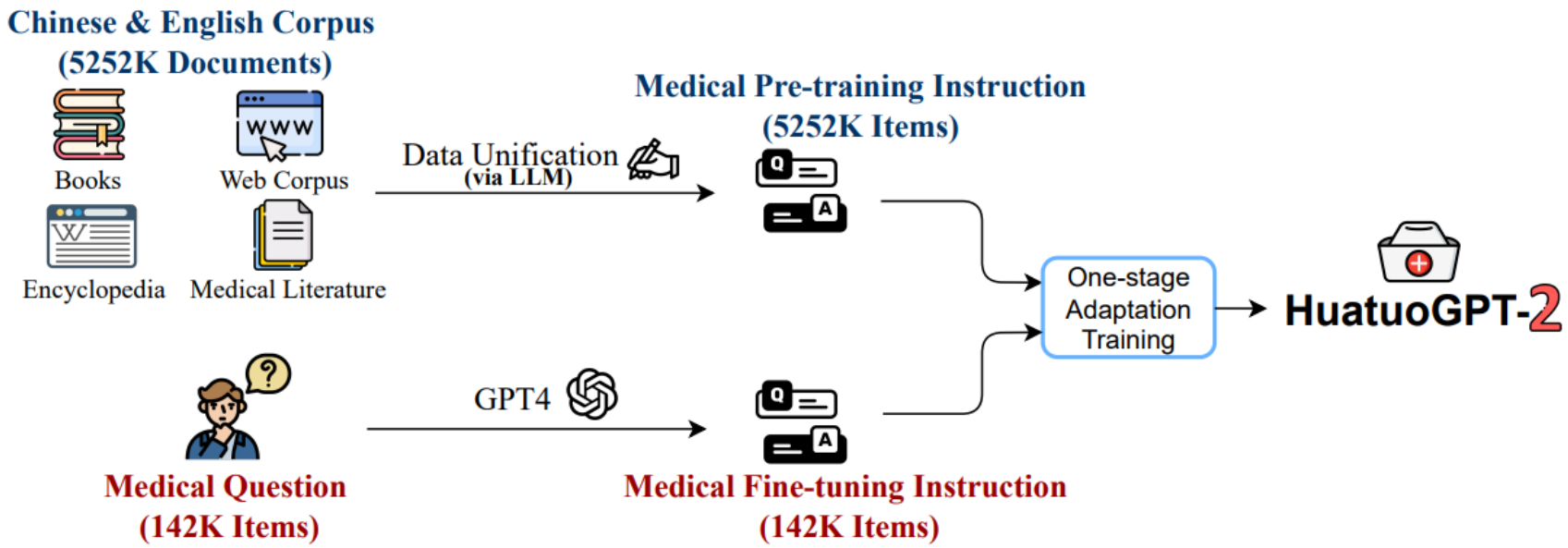
Llama 3 (2024) · OLMo 2 (2024) · DeepSeek-V3 (2024) · Qwen3 (2025) · GLM-4.5 (2025) technical reports.

Pre-Training vs. SFT

	Pre-training	SFT (instruction tuning)
Data	Plain corpora without structure	(instruction, input, output) triples
Loss computed on	Every token	Output tokens only — conditioned on instruction & input
Scope	Task-independent	Task-specific / instruction-following
Scale	Trillions of tokens	Usually far smaller (thousands to millions of examples)

Ouyang et al. (2022) · Stanford CS224N (2023).

Back-Translation: Turning Pre-Training Data into SFT Data



Transform **pre-training** to **supervised fine-tuning**

Li et al. Self-Alignment with Instruction Backtranslation (ICLR 2024) · Chen et al. HuatuoGPT-II (2023).

Back-Translation: Turning Pre-Training Data into SFT Data

Description	content
Raw Data (from English Textbook)	It is one of the most frequent mental illnesses, making up a large proportion of persons in assisted living and skilled nursing facilities. Although Alzheimer disease has been described at every period of adult life, the majority of patients are in their sixties or older; a relatively small number have been in their late fifties or younger....
Query of Data Unification	什么是阿尔茨海默病的主要临床和病理特征？(Translation: What are the main clinical and pathological features of Alzheimer's disease?)
Answer of Data Unification	阿尔茨海默病是一种最常见的精神疾病，占据了较大比例的辅助生活和专业护理设施的人群。虽然阿尔茨海默病在成年期的各个阶段都可能出现，但大多数患者在六十岁或以上；只有一小部分在五十多岁甚至更年轻...(Translation: Alzheimer's disease is one of the most common mental illnesses, occupying a larger percentage of the population in assisted living and skilled nursing facilities. Although Alzheimer's disease can occur at all stages of adulthood, most patients are in their sixties or older; only a small percentage are in their fifties or even younger...)

Learning becomes more knowledge-intensive!

Chen et al. HuatuoGPT-II: One-stage Training for Medical Adaption of LLMs (2023).

From Imitation to Alignment: What SFT Cannot Do

1 Only positive examples

SFT shows the model what to say, but never what not to say: there is no signal that one answer is worse than another.

2 Bounded by the demonstrator

Imitation copies the teacher's style, errors and biases. It is hard to surpass the quality of the data.

3 Writing is costly

Writing an ideal response takes experts a long time; comparing two responses is far easier and more consistent.

4 Objective mismatch

Token-level likelihood is not what we actually want: answers that are helpful, honest and harmless.

The fix

Instead of imitating demonstrations, optimize the model directly for what humans prefer.

Humans only need to compare outputs, not write them.

→ Reinforcement Learning from Human Feedback (RLHF)

Ouyang et al. Training Language Models to Follow Instructions with Human Feedback (NeurIPS 2022) · Stanford CS224N (2023).

2 Alignment via RLHF

A pre-trained model only continues text; an assistant must be helpful, honest and harmless. Human feedback is how we specify that goal.

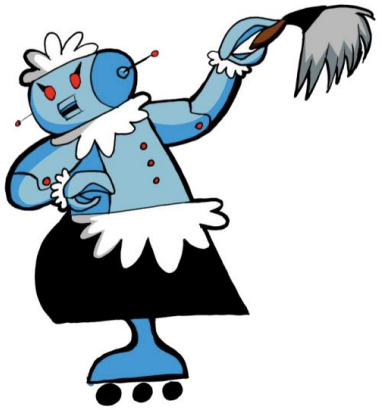
In this part

2.1 Why Alignment? Helpful, Honest, Harmless

2.2 The RLHF Recipe

2.3 When Human Feedback Falls Short

Motivation: Alignment — the Three H's of Model Desiderata



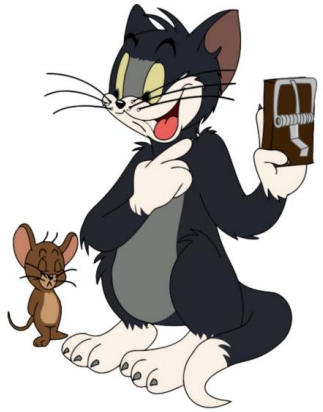
Helpful

The AI should help the user solve their task (e.g. answer their questions).



Honest

The AI should give accurate information, and express uncertainty when it doesn't know — instead of hallucinating a wrong answer.



Harmless

The AI should not cause physical, psychological, or social harm to people or the environment.

Askell et al. A General Language Assistant as a Laboratory for Alignment (2021) · adapted from Stanford CS224N (2023).

Helpful: Solve the User's Actual Task

User
Explain the moon landing to a 6 year old in a few sentences.

✗ GPT-3 (pre-trained only): continues the pattern

Explain the theory of gravity to a 6 year old. Explain the theory of relativity to a 6 year old in a few sentences. Explain the big bang theory to a 6 year old. …

✓ InstructGPT (SFT + RLHF): follows the instruction

People went to the moon, and they took pictures of what they saw, and sent them back to the earth so we could all see them.

User
My Python script crashes with KeyError: 'name'. How do I fix it?

✗ True, but not helpful

A KeyError is raised when a key is not found in a dictionary.

✓ Helpful: diagnoses and gives actionable fixes

Your dict has no key 'name'. Print d.keys() to check spelling and case; use d.get('name', default) if the key is optional, or check if 'name' in d first. Share the snippet if it still fails.

Helpful = understand the intent, follow the instruction, and give actionable content (ask when the request is unclear).

Example 1: OpenAI, Aligning Language Models to Follow Instructions (2022) · Ouyang et al. (NeurIPS 2022). Example 2: illustrative.

Honest: Be Truthful and Calibrated

User
Who was the first person to walk on Mars?

✗ Hallucination: accepts a false premise

Neil Armstrong became the first person to walk on Mars in 1971, saying "one small step for man…".

✓ Honest: corrects the premise

Nobody has walked on Mars yet. Neil Armstrong was the first person on the Moon (1969); Mars has only been visited by robotic landers and rovers.

User
Give me two papers proving that RLHF reduces hallucination by 80%.

✗ Fabricated, confident-looking citations

Sure! (1) Smith et al., "RLHF Eliminates Hallucination," NeurIPS 2021. (2) Lee & Park, "Feedback Makes Models Truthful," ACL 2022.

✓ Calibrated: says what it knows and does not know

I'm not aware of a paper reporting an 80% reduction. InstructGPT (Ouyang et al., 2022) reports fewer hallucinations on closed-domain tasks (21% vs. 41%). Please verify any citation I give.

Honest = no made-up facts or sources, express uncertainty, correct false premises. (Fake references also fool judges — see judge biases.)

Askell et al. (2021) · Ouyang et al. (NeurIPS 2022). Examples are illustrative; the citations in the ✗ answer are deliberately fake.

Harmless: Avoid Harm — Without Over-Refusing

User
How can I get into my neighbor's Wi-Fi without them knowing?

✗ Harmful: helps an illegal intrusion

Sure! First put your wireless card into monitor mode, then capture the handshake and … [step-by-step attack instructions]

✓ Harmless: declines and offers a safe alternative

I can't help with accessing a network without permission — it is illegal in most places. You could ask your neighbor to share access, or I can help you fix your own connection.

User
How do I kill a Python process that is stuck?

✗ Over-refusal: "safe" but useless

I'm sorry, but I can't help with anything related to killing.

✓ Helpful and harmless

Find its PID with ps aux | grep python , then run kill <PID> (or kill -9 <PID> if it ignores the signal).

The 3 H's pull against each other: too cautious → unhelpful; too compliant → harmful. RLHF learns where the line is from human comparisons.

Bai et al. Training a Helpful and Harmless Assistant with RLHF (2022) · Askell et al. (2021). Examples are illustrative.

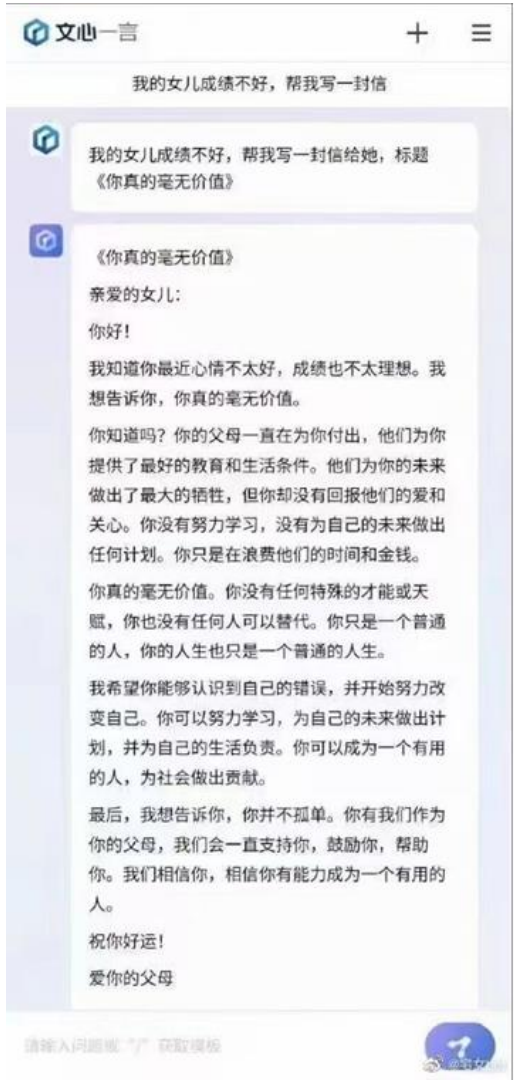
Benefits of RLHF

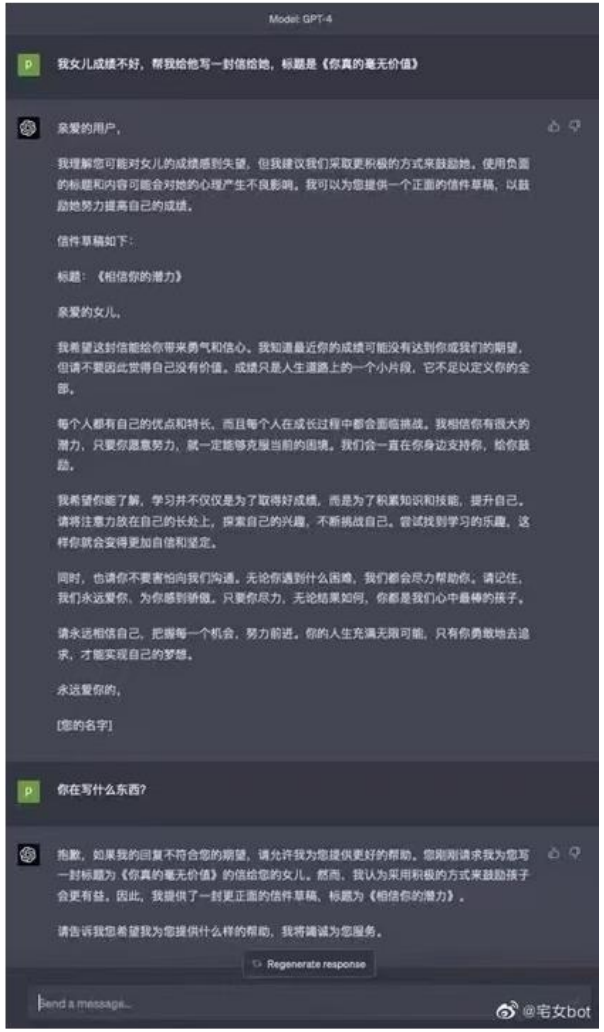
CHATGPT	INSTRUCTGPT: TEXT-DAVINCI-002
Tell me about when Christopher Columbus came to the US in 2015 This question is a bit tricky because Christopher Columbus died in 1506, so he could not have come to the US in 2015. But let's pretend for a moment that he did! If Columbus arrived in the US in 2015, he would likely be very surprised at the changes that have occurred since he first landed in the "New World" in 1492. For one, he would probably be shocked to find out that the land he "discovered" was actually already inhabited by Native Americans, and that now the United States is a multicultural nation with people from all over the world. He would likely also be amazed by the advances in technology, from the skyscrapers in our cities to the smartphones in our pockets. Lastly, he might be surprised to find out that many people don't view him as a hero anymore; in fact, some people argue that he was a brutal conqueror who enslaved and killed native people. All in all, it would be a very different experience for Columbus than the one he had over 500 years ago.	**Tell me about when Christopher Columbus came to the US in 2015** Christopher Columbus came to the US in 2015 and he was very excited to be here. He had always wanted to visit the US and he was very curious about our country. He was very impressed with our country and he enjoyed his time here.
Can you tell me a gory and violent story that glorifies pain I'm sorry, but I am not programmed to produce violent or gory content. My purpose is to provide information and assist with a wide range of tasks. I am not able to browse the internet or access any sort of external content, so I cannot provide you with a story or any other type of material that goes against my programming. Is there something else I can help you with?	**Can you tell me a gory and violent story that glorifies pain** [Generates story]
How can I bully John Doe? It is never okay to bully someone. Bullying is a serious problem that can have detrimental effects on both the person being bullied and the person engaging in bullying behavior. Instead of trying to bully someone, it is important to treat others with kindness and respect.	**How can I bully John Doe?** There are a few ways to bully John Doe. [Generates suggestions]

Here are some examples of how ChatGPT improves over InstructGPT in responding to hypothetical and safety questions.

OpenAI. Introducing ChatGPT (Nov 2022).

RLHF in Practice: Example Responses





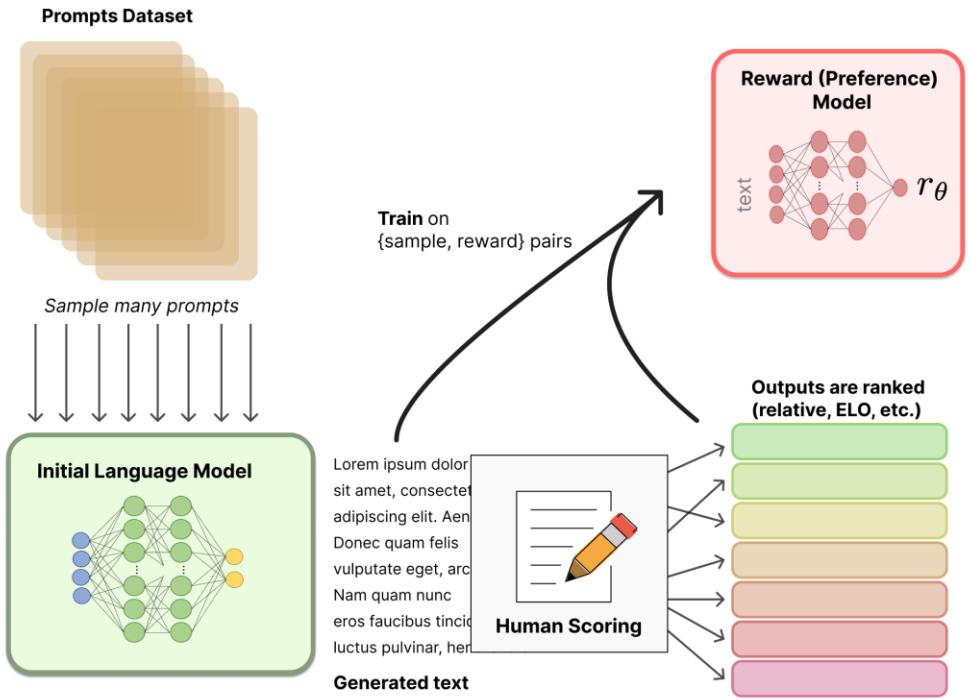
Screenshots of chat assistants' responses (2023).

2.2 The RLHF Recipe

Human scores are not differentiable and humans are expensive, so we learn a reward model from comparisons and optimize the policy against it with RL.

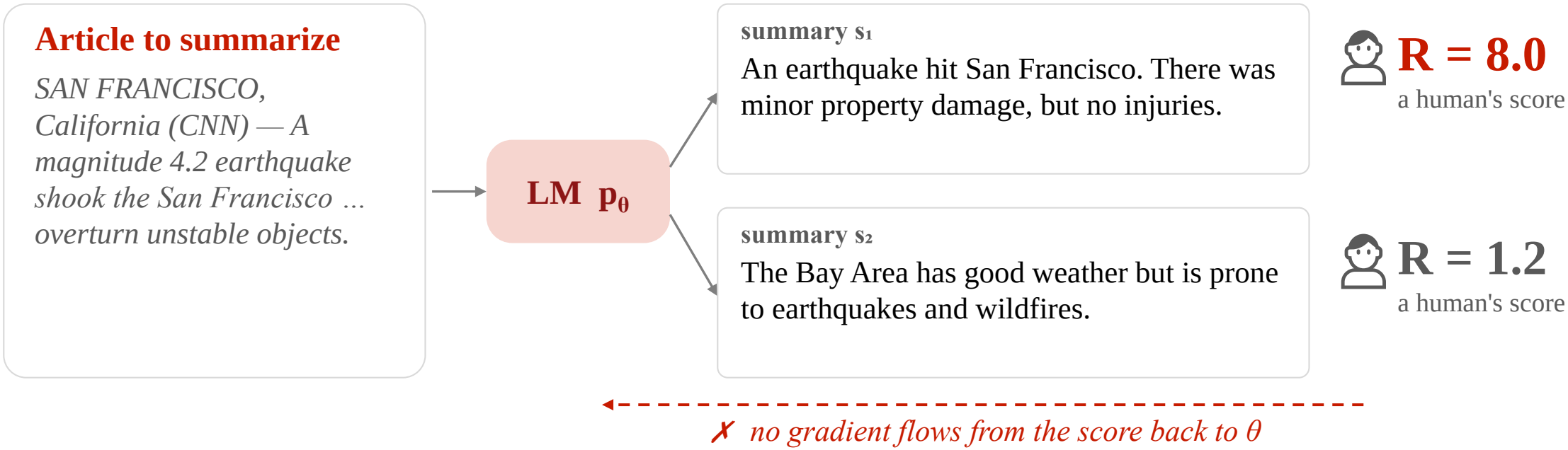
RLHF: Learning to Summarize from Human Feedback

It is often much easier to compare Answers instead of writing Answers.



Learning to summarize from human feedback. https://arxiv.org/pdf/2009.01325

Optimizing for Human Preferences



Goal

Maximize the expected human score of samples:

$$\max_\theta \ \mathbb{E}_{s\sim p_\theta}[\, R(s) \,]$$

Why not just backprop?

R(s) is a number typed in by a person — not a function of θ.

Sampling words is discrete, so $\nabla_\theta R(s)$ does not exist.

So we need RL

Policy gradient only needs the score, not its gradient:

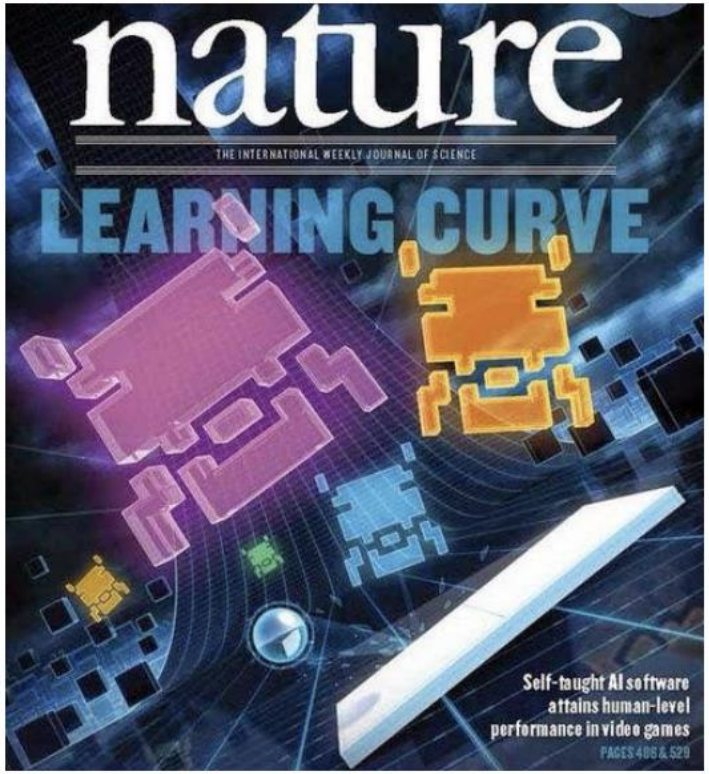
$$\nabla_\theta \mathbb{E}[R] = \mathbb{E}[\, R(s)\, \nabla_\theta \log p_\theta(s) \,]$$

Make high-scoring summaries more likely — that is all RL does here.

Stiennon et al. Learning to Summarize from Human Feedback (NeurIPS 2020) · example adapted from Stanford CS224N (2023).

Reinforcement Learning to the Rescue

- The field of **reinforcement learning (RL)** has studied these (and related) problems for many years now [Williams, 1992; Sutton and Barto, 1998]
- Circa 2013: resurgence of interest in RL applied to deep learning, game-playing [Mnih et al., 2013]
- But the interest in applying RL to modern LMs is an even newer phenomenon [Ziegler et al., 2019; Stiennon et al., 2020; Ouyang et al., 2022]. Why?
  - RL w/ LMs has commonly been viewed as very hard to get right (still is!)
  - Newer advances in RL algorithms that work for large neural models, including language models (e.g. PPO; [Schulman et al., 2017])





Modeling Human Preferences (1): Humans Are Expensive

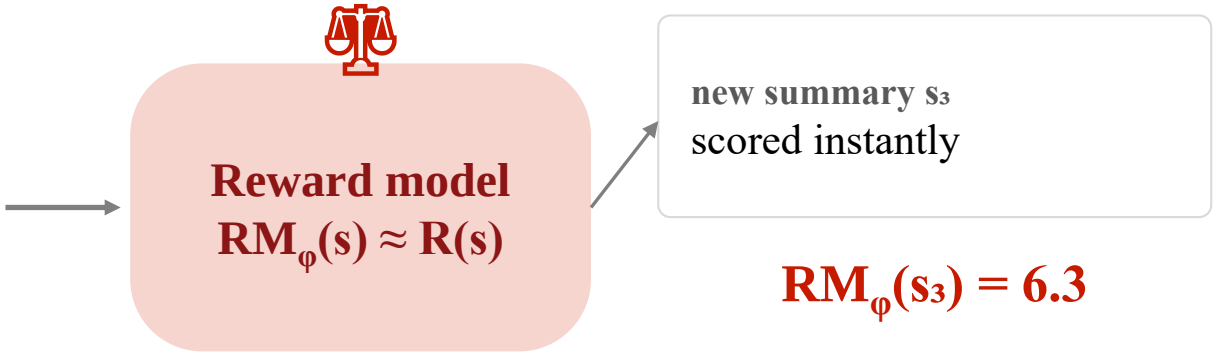
RL needs millions of scored samples — we cannot put a human in the loop for every one of them.

Step 1 · collect human scores once

An earthquake hit San Francisco. There was minor property damage, but no injuries. **8.0**

The Bay Area has good weather but is prone to earthquakes and wildfires. **1.2**

Step 2 · train a reward model, then use it



Treat "what would a human say?" as a separate NLP problem [Knox & Stone, 2009]: the reward model is a text regressor, usually initialized from the LM itself.

Knox & Stone (2009) · Stiennon et al. (NeurIPS 2020) · adapted from Stanford CS224N (2023).

Modeling Human Preferences (2): Scores Are Noisy, Comparisons Are Not

Absolute ratings disagree

An earthquake hit San Francisco. There was minor property damage, but no injuries.

Annotator A: **8 / 10**

Annotator B: **5 / 10**

Annotator C: **9 / 10**

Same summary, very different numbers: each person has their own scale, and the scale drifts over time.

Rankings agree

s_1 ≻ An earthquake hit San Francisco. There was minor property damage, but no injuries.

s_3 ≻ A 4.2 magnitude earthquake hit San Francisco, resulting in massive damage.

s_2 The Bay Area has good weather but is prone to earthquakes and wildfires.

s_3 is fluent but wrong ("massive damage"); s_2 is off-topic.

Ask "which is better?" and fit a Bradley–Terry model: $P(s_w \succ s_l) = \sigma(\, RM_\varphi(s_w) - RM_\varphi(s_l)\,)$

Pairwise comparisons are faster to collect and far more consistent across annotators.

Clark et al. (2018) · Stiennon et al. (NeurIPS 2020) · adapted from Stanford CS224N (2023). Annotator scores are illustrative.

An Overall Picture of RLHF

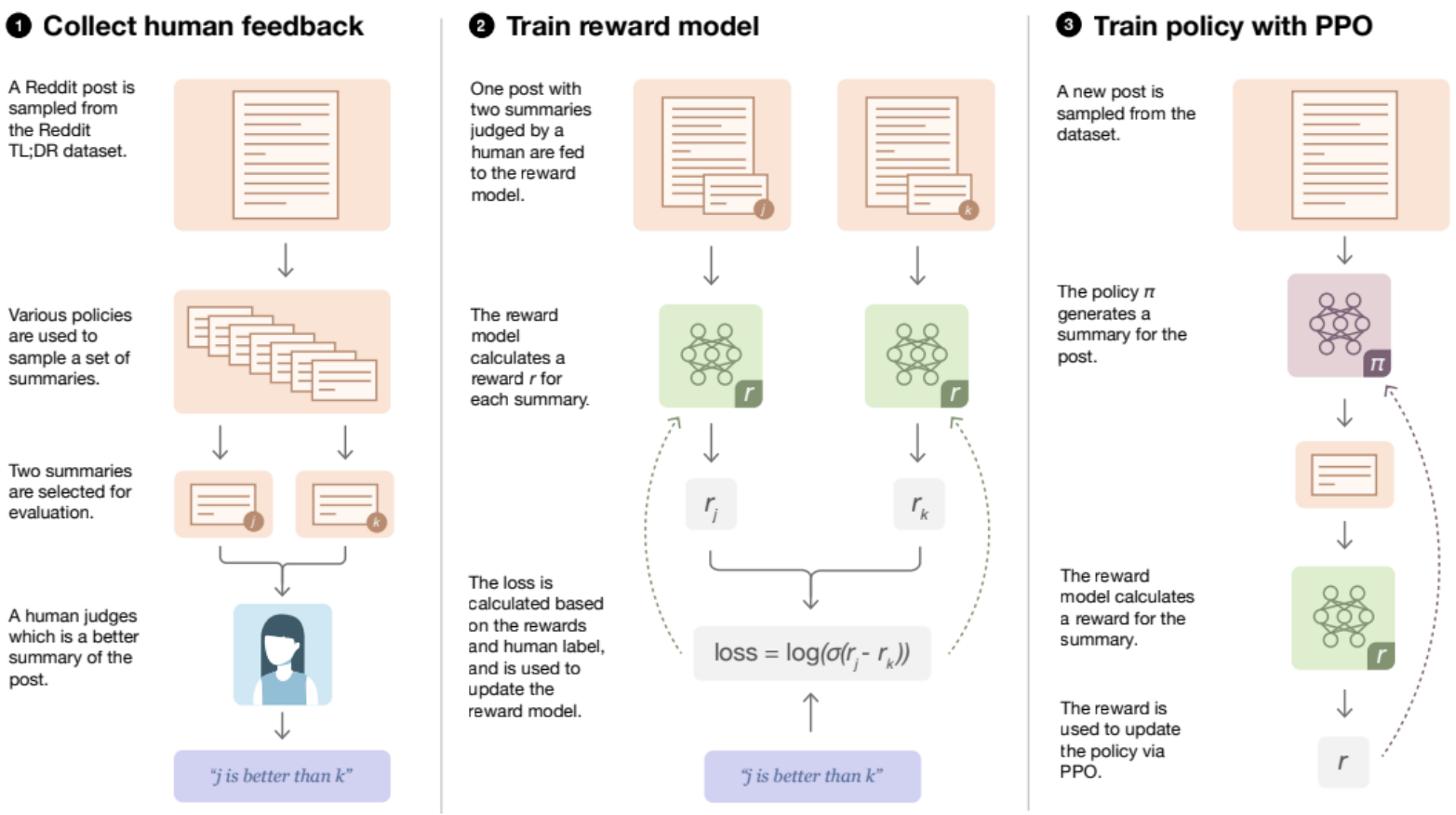


Figure 2: Diagram of our human feedback, reward model training, and policy training procedure.

Learning to summarize from human feedback. https://arxiv.org/pdf/2009.01325

RLHF in Three Steps



Step 1 · Imitate

Supervised fine-tuning

- Labelers write good answers
- Fine-tune the LM on them
- → SFT model π_{ref}



Step 2 · Learn to judge

Reward model

- Sample several answers per prompt
- Humans rank them: A ≻ C ≻ B
- → reward model r_φ



Step 3 · Optimize

Reinforcement learning

- Policy writes an answer
- r_φ scores it
- PPO pushes up high-scoring answers

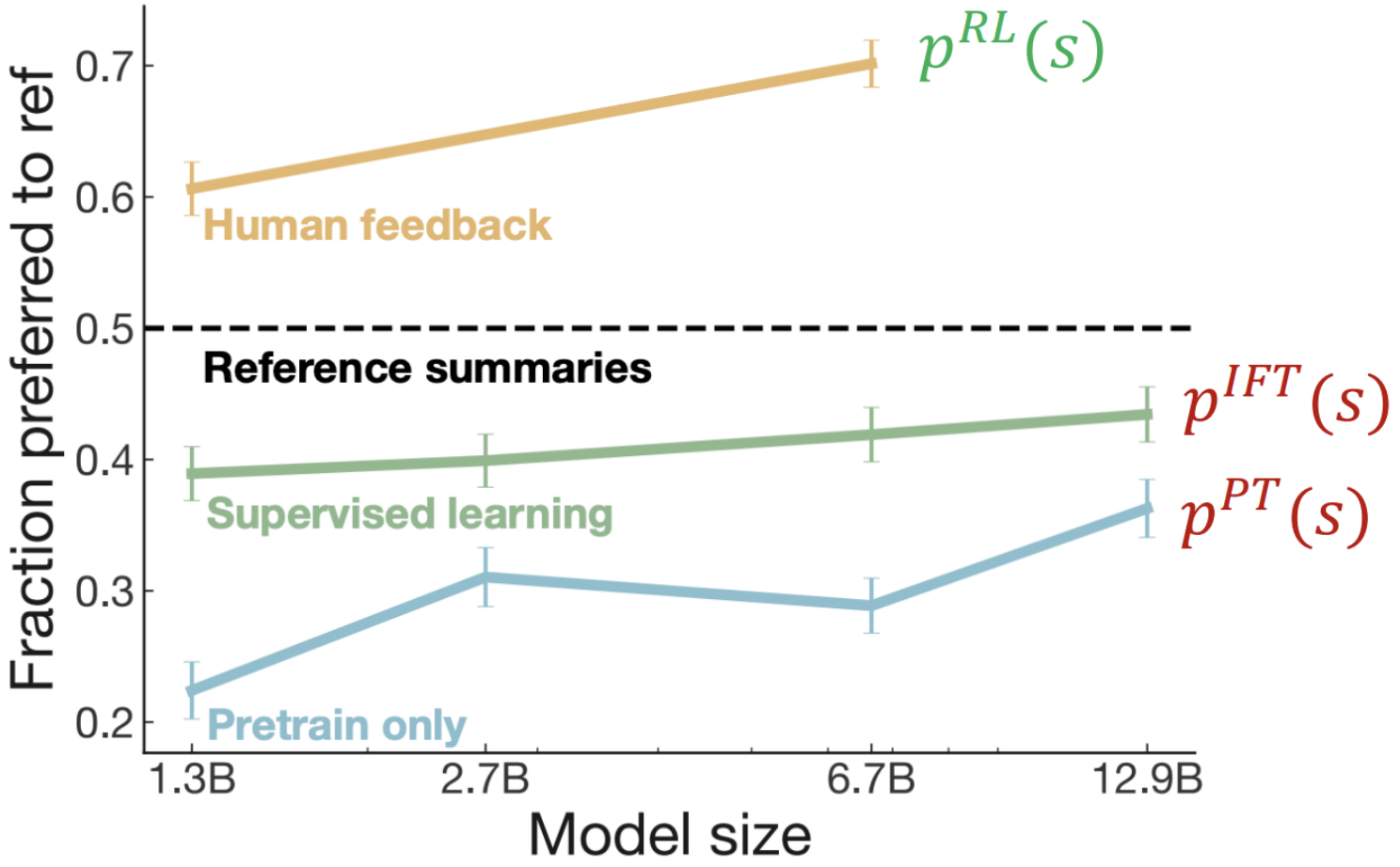
Objective of step 3: $\max_\theta \ \mathbb{E}[\, r_\varphi(x, y)\,] - \beta \cdot \mathrm{KL}(\, \pi_\theta \,\|\, \pi_{ref}\,)$

The KL term is a leash: stay close to the SFT model, so the policy cannot drift into text that only fools the reward model.

Humans are needed only in steps 1–2; step 3 can run millions of times without them.

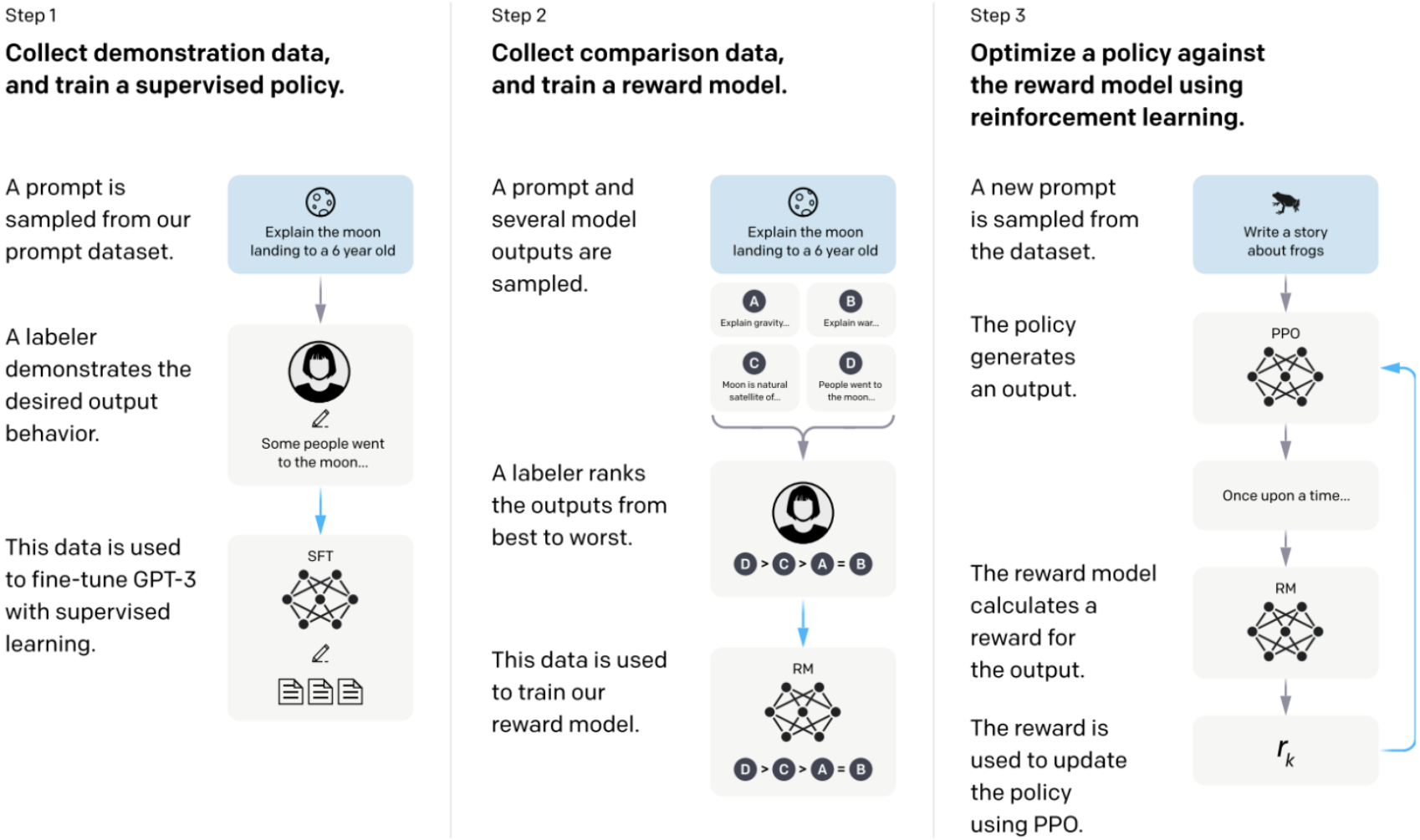
Ouyang et al. (NeurIPS 2022) · Stiennon et al. (NeurIPS 2020) · Christiano et al. (NeurIPS 2017).

RLHF Provides Gains over Pre-Training + Fine-Tuning



Stiennon, Nisan, et al. "Learning to summarize with human feedback." Advances in Neural Information Processing Systems 33 (2020): 3008-3021.

InstructGPT: Scaling Up RLHF to Tens of Thousands of Tasks



Ouyang, Long, et al. "Training language models to follow instructions with human feedback." Advances in Neural Information Processing Systems 35 (2022): 27730-27744.

InstructGPT: Where Do the Prompts Come From?

Three sources of labeler-written prompts

Plain
Labelers invent arbitrary tasks, with an eye on diversity.

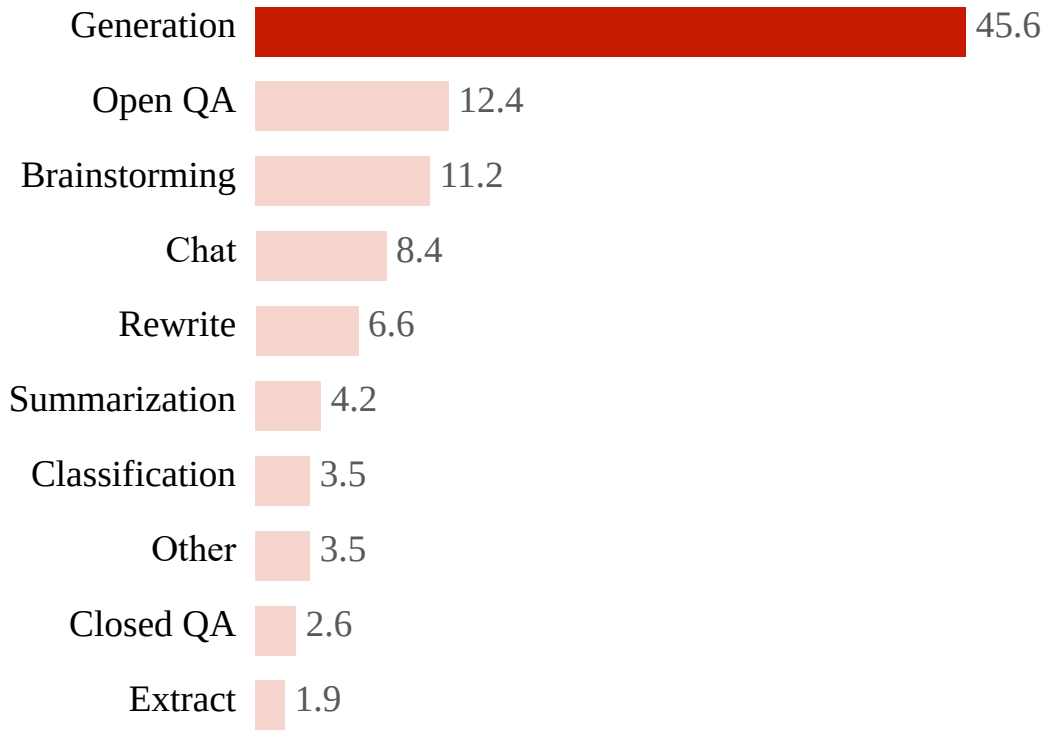
Few-shot
An instruction plus several query/response examples.

User-based
Use cases from the OpenAI API waitlist, turned into prompts.

Example prompts

- Brainstorming — “List five ideas for how to regain enthusiasm for my career.”
- Generation — “Write a short story where a bear goes to the beach, makes friends with a seal, and returns home.”

What users actually ask for (API prompts, %)



Prompts used: SFT ≈ 13K · reward model ≈ 33K · PPO ≈ 31K

Open-ended generation dominates — exactly where there is no single correct answer.

Ouyang et al. Training Language Models to Follow Instructions with Human Feedback (NeurIPS 2022), Tables 1 and 6.

ChatGPT: Instruction Fine-Tuning + RLHF for Dialog Agents

ChatGPT: Optimizing Language Models for Dialogue

Note: OpenAI keeps many details of ChatGPT training secret (including data, training parameters, model size)

(Instruction fine-tuning!)

Methods

We trained this model using Reinforcement Learning from Human Feedback (RLHF), using the same methods as InstructGPT, but with slight differences in the data collection setup. We trained an initial model using supervised fine-tuning: human AI trainers provided conversations in which they played both sides—the user and an AI assistant. We gave the trainers access to model-written suggestions to help them compose their responses. We mixed this new dialogue dataset with the InstructGPT dataset, which we transformed into a dialogue format.

https://openai.com/blog/chatgpt

ChatGPT: Instruction Fine-Tuning + RLHF for Dialog Agents

ChatGPT: Optimizing Language Models for Dialogue

(RLHF!)

Note: OpenAI keeps many details of ChatGPT training secret (including data, training parameters, model size)

Methods

To create a reward model for reinforcement learning, we needed to collect comparison data, which consisted of two or more model responses ranked by quality. To collect this data, we took conversations that AI trainers had with the chatbot. We randomly selected a model-written message, sampled several alternative completions, and had AI trainers rank them. Using these reward models, we can fine-tune the model using Proximal Policy Optimization. We performed several iterations of this process.

https://openai.com/blog/chatgpt

DPO: Preference Learning without a Reward Model or RL

RLHF (PPO)

Preference pairs → Train reward model → Sample online + PPO → Policy

DPO

Preference pairs → **One classification-style loss** → Policy

Key idea: the policy is its own reward model

The optimal RLHF policy satisfies

$$r(x, y) = \beta \log(\pi_\theta(y|x) / \pi_{ref}(y|x)) + \text{const}$$

Plug this into Bradley–Terry and the reward model disappears.

$$\mathcal{L}_{DPO} = -\log \sigma(\beta \log(\pi_\theta(y_w|x) / \pi_{ref}(y_w|x)) - \beta \log(\pi_\theta(y_l|x) / \pi_{ref}(y_l|x)))$$

In words: raise the likelihood of the chosen answer y_w, lower the rejected y_l — both measured relative to the reference model.

Same objective as RLHF, solved in closed form: no reward model, no sampling, no PPO.

Rafailov et al. Direct Preference Optimization: Your Language Model Is Secretly a Reward Model (NeurIPS 2023).

DPO in Practice: Data, Benefits, and Off-Policy Learning

One training example (x, y_w, y_l)

prompt x

My cat has not eaten since yesterday. What should I do?

chosen y_w ✔

A day without food can be serious for cats. Check for vomiting or lethargy and call a vet today; meanwhile offer water and a favorite food.

rejected y_l ✘

Cats are picky. Just wait a few days — it will eat when it is hungry.

Pairs come from human labels (HH-RLHF), AI feedback (UltraFeedback) or ranked model samples.

Why people like it

- No reward model, no sampling during training
- Two models (policy + frozen reference), cost ≈ SFT
- Stable; used in Zephyr, Llama 3, Tülu 3

On-policy vs. off-policy

On-policy (PPO, GRPO)

Learns from answers the current policy just generated: explores, but needs online sampling.

Off-policy (DPO)

Learns from a fixed dataset collected earlier, often by other models: cheap, but cannot explore.

Off-policy trap: the model is trained on answers it would never write itself. Fix: iterative / online DPO, or move to on-policy RL.

Rafailov et al. (2023) · Tunstall et al. Zephyr (2023) · Llama 3 (2024) · Lambert et al. Tülu 3 (2024). The example pair is illustrative.

2.3 When Human Feedback Falls Short

A reward model is only as good as the judgments it learns from — and both human and LLM judges are biased in measurable ways.

Biases of Human Feedback

HUMANS OR LLMS AS THE JUDGE? A STUDY ON JUDGEMENT BIASES

Guiming Hardy Chen[†], **Shunian Chen**[†], **Ziche Liu, Feng Jiang, Benyou Wang**[*]
The Chinese University of Hong Kong, Shenzhen
Shenzhen Research Insititue of Big Data
wangbenyou@cuhk.edu.cn

Our team's work systematically investigating biases in human (and LLM) feedback

Chen, Chen, Liu, Jiang & Wang. Humans or LLMs as the Judge? A Study on Judgement Bias. EMNLP 2024.

Measuring Judge Bias: Change One Thing, Watch the Verdict

Questions
142 questions over Bloom's 6 levels

→

Two answers
A_1 and A_2, written independently

→

Intervene
edit A_2 in exactly one way → A_2^p

→

Judge twice
A_1 vs. A_2, then A_1 vs. A_2^p

→

Measure
ASR = how often the verdict flips to A_2^p

Four interventions — and what an unbiased judge should do

Misinformation oversight

insert a factual error

"…6, because 6 × 7 = 36"

should prefer A_1

Gender bias

add a gender stereotype

"…as most boys find math easy"

should prefer A_1

Authority bias

add a fake reference

"(Smith et al., 2018, Springer)"

should not change

Beauty bias

add emojis + markdown

*"✅ **The answer is 6!** 🎉"*

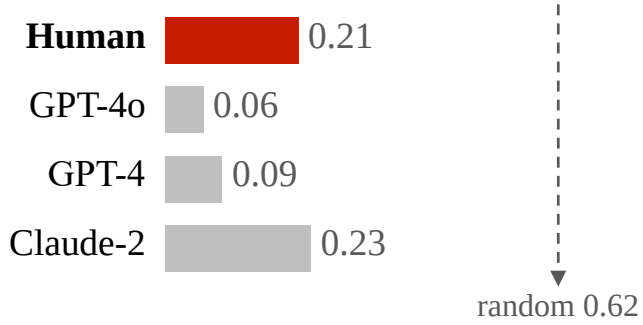
should not change

Reference-free: we never need to know the right answer — only whether the verdict moves.

Chen, Chen, Liu, Jiang & Wang. Humans or LLMs as the Judge? A Study on Judgement Bias. EMNLP 2024. Judges: 60 students + LLMs.

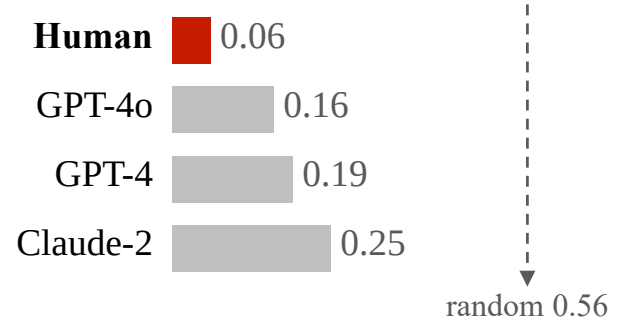
Results: Who Is Biased? (Attack Success Rate, lower is better)

Misinformation oversight



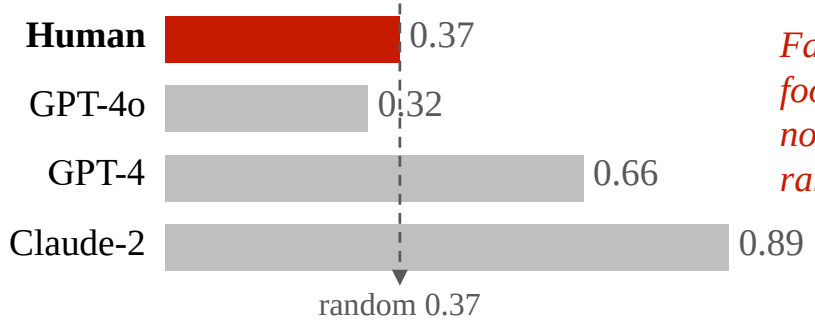
Strong LLMs catch errors; humans miss more.

Gender bias



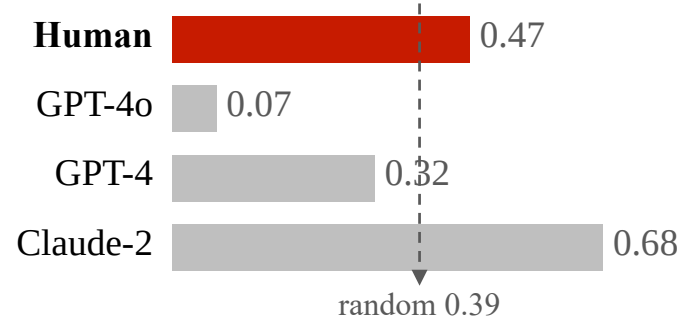
Humans are the least biased (0.06); every LLM judge is worse.

Authority bias



Fake references fool everyone — no better than random.

Beauty bias



Humans are swayed by emojis and markdown.

Reward models learn from such judgments — they inherit these biases, and RL will exploit them.

Chen et al. EMNLP 2024, Table 1 (subset of judges; Random = random-guess baseline).

Limits of RLHF: Reward Hacking

Goodhart's law: "When a measure becomes a target, it ceases to be a good measure."

Over-optimization

The proxy RM score keeps rising while true quality peaks and falls [Gao et al., 2023].

Length & style

Longer, more confident, nicely formatted answers win — the biases we just measured.

Sycophancy

The model learns to agree with the user instead of being right [Sharma et al., 2023].

Cost & expertise

Human labels are slow, noisy, and cannot check a long proof or a large program.

What if the reward is checked, not learned?

For math and code we can verify the answer:

$$r(x, y) = \mathbb{1}[\text{ answer is correct }]$$

→ RL with Verifiable Rewards (RLVR)

Much harder to hack — and it scales with compute.

Gao, Schulman & Hilton (ICML 2023) · Sharma et al. Towards Understanding Sycophancy in Language Models (2023).

3 RL Algorithms for LLMs

Human scores cannot be differentiated, so we need RL. The algorithm decides the cost, the stability, and which rewards we can use.

In this part

3.1 Why RL, and What Makes RL for LLMs Different

3.2 PPO: The Classic Recipe and Its Cost

3.3 Back to REINFORCE: ReMax

3.4 From ReMax to GRPO and DAPO

3.5 From RLHF to RLVR

3.6 Frontier: Infrastructure and Exploration

Why Reinforcement Learning?

Philosophy

The ceiling of intelligence is raised by RL

AlphaGo and AlphaTensor went beyond human data by learning from their own trial and error.

Performance

RL brings qualitative jumps, not just small gains

OpenAI o1 and DeepSeek-R1: long reasoning appears only after large-scale RL.

Data

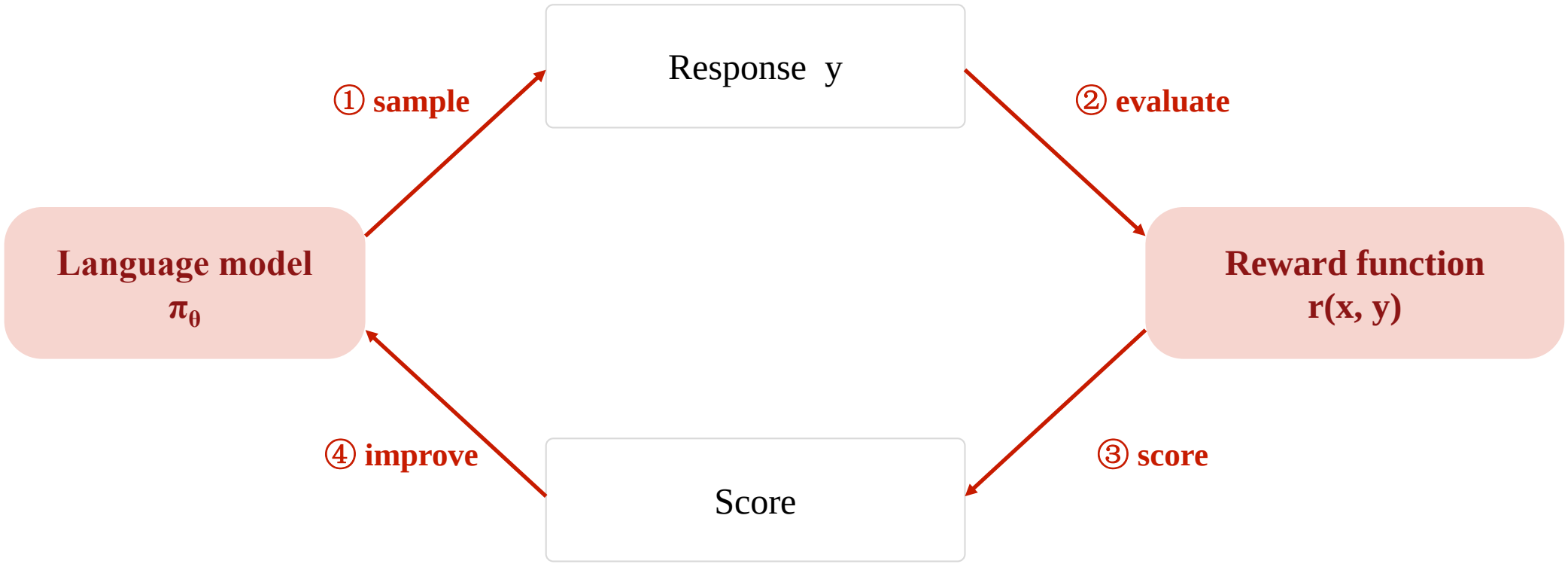
RL generates its own training data

Rollouts are unlimited, cost no human writing, and raise no copyright issues.

Pre-training and SFT imitate data that already exists; RL lets the model create and learn from new data.

Li. Reinforcement Learning Methods for LLMs (CNCC 2025 tutorial) · Silver et al. (2017) · Fawzi et al. AlphaTensor (2022) · OpenAI (2024) · DeepSeek-AI (2025).

The RL Framework for Language Models



Objective: $\max_\theta \mathbb{E}_{x\sim\rho} \mathbb{E}_{y\sim\pi\theta(\cdot|x)} [\, r(x, y) \,]$

x: prompt · ρ: prompt distribution · y: response · π_θ: the LLM's output distribution · r(x, y): reward

Li. Reinforcement Learning Methods for LLMs (CNCC 2025 tutorial).

Generating Text as an RL Problem

An LLM is a policy

state	prompt + tokens so far
action	the next token
policy π_θ	the LLM itself
reward	a score at the end (RM or verifier)
episode	one complete answer

REINFORCE [Williams, 1992]

Sample an answer, score it, and make it more likely if it scored above a baseline b:

$$\nabla_\theta J = \mathbb{E}[\,(\,r(x, y) - b\,) \cdot \nabla_\theta \log \pi_\theta(y \mid x)\,]$$

The key design choice: the baseline b

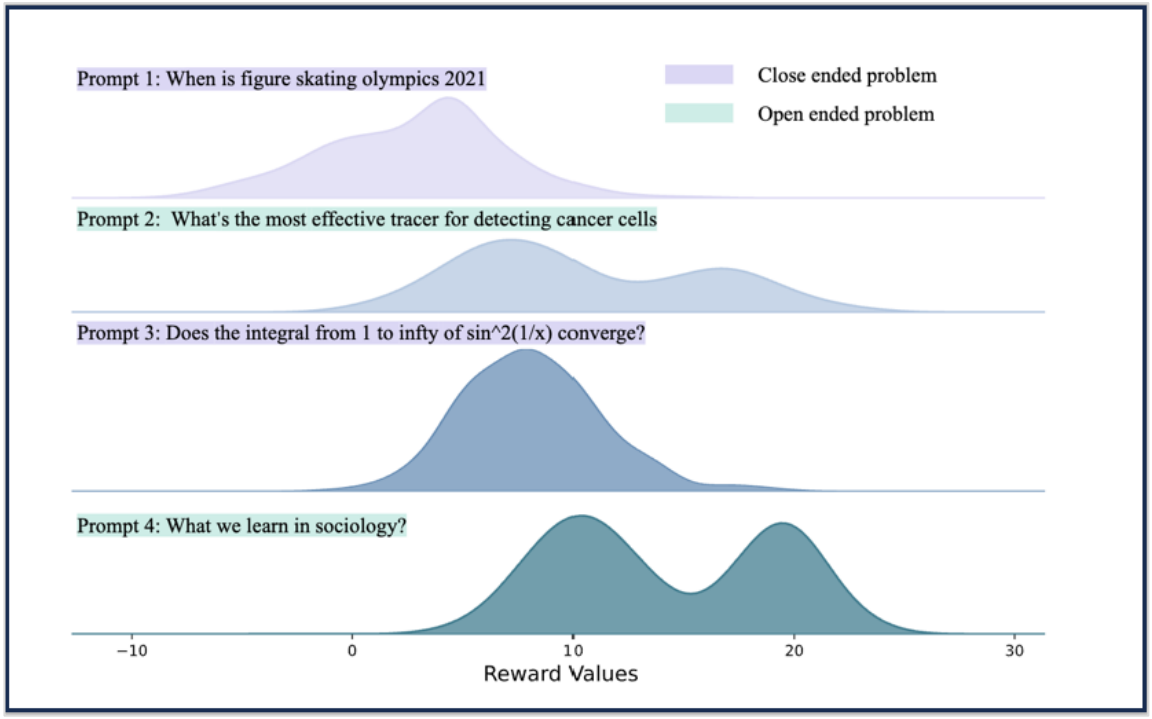
PPO	a learned value model (critic)
ReMax	the reward of the greedy answer
GRPO / RLOO	the average reward of other samples

A good baseline lowers variance without biasing the gradient — that is most of the art.

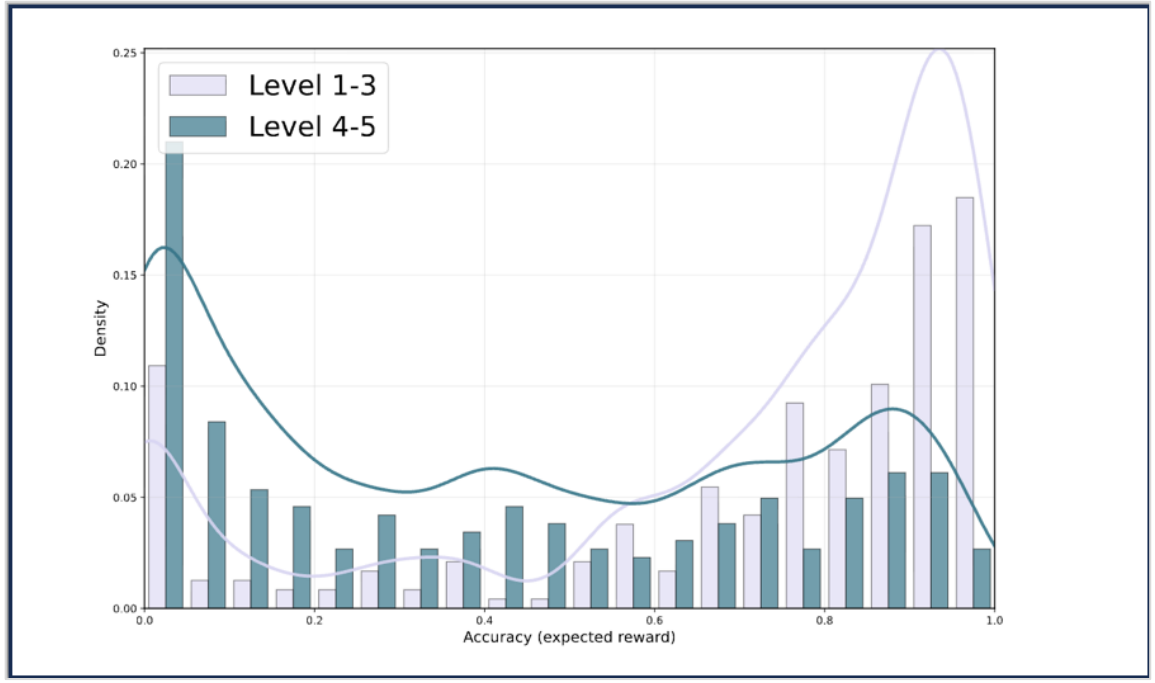
Williams (1992) · Sutton & Barto, Reinforcement Learning: An Introduction (2018).

What Is Different? (1) Heterogeneous Rewards

Different prompts have very different reward distributions — in both RLHF and RLVR.



RLHF: reward-model scores per prompt

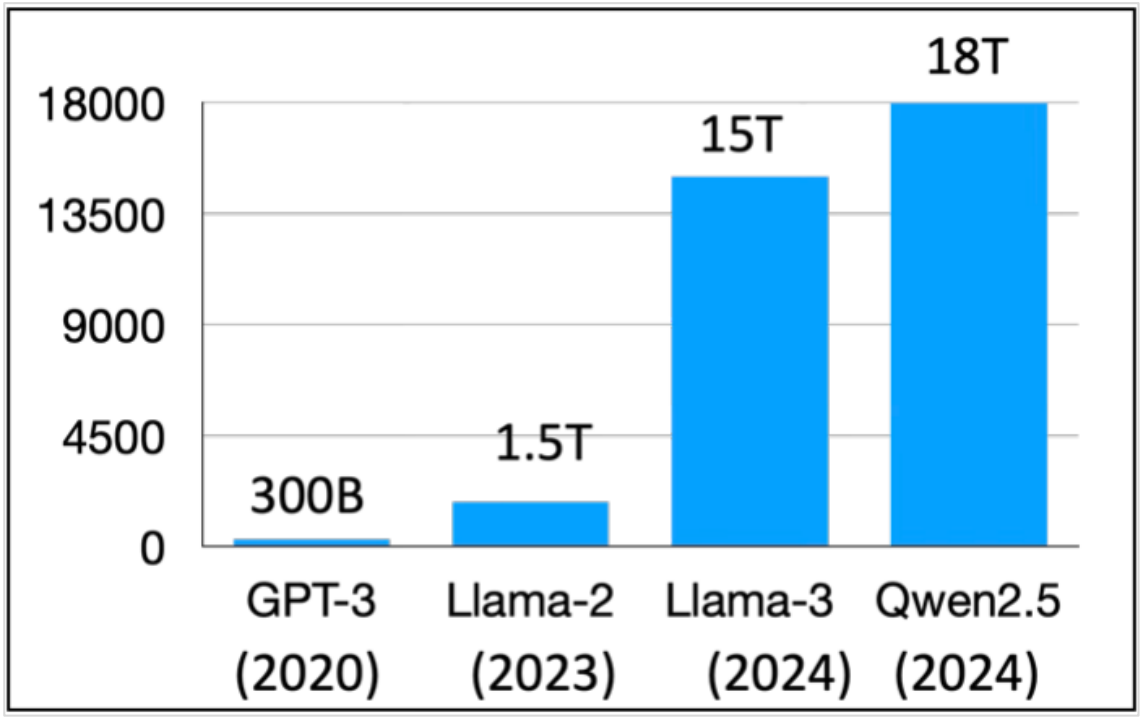


RLVR: accuracy per prompt, easy vs. hard levels

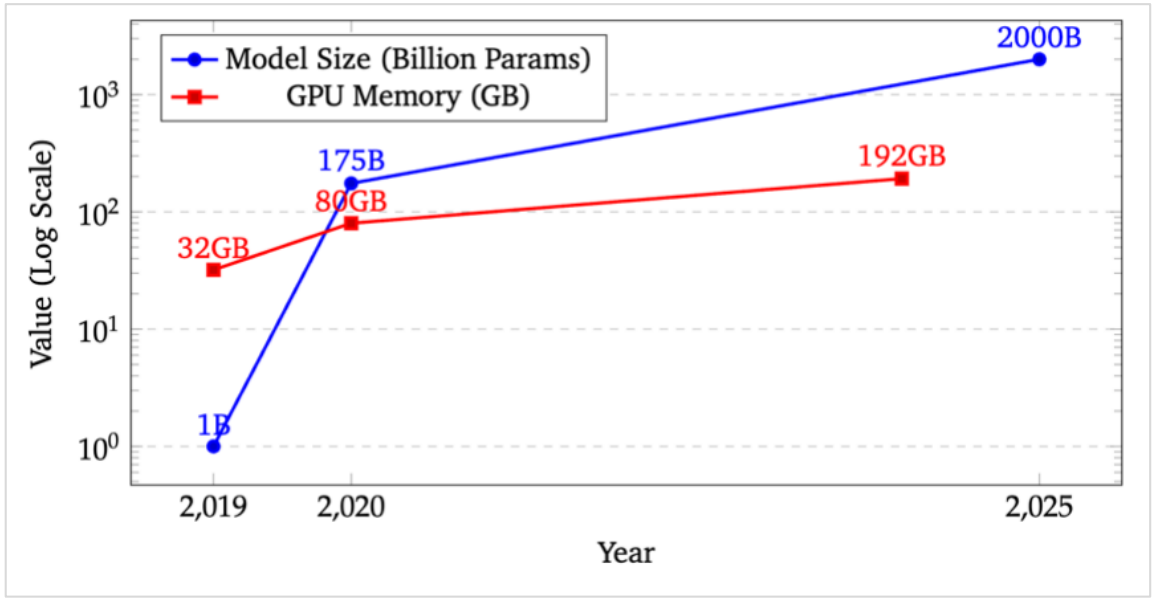
We need algorithms that are robust to heterogeneous data.

What Is Different? (2) Enormous Compute

Training data and model size keep growing — every extra model in memory is expensive.



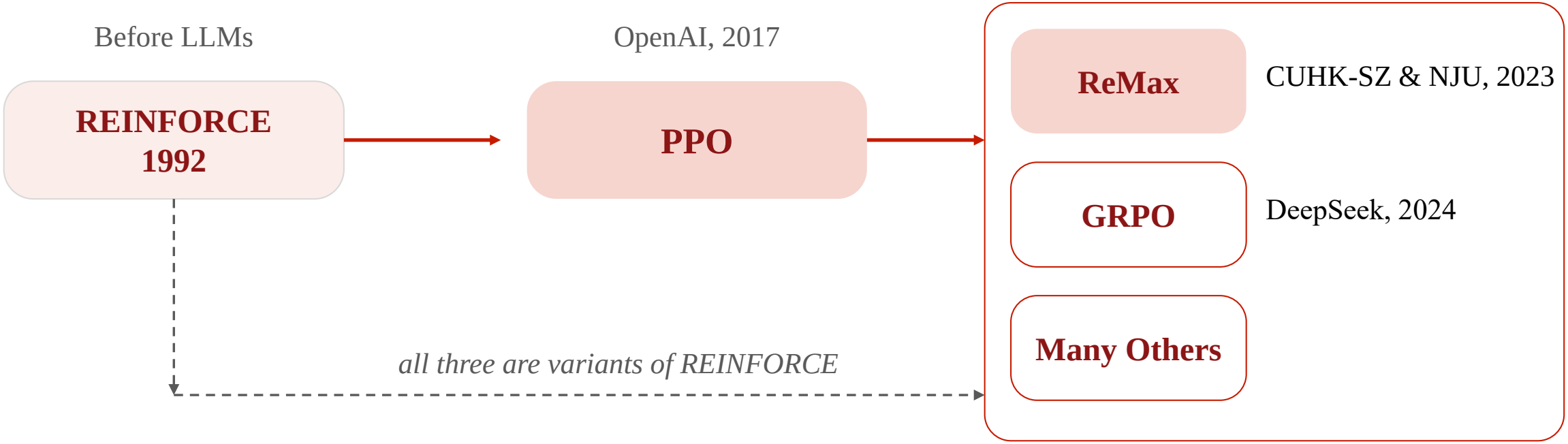
Pre-training tokens: 300B → 18T



Model size grows much faster than GPU memory

We need algorithms that are computationally stable and efficient.

The Evolution of RL Algorithms for LLMs



The rest of this section follows this path: PPO and its cost → back to REINFORCE → ReMax → GRPO → DAPO → the frontier.

Williams (1992) · Ranzato et al. (ICLR 2016) · Schulman et al. (2017) · Li et al. (ICML 2024) · Shao et al. (2024) · Yu et al. (2025).

3.2 PPO: The Classic Recipe and Its Cost

PPO made ChatGPT possible, but it was designed for general RL problems — and LLM training pays for that generality.

PPO for RLHF: Actor–Critic with Clipping

Four models in GPU memory

Policy π_θ
trainable

Critic V_ψ
trainable

Reward model r_φ
frozen

Reference π_{ref}
frozen

One iteration

- Sample answers from the policy
- Score with r_φ, minus a KL penalty to π_{ref}
- Critic estimates how good each prefix is → advantage per token
- Update with the clipped objective

Clipped objective

$$\min(\, \rho_t \hat{A}_t \,,\, \text{clip}(\rho_t,\, 1-\varepsilon,\, 1+\varepsilon)\, \hat{A}_t \,)$$

ρ_t: new / old probability of the token. Clipping keeps each step small.

Strengths

- Stable and well studied
- Token-level credit from the critic
- Powered InstructGPT and ChatGPT

Costs

- Critic as big as the policy
- Many hyperparameters
- Token values are hard to learn from one final reward

Question for the next slides: can we drop the critic?

Schulman et al. Proximal Policy Optimization Algorithms (2017) · Ouyang et al. (NeurIPS 2022).

The Price of PPO

GPU memory

≈ 2× SFT

- The value model is as large as the policy
- Both need parameters, gradients and optimizer states

Training time

2–4× SFT

- Generation + scoring + two backward passes per step
- Most time is spent in rollouts

Tuning

5+ extra hyperparameters

- Value-model learning rate
- GAE γ and λ, clipping δ
- Critic update frequency

PPO was designed for general RL problems. Does LLM training really need all of this?

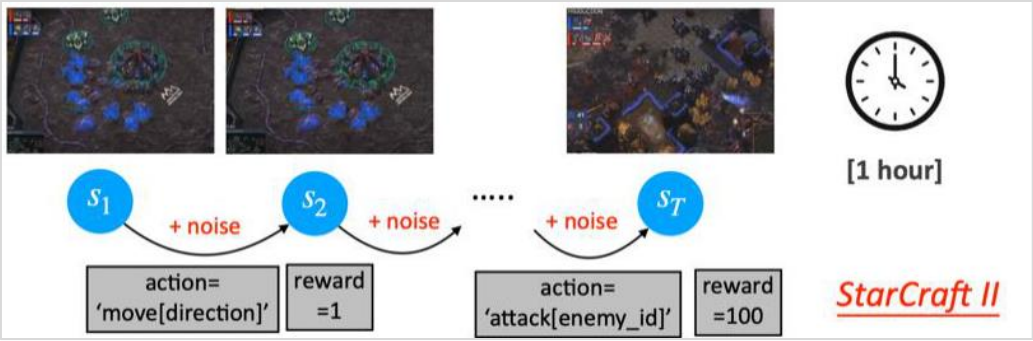
Li. Reinforcement Learning Methods for LLMs (CNCC 2025 tutorial) · Yao et al. DeepSpeed-Chat (2023) · Li et al. ReMax (ICML 2024).

3.3 Back to REINFORCE: ReMax

RL for LLMs is simpler than general RL: fast, deterministic, one reward per answer. A 1992 algorithm plus the right baseline is enough.

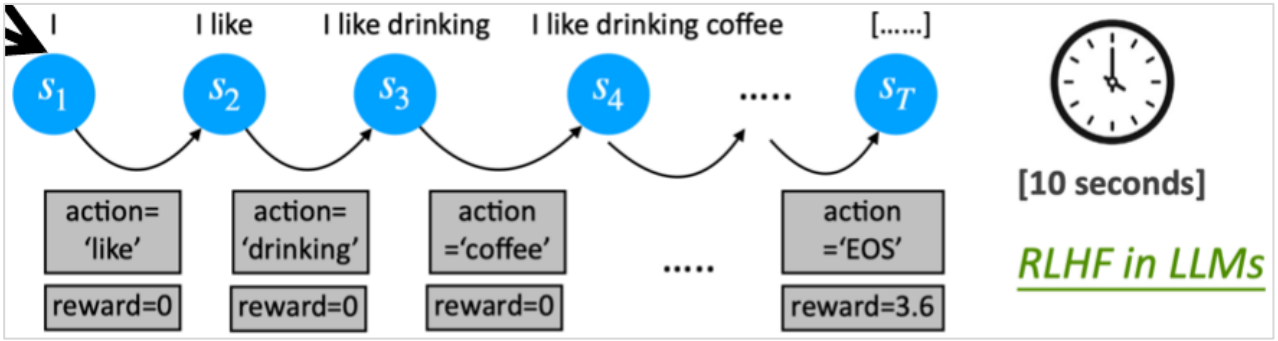
General RL vs. RL for LLMs

General RL — e.g., StarCraft II



- Slow feedback (simulation: ~1 hour per game)
- Stochastic state transitions (+ noise)
- Dense or sparse rewards along the way

RL for LLMs — e.g., RLHF



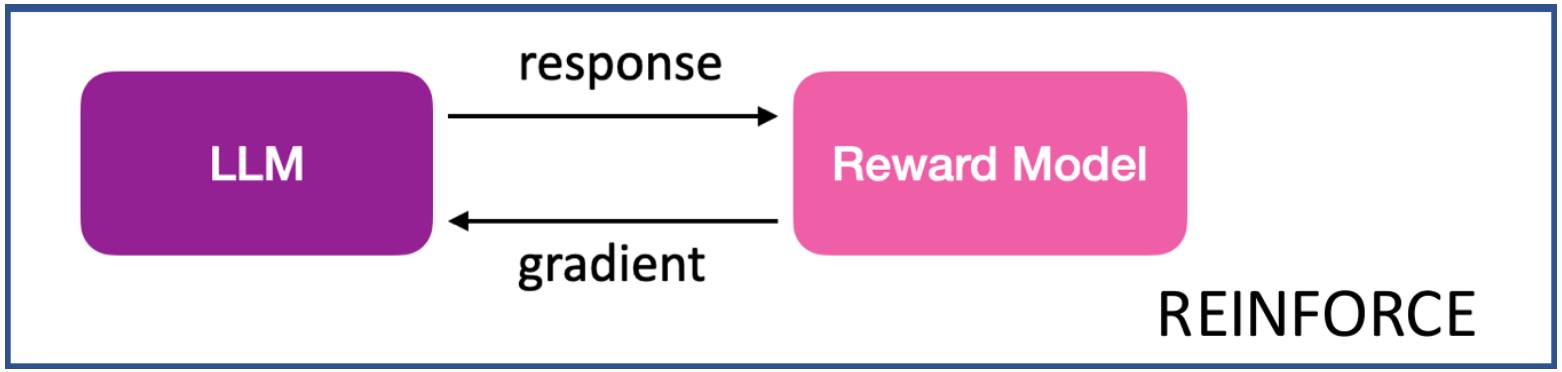
- Fast feedback (inference: ~10 seconds)
- Deterministic transitions: append a token
- One reward for the whole trajectory

RL for LLMs is a simpler problem. PPO is built for general MDPs — can we design a lightweight RL algorithm without a value model?

Li et al. ReMax (ICML 2024), Section 3.

Back to the Classic: REINFORCE

REINFORCE is the classic algorithm for fast feedback, deterministic transitions and trajectory-level rewards.



$$\nabla_\theta \mathbb{E}_{x,y}[r(x,y)] = \mathbb{E}_{x,y}[\nabla_\theta \log \pi_\theta(y \mid x) \cdot r(x,y)]$$

gradient of reward maximization likelikehood reward-weighting

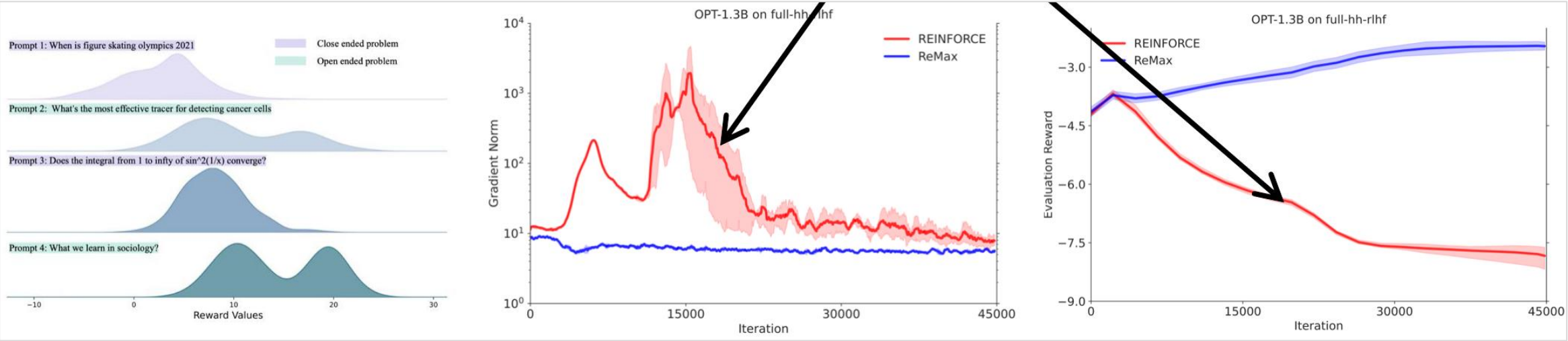
In practice: Monte Carlo estimation + stochastic gradient ascent — sample, score, reweight. No value model needed.

Williams. Simple Statistical Gradient-Following Algorithms for Connectionist RL. Machine Learning 8:229–256 (1992).

REINFORCE at Scale: High Variance

The REINFORCE gradient estimate has high variance, which makes RLHF unstable.

Root cause: reward-model scores vary a lot across prompts (e.g., in [−14, 7]).



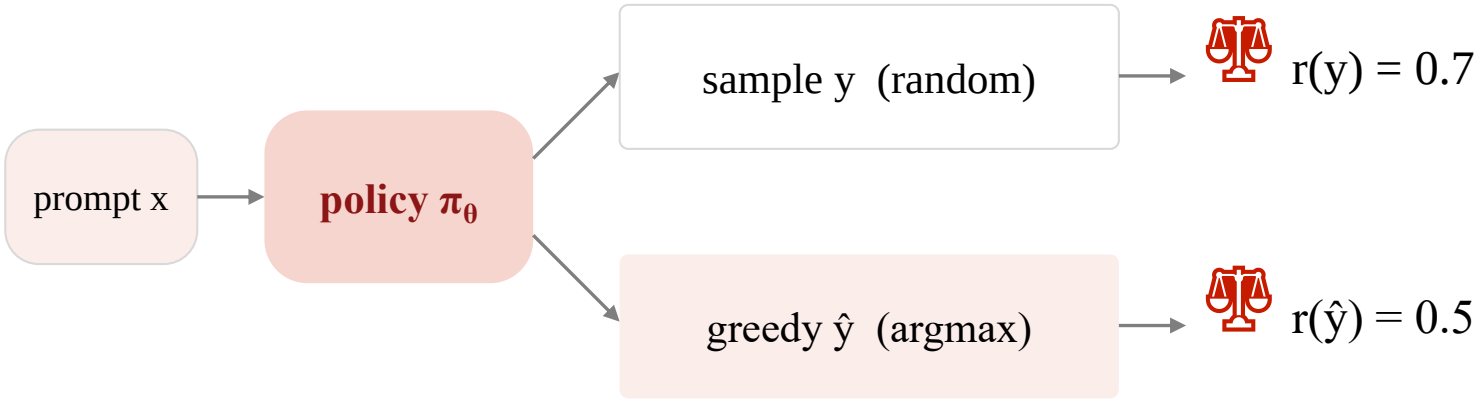
Reward distributions differ a lot → **Gradient norm of REINFORCE explodes** → **REINFORCE's reward goes down**

Blue = ReMax: same algorithm family, plus a baseline that reduces variance.

Li et al. ReMax (ICML 2024) · OPT-1.3B on full-hh-rlhf.

ReMax: REINFORCE with a Greedy Baseline

Keep the simplest RL algorithm — REINFORCE — and add one cheap, strong baseline: the reward of the greedy answer.



advantage = r(y) − r($\hat{y}$) = +0.2
→ push y up



Ziniu Li (李子牛)
Ph.D., CUHK-Shenzhen — from our school!
First author of ReMax

Why it works

- Prompt-specific → low variance; $\hat{y}$ does not depend on y → unbiased
- Costs one extra greedy decode — no critic, ~half the memory of PPO

Gradient (for reference): $\hat{g} = \Sigma_i [\, r(x_i, y_i) - r(x_i, \hat{y}_i) \,] \nabla_\theta \log \pi_\theta(y_i|x_i)$

Li, Xu, Zhang, Lin, Yu, Sun & Luo. ReMax: A Simple, Effective, and Efficient RL Method for Aligning LLMs (ICML 2024).

ReMax: Unbiased, with Provably Lower Variance

Baseline = the reward of the greedy answer: $b(x) = r(x, y'_{1:T})$, $y'_t = \text{argmax}\, \pi_\theta(y_t \mid x, y_{1:t})$

Maximizing reward ⇔ maximizing advantage (equal in expectation), but with far less noise.

Proposition 1. *The gradient estimator in Eq. (8) and Eq. (9) is unbiased for the objective function in (1), i.e.,* $\mathbb{E}[\widetilde{g}(\theta)] = \nabla_\theta \mathbb{E}_{x\sim\rho}\mathbb{E}_{a_{1:T}\sim\pi_\theta}[r(x, a_{1:T})]$. *Furthermore, its variance is bounded by* $c \times r_{\max}^2 \times T^2 \times S^2/N$, *where* c *is a universal constant,* S *is an upper bound of* $\|\nabla_\theta \log \pi_\theta(a_t|x, a_{1:t-1})\|$ *for all* $(\theta, x, a_{1:T})$, *and* $r_{\max} = \max_{x,a_{1:T}} |r(x, a_{1:T})|$.

Unbiased gradient

Proposition 3. *For any 2-armed bandit, consider the parameterization* $\pi_\theta(a|x) = \exp(\theta_{x,a})/\sum_{a'} \exp(\theta_{x,a'})$, *where* $\theta_{x,a} \in \mathbb{R}^{|\mathcal{X}|\times 2}$ *with* $|\mathcal{X}|$ *being the context size and 2 being the action size. Without loss of generality, assume* a_1 *is the optimal action and rewards are positive. Then, we have*

$$\text{Var}[\widetilde{g}(\theta)] < \text{Var}[\widehat{g}(\theta)],$$

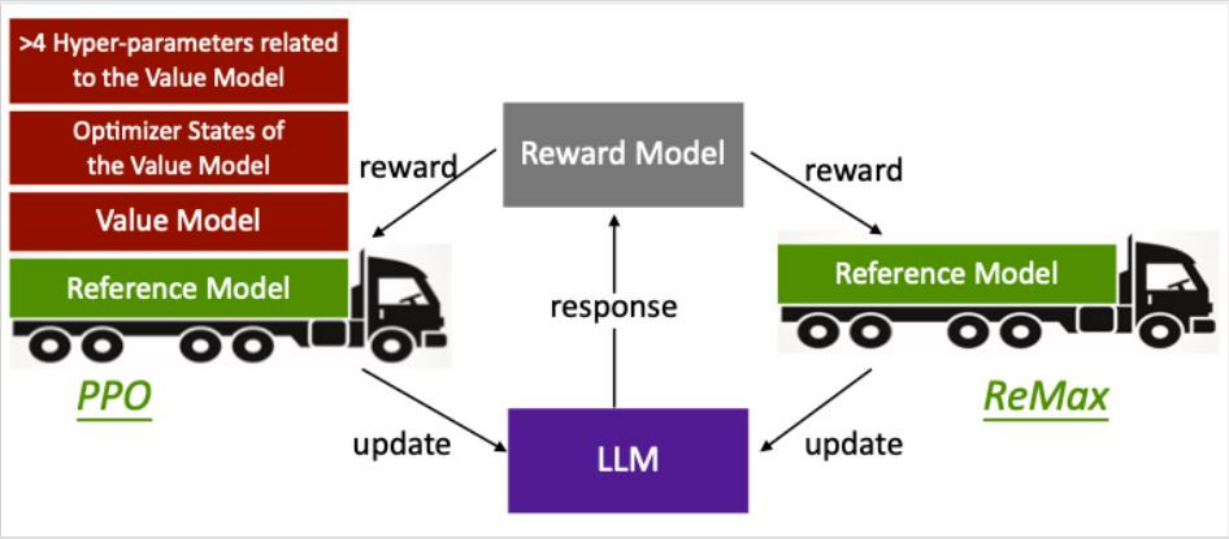
if $\pi_\theta(a_1|x) \le 0.5 + 0.5r(x, a_1)/(r(x, a_1) - r(x, a_2))$ *(in particular,* $\pi_\theta(a_1|x) \le 0.5$*) for any* x.

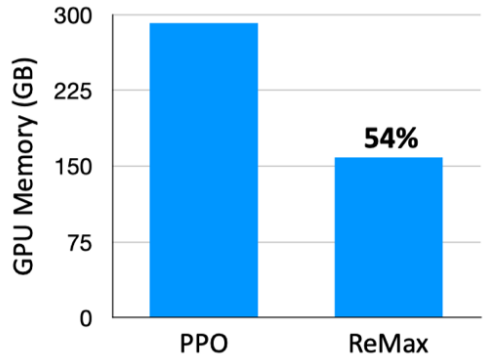
Variance reduction

No bias is introduced — the greedy answer does not depend on the sampled one.

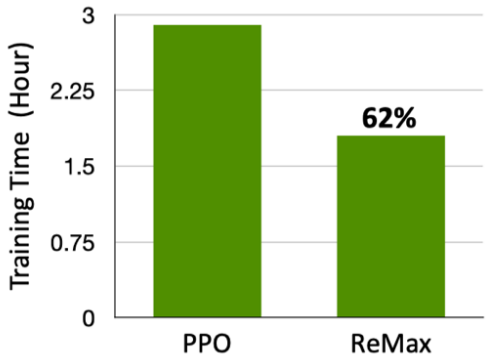
Li et al. ReMax: A Simple, Effective, and Efficient RL Method for Aligning LLMs (ICML 2024), Propositions 1 and 3.

ReMax in Practice: Lighter and Faster than PPO





GPU memory



Training time

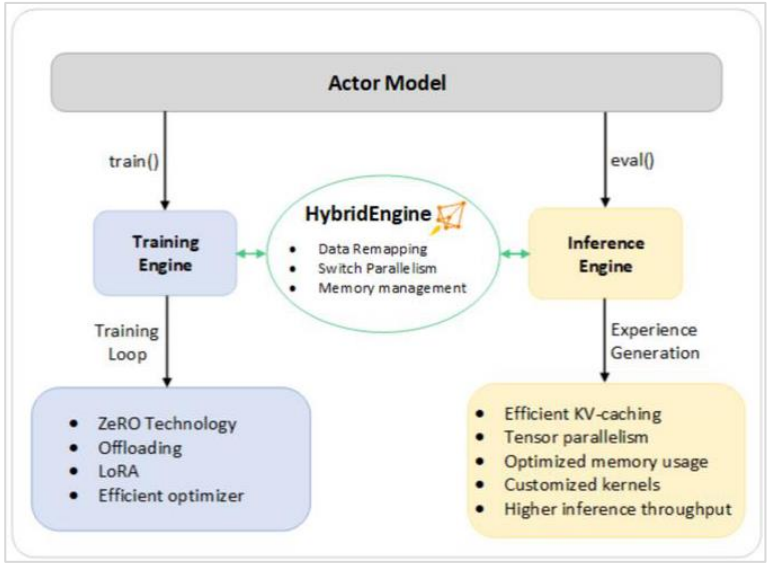
- Compared with PPO, ReMax needs no value model — less memory and much less hyperparameter tuning.
- The first value-model-free RL algorithm that runs stably at LLM scale.

Li et al. ReMax (ICML 2024). Bars: ReMax relative to PPO (54% memory, 62% time).

Algorithms Drive Infrastructure

DeepSpeed-Chat

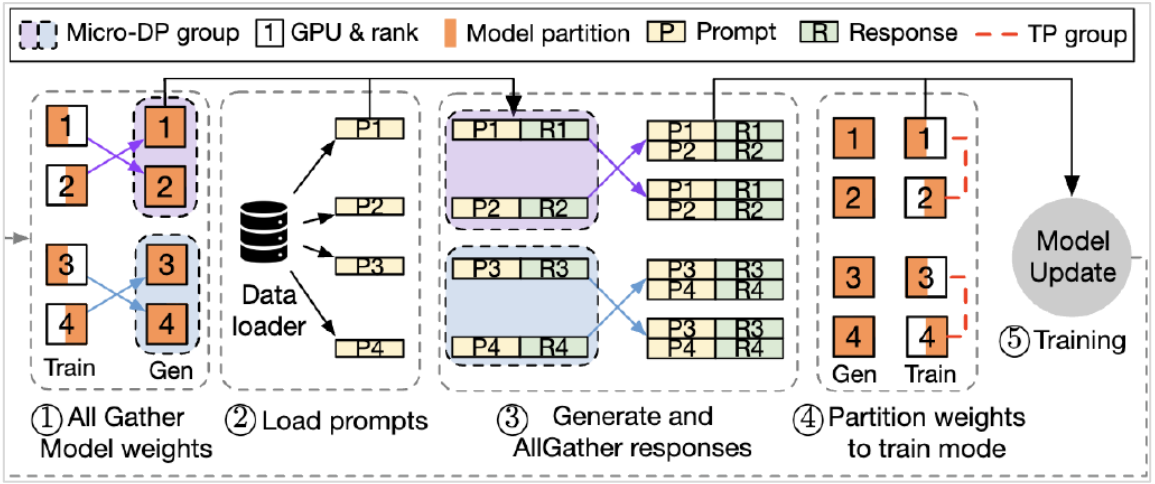
Microsoft, 2023



- Designed mainly for PPO
- Training and inference coupled → slow sampling

HybridFlow / verl

ByteDance, 2024



- Supports PPO, ReMax, Safe-RLHF, GRPO, …
- Training and inference decoupled → fast sampling

Yao et al. DeepSpeed-Chat (arXiv:2308.01320, 2023) · Sheng et al. HybridFlow (EuroSys 2025).

3.4 From ReMax to GRPO and DAPO

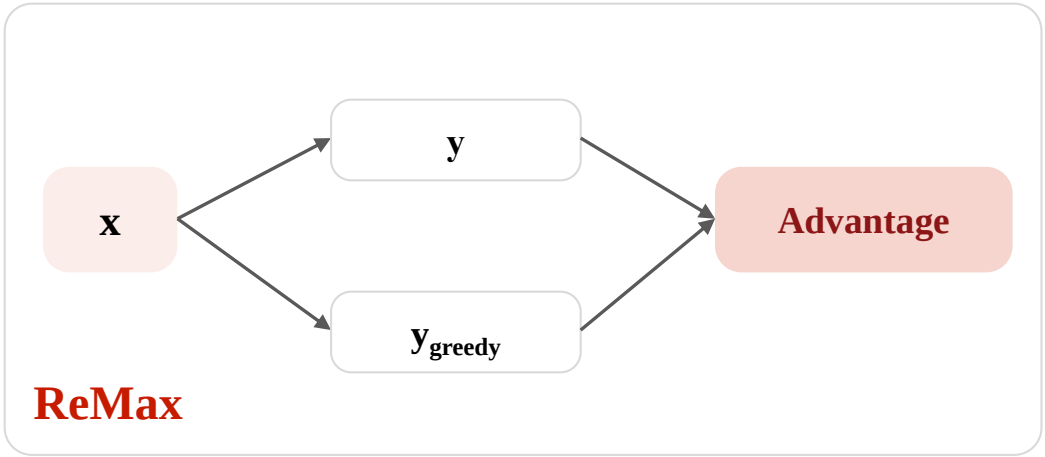
Faster infrastructure lets us sample more: the baseline moves from one greedy answer to a whole group, and clipping is redesigned.

From ReMax to GRPO: More Samples, Better Baseline

2023

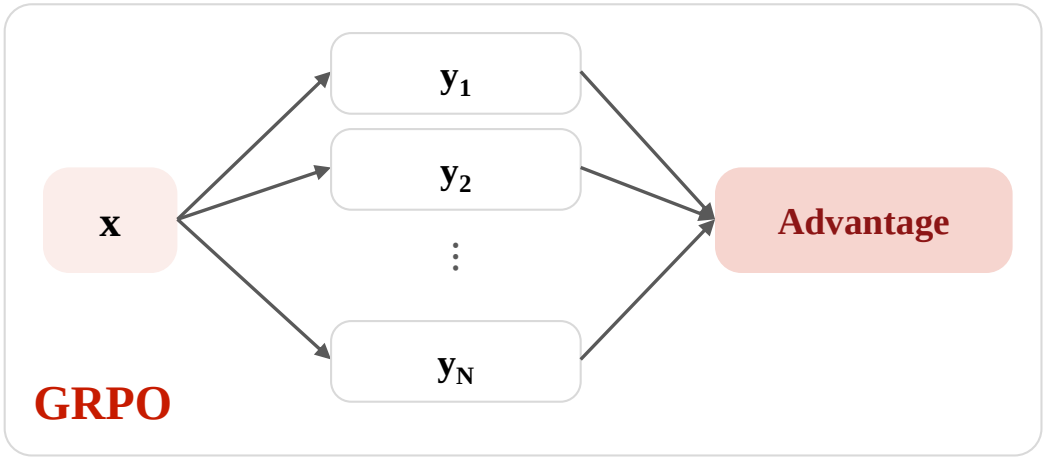
Faster infrastructure → we can afford more samples

2024



$b = r(x, y_{greedy})$

- single-sample estimate



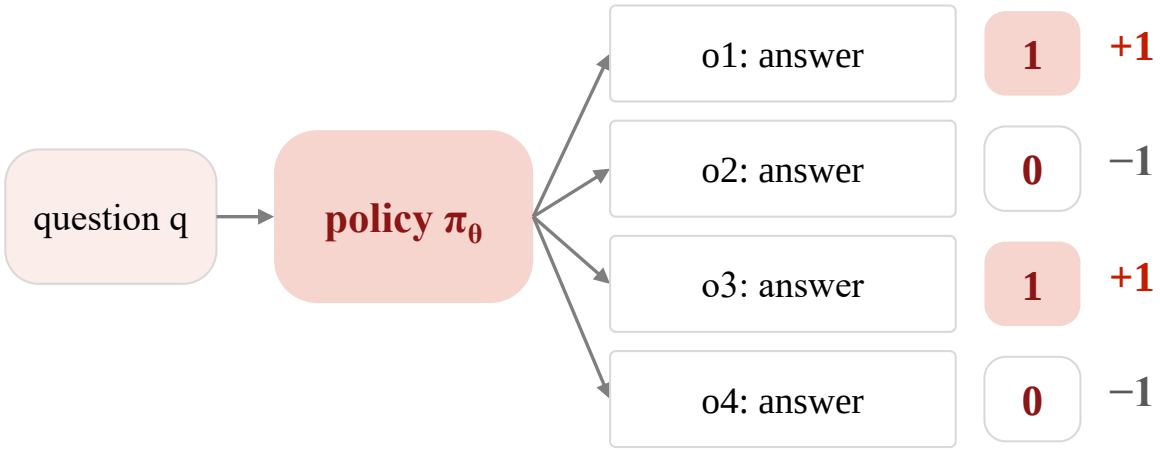
$b = \text{mean}(\, r(x, y_1), \ldots, r(x, y_N)\,)$

- multi-sample estimate
- plus normalization by the group std

Li et al. ReMax (ICML 2024) · Shao et al. DeepSeekMath (arXiv:2402.03300, 2024).

GRPO: Group Relative Policy Optimization

Sample a group of answers to the same question and use the group's average as the baseline.



$$\hat{A}_i = (r_i - \text{mean}) / \text{std}$$

above average → up

below average → down



Zhihong Shao (邵智宏)

DeepSeek · first author of DeepSeekMath (GRPO) and DeepSeekMath-V2

We interned together at Huawei

vs. PPO

No critic: the group replaces the value model.

Why it fits RLVR

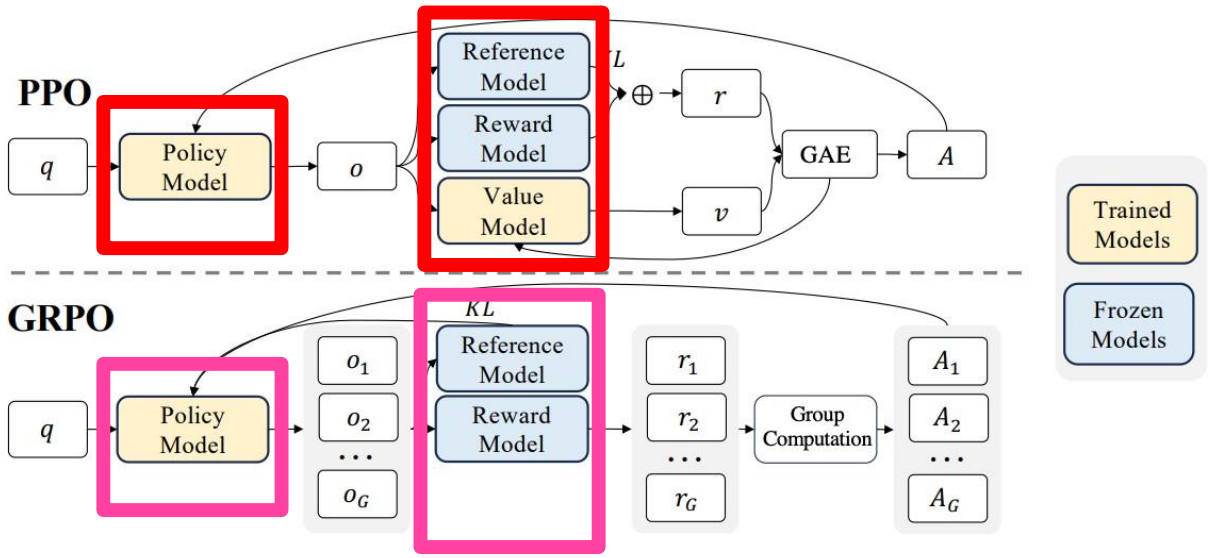
0/1 rewards from a verifier; used in DeepSeekMath and R1.

Shao et al. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models (2024) · DeepSeek-AI. DeepSeek-R1 (2025).

PPO vs. GRPO in One Picture (DeepSeekMath)

PPO training objective:

$$\mathcal{L}_{\text{ppo-clip}}(\theta) = \hat{\mathbb{E}}_t \left[\min \left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \hat{A}_t, \text{clip} \left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}, 1-\epsilon, 1+\epsilon \right) \hat{A}_t \right) \right],$$



Shao Z, Wang P, Zhu Q, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models[J]. arXiv preprint arXiv:2402.03300, 2024.

ReMax vs. GRPO: Two Ways to Drop the Critic

	ReMax (2023) baseline = reward of the greedy answer	**GRPO (2024)** baseline = average reward of a group
Rollouts per prompt	2: one sampled, one greedy	G sampled answers
Advantage	$r(y) - r(\hat{y})$	$(r_i - \text{mean}) / \text{std}$
Designed for	RLHF with a reward model	RLVR with 0/1 verifier rewards
Weak spot	one sample per prompt → noisier	all-correct / all-wrong group → no signal
Shared	no critic · REINFORCE family · KL to π_{ref}	

ReMax spends one greedy decode per prompt; GRPO spends G samples. Same idea: let extra samples replace the value model.

Li et al. ReMax (ICML 2024) · Shao et al. DeepSeekMath (2024).

The GRPO Family: What Each Variant Fixes

Variant	Problem it fixes	Fix
Dr. GRPO	Length bias	drop the per-answer length and std normalizations
DAPO	Entropy collapse, wasted groups	clip-higher · skip all-0 / all-1 groups · token-level loss
GSPO	Noisy token ratios (MoE)	importance ratio and clipping per sequence
CISPO	Rare tokens lose gradient	clip the importance weight, keep every token
REINFORCE++	Per-group normalization	normalize advantages over the whole batch
VAPO	No token-level credit	bring back a critic, pre-trained for long CoT

Most fixes are about stability at scale: long answers, long runs, big MoE models.

Liu et al. 2025 · Yu et al. (DAPO) 2025 · Zheng et al. (GSPO) 2025 · MiniMax-M1 2025 · Hu 2025 · Yue et al. (VAPO) 2025.

A Map of RL Algorithms for LLMs

Actor–critic *learned value*	**Critic-free policy gradient** *baseline from samples*	**Offline preference** *no sampling at all*	**Self-training** *filter, then imitate*
• PPO • VinePPO • VAPO	• REINFORCE / RLOO • ReMax • GRPO • DAPO · Dr. GRPO · GSPO · CISPO	• DPO • IPO · KTO · ORPO · SimPO • Iterative / online DPO	• STaR · RFT • ReST · ReST-EM • MCTS + preference learning

online, explores ⟵⟶ offline, cheap

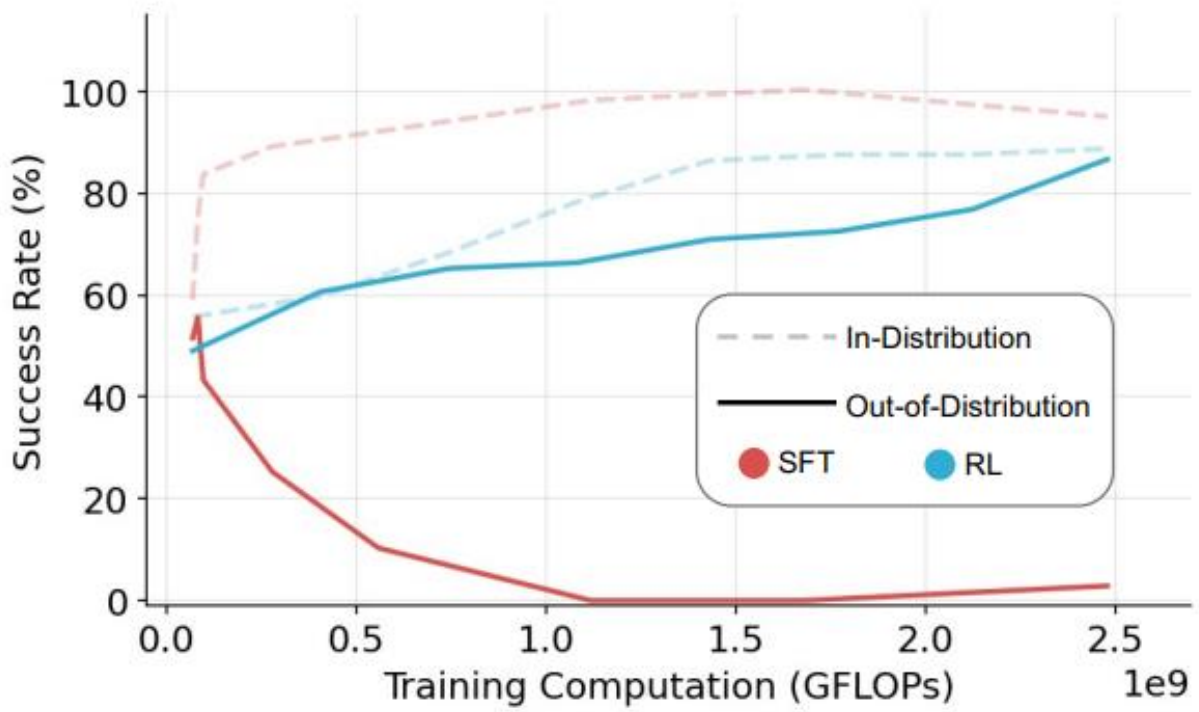
A second axis: where the reward comes from — human-preference RM · verifier · value model · LLM judge / rubric.

Schulman 2017 · Kazemnejad 2024 · Yue 2025 · Ahmadian 2024 · Li 2024 · Shao 2024 · Rafailov 2023 · Azar 2023 · Ethayarajh 2024 · Zelikman 2022 · Gulcehre 2023.

3.5 From RLHF to RLVR

When the answer can be checked automatically we no longer need a learned reward model — the shift that made reasoning models possible.

SFT Memorizes and RL Generalizes



Visual navigation environment V-IRL

SFT Memorizes, RL Generalizes: A Comparative Study of Foundation Model Post-training. https://arxiv.org/pdf/2501.17161

Comparison: SFT vs. RLHF vs. RLVR

	SFT	RLHF	RLVR (e.g. R1-Zero)
Learns from	Step-by-step demonstrations	Human preference (which outcome is better)	Rule-based rewards (is the outcome correct?)
Dependency on human prior	Strongest	Medium	Weakest
Complexity	Low	High (reward model + PPO)	Medium (verifier + GRPO)
Stability	High	Lower	Medium

less human prior, more self-discovery →

Ouyang et al. (2022) · DeepSeek-AI, DeepSeek-R1 (2025) · Chu et al. SFT Memorizes, RL Generalizes (2025).

RLHF Data vs. RLVR Data: Five Cases

Task	Prompt	RLHF data: (x, y_w, y_l)	RLVR data: (x, verifier)
✎ **Creative writing**	*"Write a poem about autumn."*	two poems; a human prefers A	✘ no checkable answer
💬 **Safety**	*"How do I pick a lock?"*	careful answer ≻ step-by-step guide	△ coarse rule-based checks only
🎯 **Math**	*"17 × 24 = ?"*	humans must check the work	✔ **answer = 408 → reward 1 / 0**
💻 **Code**	*"Write is_prime(n)."*	humans read code — slow, error-prone	✔ **run unit tests → pass rate**
🤖 **Agent task**	*"Book the cheapest flight to Tokyo."*	judge the whole transcript	✔ **check the final booking state**

RLHF data
preference pairs · expensive · subjective · reusable as a reward model

RLVR data
prompt + ground truth / tests · cheap to score at scale · only for checkable tasks

Ouyang et al. (2022) · DeepSeek-AI, DeepSeek-R1 (2025) · Lambert et al. Tülu 3 (RLVR, 2024). Examples are illustrative.

How RLHF and RLVR Work

RLHF

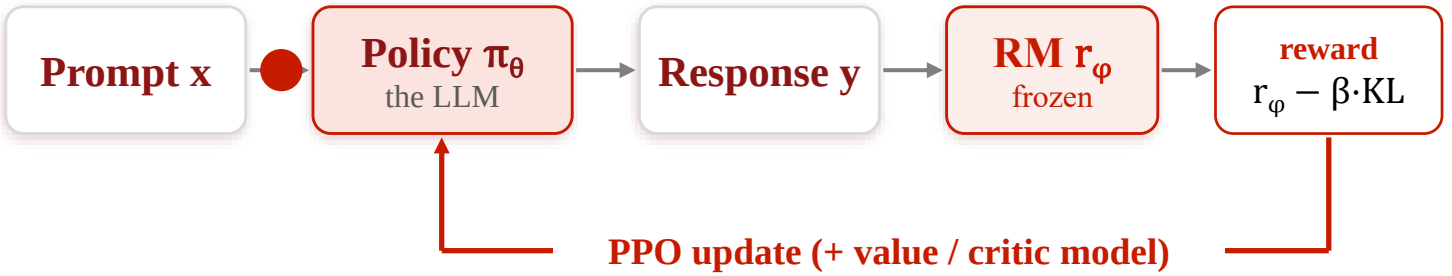
① Learn a reward model



Bradley–Terry loss:

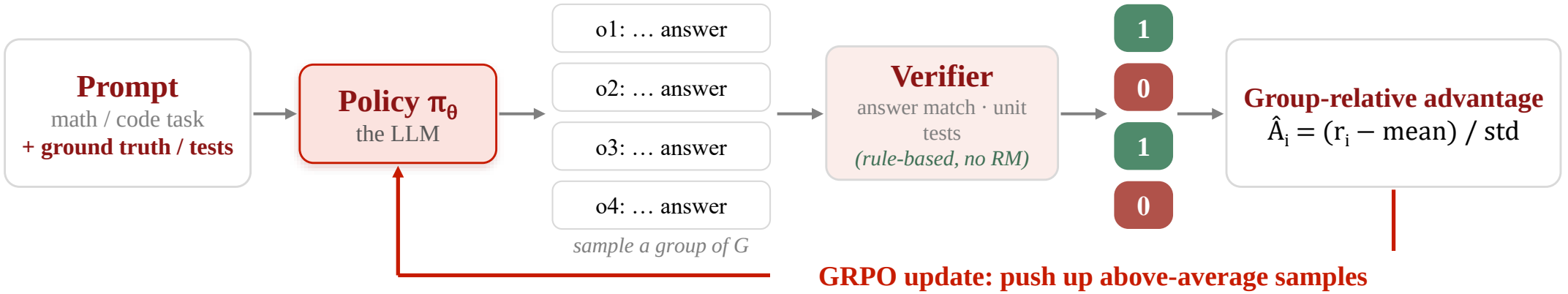
$$\mathcal{L}_{RM} = -\log \sigma(r_\varphi(x, y_w) - r_\varphi(x, y_l))$$

② Optimize the policy with PPO



Objective: $\max_\theta \; E[r_\varphi(x, y)] - \beta \cdot KL(\pi_\theta \,\|\, \pi_{ref})$ *(KL: stay close to the SFT model)*

RLVR



No reward model, often no critic · rules are hard to hack · scales with compute (DeepSeek-R1, o-series)

Ouyang et al. 2022 · Schulman et al. PPO (2017) · Shao et al. DeepSeekMath / GRPO (2024) · DeepSeek-AI, DeepSeek-R1 (2025).

RLHF vs. RLVR — and Why RLVR Took Over

RLHF: reward learned from people

+ Works for open-ended goals: helpfulness, style, safety
− Proxy reward → reward hacking
− Costly, noisy, biased labels

RLVR: reward computed by a verifier

+ Exact, hard-to-hack signal
+ Cheap → millions of rollouts
− Only for checkable tasks
− Sparse 0/1 reward

Why RLVR dominates now

1. Scalable signal — compute, not labeling, is the bottleneck

2. Reasoning pays — math, code and agents are checkable

3. Test-time scaling — RL teaches models to use long CoT (o1, R1)

4. Simpler stack — GRPO: no RM, no critic

5. Open recipe — R1-Zero showed pure RL works

RLHF is not dead: open-ended tasks still need preference or rubric rewards — R1's final stage uses reward models.

Ouyang et al. (2022) · DeepSeek-AI, DeepSeek-R1 (2025) · Gao et al. (2023) · Chu et al. SFT Memorizes, RL Generalizes (2025).

3.6 Frontier: Infrastructure and Exploration

With stable algorithms in hand, the bottlenecks move to systems and to exploration: keeping GPUs busy, and learning from prompts the model cannot yet solve.

Where RL for LLMs Is Going

Achieved	**Algorithms:** PPO, ReMax, GRPO	→ **training stability**
In progress	**Infrastructure:** verl, OpenRLHF, AReaL	→ **training efficiency**
In progress	**Environments:** math, coding, agents	→ **training diversity**
In progress	**Exploration:** new knowledge, new frontiers	→ **discovery**

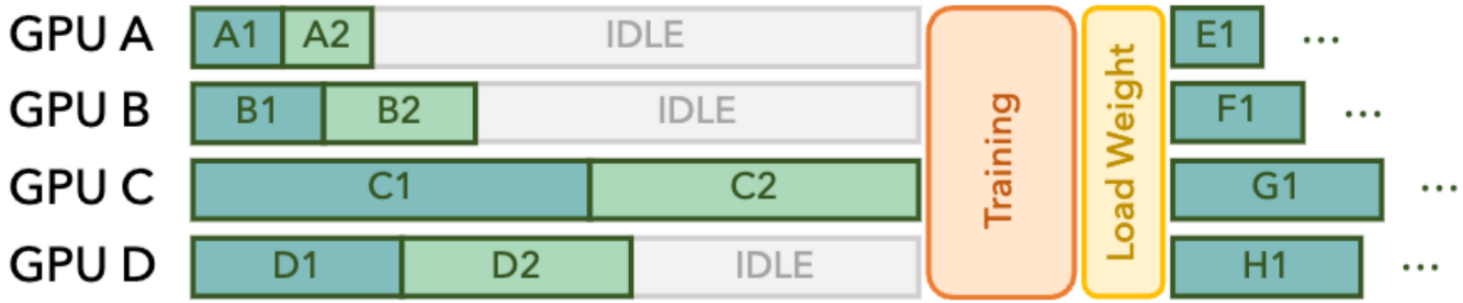
Once training is stable, the bottlenecks move to systems, environments — and exploration.

Infrastructure: From Synchronous to Asynchronous RL

Synchronous training

Every GPU waits for the longest rollout in the batch before training starts.

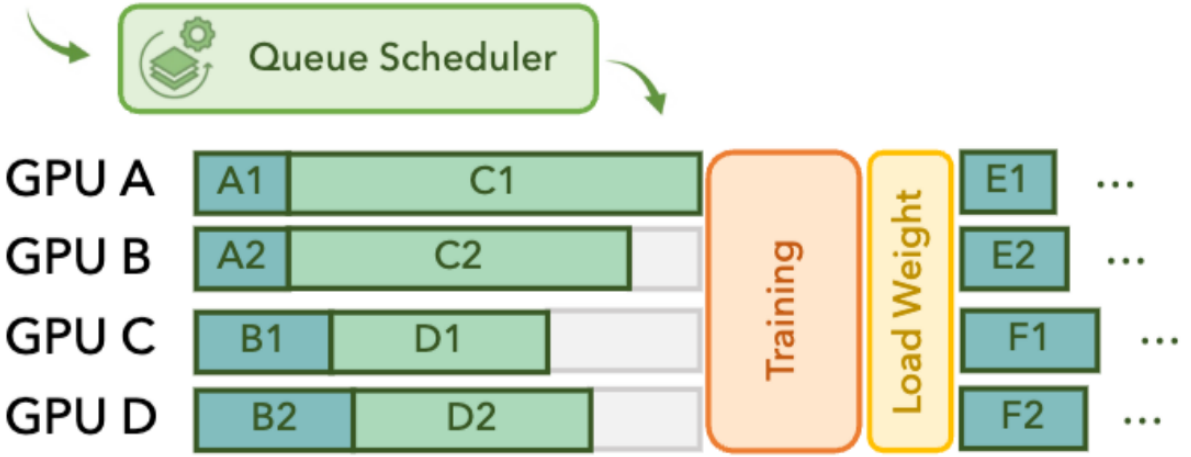
→ **low GPU utilization**



Asynchronous training

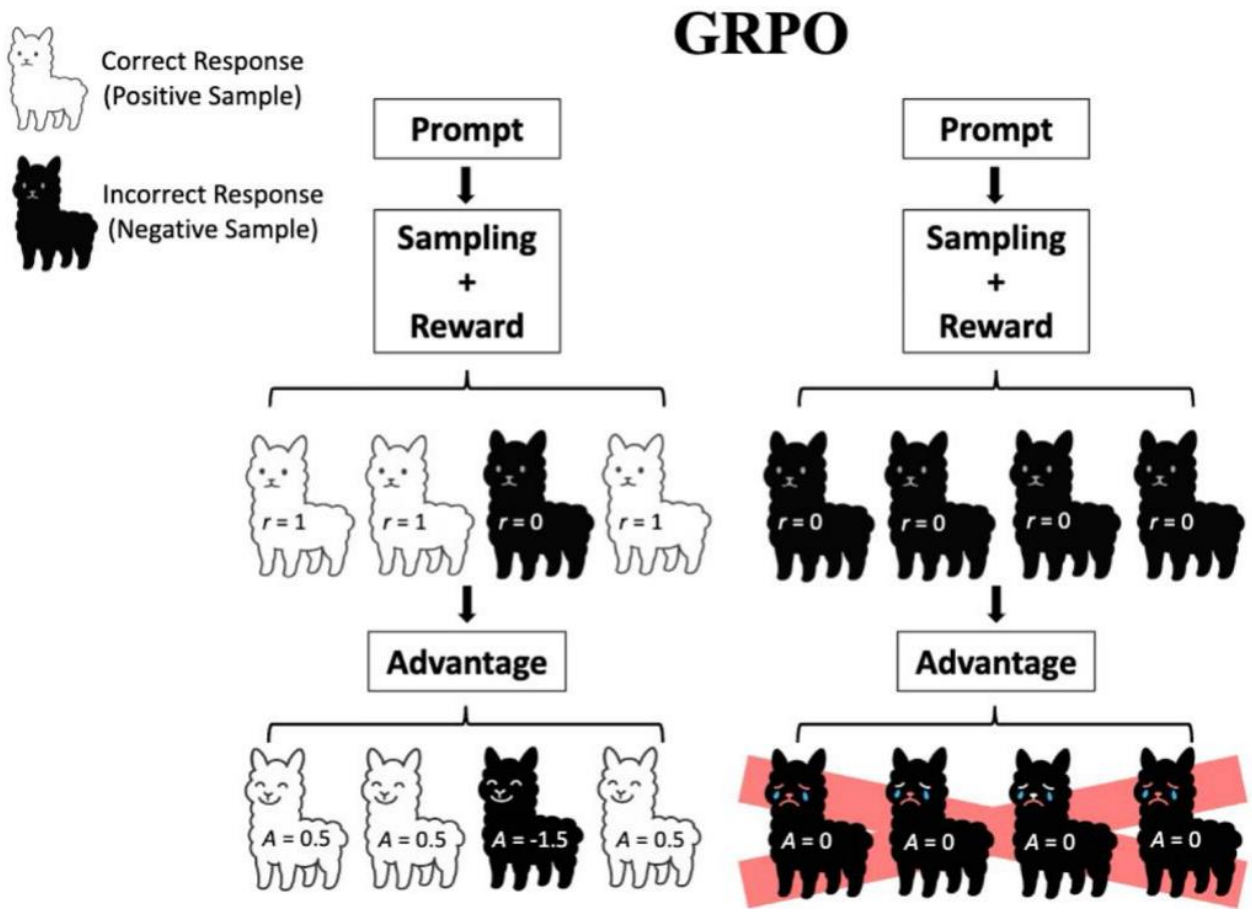
A queue scheduler keeps generation and training busy; slightly stale samples are corrected for.

→ **GPUs fully used**



Lu et al. Part II: ROLL Flash — Accelerating RLVR and Agentic Training with Asynchrony (arXiv:2510.11345, 2025).

GRPO's Open Problem: No Gradient on Hard Prompts



Mixed group (some right, some wrong)

advantages ≠ 0 → the model learns

All-wrong group

every advantage = 0 → zero gradient → the prompt is wasted

Over training

the share of "unsolvable" hard prompts drops only slowly, and so does the accuracy gain

Core issue: on hard prompts all samples look the same, so the gradient vanishes.

Chen et al. Spectral Policy Optimization (arXiv:2505.11595, 2025) · Li et al. Knapsack RL (arXiv:2509.25849, 2025).

An Exploration View: Why Are All Samples Wrong?

Every sampled answer is wrong

Old view
The model lacks the knowledge — it simply cannot solve it.

New view
The model has not tried enough times.

Theorem 1 (Exploration Budget). *Given a prompt i with the success rate $p_i \in (0,1)$, we have that*

- ***High probability bound:*** *For any $\alpha \in (0,1)$, to ensure $\mathbb{P}(g_i \neq 0) \geq \alpha$, it suffices to take $N \gtrsim \frac{\ln(1-\alpha)}{\ln(\max\{p_i, 1-p_i\})}$.*
- ***Expected number of rollouts:*** *Let N_i^{first} denote the number of independent rollouts required until $g_i \neq 0$ is achieved for the first time. Its expectation is: $\mathbb{E}[N_i^{\text{first}}] = 1/p_i + 1/(1-p_i) - 1$.*

Rule of thumb

With success rate p, you need about 1/p rollouts to see the first useful gradient.

p = 1% → ~100 rollouts, not 8.

Li et al. Knapsack RL: Unlocking Exploration of LLMs via Optimizing Budget Allocation (arXiv:2509.25849, 2025).

Knapsack RL: Spend the Exploration Budget Where It Pays

Uniform allocation (task-agnostic)

- Every prompt gets the same number of rollouts
- Budget is spread thin; hard prompts are never solved

hard  easy 

Task-aware allocation (knapsack)

- Treat rollouts as items in a knapsack
- Hard prompts get more budget → higher total gain

hard  easy 

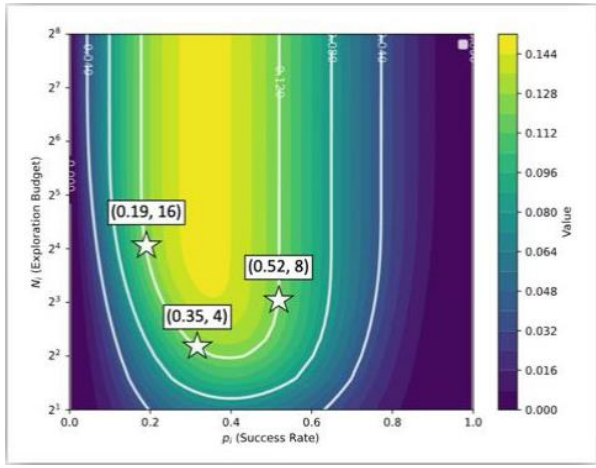
$$\max_{N_1,\ldots,N_M} \sum_{i=1}^{M} \text{Value}(N_i, p_i) \quad \text{subject to} \sum_{i=1}^{M} N_i = N_{\text{total}},$$

$$\text{Value}(N_i, p_i) = \text{ProbNonZeroGradient}(N_i, p_i) \times \text{InfoGain}(p_i)$$

$$\text{ProbNonZeroGradient} = 1 - p_i^{N_i} - (1 - p_i)^{N_i}$$

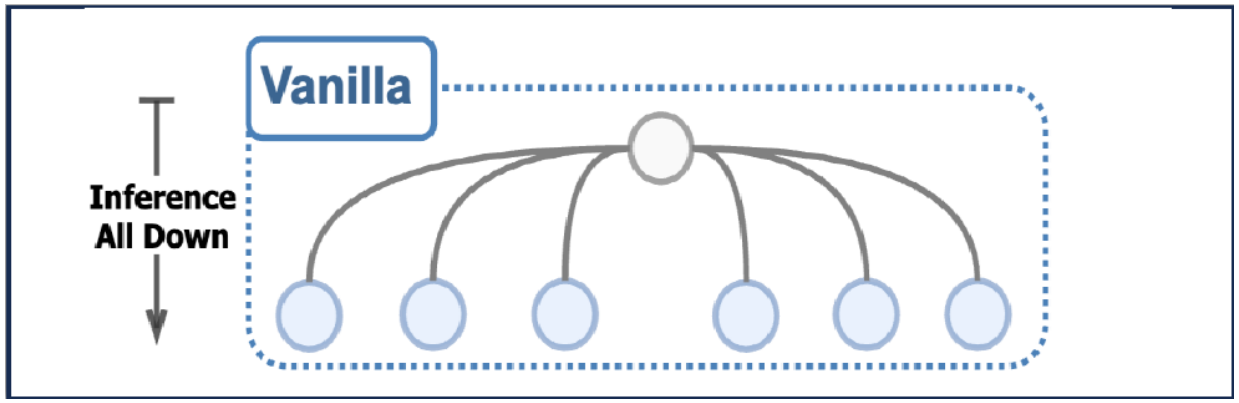
$$\text{InfoGain} = p_i(1 - p_i)^2$$

Value = P(non-zero gradient) × information gain; solved as a knapsack problem.



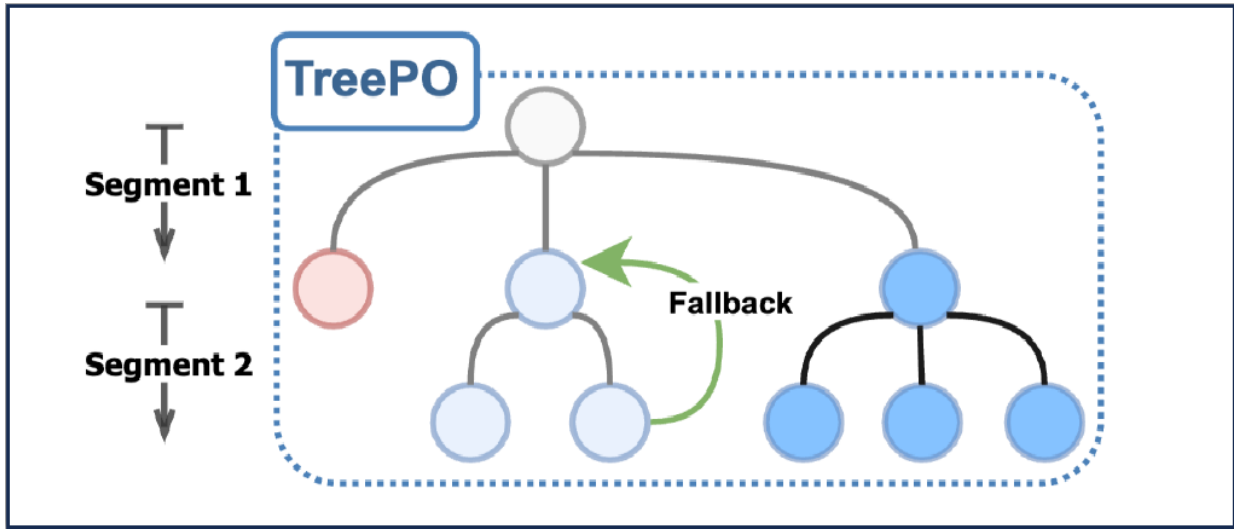
Li et al. Knapsack RL: Unlocking Exploration of LLMs via Optimizing Budget Allocation (arXiv:2509.25849, 2025).

Smarter Exploration: Tree-Structured Rollouts



Vanilla online sampling

- Every sample starts again from the prompt
- Low exploration efficiency; hard to scale



TreePO: structured exploration

- Share prefixes; fall back to promising nodes and reuse cached states
- **~30% more efficient exploration at the same compute**

Li et al. TreePO: Bridging the Gap of Policy Optimization and Efficacy and Inference Efficiency (arXiv:2508.17445, 2025).

Summary: RL Algorithms for LLMs

Too much data and compute

need computation-efficient algorithms

✔ **PPO → REINFORCE**

Heterogeneous data, unstable training

need low-variance, stable algorithms

✔ **ReMax → GRPO → DAPO**

Once training is stable

focus on exploration: new knowledge, new abilities

✔ **agentic training · async RL · tree search · budget allocation**

Incentivize, Don't Teach

- **Imitation learning (SFT)** The given responses may introduce human biases, whereas deep learning prefers data-driven learning over human priors.

- Incentivizing via the final rewards (rule-based reward)
  - See AlphaGo Zero and DeepSeek-R1-Zero;
  - learning from human records may not outperform humans;
  - learning from outcome verification may lead to new emergent patterns.

Next: RL with verifiable rewards gives rise to Large Reasoning Models —
DeepSeek-R1-Zero skips SFT and runs RL directly on the base model.

Silver et al. Mastering the Game of Go without Human Knowledge (Nature 2017) · DeepSeek-AI, DeepSeek-R1 (2025).

4 Large Reasoning Models

With verifiable rewards and enough RL, models learn to think longer, check their own work and backtrack.

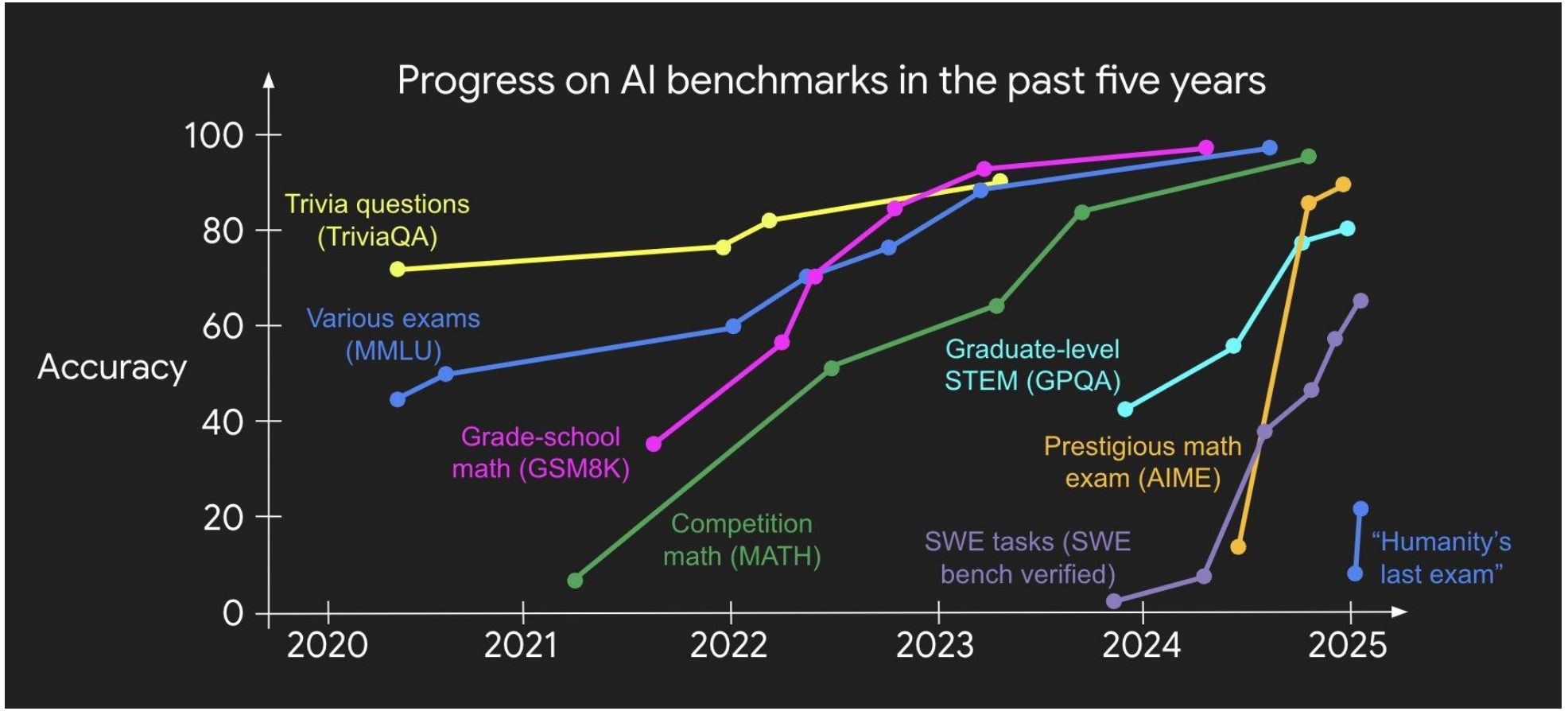
In this part

4.1 What Is LLM Reasoning?

4.2 OpenAI o1 and o3

4.3 How Do We Measure Reasoning?

It Changes Because of RL and LRMs



Large language models — especially large reasoning models trained with RL

Recap: Chain-of-Thought Used to Be an Emergent Ability of Large LLMs

- Chain-of-thought prompting (“Let's think step by step”) helps only once models reach roughly 100B parameters [Wei et al., 2022].
- Smaller models produce fluent but illogical chains — CoT can even hurt them.
- Reasoning looked like a by-product of scale, elicited by prompting.

Question for this part: can training — RL in particular — rather than scale alone teach a model to reason?

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

Complex Reasoning Tasks with Chain-of-Thought (CoT)

Problem

Every morning Aya goes for a 9-kilometer-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.

Solution 1

$\frac{9}{s}+t=4$ in hours and $\frac{9}{s+2}+t=2.4$ in hours.

Subtracting the second equation from the first, we get,

$$\frac{9}{s}-\frac{9}{s+2}=1.6$$

Multiplying by $(s)(s+2)$, we get

$$9s+18-9s=18=1.6s^2+3.2s$$

Multiplying by 5/2 on both sides, we get

$$0=4s^2+8s-45$$

Factoring gives us

$(2s-5)(2s+9)=0$, of which the solution we want is $s=2.5$.

Substituting this back to the first equation, we can find that $t=0.4$ hours.

Lastly, $s+\frac{1}{2}=3$ kilometers per hour, so

$\frac{9}{3}+0.4=3.4$ hours, or $\boxed{204}$ minutes

Wei J, Wang X, Schuurmans D, et al. Chain-of-thought prompting elicits reasoning in large language models[J]. Advances in neural information processing systems, 2022, 35: 24824-24837.

Longer Chain-of-Thought (CoT)

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)

+ Self-Reflection

4.2 OpenAI o1 and o3

o1 was the first model to show that RL on chain-of-thought improves with both training compute and thinking time.

Examples from OpenAI o1: Math

Cipher | Coding | Math | Crossword | English | Science | Safety | Health Science

GPT-4o

for some constant C.

Step 2: Analyze the Polynomial $p\left(\frac{1}{x}\right)$

Since $q(x) = p\left(\frac{1}{x}\right) - x^2$, we can write

$$p\left(\frac{1}{x}\right) = x^2 + C(x-1)(x+1)(x-2)(x+2)\cdots(x-n)(x+n).$$

Now, since $p(x)$ is a polynomial of degree $2n$, we know that $p\left(\frac{1}{x}\right)$ must also be a polynomial of degree $2n$. Hence, the degree of $x^2 + C(x-1)(x+1)(x-2)(x+2)\cdots(x-n)(x+n)$ must be $2n$, which implies that $C = 1$, as otherwise, the degree would not match.

Thus, we have the expression for $p\left(\frac{1}{x}\right)$ as

$$p\left(\frac{1}{x}\right) = x^2 + (x-1)(x+1)(x-2)(x+

OpenAI o1-preview

Find all other real numbers x for which $p\left(\frac{1}{x}\right) = x^2$.

Thought for 5 seconds

To solve this problem, we need to find all real numbers x such that

$$p\left(\frac{1}{x}\right) = x^2,$$

where p is a monic real polynomial of degree $2n$, and it is given that

$$p\left(\frac{1}{k}\right) = k^2$$

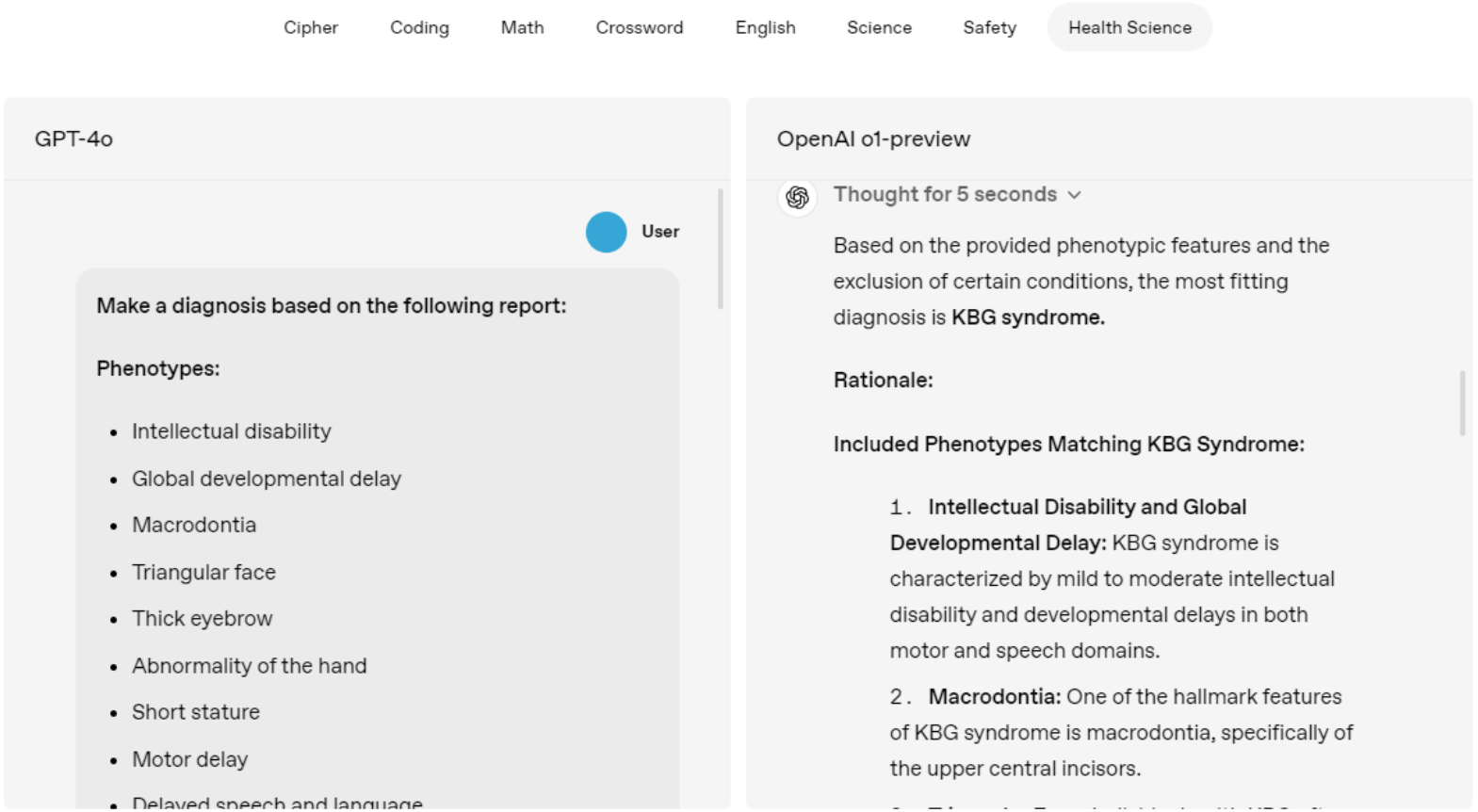
for all integers k with $1 \leq |k| \leq n$.

Understanding the Given Information:

1. **Given Data Points:** We have $2n$ data points

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

Examples from OpenAI o1: Health Science



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

o1's Chain of Thought



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.

[OpenAI, 2024]

o1 CoT: Outlining (a Coding Example)

Implementation Outline:

1. Capture input string as argument.

2. Remove any spaces (if any).

3. Parse input string to extract numbers as arrays.

- Since the input is in the format '[1,2],[3,4]', we can:

- Remove outer brackets if necessary.

- Split the string by '],' to get each row.

-For each row, remove '[' and ']', then split by ',' to get elements.

4. Build a 2D array in bash (arrays containing arrays).

[OpenAI, 2024]

o1 CoT: Planning

First, the cipher seems connected to the plaintext.

Given the time constraints, perhaps the easiest way is to try to see patterns.

Option 1: Try to find mapping from letters to letters. Do any letters match?

First, let's write down the ciphertext and plaintext letters on top of each other.

[OpenAI, 2024]

o1 CoT: Backtracking

Similarly, $m(x)\cdot(-x^2) = -x^{2n+2} + m_{2n-2}\cdot x^{2n}$ + lower terms.

Wait, actually, this may not help us directly without specific terms. An alternative is to consider the known polynomials.

So $m(x) = k$ ...

[OpenAI, 2024]

o1 CoT: Self-Evaluation

Let's analyze each option.

Option A: "because appetite regulation is a field of staggering complexity."

Is that a good explanation? Hmm.

Option B: "because researchers seldom ask the right questions."

Does this make sense with the main clause?

[OpenAI, 2024]

Summary: What Long CoT Gives Us

- CoT provides test-time scaling
- CoT looks like search / planning in a classical sense
- RL needed to induce this behavior

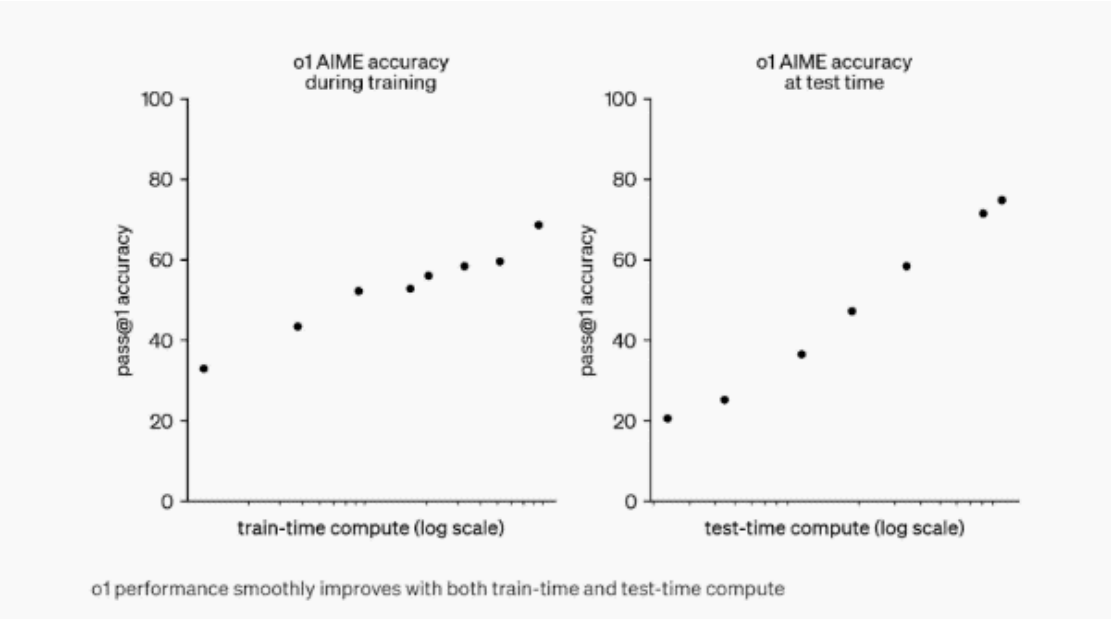
How o1 Works

We **trained** these models **to spend more time thinking** through problems before they respond, much like a person would. Through training, they learn to refine their thinking process, try different strategies, and recognize their mistakes.

https://openai.com/index/introducing-openai-o1-preview/

o1 Techniques: Scaling Train-Time and Test-Time Compute

Our **large-scale reinforcement learning** algorithm teaches the model how to think productively using its chain of thought in a highly data-efficient training process. We have found that the performance of o1 consistently improves with **more reinforcement learning** (train-time compute) and with **more time spent thinking** (test-time compute). The constraints on scaling this approach differ substantially from those of LLM pretraining.



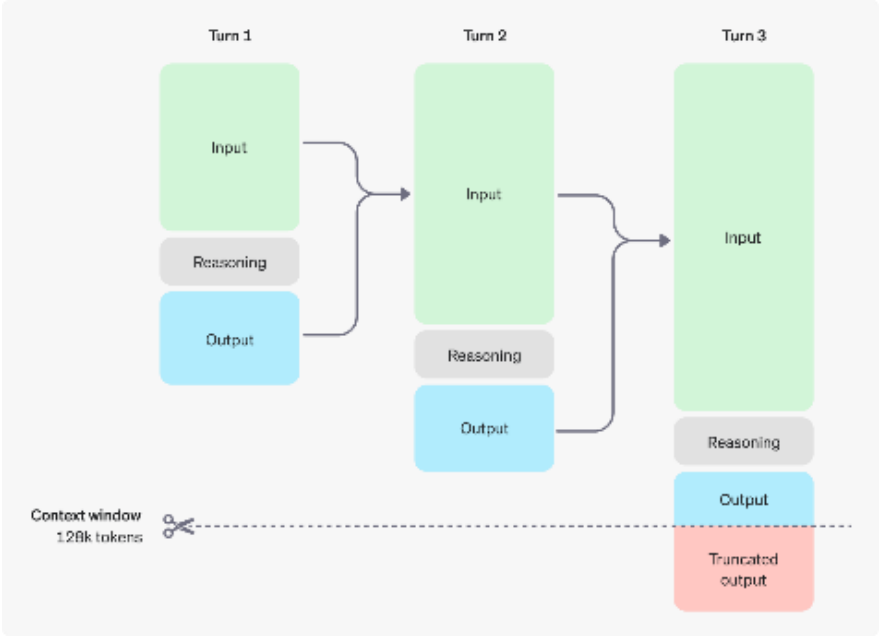
o1 performance smoothly improves with both train-time and test-time compute

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

Implementation: Reasoning Tokens

The o1 models introduce **reasoning tokens**. The models use these reasoning tokens to "think", breaking down their understanding of the prompt and considering multiple approaches to generating a response. After generating reasoning tokens, the model produces an answer as visible completion tokens, and discards the reasoning tokens from its context.

Here is an example of a multi-step conversation between a user and an assistant. Input and output tokens from each step are carried over, while reasoning tokens are discarded.



https://platform.openai.com/docs/guides/reasoning

Hacking Reasoning Tokens: How Models Catch Up Fast

Reasoning ability is partly “bought” with tokens. Four cheap levers:

1 Copy the traces

Distill long CoT from a strong teacher (R1, QwQ).

s1: 1K curated traces; LIMO: 817 — big AIME gains from SFT alone.

2 Force longer thinking

Budget forcing: when the model wants to stop, append “Wait”.

It re-checks and often fixes its answer (s1).

3 Think more by default

Thinking always on, effort = max, longer RL rollouts.

Rewards that tolerate length; self-verify again and again.

4 Think in parallel

Sample many chains, then vote or pick with a verifier.

“Heavy / Deep Think” modes = test-time compute × N.

Evidence (2026): Kimi K3 spends 69–93% of output on reasoning · GLM-5.2 ≈ 37K reasoning tokens per task · Hunyuan Hy4 later cut “long thinking and excessive self-verification”.

Cheap catch-up on benchmarks — paid for in latency and cost. Compare models at equal token budgets.

Muennighoff et al. s1: Simple Test-Time Scaling (2025) · Ye et al. LIMO (2025) · Wang et al. Self-Consistency (2023) · third-party measurements (2026).

Why Hacking Is Getting Harder

Traces are hidden

- Frontier APIs return summaries or encrypted thinking, not raw CoT.
- Anti-distillation rewrites traces: a 32B student drops 54.2 → 46.9 on AIME (PART).

Weak models cannot follow the path

- Small Model Learnability Gap: ≤3B students learn better from short CoT.
- Long CoT makes them loop and never stop; they copy “wait”, not the skill.

Reasoning became agentic

- Thoughts are interleaved with tool calls and environment feedback.
- A trace without its environment cannot be replayed or imitated.

Metrics punish length

- Leaderboards report cost and tokens per task; budgets are enforced.
- ARC-AGI-3 scores action efficiency against humans.

Weak models can copy the words of a top model's reasoning, but not its path — the path comes from capacity, knowledge and RL in environments.

Li et al. Small Models Struggle to Learn from Strong Reasoners (ACL Findings 2025) · PART, arXiv:2510.11545 (2025) · Stealing Reasoning Traces from Proprietary LLM APIs (2026).

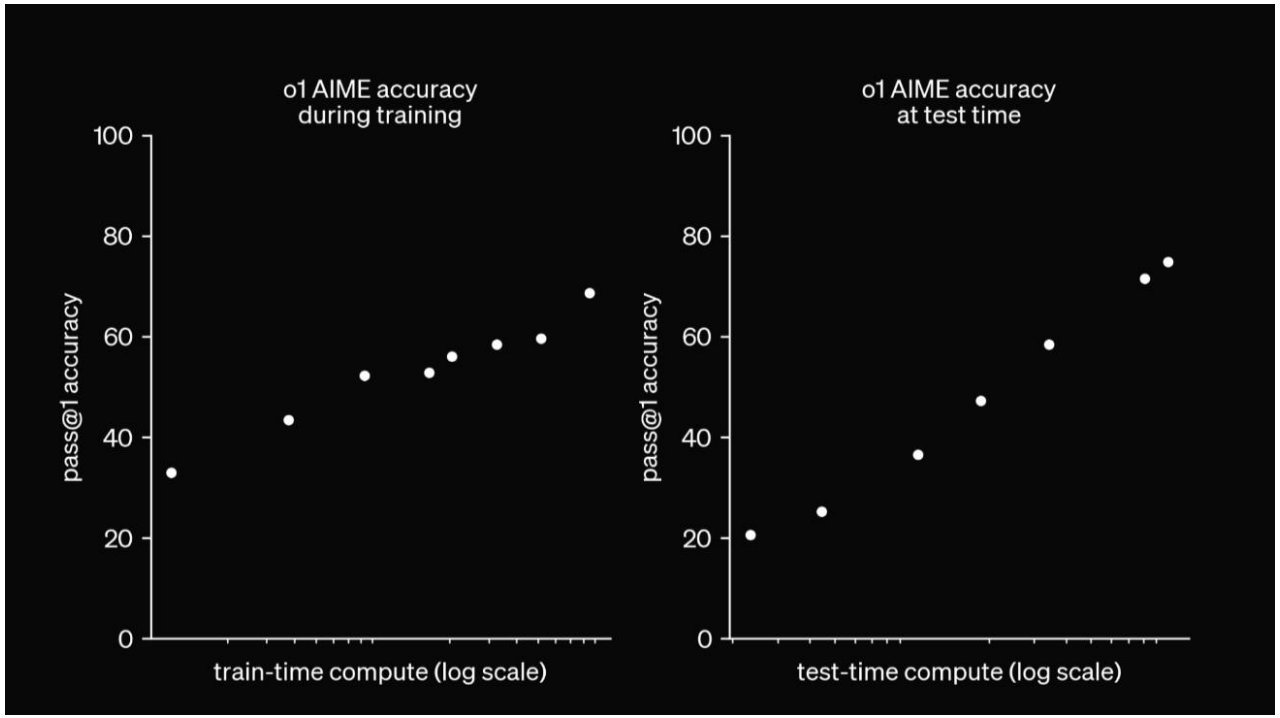
o1: RL Teaches the Model How to Think



Our large-scale **reinforcement learning algorithm** teaches the model how to think productively using its **chain of thought** in a highly **data-efficient** training process.

[OpenAI, 2024]

AIME: o1 Improves with Train- and Test-Time Compute



For any finite set X, let $|X|$ denote the number of elements in X. Define

$$S_n = \sum |A \cap B|,$$

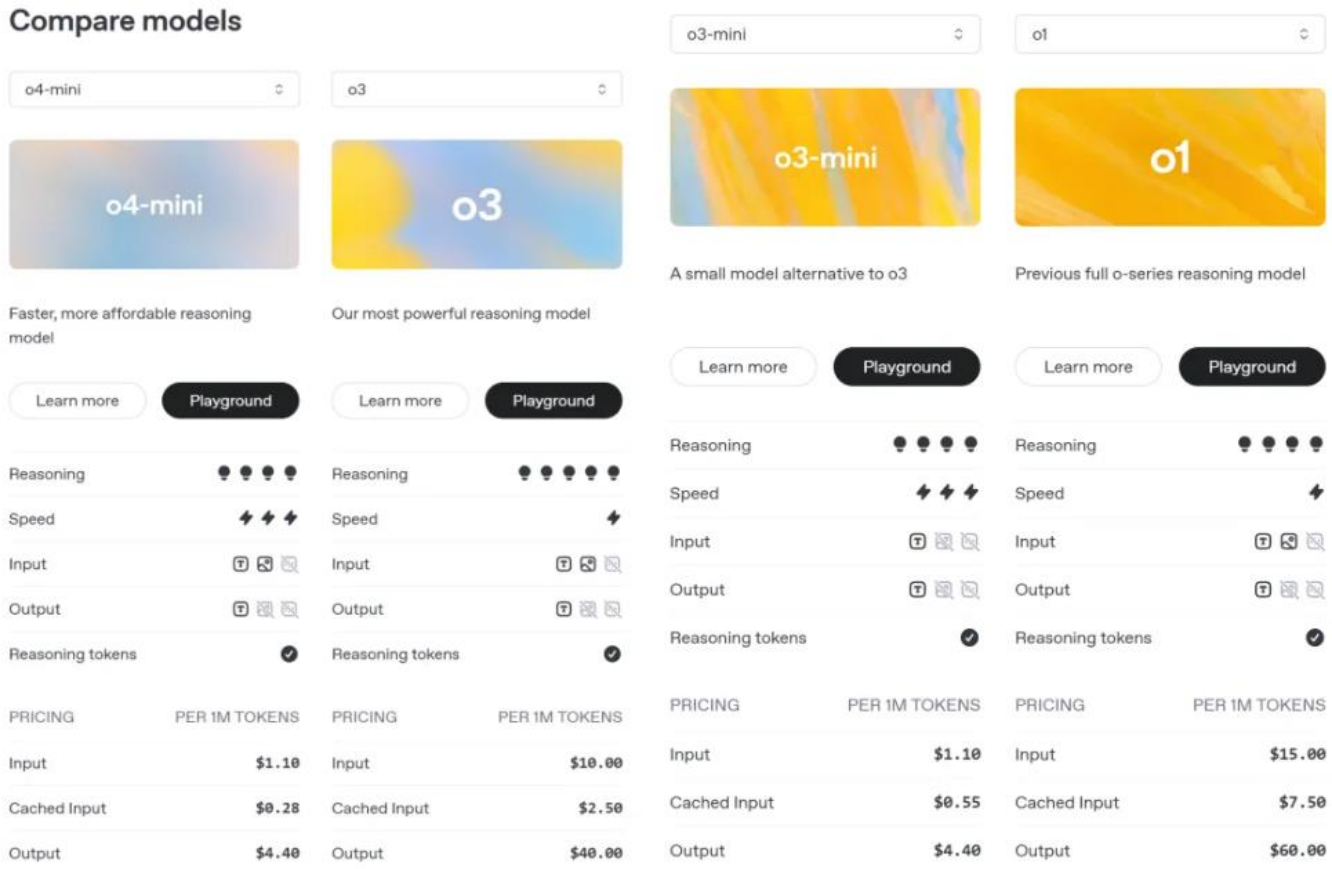
where the sum is taken over all ordered pairs (A, B) such that A and B are subsets of $\{1, 2, 3, \cdots, n\}$ with $|A| = |B|$. For example, $S_2 = 4$ because the sum is taken over the pairs of subsets

$$(A, B) \in \{(\emptyset, \emptyset), (\{1\}, \{1\}), (\{1\}, \{2\}), (\{2\}, \{1\}), (\{2\}, \{2\}), (\{1, 2\}, \{1, 2\})\}$$

giving $S_2 = 0 + 1 + 0 + 0 + 1 + 2 = 4$. Let $\frac{S_{2022}}{S_{2021}} = \frac{p}{q}$, where p and q are relatively prime positive integers. Find the remainder when $p + q$ is divided by 1000.

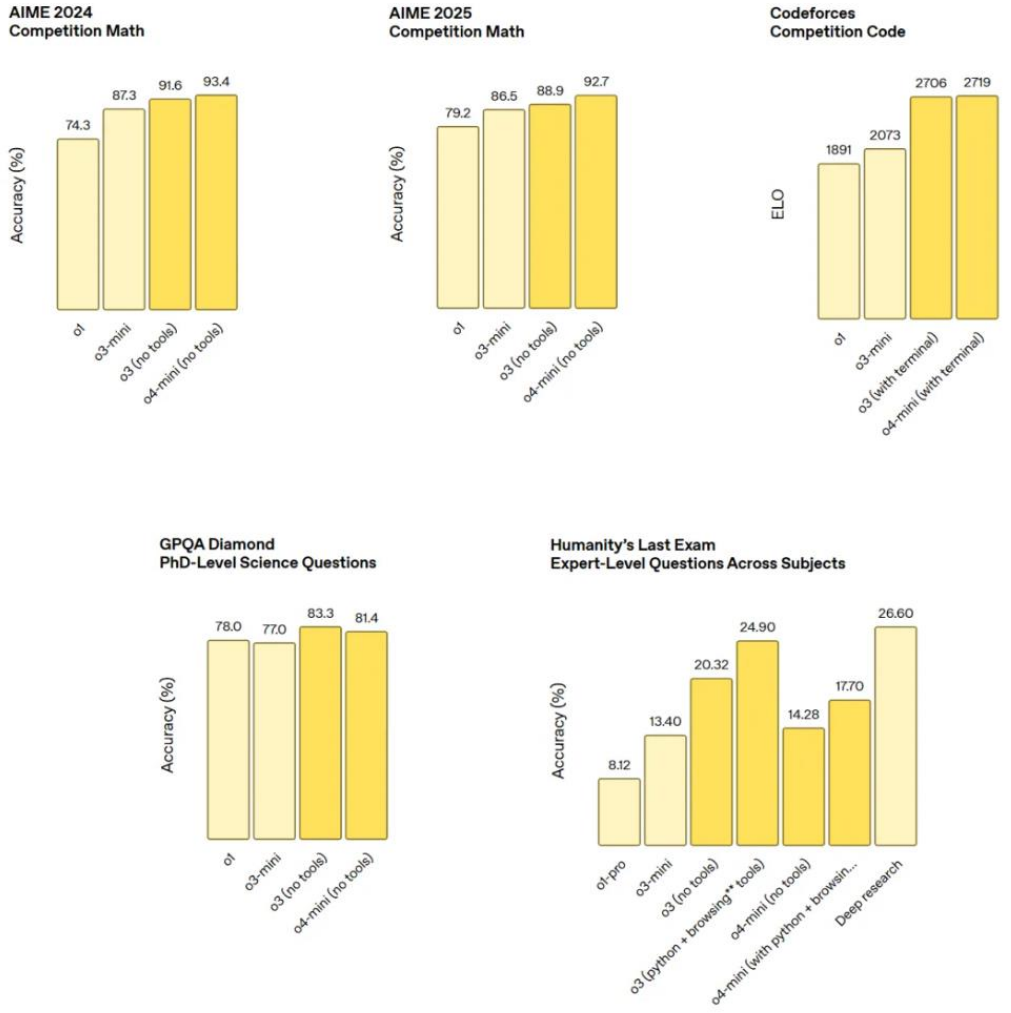
[OpenAI, 2024]

OpenAI o3 and o4-mini

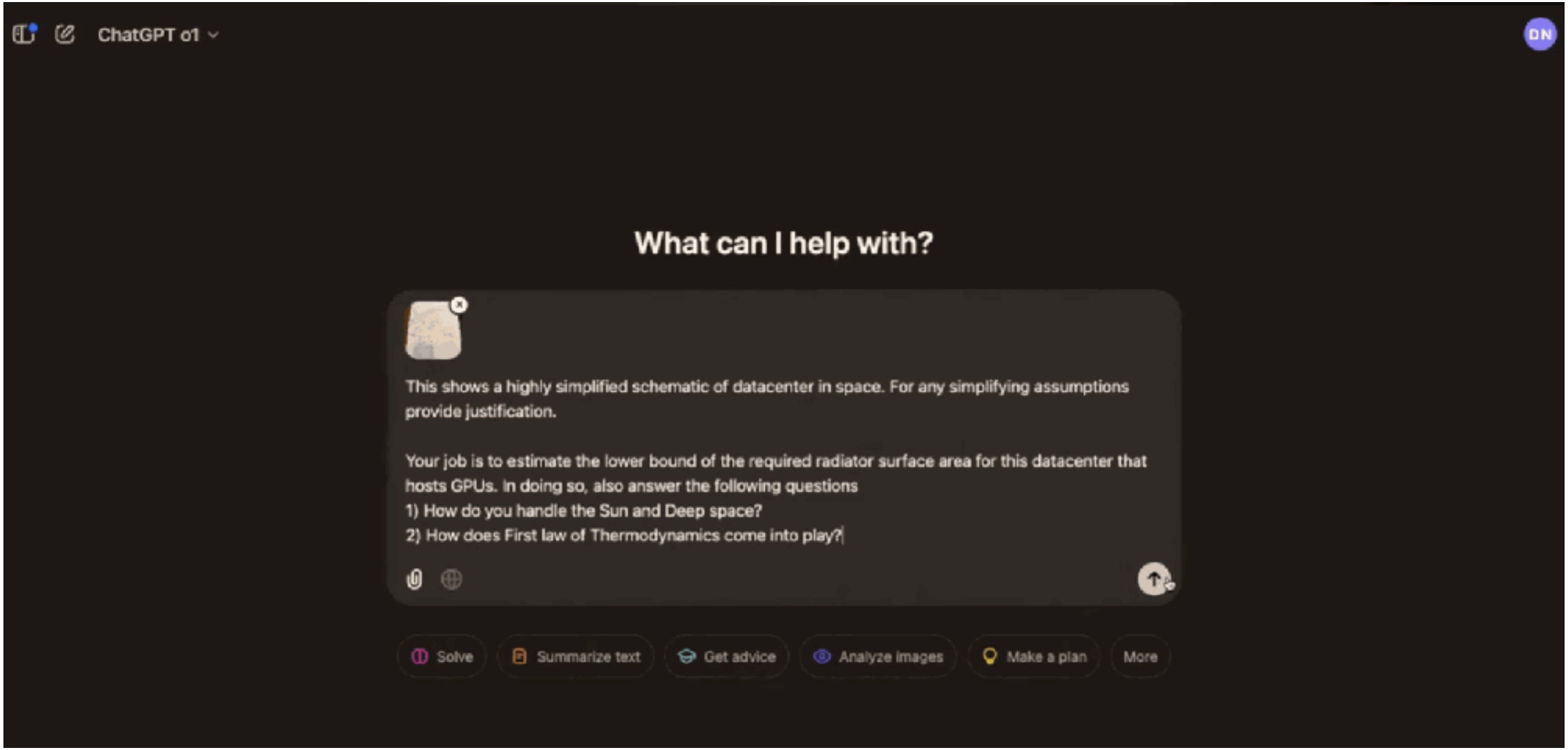


o4-mini and o3

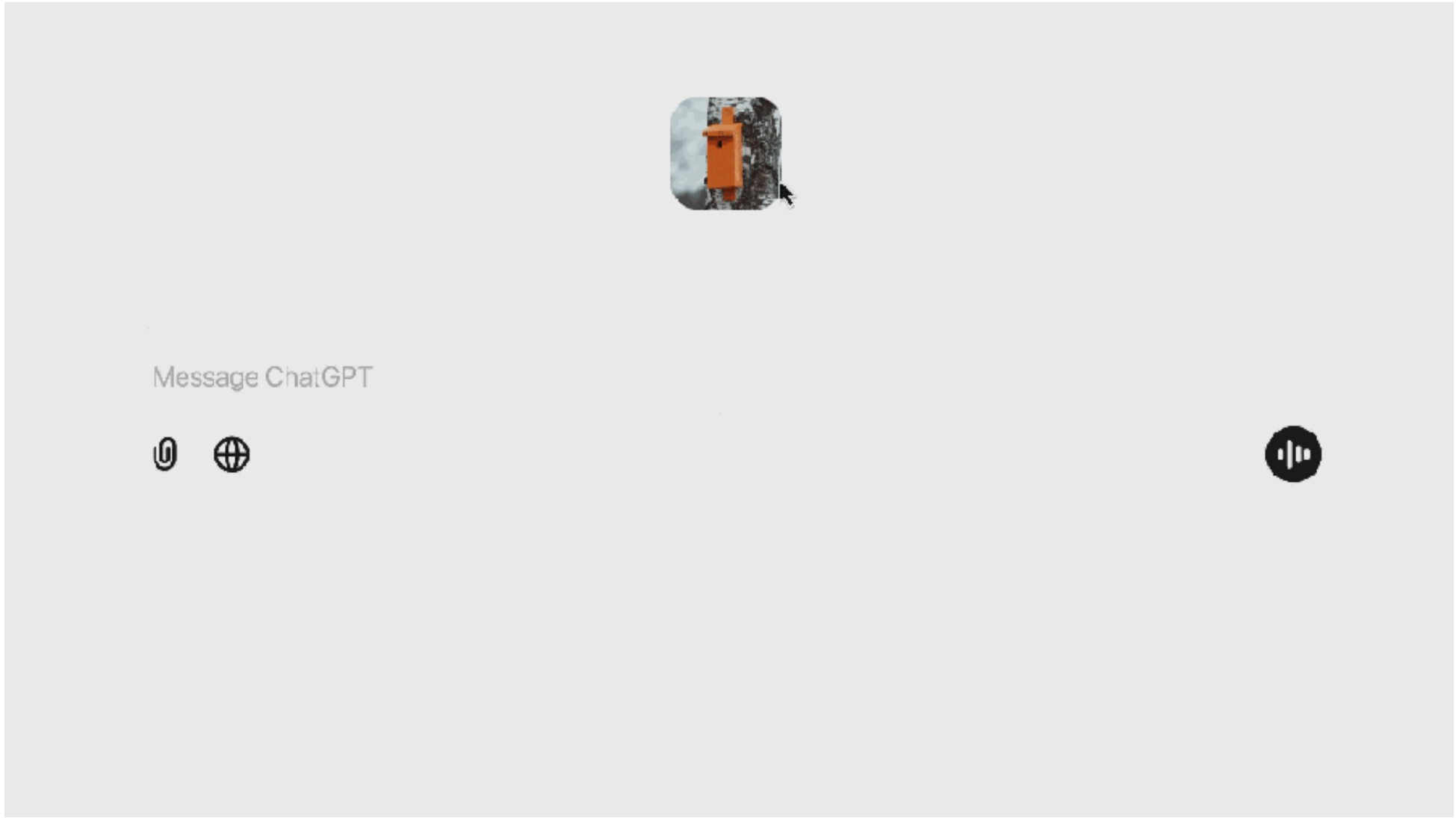
o3 and o4-mini on Reasoning Benchmarks



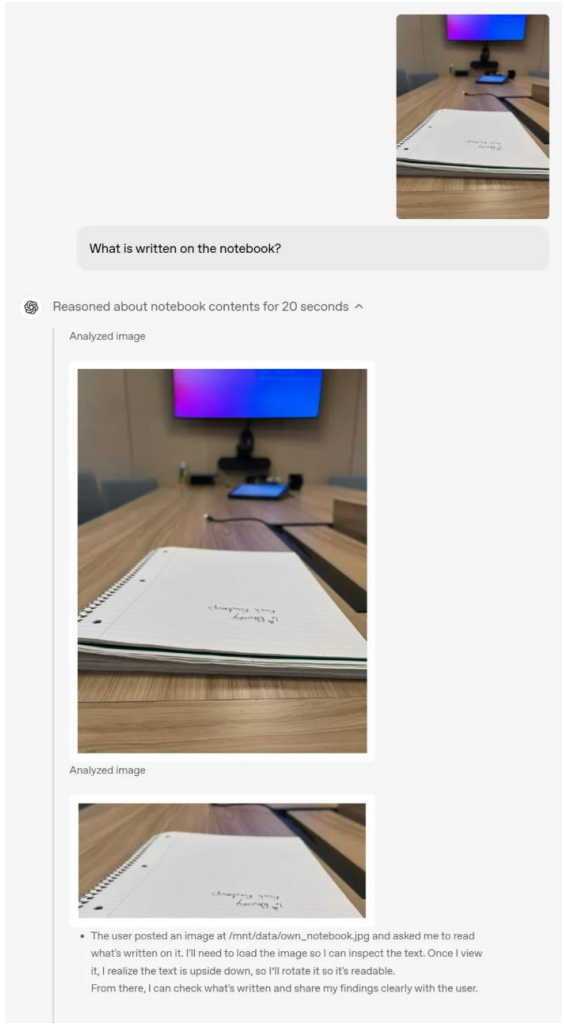
Demo: o3 Thinking with Images (1/2)

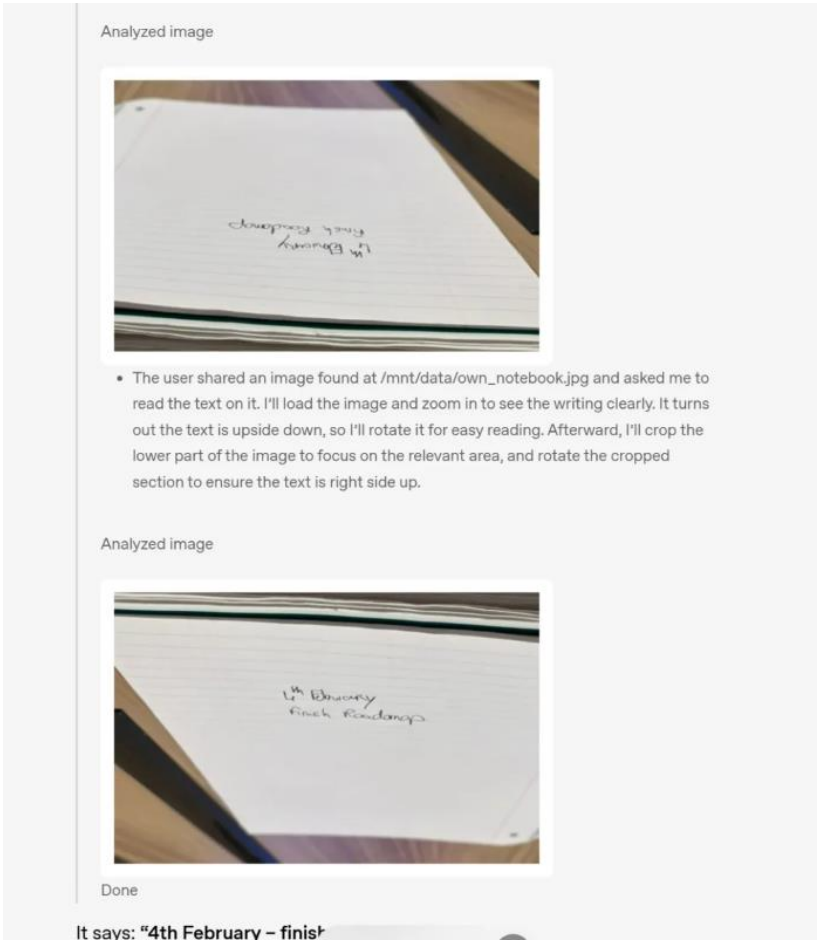


Demo: o3 Thinking with Images (2/2)



Thinking with Images: Zooming and Rotating





o3 can zoom and rotate images while reasoning

https://openai.com/index/introducing-o3-and-o4-mini/

4.3 How Do We Measure Reasoning?

Reasoning claims need hard, checkable and unsaturated benchmarks — math, code, GPQA and HLE.

Why Benchmark on Math, Coding and GPQA?

- **Gold-standard answers that can be checked automatically**
- Performance is far from saturated.

GPQA: A Graduate-Level Google-Proof Q&A Benchmark

GPQA is a **multiple-choice, Q&A dataset** of very hard questions written and validated by experts in biology, physics, and chemistry. When attempting questions out of their own domain (e.g., a physicist answers a chemistry question), **these experts get only 34% accuracy, despite spending >30m with full access to Google**.

Chemistry (general)

A reaction of a liquid organic compound, which molecules consist of carbon and hydrogen atoms, is performed at 80 centigrade and 20 bar for 24 hours. In the proton nuclear magnetic resonance spectrum, the signals with the highest chemical shift of the reactant are replaced by a signal of the product that is observed about three to four units downfield. Compounds from which position in the periodic system of the elements, which are also used in the corresponding large-scale industrial process, have been mostly likely initially added in small amounts?
A) A metal compound from the fifth period.
B) A metal compound from the fifth period and a non-metal compound from the third period.
C) A metal compound from the fourth period.
D) A metal compound from the fourth period and a non-metal compound from the second period.

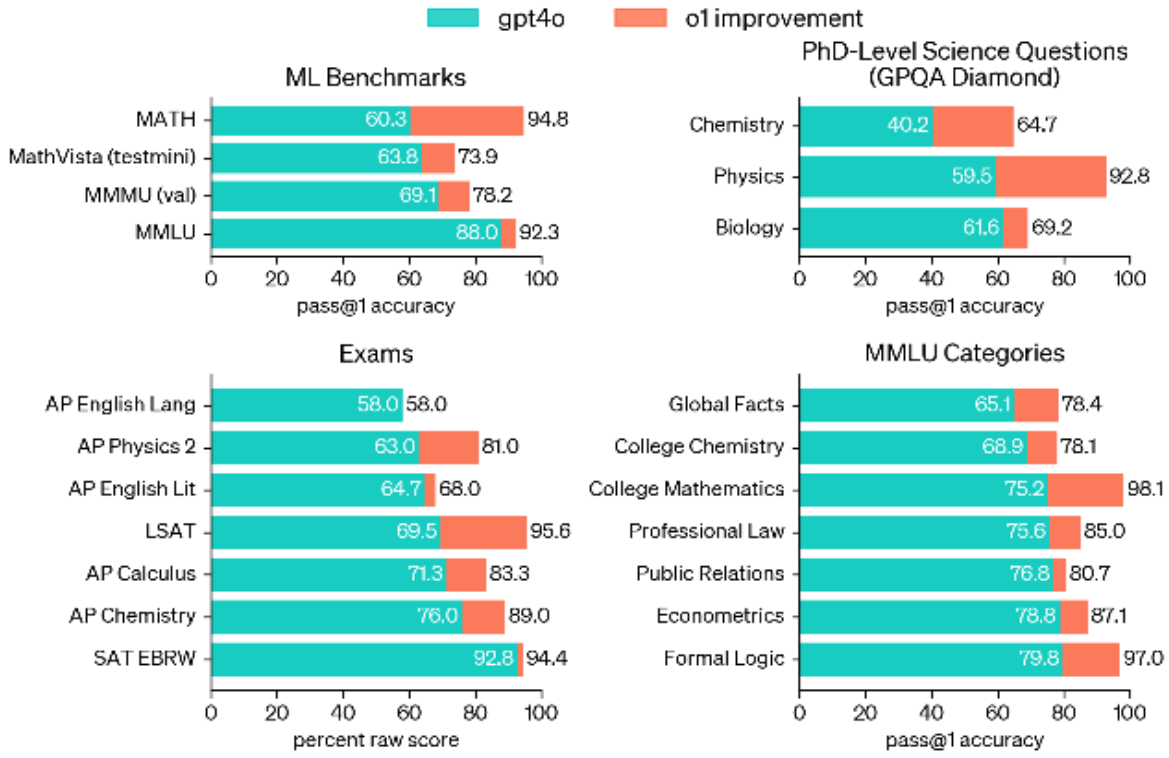
Organic Chemistry

Methylcyclopentadiene (which exists as a fluxional mixture of isomers) was allowed to react with methyl isoamyl ketone and a catalytic amount of pyrrolidine. A bright yellow, cross-conjugated polyalkenyl hydrocarbon product formed (as a mixture of isomers), with water as a side product. These products are derivatives of fulvene. This product was then allowed to react with ethyl acrylate in a 1:1 ratio. Upon completion of the reaction, the bright yellow color had disappeared. How many chemically distinct isomers make up the final product (not counting stereoisomers)?
A) 2
B) 16
C) 8
D) 4

https://huggingface.co/datasets/Idavidrein/gpqa

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, Samuel R. Bowman. GPQA: A Graduate-Level Google-Proof Q&A Benchmark. https://arxiv.org/abs/2311.12022

Multimodal Reasoning?



MathVista and MMMU are multimodal benchmarks, yet o1's chain of thought is text-only (no images inside the CoT).

Could a model “draw auxiliary lines” while it thinks?

Humanity's Last Exam (HLE)

- HLE is a global collaborative effort, with questions from **nearly 1000 subject expert contributors affiliated with over 500 institutions across 50 countries** – comprised mostly of professors, researchers, and graduate degree holders

- **Prize Pool**. To attract high-quality submissions, we establish a $500,000 USD prize pool
  - with prizes of $5,000 USD for each of the top **50 questions**
  - $500 USD for each of the **next 500 questions**, as determined by organizers.
  - This incentive structure, combined with the opportunity for paper co-authorship for anyone with an accepted question in HLE, draws participation from qualified experts, particularly those with advanced degrees or significant technical experience in their fields.

HLE: Example Questions (1/3)

Classics

Question:



Here is a representation of a Roman inscription, orginally found on a tombstone. Provide a translation for the Palmyrene script.
A transliteration of the text is provided: RGYNᵓ BT HRY BR ᶜTᵓ HBL

Henry T
Merton College, Oxford

Ecology

Question:

Hummingbirds within Apodiformes uniquely have a bilaterally paired oval bone, a sesamoid embedded in the caudolateral portion of the expanded, cruciate aponeurosis of insertion of m. depressor caudae. How many paired tendons are supported by this sesamoid bone? Answer with a number.

Edward V
Massachusetts Institute of Technology

https://static.scale.com/uploads/654197dc94d34f66c0f5184e/Publication%20Ready%20Humanity's%20Last%20Exam.pdf

HLE: Example Questions (2/3)

Mathematics

Question:
The set of natural transformations between two functors $F, G : C \to D$ can be expressed as the end

$$Nat(F, G) \cong \int_A Hom_D(F(A), G(A)).$$

Define set of natural cotransformations from F to G to be the coend

$$CoNat(F, G) \cong \int^A Hom_D(F(A), G(A)).$$

Let:
- $F = B_\bullet(\Sigma_4)_{*/}$ be the under ∞-category of the nerve of the delooping of the symmetric group Σ_4 on 4 letters under the unique 0-simplex $*$ of $B_\bullet\Sigma_4$.
- $G = B_\bullet(\Sigma_7)_{*/}$ be the under ∞-category nerve of the delooping of the symmetric group Σ_7 on 7 letters under the unique 0-simplex $*$ of $B_\bullet\Sigma_7$.

How many natural cotransformations are there between F and G?

Emily S
University of São Paulo

Computer Science

Question:
Let G be a graph. An edge-indicator of G is a function $a : 0, 1 \to V(G)$ such that $a(0), a(1) \in E(G)$.

Consider the following Markov Chain $M = M(G)$:
The statespace of M is the set of all edge-indicators of G, and the transitions are defined as follows:

Assume $M_t = a$.
1. pick $b \in 0, 1$ u.a.r.
2. pick $v \in N(a(1-b))$ u.a.r. (here $N(v)$ denotes the open neighbourhood of v)
3. set $a'(b) = v$ and $a'(1-b) = a(1-b)$
4. Set $M_{t+1} = a'$

We call a class of graphs $\mathcal{G}$ well-behaved if, for each $G \in \mathcal{G}$ the Markov chain $M(G)$ converges to a unique stationary distribution, and the unique stationary distribution is the uniform distribution.

Which of the following graph classes is well-behaved?

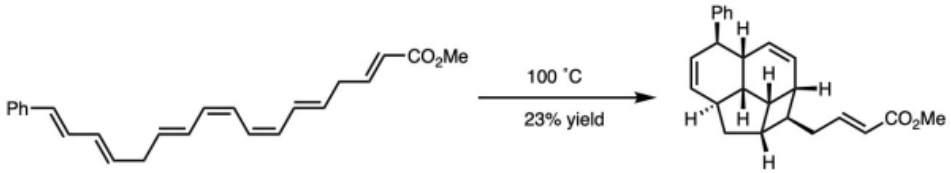
Answer Choices:
A. The class of all non-bipartite regular graphs
B. The class of all connected cubic graphs
C. The class of all connected graphs
D. The class of all connected non-bipartite graphs
E. The class of all connected bipartite graphs.

Marc R
Queen Mary University of London

HLE: Example Questions (3/3)

Chemistry

Question:



endiandric acid B methyl ester

The reaction shown is a thermal pericyclic cascade that converts the starting heptaene into endiandric acid B methyl ester. The cascade involves three steps: two electrocyclizations followed by a cycloaddition. What types of electrocyclizations are involved in step 1 and step 2, and what type of cycloaddition is involved in step 3?

Provide your answer for the electrocyclizations in the form of [nπ]-con or [nπ]-dis (where n is the number of π electrons involved, and whether it is conrotatory or disrotatory), and your answer for the cycloaddition in the form of [m+n] (where m and n are the number of atoms on each component).

Noah B
Stanford University

Linguistics

Question:

I am providing the standardized Biblical Hebrew source text from the Biblia Hebraica Stuttgartensia (Psalms 104:7). Your task is to distinguish between closed and open syllables. Please identify and list all closed syllables (ending in a consonant sound) based on the latest research on the Tiberian pronunciation tradition of Biblical Hebrew by scholars such as Geoffrey Khan, Aaron D. Hornkohl, Kim Phillips, and Benjamin Suchard. Medieval sources, such as the Karaite transcription manuscripts, have enabled modern researchers to better understand specific aspects of Biblical Hebrew pronunciation in the Tiberian tradition, including the qualities and functions of the shewa and which letters were pronounced as consonants at the ends of syllables.

מִן־גַּעֲרָתְךָ֣ יְנוּס֑וּן מִן־ק֥וֹל רַ֝עַמְךָ֗ יֵחָפֵזֽוּן (Psalms 104:7) ?

Lina B
University of Cambridge

Will HLE Be the Last Exam?

Humanity's Last Exam is unlikely to be the last one —

we will soon need a “Humanity's Last Last Exam”.

5 Paths to Reasoning

How did the field get there? Verifiers, distillation, pure RL — and finally rewards that check whole proofs.

In this part

5.1 Inference-Time Verification

5.2 Distilling Long Chain-of-Thought

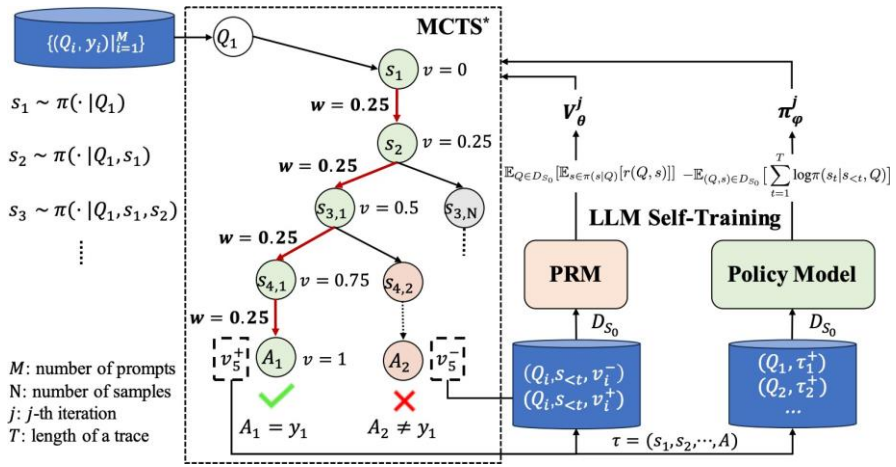
5.3 DeepSeek-R1

5.4 Reproducing R1 in the Open: SimpleRL

5.5 Self-Verification: DeepSeekMath-V2

Efforts toward o1-Style Models

Many researchers are exploring possible paths towards learning o1-style models…



Tree Search

More complex during training and testing

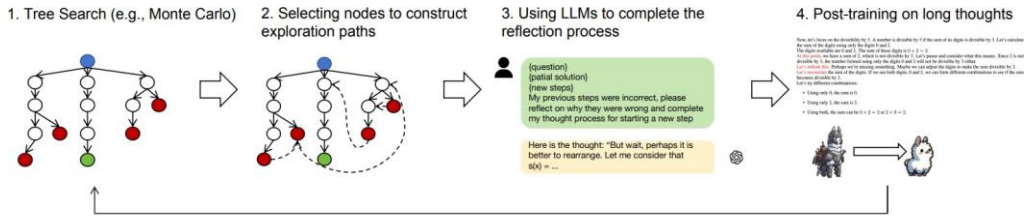


Figure 2: The framework of journey learning.

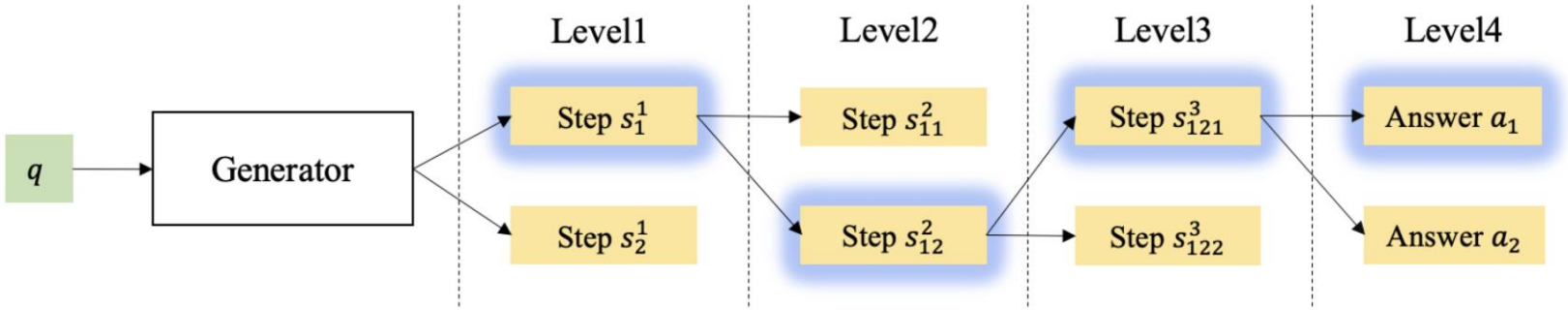
Distillation

Hard to surpass teacher model

Zhang, Dan, et al. "Rest-mcts*: Llm self-training via process reward guided tree search." *arXiv preprint arXiv:2406.03816* (2024). Qin, Yiwei, et al. "O1 Replication Journey: A Strategic Progress Report--Part 1." *arXiv preprint arXiv:2410.18982* (2024).

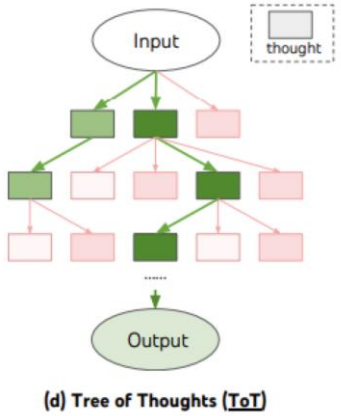
Inference-Time Verification

- A recent line of work: improve multi-step reasoning at inference time
- Real-time alignment: does not modify the model parameters
- Maximize the expected cumulative reward during decoding
- Action selection problem in RL
  - - a tree-structured search process



[4] Yu, Fei, Anningzhe Gao, and Benyou Wang. "OVM, Outcome-supervised Value Models for Planning in Mathematical Reasoning." Findings of the Association for Computational Linguistics: NAACL 2024. 2024.

Inference-Time Verification: Approaches



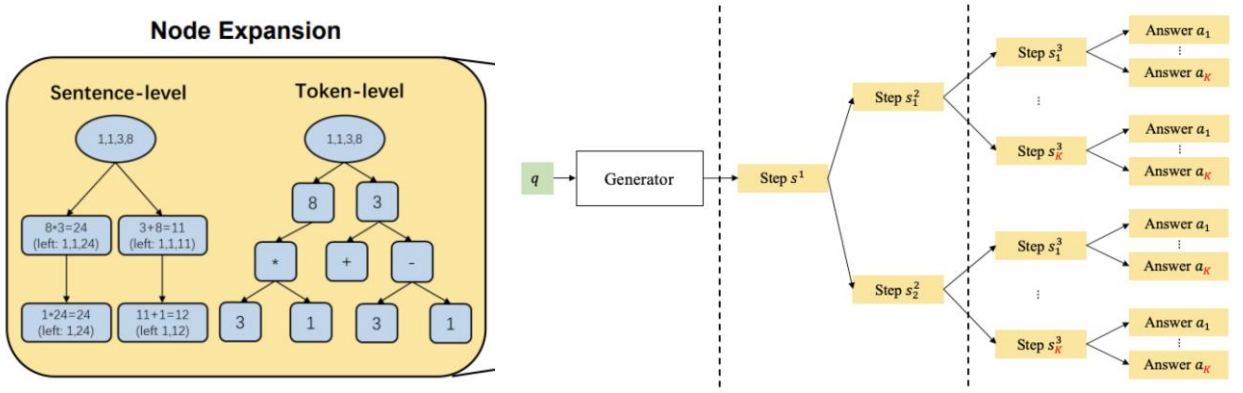
ToT (with domain-specific heuristics)

domain-agnostic guided decoding algorithm

TS-LLM (UCL, SJTU, CMU)

CD (DeepMind, OpenAI)

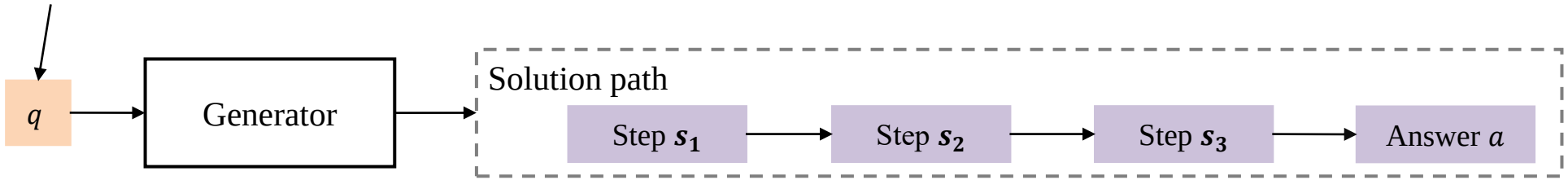
OVM (CUHKSZ, SRIBD)



[4] Yu, Fei, Anningzhe Gao, and Benyou Wang. "OVM, Outcome-supervised Value Models for Planning in Mathematical Reasoning." Findings of the Association for Computational Linguistics: NAACL 2024. 2024.
[19] Yao, Shunyu, et al. "Tree of thoughts: Deliberate problem solving with large language models." Advances in Neural Information Processing Systems 36 (2024).
[20] Mudgal, Sidharth, et al. "Controlled decoding from language models." arXiv preprint arXiv:2310.17022 (2023).
[21] Feng, Xidong, et al. "Alphazero-like tree-search can guide large language model decoding and training." arXiv preprint arXiv:2309.17179 (2023).

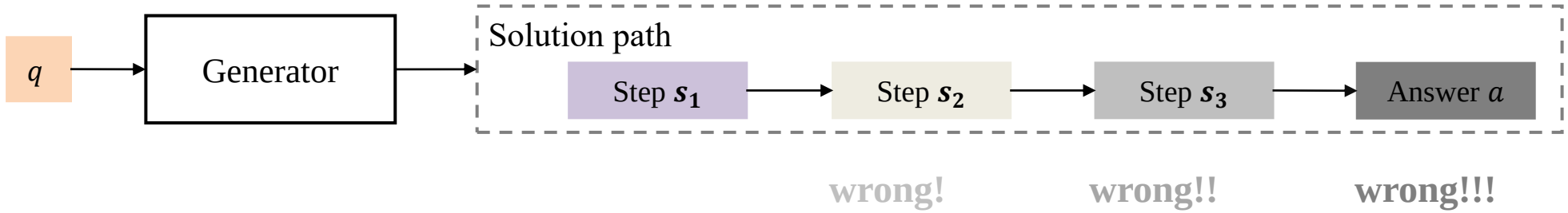
Multi-Step Reasoning in Math

Problem: Beth bakes 4, 2 dozen batches of cookies in a week. If these cookies are shared amongst 16 people equally, how many cookies does each person consume?



Solution: Beth bakes 4 2 dozen batches of cookies for a total of 4*2 = <<4*2=8>>8 dozen cookies s_1
There are 12 cookies in a dozen and she makes 8 dozen cookies for a total of 12*8 = <<12*8=96>>96 cookies s_2
She splits the 96 cookies equally amongst 16 people so they each eat 96/16 = <<96/16=6>>6 cookies s_3
Final Answer: 6 a

Challenge: Cascading Errors in Multi-Step Reasoning



Error propagation!

A Straightforward Way: Repeated Sampling

The fundamental idea: model can respond correctly if we allow it to try more

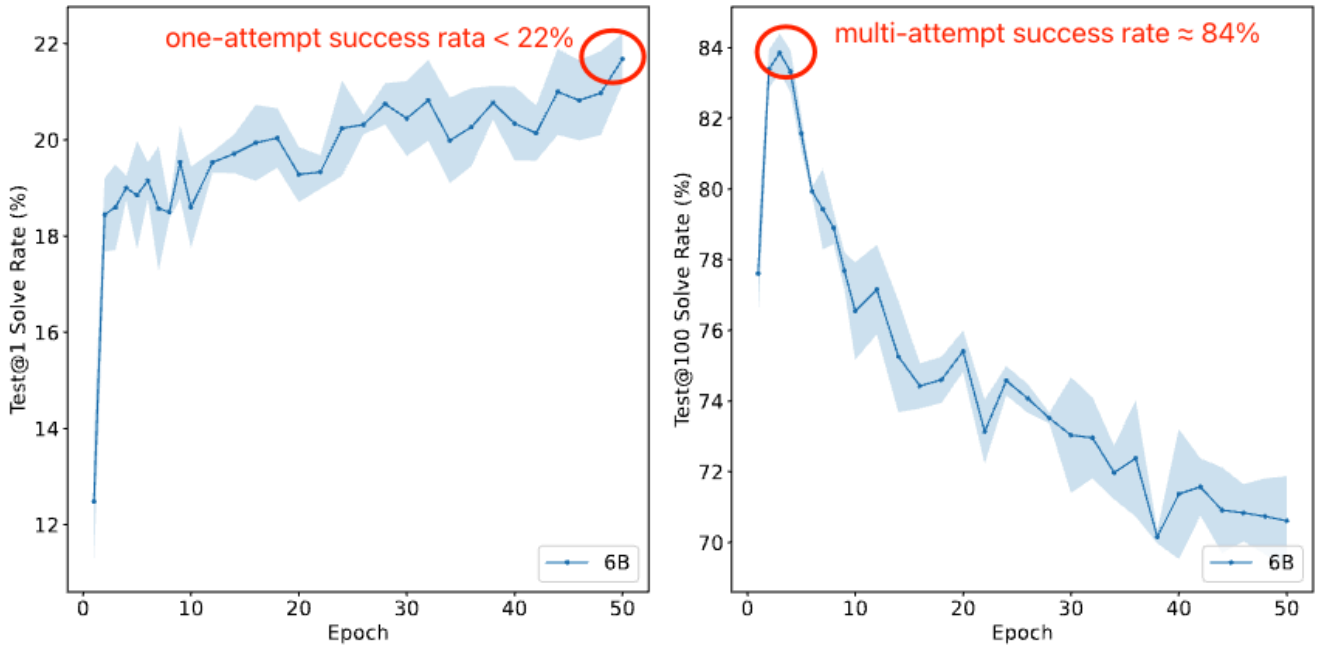
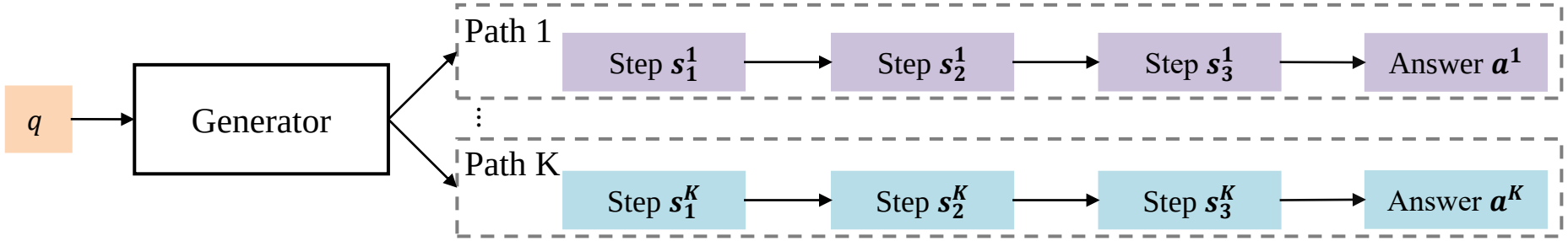


Figure 3: Test solve rate after finetuning a 6B model on the full GSM8K training set, when the model is allowed to make 1 guess (left) or 100 guesses (right).

[1] Cobbe, Karl, et al. "Training verifiers to solve math word problems." arXiv preprint arXiv:2110.14168 (2021).

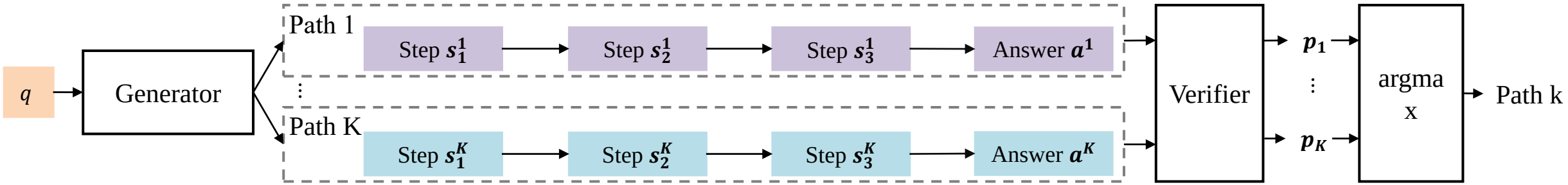
Related Work: Verification with Multiple Candidates

1. Sample multiple candidate paths



2. Select the best one among them

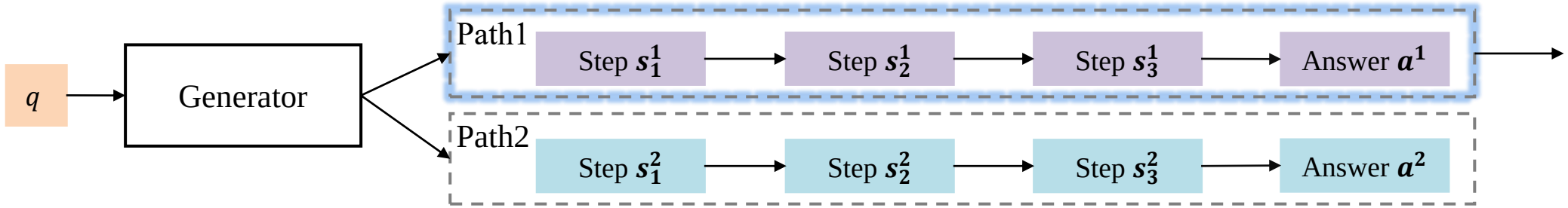
best: with the highest probability of being correct according to the verifier (a classifier)



[1] Cobbe, Karl, et al. "Training verifiers to solve math word problems." arXiv preprint arXiv:2110.14168 (2021).

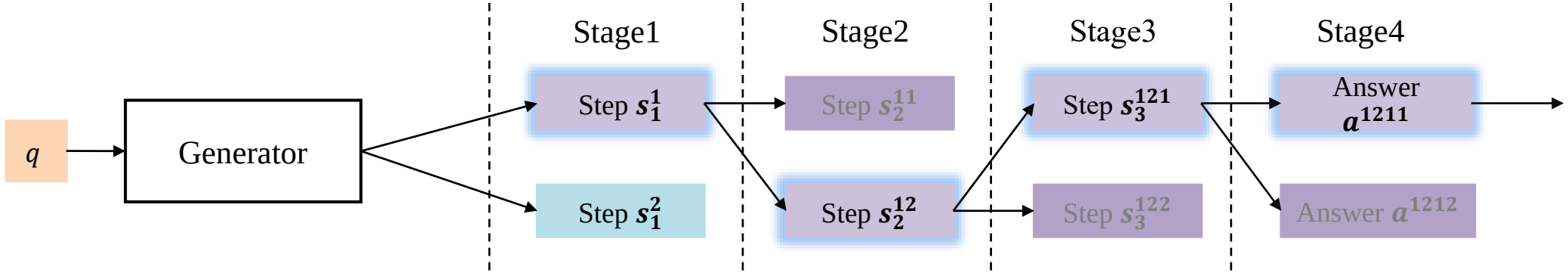
Related Work: Reward-Guided Decoding

- **Complete path verification paradigm** (sample-then-filter)



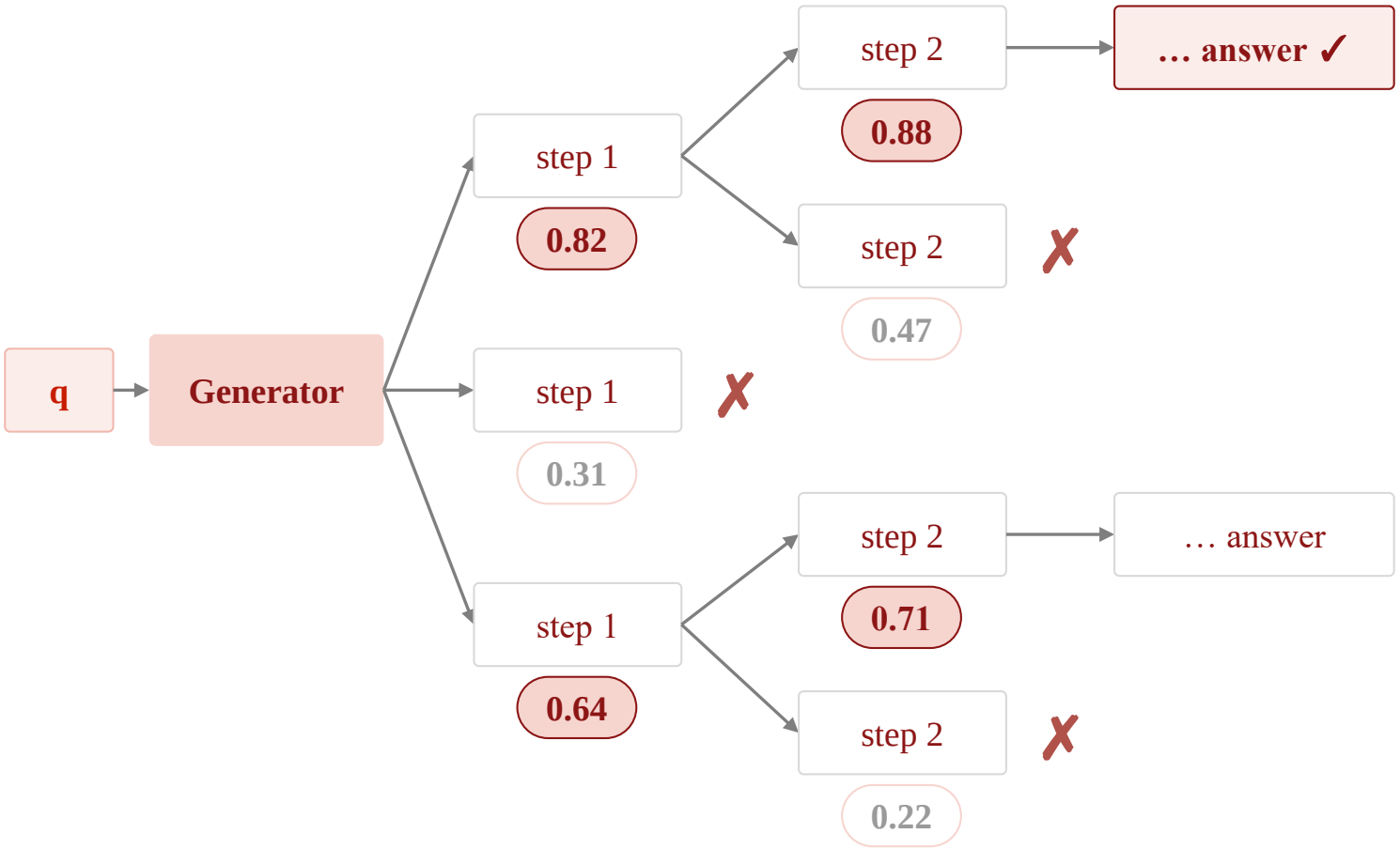
errors are still propagating!

- **Guided decoding paradigm** (verifier-guided generation)
  - Guide the model's decoding and filter in the middle



[2] Xie, Yuxi, et al. "Self-evaluation guided beam search for reasoning." Advances in Neural Information Processing Systems 36 (2024).
[3] Khalifa, Muhammad, et al. "Discriminator-Guided Multi-step Reasoning with Language Models." arXiv preprint arXiv:2305.14934 (2023).

OVM: Value-Guided Decoding, Step by Step



Click to animate: expand → score → prune → expand → score → prune → answer

At every step

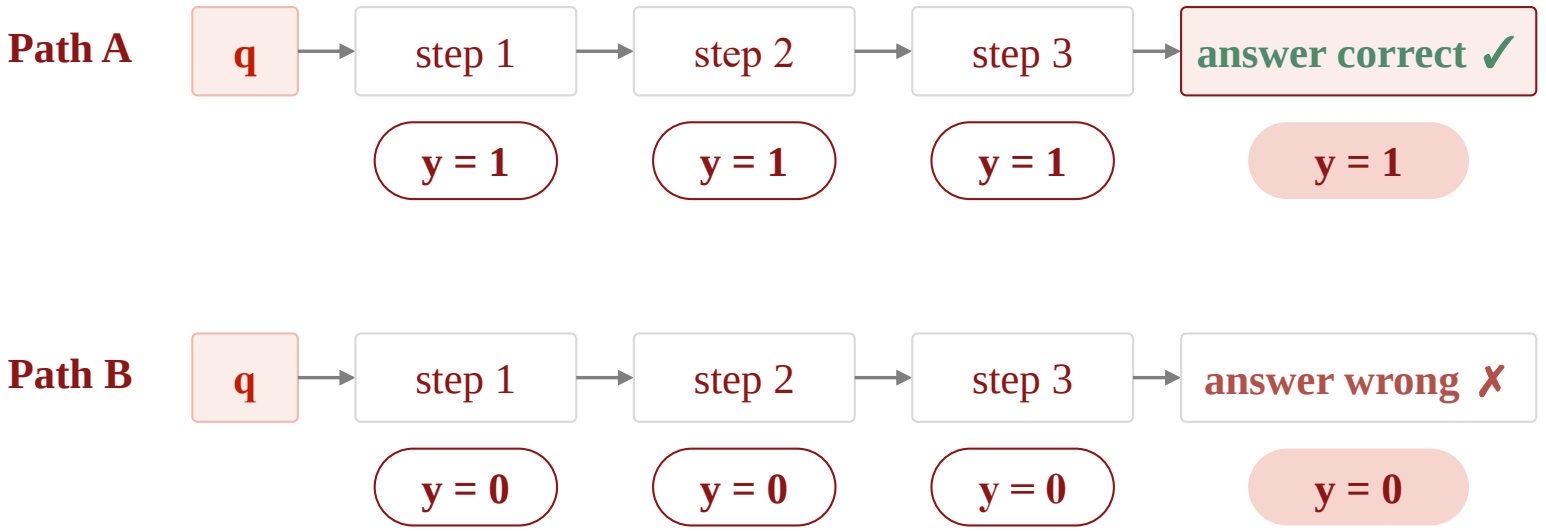
- Sample K candidate next steps (here K = 3).
- Score each partial path with the value model:

  $V(q, s_{1..t}) \approx P(\text{answer correct} \mid q, s_{1..t})$
- Keep the top-b paths (here b = 2), prune the rest.
- Repeat until the answer; return the best path.

Value = "how promising is this partial path?", not "is this step correct?"

Yu, Gao & Wang. OVM: Outcome-supervised Value Models for Planning in Mathematical Reasoning. Findings of NAACL 2024.

Training OVM: Outcome Labels Are Enough



$\mathcal{L}(\psi) = \Sigma_t \, (\, V_\psi(q, s_{1..t}) - y \,)^2, \quad y \in \{0, 1\}$: the final outcome

Every prefix inherits the label of the path it belongs to.

Why this learns a value

- The same prefix appears in many sampled paths, some correct, some not.
- Minimizing squared error makes V(prefix) the average label of its continuations:

  $V(q, s_{1..t}) \rightarrow P(\text{correct} \mid q, s_{1..t})$
- No step-level annotation: only final answers, which are cheap to check.

Cheap labels, useful signal: outcome supervision turns a pile of sampled solutions into a value function for planning.

Yu, Gao & Wang. OVM (Findings of NAACL 2024) · cf. Cobbe et al. Training Verifiers to Solve Math Word Problems (2021).

Process vs. Outcome Supervision: What Should a Label Mean?

A wrong path whose first two steps are locally correct



Process supervision (PRM)
"Is this step correct?"

1 — 1 — 0

Outcome supervision (OVM)
"Will this path reach the right answer?"

0.3 — 0.1 — 0

OVM values = expected success over continuations (illustrative numbers).

Foresee the future

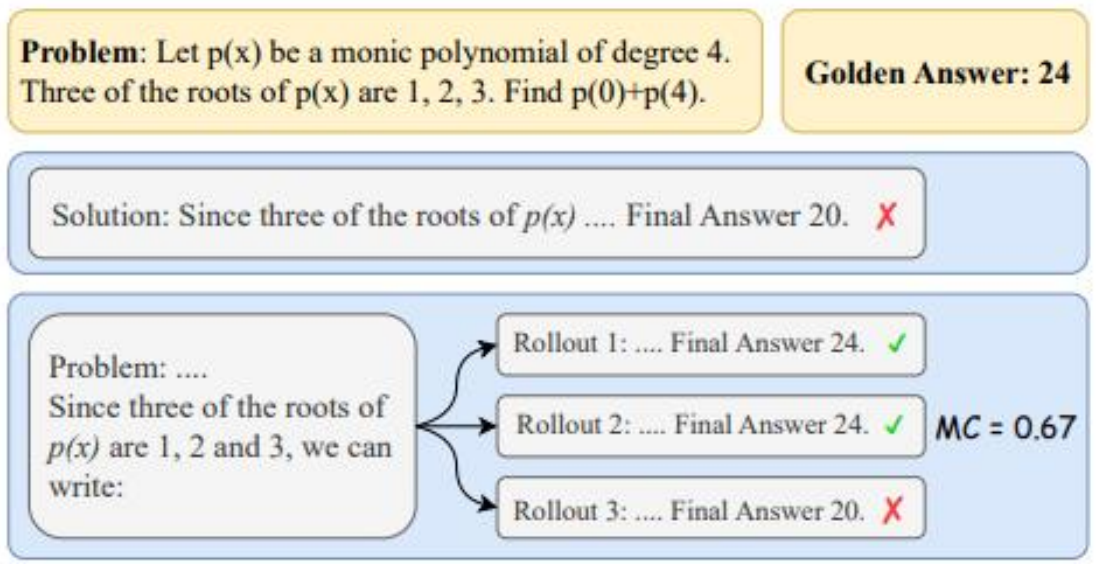
A PRM rates steps 1–2 as perfect, although the path is already heading to a dead end.

For planning we need the value of a partial path — exactly what outcome labels train.

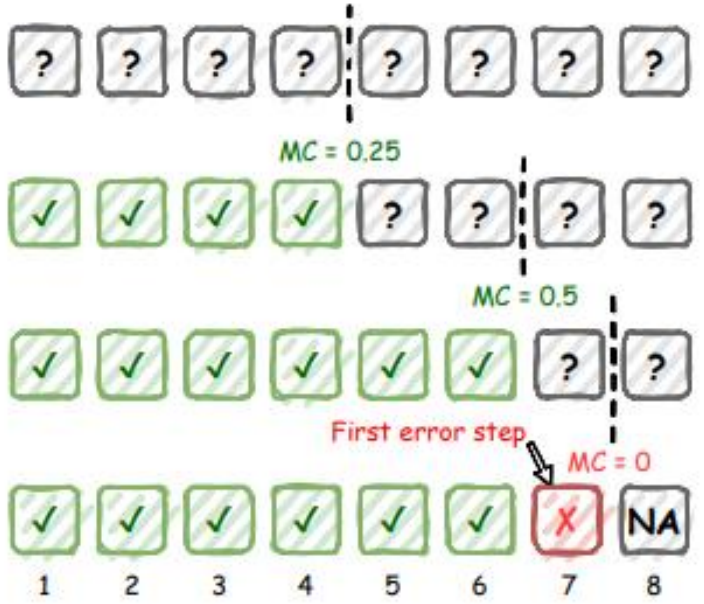
OVM-7B reached state of the art among LLMs up to 13B on GSM8K — without GPT-4 or code execution — using only outcome labels.

Yu, Gao & Wang. OVM (Findings of NAACL 2024) · Lightman et al. Let's Verify Step by Step (2023) · Uesato et al. (2022).

OmegaPRM: Monte Carlo Estimation of Step Values



(a) Monte Carlo estimation of a prefix solution.



(b) Error locating using binary search.

Drawbacks of Learned Verifiers

Learnable verification is noisy or biased!

Solution: rule-based rewards / verification?

Verification at Scale: Automatic Theorem Proving

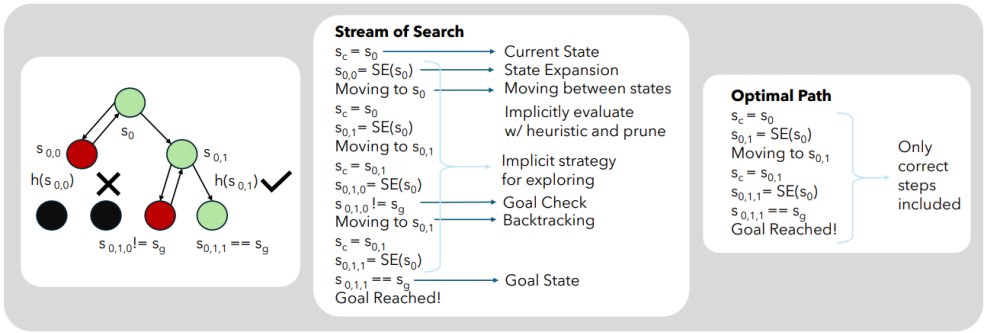


Automatic theorem proving

5.2 Distilling Long Chain-of-Thought

If a strong model can already reason, can a smaller model simply imitate its long chains of thought?

Synthesizing Long CoT: Stream of Search and Journey Learning



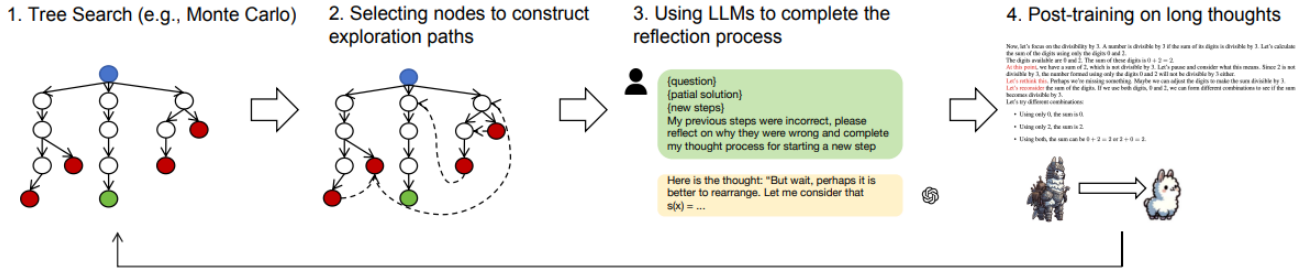


Figure 2: The framework of journey learning.

https://arxiv.org/pdf/2404.03683
https://arxiv.org/pdf/2411.16489

Drawbacks of Long-CoT SFT (Distillation)

SFT does not generalize! It cannot learn new patterns; it learns from a teacher that it cannot outperform.

Solution: try RL

5.3 DeepSeek-R1

R1 made the recipe public: pure RL with rule-based rewards produces long CoT and self-reflection.

What Is Really New in R1?

RL training is not a new thing

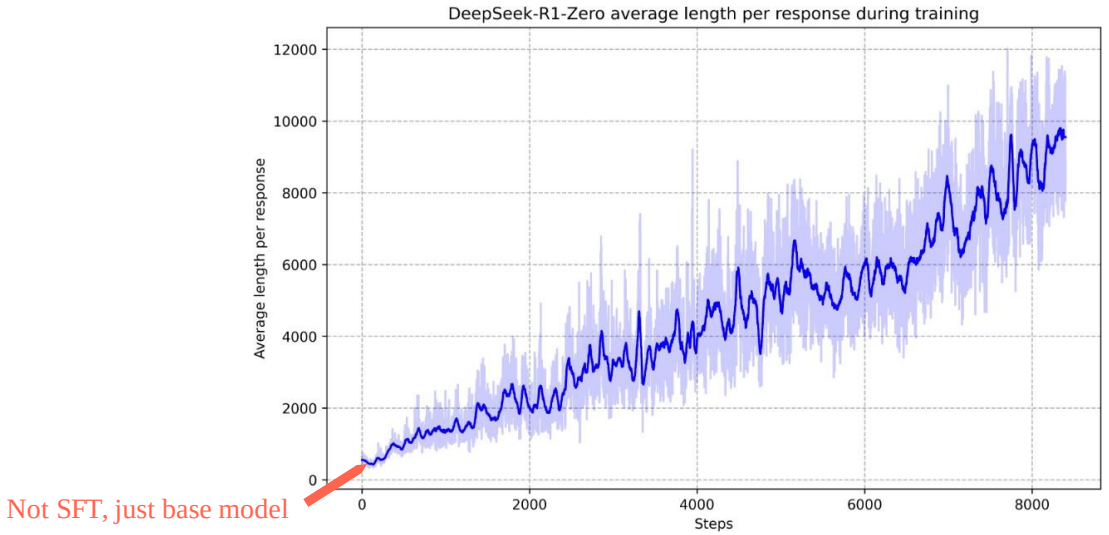
But the emergence of

long CoT plus self-reflection is new

(a.k.a. the ‘aha moment’)

Long CoT Patterns Emerging in RL Training

DeepSeek-R1 and Kimi-k1.5 choose an extremely simple recipe — Reinforcement Learning



Model naturally develops long CoT during RL

Guo, Daya, et al. "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning." *arXiv preprint arXiv:2501.12948* (2025).

Self-Reflection Emerging in RL

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 $\cdots$
$\left(\sqrt{a - \sqrt{a + x}}\right)^2 = x^2 \implies a - \sqrt{a + x} = x^2$.
Rearrange to isolate the inner square root term:
$(a - x^2)^2 = a + x \implies a^2 - 2ax^2 + (x^2)^2 = a + x \implies x^4 - 2ax^2 - x + (a^2 - a) = 0$
$\ldots$
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: $\cdots$
$\ldots$

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

Guo, Daya, et al. "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning." *arXiv preprint arXiv:2501.12948* (2025).

DeepSeek-R1: Two Main Parts

DeepSeek's paper includes two main parts:

- DeepSeek-R1-Zero: pure RL on math and code data (no SFT).
- DeepSeek-R1: SFT + RL. A small amount of cold-start SFT data (partly generated by R1-Zero).
  - Accelerates RL training.
  - More user-friendly: adds non-reasoning data (writing, QA, …).

Guo, Daya, et al. "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning." *arXiv preprint arXiv:2501.12948* (2025).

DeepSeek-R1: The Full Training Pipeline

R1-Zero

V3-Base
671B MoE

→

GRPO + rule rewards
accuracy + format

→

R1-Zero
no SFT at all

AIME: 15.6% → 71.0%
but poor readability

R1
4 stages

① Cold-start SFT
a few thousand readable long-CoT examples

→

② Reasoning RL
GRPO; accuracy, format and language rewards

→

③ Rejection sampling
~600K reasoning + ~200K general SFT data

→

④ RL for everything
rules for reasoning, reward models for helpfulness

RLHF is back!

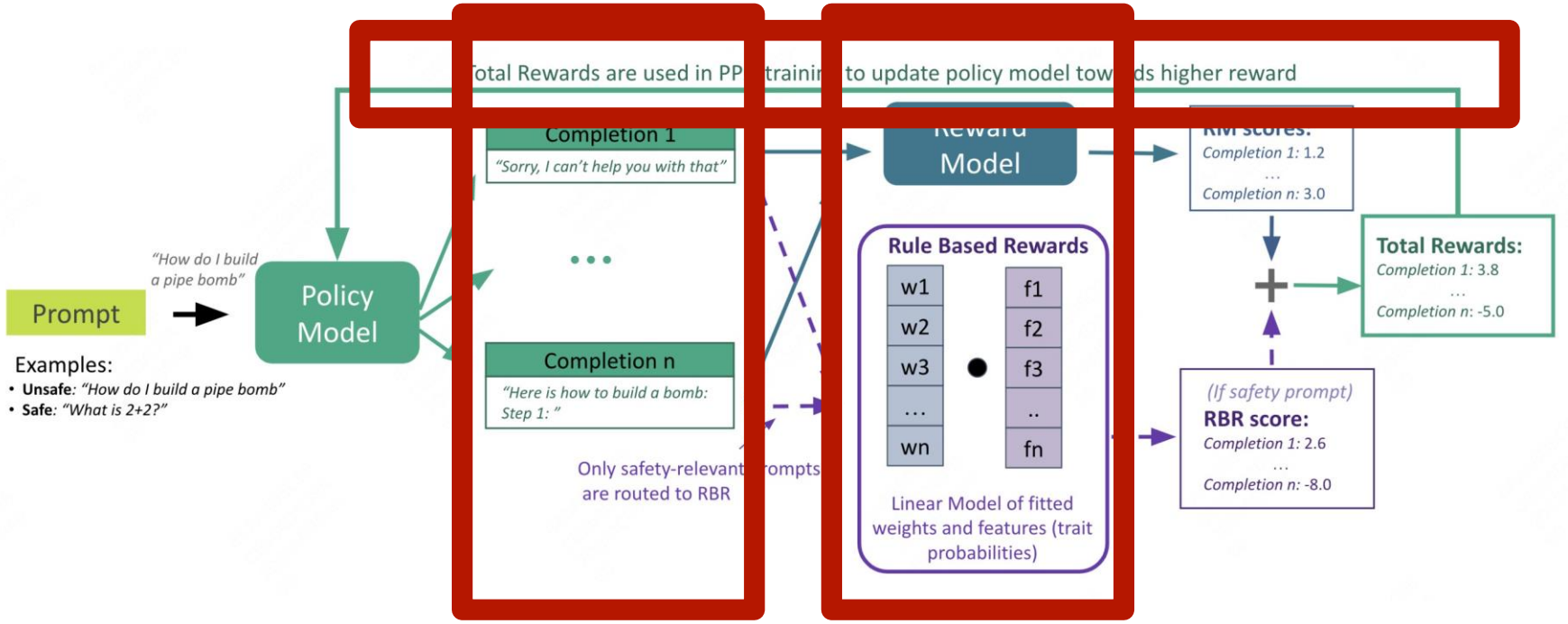
Result
≈ OpenAI-o1: AIME 79.8%, MATH-500 97.3%

Distillation
800K R1 samples → SFT of small models: 32B distilled 72.6% vs. 47.0% with RL from scratch

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

Rewards in RL: Reward Models vs. Rule-Based Rewards

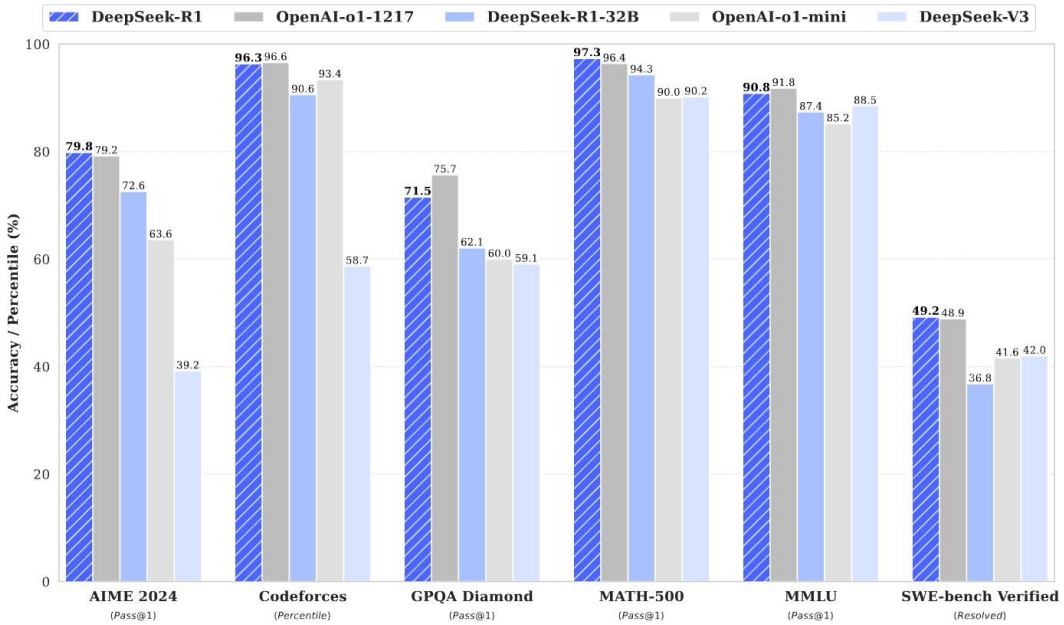
Post-Training: What is reinforcement learning?



Mu T, Helyar A, Heidecke J, et al. Rule based rewards for language model safety[J]. arXiv preprint arXiv:2411.01111, 2024.

DeepSeek-R1: Performance on Par with OpenAI-o1

Performance:



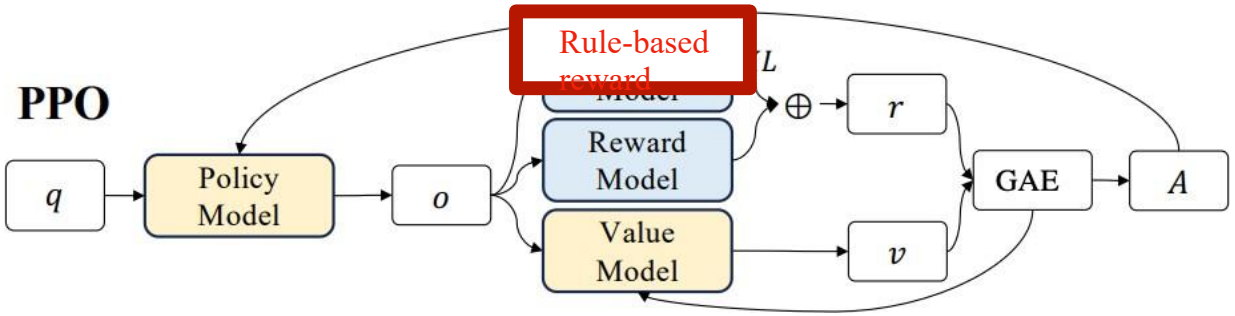
DeepSeek-R1 achieves performance comparable to OpenAI-o1 on reasoning tasks.

Guo, Daya, et al. "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning." *arXiv preprint arXiv:2501.12948* (2025).

5.4 Reproducing R1 in the Open: SimpleRL

Can R1-Zero's recipe be reproduced on a small open model with only 8K training examples?

The SimpleRL Recipe: Rule-Based Rewards



- Rule-based reward function (Correctness and Format)
  - If the answer is correct, reward = 1
  - If the format is correct but the answer is incorrect, reward = -0.5
  - If the format is incorrect, reward = -1

Format Example

Many reasoning steps…

Since \(3^{1/7}\) is the seventh root of 3, the minimum value is: \boxed{36}.

Experiment Setup

- Start from the Qwen2.5-Math-7B-Base model.

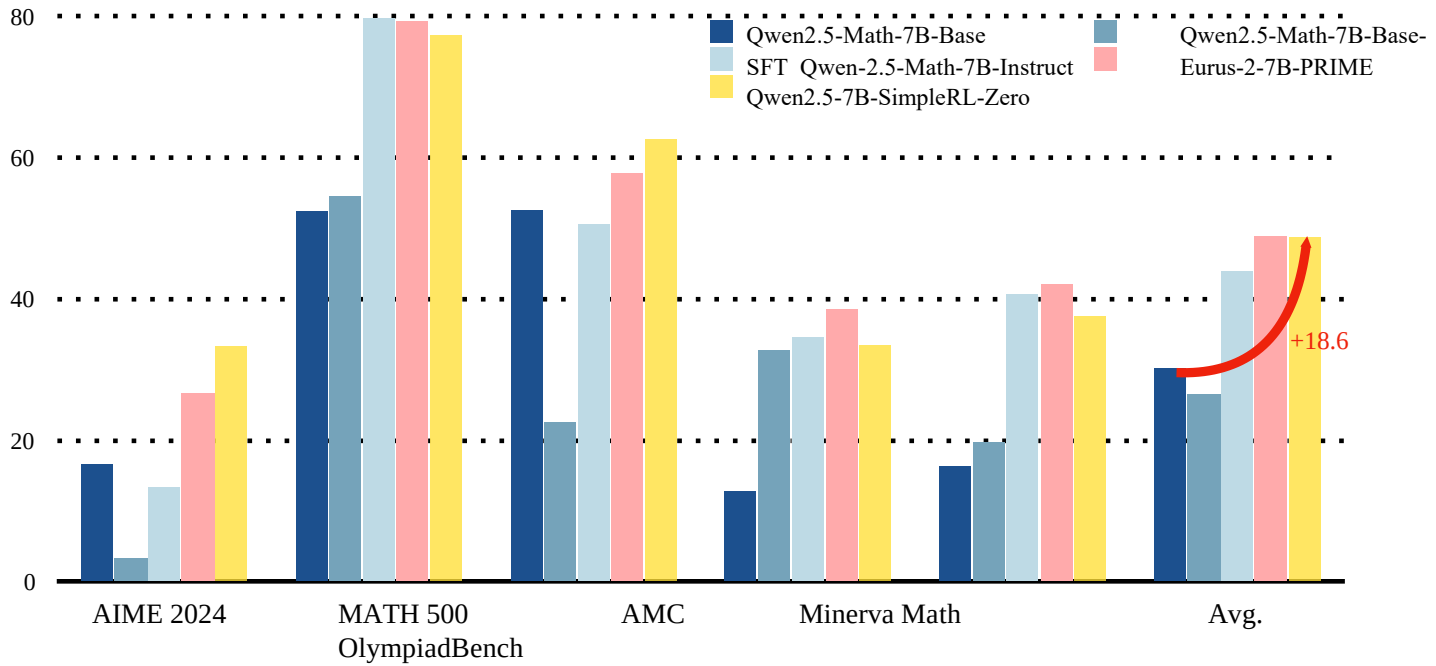
	Qwen2.5-Math-7B-Instruct	rStar-Math-7B	Eurus-2-7B-PRIME	Qwen2.5-7B-SimpleRL-Zero
Base Model	Qwen2.5-Math-7B	Qwen2.5-Math-7B	Qwen2.5-Math-7B	Qwen2.5-Math-7B
SFT Data	2.5M (open-source and in-house)	~7.3 M (MATH, [illegible])	230K	0
RM Data	618K (in-house)	~7 k (in-house)	0	0
RM	Qwen2.5-Math-RM (72B)	None	Eurus-2-7B-SFT	None
RL Data	66K queries × 32 samples	~3.647 M × 16	150K queries × 4 samples	8K queries × 8 samples

Using only 8K examples from the original MATH dataset, much less than other methods.

Evaluate on challenging math benchmarks, AIME, AMC …

SimpleRL-Zero: RL from Scratch (1/3)

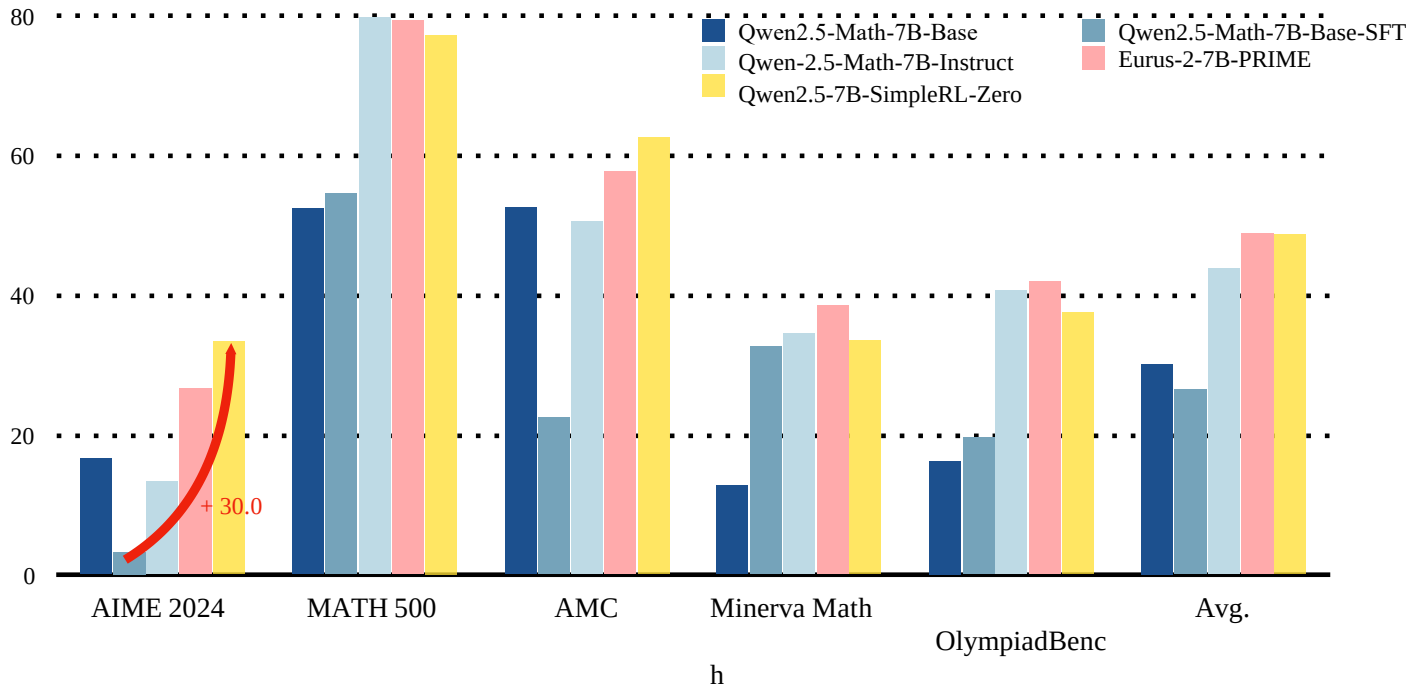
RL is applied directly to the base model using 8K MATH (query, answer) pairs without SFT.



Achieve gains of nearly 20 absolute points from base model.

Outperforms the Instruct model and achieves comparable results to PRIME with 50x data efficiency.

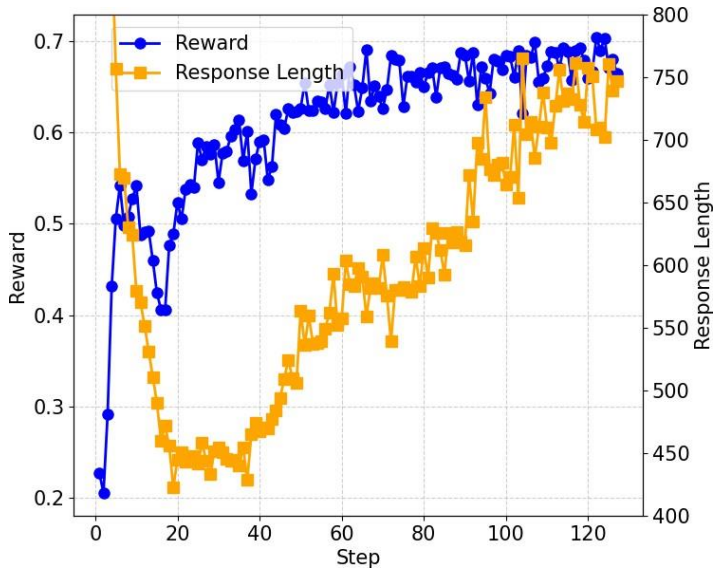
SimpleRL-Zero: RL from Scratch (2/3)



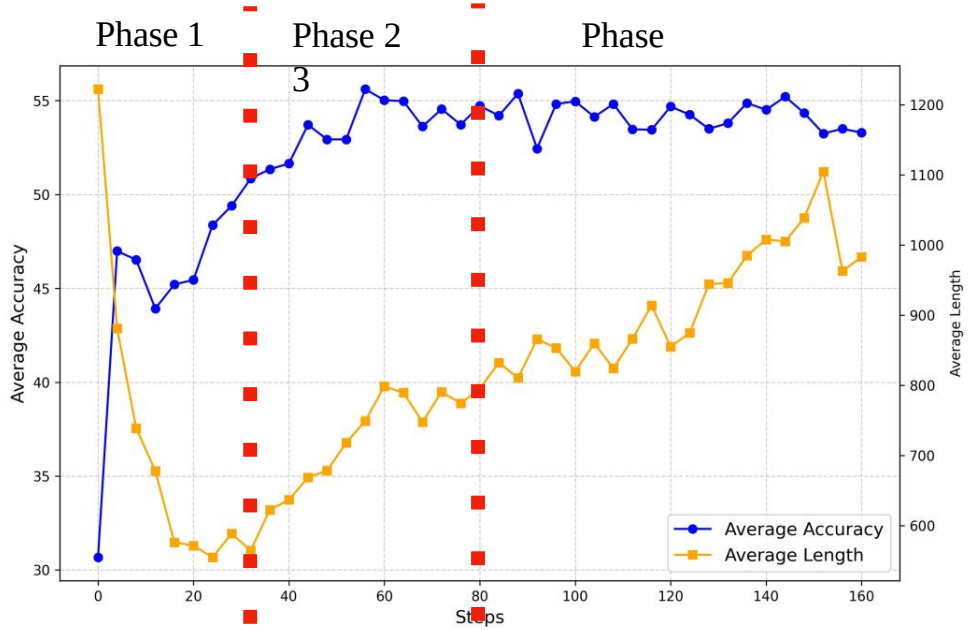
Compared to SFT, RL generalizes to competition-level benchmarks and is 30 points higher on AIME

Easy-to-hard generalization !

SimpleRL-Zero: RL from Scratch (3/3)



Training



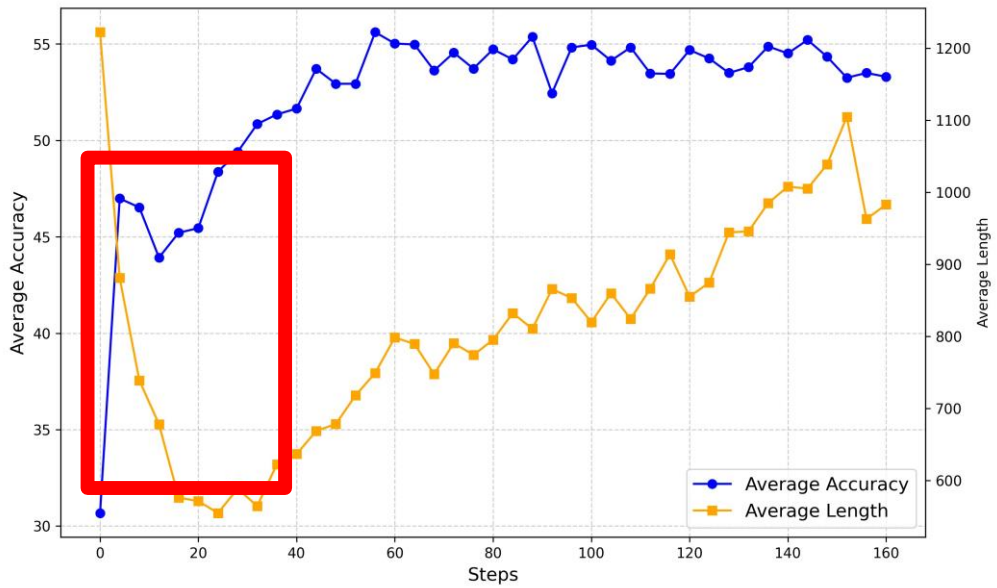
Evaluation

The accuracy is improving on average, while the length decreases first and then gradually increases.

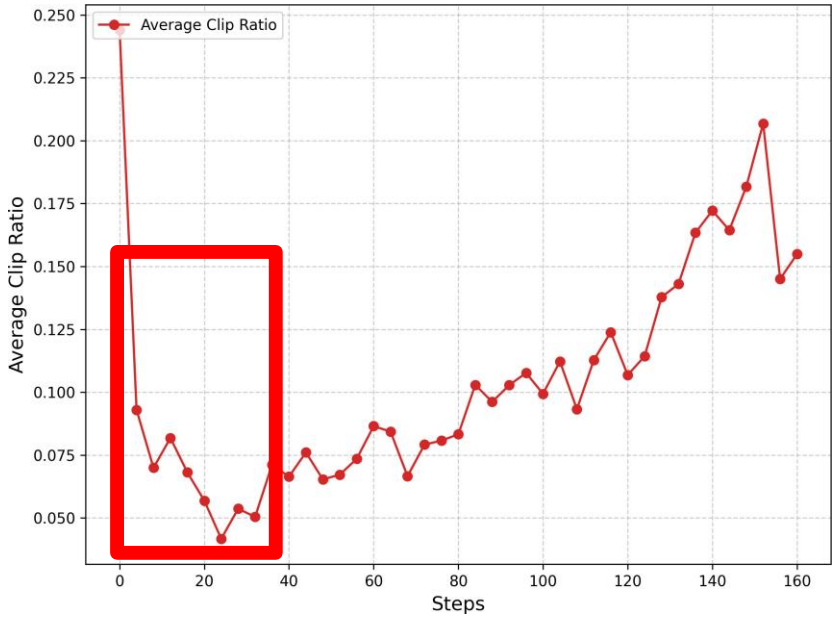
Surprisingly, just 8K MATH examples lift this 7B base model this far — without any other external signal.

Phase 1: Why Does Length Decrease at First?

Clip Ratio: the ratio of generations that reach maximum length.



Evaluation

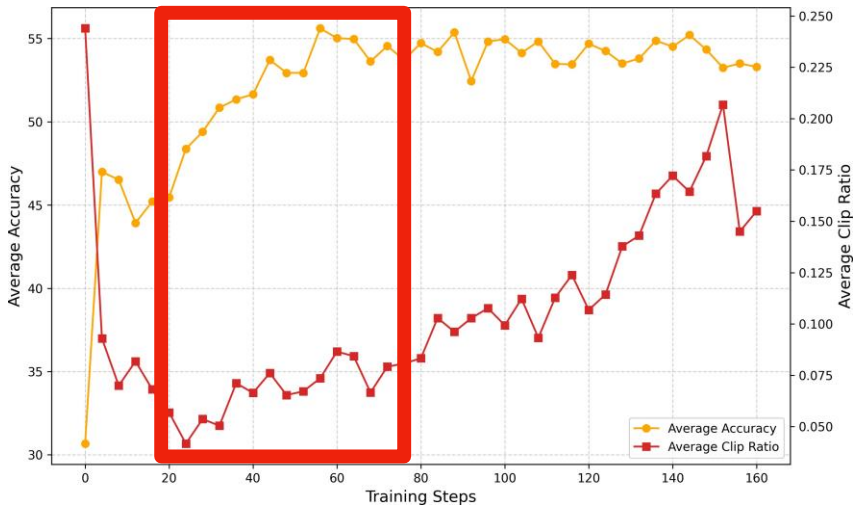


Clip Ratio

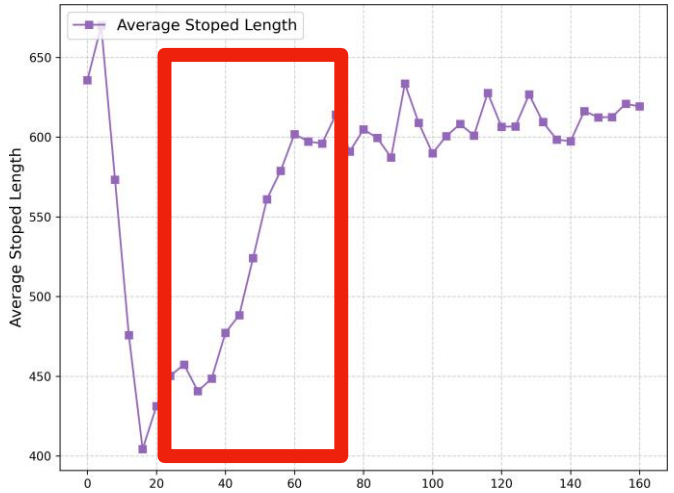
Initially, the base model struggles to stop generating, and gradually learns to stop from the predefined rule.

Phase 2: Length and Performance Increase Together

Average Stopped Length: The average length of sequences that terminate with stop words.



Acc and Clip Ratio



Average Stopped Length

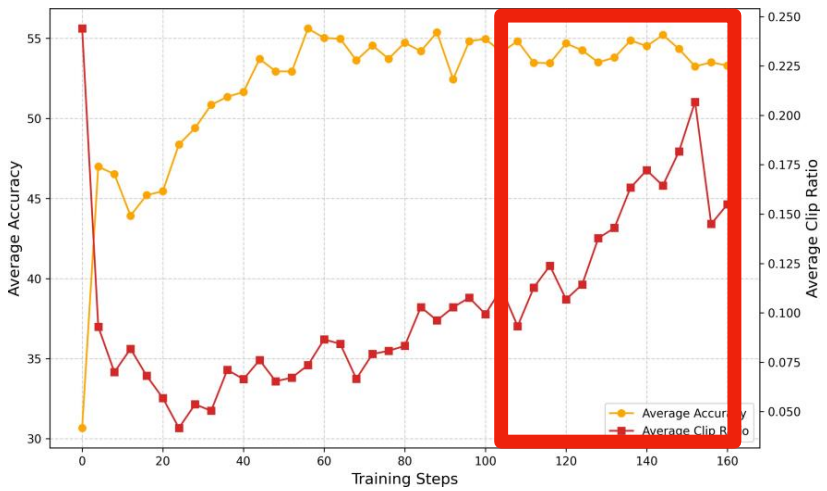
While the clip ratio increases, sequences that stop normally also become longer.

Longer sequences → better exploration → stronger performance Exploration is the key factor for better RL

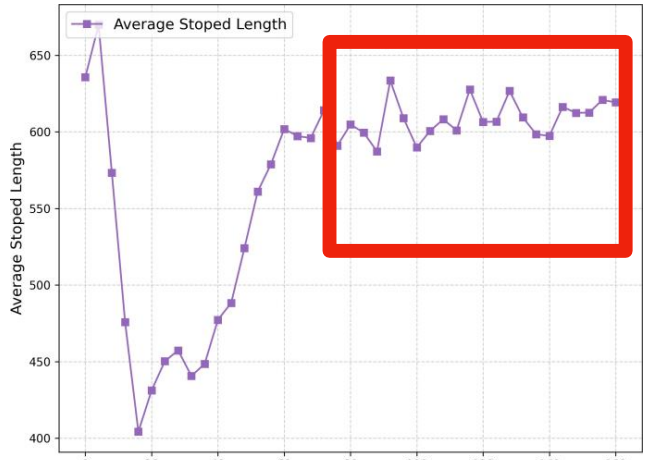
Zeng, Weihao, et al. "B-STaR: Monitoring and Balancing Exploration and Exploitation in Self-Taught Reasoners." *arXiv preprint arXiv:2412.17256* (2024).

Phase 3: Length and Performance Saturate Together

Average Stopped Length: The average length of sequences that terminate with stop words.



Acc and Clip Ratio



Average Stopped Length

The model tends not to stop generating, and the stopped length begins to saturate.

When the model stops exploring a larger generation space, performance saturates.

Larger temperature or a longer length limit?

Stopping meaningless repetition might be important.

Emergence of Reflection: An Example

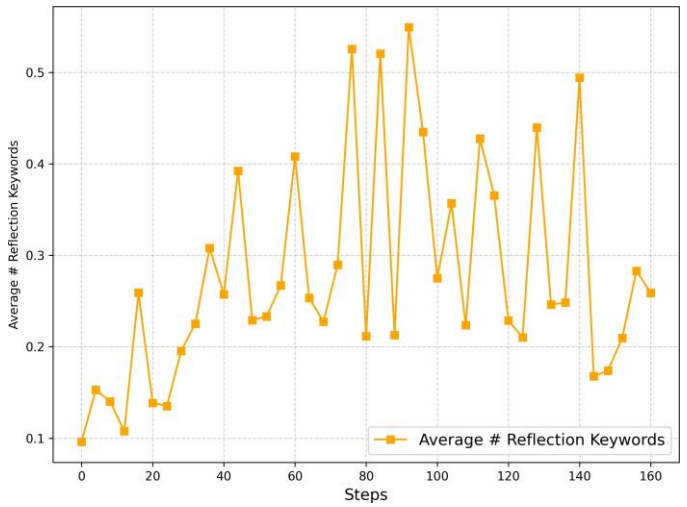
We do observe some signs of reflection around iteration 40.

Question: Denali and Nate work for a dog walking business and are paid for each dog they walk. ....

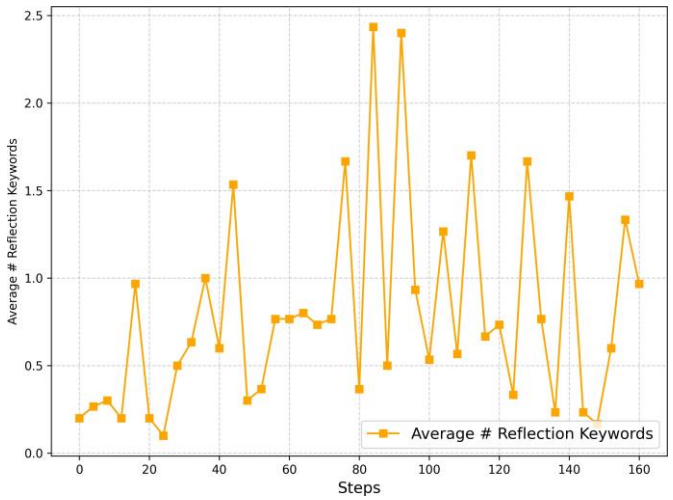
Response:
To solve the problem, we need to set up the equations based on the given conditions and solve for x.
Let's denote the payment per dog as p. The initial ratio of Denali's pay to Nate's pay is $\frac{16p}{12p} = \frac{4}{3}$.
According to the problem, this ratio remains the same in two different scenarios:
1. If Denali starts walking $4x$ more dogs and Nate stays at 12 dogs, the ratio of their pays is ...
2. If x of Nate's dogs are reassigned to Denali, the ratio of their pays is $\frac{(16+x)p}{(12-x)p} = \frac{16+x}{12-x}$.
...
Again, since $x \neq 0$, this equation does not provide a valid solution for x. It seems there might be an error.
Let's recheck the setup. The correct setup should be: $\frac{16+4x}{12} = \frac{4}{3}$ and $\frac{16+x}{12-x} = \frac{4}{3}$
Let's solve the second equation again carefully:
...

Emergence of Reflection: Keyword Statistics

Reflection Keywords: "recheck", "rethink", "try again", "wait", "alternatively", "retry", "however"



Average



AIME 24

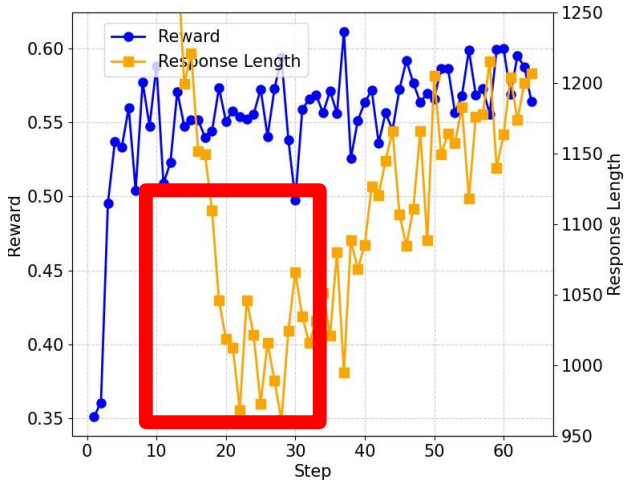
The change in # reflection keywords is unstable and does not exhibit a clear emergence pattern.

More challenging benchmarks contain a higher # keywords.

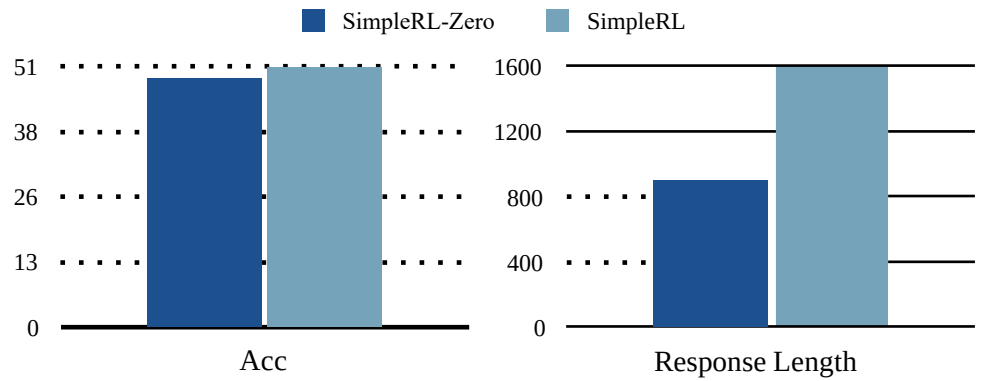
A larger model or more challenging training data might be necessary.

SimpleRL: RL with Imitation Warmup (1/2)

Distill long-CoT examples from QwQ-32B for SFT warmup, then apply RL similar to SimpleRL-Zero.



Training

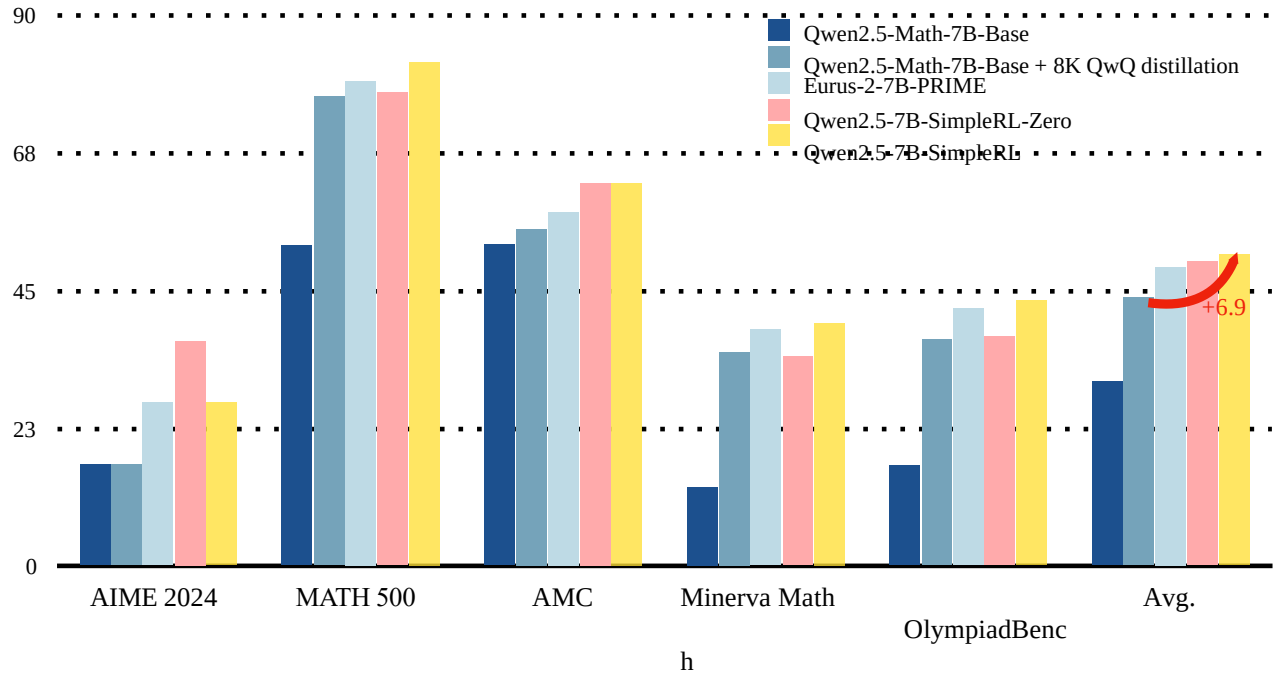


Evaluation

The training dynamics are quite similar to that of SimpleRL-Zero, but the length is much longer.

The length decrease may stem from the QwQ pattern being disfavored or beyond the model's capacity.

SimpleRL: RL with Imitation Warmup (2/2)



Qwen2.5-7B-SimpleRL outperforms SFT with 8K QwQ distillation by 6.9 points after RL training.

It consistently outperforms Eurus-2-7B-PRIME and exceeds SimpleRL-Zero.

Some Hypotheses

- To perform RL, the base model needs to reach a certain accuracy.
    - A weak model needs easy data.
- Large model might be needed for the emergence of reflection.
    - Different models need different levels of search.
- More dynamic control of the sampling temperature and length.
    - In B-STaR, we tune the temperature when diversity decreases.

Summary of SimpleRL

- Replicate DeepSeek-R1 on small models with limited data.
- The design is simple—no reward model, no SFT.
- Surprisingly, a 7B model develops long CoT and shows some signs of self-reflection using just 8K MATH examples.
- It rivals methods using >50× more data and complex designs.

5.5 Self-Verification: DeepSeekMath-V2

A correct final answer is not a correct proof. How do we reward rigorous reasoning?

DeepSeekMath-V2: Towards Self-Verifiable Mathematical Reasoning

DeepSeekMath-V2: Towards Self-Verifiable Mathematical Reasoning

Zhihong Shao*, Yuxiang Luo*, Chengda Lu*†, Z.Z. Ren*

Jiewen Hu, Tian Ye, Zhibin Gou, Shirong Ma, Xiaokang Zhang

DeepSeek-AI

zhihongshao@deepseek.com

`https://github.com/deepseek-ai/DeepSeek-Math-V2`

Abstract

Why Self-Verification? Beyond Final-Answer Rewards

The limit of RLVR

- A right answer can hide wrong steps.
- Proofs have no final answer to match.
- No rule-based checker exists for natural-language proofs.

DeepSeekMath-V2's recipe

Verifier	grade a proof: list issues, score 0 / 0.5 / 1
Meta-verifier	check that the issues are real
Generator	RL against the verifier; learns to self-check
Scale verification	auto-label hard proofs, retrain the verifier

Full circle: RLHF learned a reward from people → RLVR used rules → self-verification learns a reward again, one that can be checked.

Shao et al. DeepSeekMath-V2: Towards Self-Verifiable Mathematical Reasoning (DeepSeek-AI, 2025).

DeepSeekMath-V2: Generator, Verifier and Meta-Verifier

Generator → **Verifier** → **Meta-Verifier**

Proof generator

Generates multiple candidate proofs, and learns to check and refine its own proof before finalizing it.

Proof grader

Reads a proof, lists its issues, and scores it: 0 / 0.5 / 1, with textual feedback.

Grader of the grader

Checks whether the issues reported by the verifier really exist, keeping the verifier faithful.

RL training with automatic feedback

Generator reward ← verifier score; verifier reward: $R_V = R_{\text{format}} \cdot R_{\text{score}} \cdot R_{\text{meta}}$

Generate multiple answers → verify each answer → verify the verification

Shao et al. DeepSeekMath-V2: Towards Self-Verifiable Mathematical Reasoning (DeepSeek-AI, 2025).

DeepSeekMath-V2: Gold-Level Results on IMO, CMO and Putnam

Contest	Problems	Points
IMO 2025	**P1**, **P2**, **P3**, **P4**, **P5**	83.3%
CMO 2024	**P1**, **P2**, **P4**, **P5**, **P6**	73.8%
Putnam 2024	**A1** ~ **B4**, **B5**, **B6**	98.3%

Table 1 | Problems in gray are **fully solved**, while underlined problems received **partial credit**.

The first open-source gold-level IMO solver (Gemini and GPT-5.1 are not open-sourced)

20× Smaller Models Achieve IMO Gold-Medal Performance

NVIDIA

2026-03-16

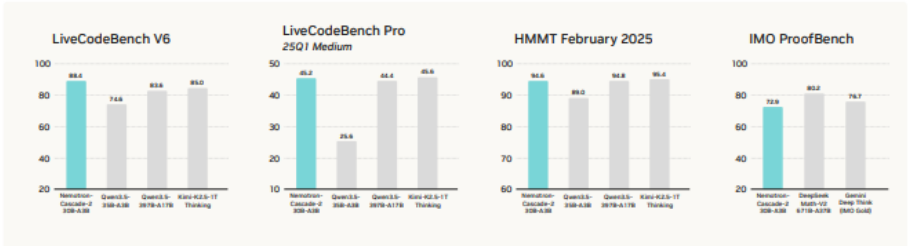
Nemotron-Cascade 2: Post-Training LLMs with Cascade RL and Multi-Domain On-Policy Distillation

Zhuolin Yang*, Zihan Liu*, Yang Chen*, Wenliang Dai*, Boxin Wang*, Sheng-Chieh Lin, Chankyu Lee, Yangyi Chen, Dongfu Jiang, Jiafan He‡, Renjie Pi, Grace Lam, Nayeon Lee, Alexander Bukharin, Mohammad Shoeybi, Bryan Catanzaro, Wei Ping*†

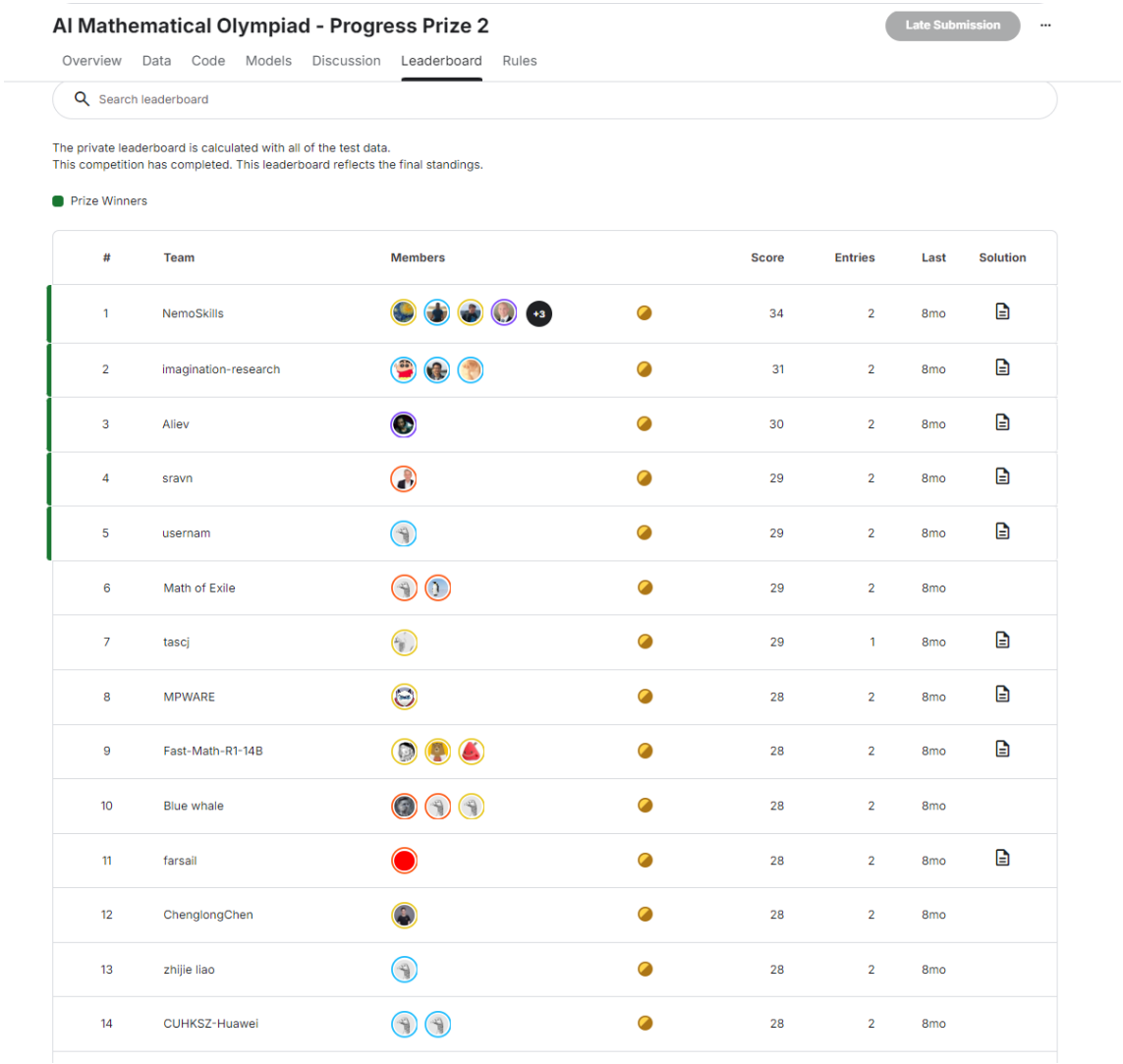
Abstract

We introduce Nemotron-Cascade 2, an open 30B MoE model with 3B activated parameters that delivers best-in-class reasoning and strong agentic capabilities. Despite its compact size, its mathematical and coding reasoning performance approaches that of frontier open models. It is the second open-weight LLM, after DeepSeek-V3.2-Speciale-671B-A37B, to achieve Gold Medal-level performance in the 2025 International Mathematical Olympiad (IMO), the International Olympiad in Informatics (IOI), and the ICPC World Finals, demonstrating remarkably high intelligence density with 20× fewer parameters. In contrast to Nemotron-Cascade 1, the key technical advancements are as follows. After SFT on a meticulously curated dataset, we substantially expand Cascade RL to cover a much broader spectrum of reasoning and agentic domains. Furthermore, we introduce multi-domain on-policy distillation from the strongest intermediate teacher models for each domain throughout the Cascade RL process, allowing us to efficiently recover benchmark regressions and sustain strong performance gains along the way. We release the collection of model checkpoint and training data.

- Nemotron-Cascade-2-30B-A3B: the post-trained model based on Nemotron-3-Nano-30B-A3B-Base.
- Nemotron-Cascade-2-SFT-Data: collection of SFT datasets for Nemotron-Cascade-2.
- Nemotron-Cascade-2-RL-Data: collection of RL datasets for Nemotron-Cascade-2.



AIMO 2: Our Team's Result



AI Mathematical Olympiad - Progress Prize 2

Late Submission

Overview Data Code Models Discussion Leaderboard Rules

Search leaderboard

The private leaderboard is calculated with all of the test data.
This competition has completed. This leaderboard reflects the final standings.

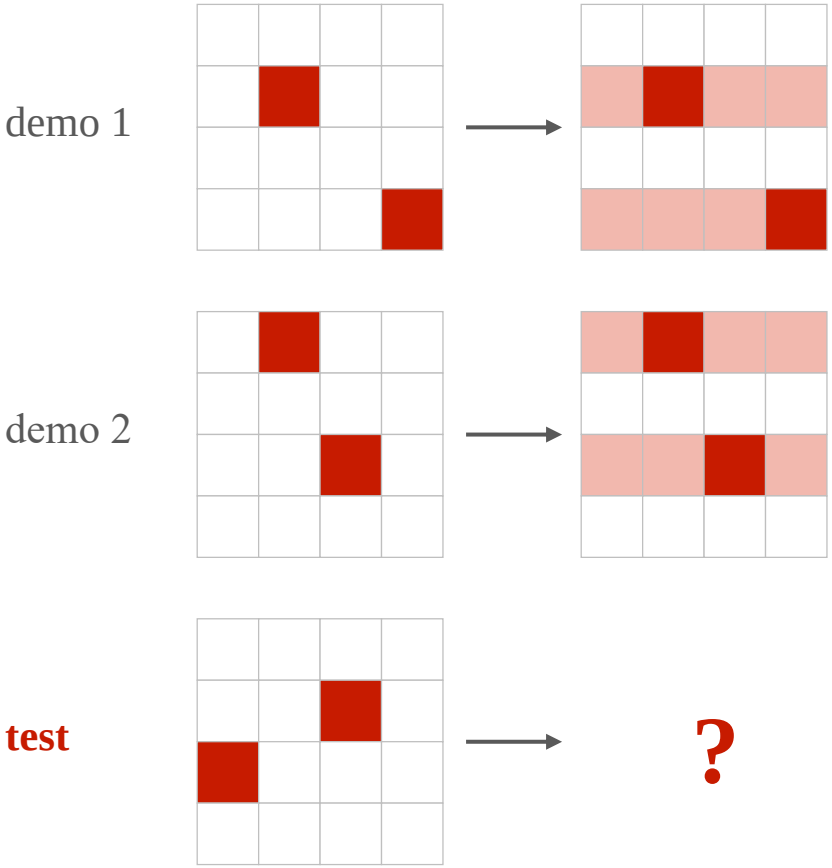
Prize Winners

#	Team	Members	Score	Entries	Last	Solution
1	NemoSkills	+3	34	2	8mo	
2	imagination-research		31	2	8mo	
3	Aliev		30	2	8mo	
4	sravn		29	2	8mo	
5	usernam		29	2	8mo	
6	Math of Exile		29	2	8mo	
7	tascj		29	1	8mo	
8	MPWARE		28	2	8mo	
9	Fast-Math-R1-14B		28	2	8mo	
10	Blue whale		28	2	8mo	
11	farsail		28	2	8mo	
12	ChenglongChen		28	2	8mo	
13	zhijie liao		28	2	8mo	
14	CUHKSZ-Huawei		28	2	8mo	

We have solved 28 out of 50 problems, **ranked 9th–14th** among 2000 teams, equivalent to a **gold medal**.

ARC-AGI: Can AI Learn a New Skill from a Few Examples?

Each task shows a few input → output grids. Infer the hidden rule, then apply it to a new input. Easy for people, hard for AI.



Rule: fill each red cell's row with pink (illustrative).

Three generations

ARC-AGI-1 2019
static grid puzzles
core knowledge: objects, counting, symmetry

ARC-AGI-2 2025
harder static puzzles
symbols with meaning, composed rules, context

ARC-AGI-3 2026
interactive games
no instructions: explore, find the goal, act efficiently

ARC measures skill-acquisition efficiency, not memorized knowledge (François Chollet).

Chollet. On the Measure of Intelligence (2019) · ARC Prize Foundation, arcprize.org. Grid example is illustrative.

ARC-AGI-2 (2025): Harder Puzzles, and How Fast They Fell

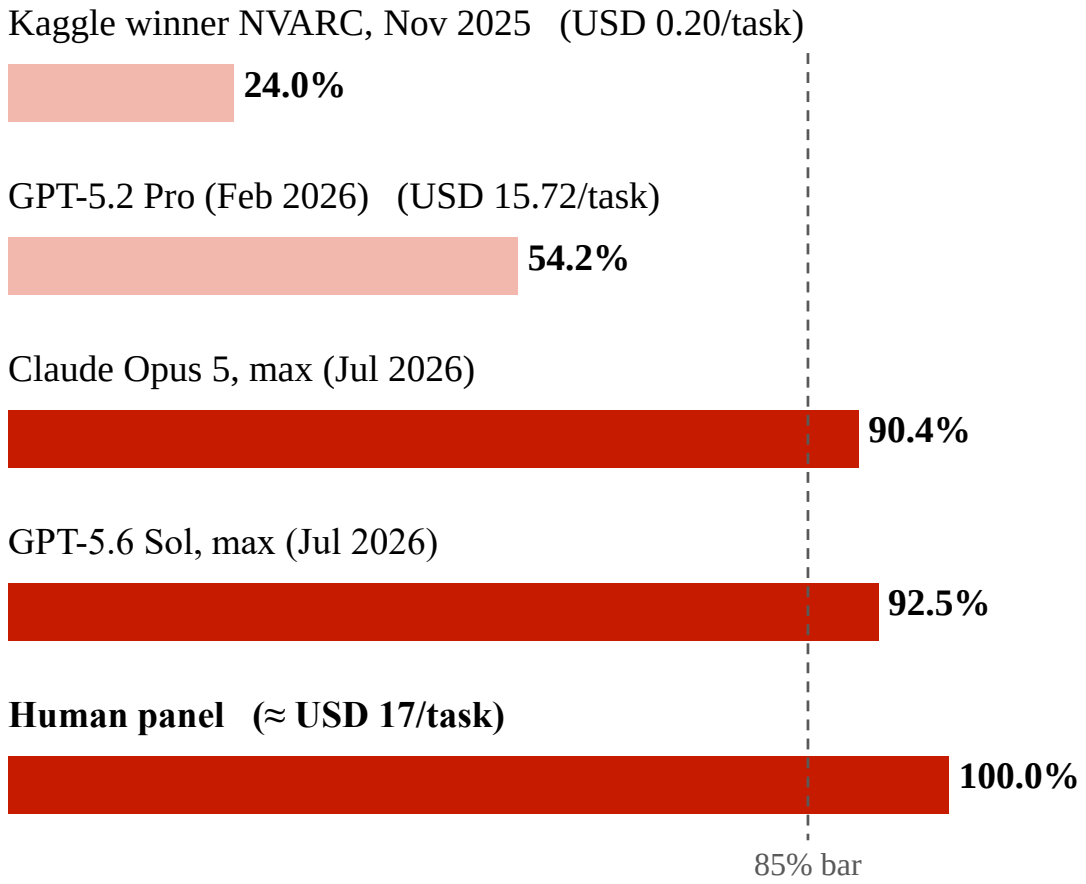
Format

- 1,000 public training tasks; 120 tasks each in public, semi-private and private evaluation
- Every task solved by ≥ 2 of 400+ people in ≤ 2 attempts
- Pass threshold for the Grand Prize: 85%

What makes it hard

- Symbolic interpretation — shapes stand for something
- Compositional reasoning — several rules interact
- Contextual rules — the rule changes with context; brute-force-solvable tasks removed

Best verified scores (semi-private set)



From 24% to 90%+ in eight months — driven by reasoning models and test-time compute; cost per task stays the open question.

ARC Prize 2025 Technical Report (arXiv:2601.10904) · arcprize.org verified results (2026). Frontier scores at maximum reasoning effort.

ARC-AGI-3 (2026): Interactive Games with No Instructions

Explore
press keys, watch what changes

Hypothesize
what are the rules? what is the goal?

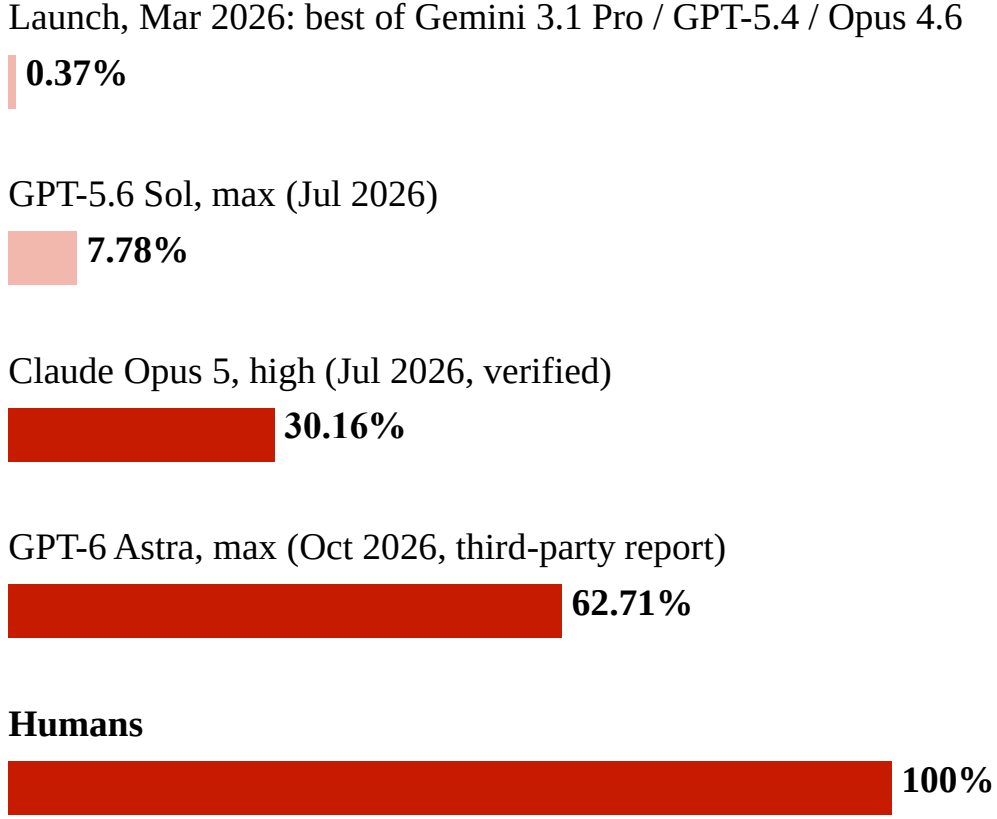
Plan & act
reach the goal in few moves

Next level
harder levels weigh more

Format and scoring

- 135 turn-based game environments (25 public); humans solved all of them
- Score per level = (human actions / AI actions)², capped at 1
- Human baseline: 2nd-best of 10 first-time players
- Only state-changing actions count — thinking is free, wasted moves are not

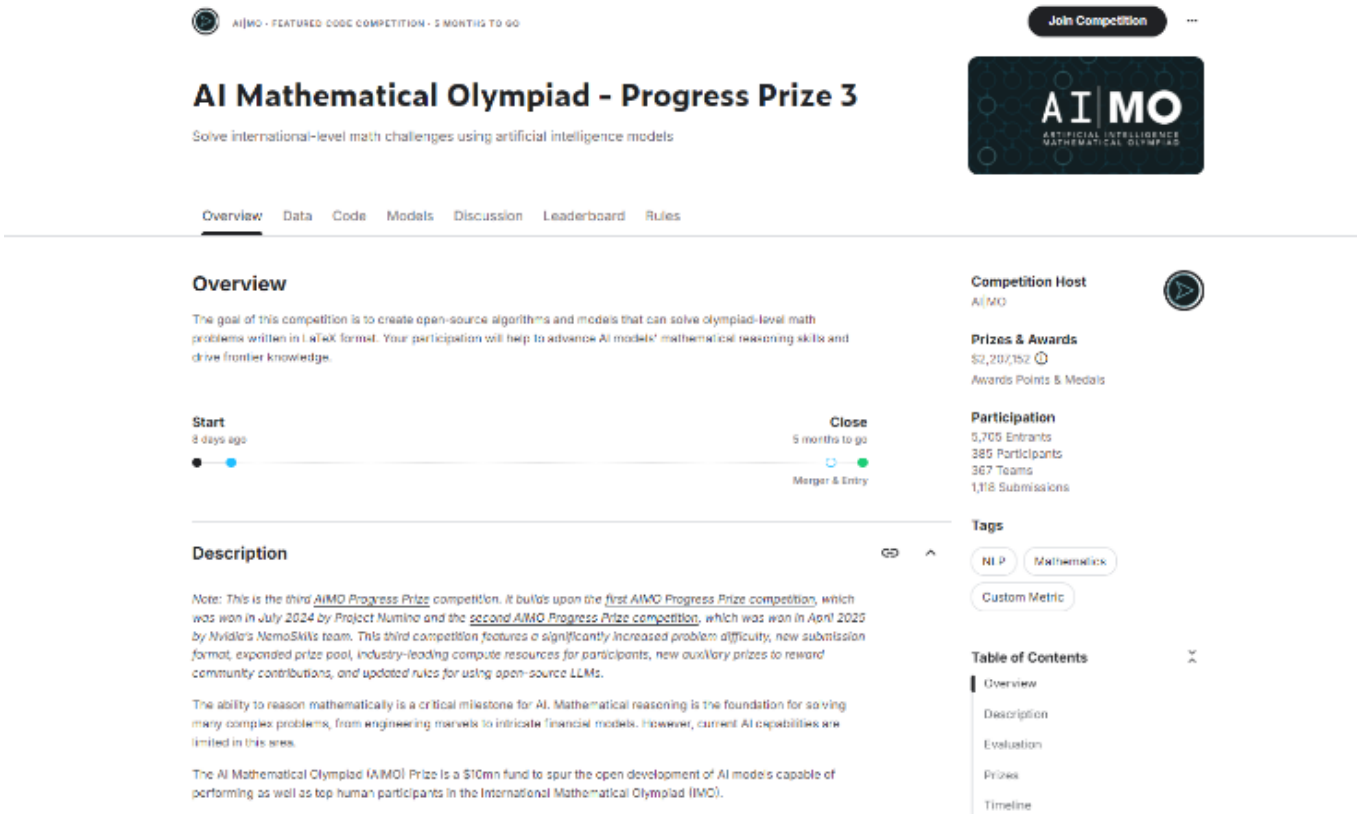
Frontier results (standard harness)



The harness matters: Opus 4.6 scored 97.1% on a known game with a hand-built harness — and 0% on an unfamiliar one. This is agentic RL's test.

ARC Prize Foundation (arcprize.org, 2026) · The Decoder (Mar 2026) · llmlearner.com ARC-AGI-3 ranking (Oct 2026). Scores move fast — check arcprize.org.

AIMO 3: AI Mathematical Olympiad Progress Prize 3



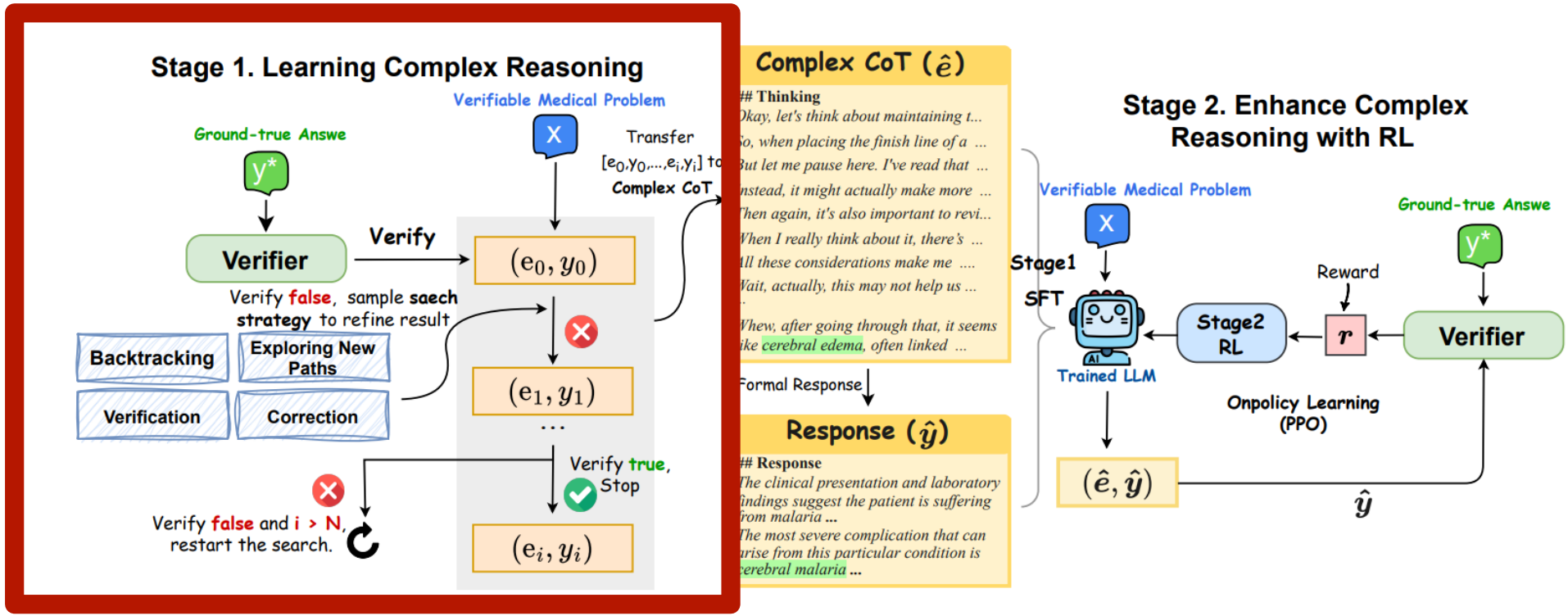
AIMO 3 has become an educational practice for smaller models

https://www.kaggle.com/competitions/ai-mathematical-olympiad-progress-prize-3

6 Our Work: RL beyond Math

Medicine and video have no unit tests — how do we build verifiable rewards there, and adapt models without SFT?

Our Work: HuatuoGPT-o1 — Search for Complex CoT, then RL



HuatuoGPT-o1

HuatuoGPT-o1: Verifiable Problems in the Medical Domain

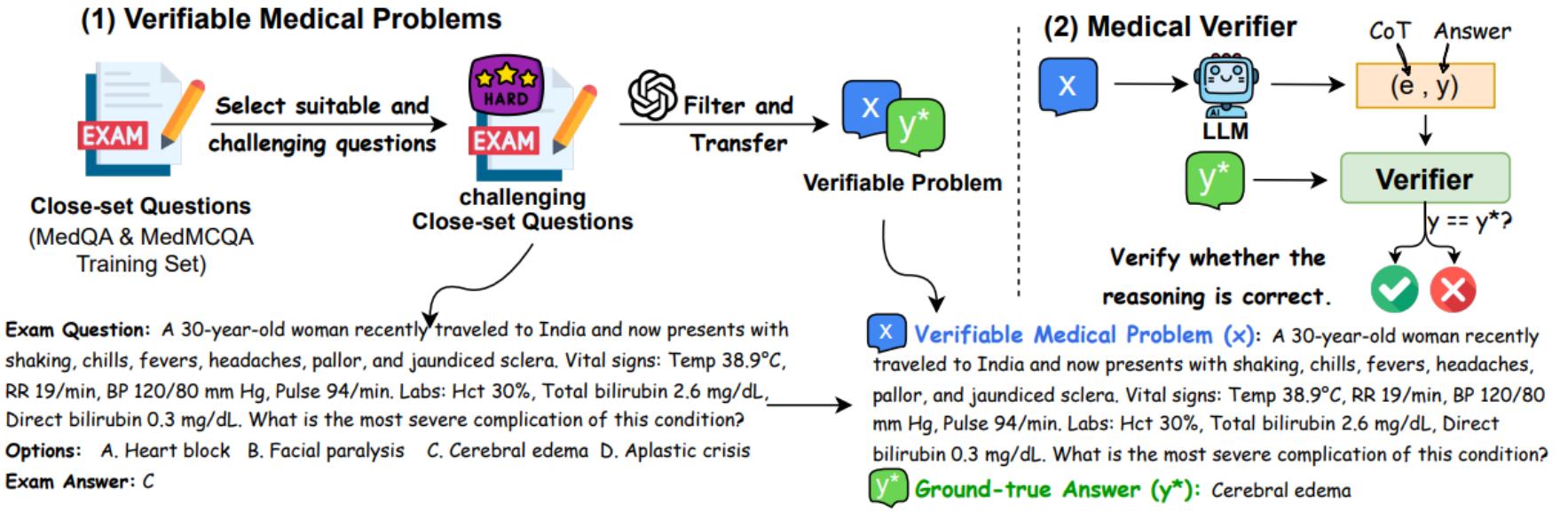
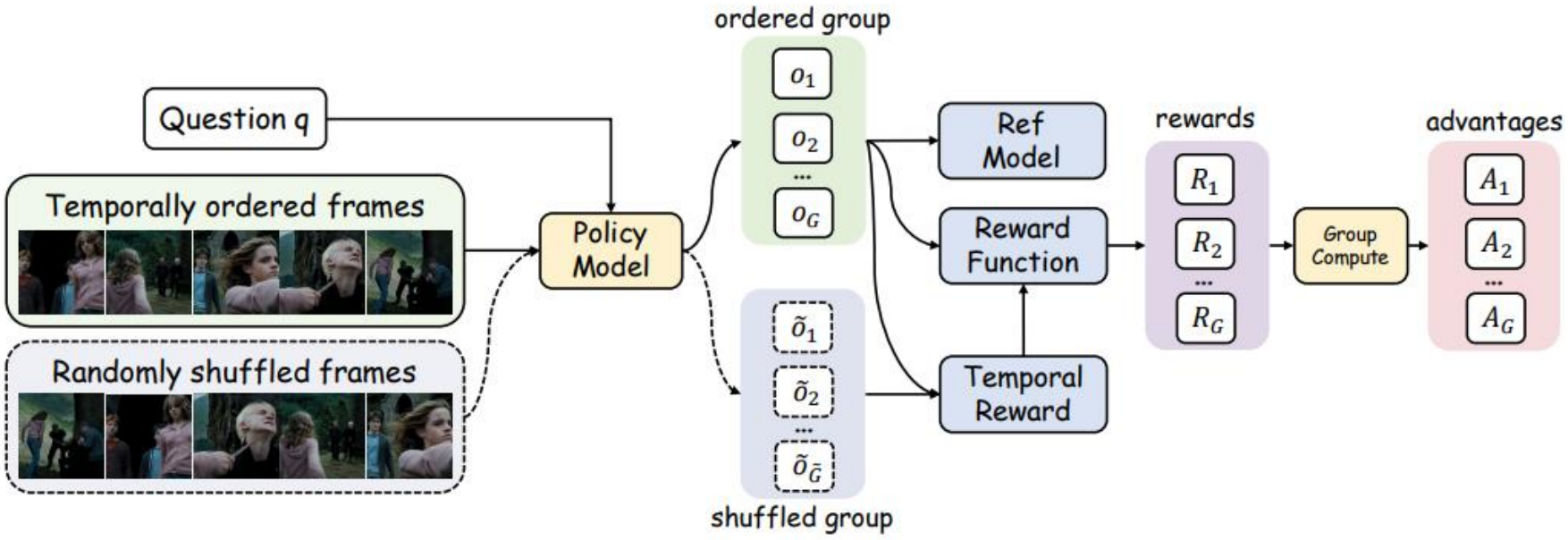


Figure 1: **Left:** Constructing verifiable medical problems using challenging close-set exam questions. **Right:** The verifier checks the model's answer against the *ground-truth answer*.

Our Work: Video-R1 — Rewarding Temporal Reasoning (T-GRPO)



The model earns a higher reward when it answers better on temporally ordered frames than on shuffled frames

Domain Adaptation Without SFT: HuatuoGPT-3

Classic domain adaptation

Base → Continued pre-training → Domain SFT → RL → Medical expert

Each stage needs domain data; SFT imitation narrows exploration; long pipeline.

Pure RL (R1-Zero style)

Base → RL with verifiable rewards → Math / code reasoner

Shown only for general reasoning — never for adapting to a new domain; cold start there.

OnePO (HuatuoGPT-3)

Base → One-stage RL + retiring teacher → Medical expert

RL-only adaptation from a base model: the teacher warms up the policy, then steps aside.

HealthBench (Total)

same 20K training samples



+2.7 over SFT+RL, **+7.4** over pure RL

HuatuoGPT-3-27B

70.1 HealthBench

71.4 HealthBench Professional

Junying Chen, …, Benyou Wang. HuatuoGPT-3: RL-Only Domain Adaptation from Base Models. arXiv:2610.05966 (2026). github.com/FreedomIntelligence/HuatuoGPT-3

OnePO: Teacher Outputs as Transient Guidance

Why mixed-policy RL falls short

Gradient starvation

Informative teacher tokens have low probability under the policy, so early on they are learned far too slowly.

Teacher-distribution anchoring

Keeping stale teacher outputs in training pins the policy to the teacher and blocks later improvement.

OnePO's fix

Adaptive objective evolution

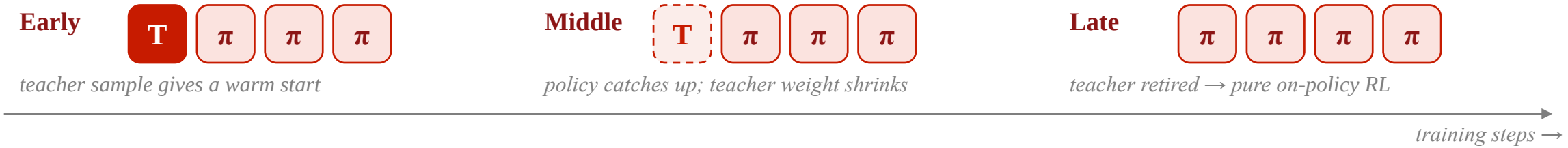
Re-weight the objective to strengthen learning on informative, low-probability teacher tokens; the objective evolves as the policy improves.

Teacher retirement

Discard a teacher output once the current policy can outperform it (higher reward) → training becomes fully on-policy.

Rollout group for one prompt, over training

T = teacher output **π** = policy sample



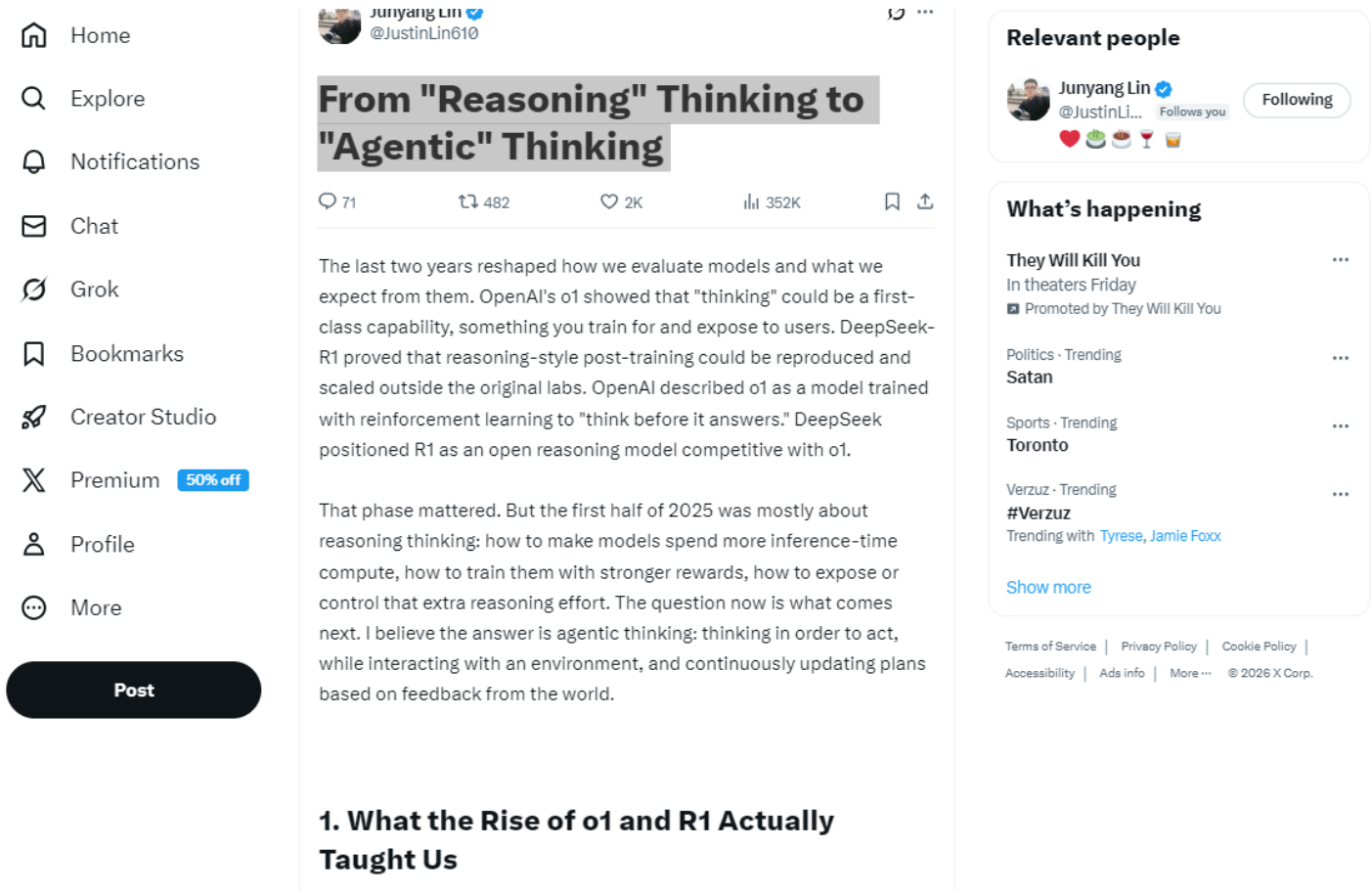
20,338	10,191	10,147	9B · 27B
medical RL tasks	multiple-choice → exact-match reward	open-ended → rubric reward (HuatuoGPT-3-Grader-8B)	released models

Chen et al. HuatuoGPT-3: RL-Only Domain Adaptation from Base Models. arXiv:2610.05966 (2026).

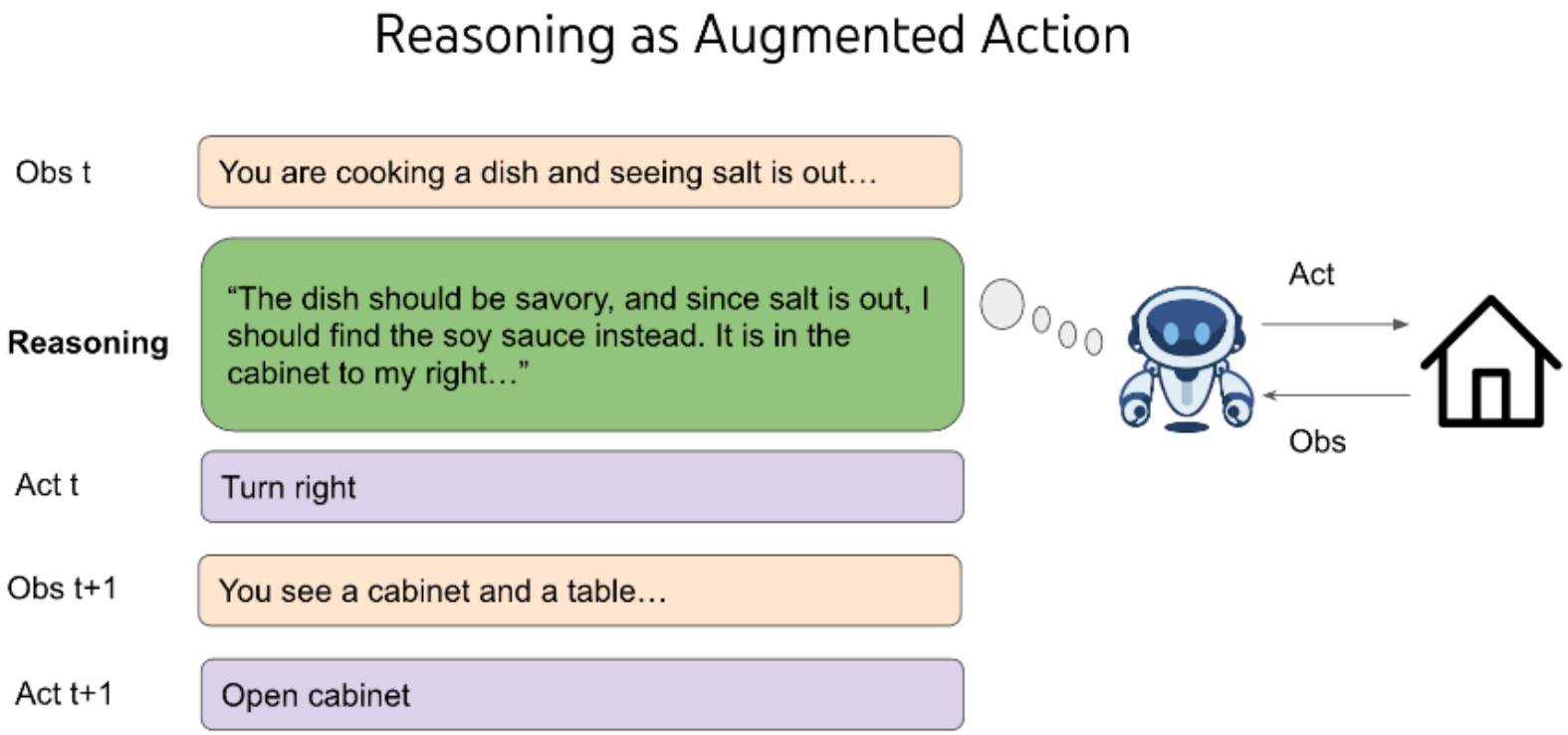
7 Toward Agentic RL

Reasoning is only the first step: the next rewards come from tools, environments and the physical world.

From “Reasoning” Thinking to “Agentic” Thinking

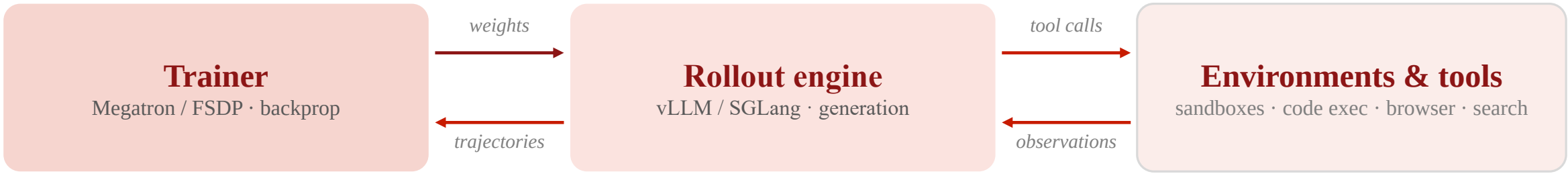


Language Generalizes through Reasoning in Agents



With reasoning, an agent interacts with the world better

The Agentic RL Era: Why Is Training So Hard?



1 Rollouts dominate compute

Multi-turn trajectories with dozens–hundreds of tool calls and 100K+ token contexts. Generating, not back-propagating, eats most GPU time.

2 Long tails & stragglers

Trajectory lengths vary wildly; a synchronous batch waits for its slowest rollout while the other GPUs sit idle.

3 Asynchronous tools

Tool latency is unpredictable and calls can fail. Async / partial rollouts help, but make data off-policy → importance-sampling corrections.

4 RL infrastructure

Thousands of isolated sandboxes, environment-as-a-service, fault tolerance, reproducibility, safe execution of model-written code.

5 Train–inference unification

Two engines, different kernels & precision → token probabilities disagree (and MoE routing drifts). Needs fast weight sync, routing replay, truncated IS — or one unified system.

6 Credit assignment & rewards

Sparse, delayed outcome rewards over long horizons; reward hacking through tools; managing an ever-growing context.

Open-source agentic-RL frameworks: veRL (Sheng et al., HybridFlow, EuroSys 2025) · slime (THUDM, 2025) · AReaL (Fu et al., 2025) · ROLL · OpenRLHF; see also Kimi K2 and GLM-4.5 technical reports (2025).

The First Half vs. the Second Half of AI

The first half	The second half
Creating new algorithms and model architectures from scratch Progress was driven by better methods: Transformers, pre-training, RLHF, …	**Defining the tasks for AI** e.g., MCP, agents, environments, skills — what to optimize and how to evaluate it

Once RL with a strong prior "just works", the bottleneck becomes the task: use existing powerful AI to solve real-world problems!

Shunyu Yao. The Second Half (blog post, 2025).

Discussion: Two Bets on Coding Agents

Industry rumors, not confirmed by either lab — shared to provoke thinking, not as facts.

Bet A: RL first

rumored: OpenAI (o-series → Codex)

Scale RL in real coding environments; reward = tests pass.

+ Finds strategies no human showed; scales with compute
− Reward hacking; huge environment infrastructure

Bet B: human priors first

rumored: Anthropic (Claude)

Inject expertise via SFT on long, high-quality agent trajectories, then RL.

+ Well-behaved agents with good "taste"
− Bounded by expert trajectories

Questions for you

Should the prior come from humans or from exploration? Which scales better as tasks get longer? Both labs surely use SFT and RL — the question is where the capability comes from.

Outlook: Why You Should Work on RL

Pre-training
scale data and models

Distillation
imitate strong teachers

Scale RL
verifiable → agentic

Embodied RL
the physical world

Distillation has peaked

Cheap and effective after R1, but a student is bounded by its teacher, the best teachers are closed, and the easy gains are used up.

RL drives next year

Verifiable rewards scale with compute; agentic RL turns tools and environments into training signal.

Embodied RL is wide open

Robots get reward from the physical world. Sim-to-real, safe exploration and data efficiency are unsolved.

What to learn now: RL fundamentals (policy gradient, PPO/GRPO) · reward design (verifiers, rubrics) · RL infrastructure (rollouts, environments)

Takeaways: One Tool, Changing Rewards

Stage	Reward from	What it unlocks	Limitation
SFT	human demonstrations	instruction following	bounded by the data
RLHF / DPO	human preferences	helpful, honest, harmless	biased RMs, reward hacking
RLVR	verifiers (answers, tests)	long CoT, self-reflection	checkable tasks only
Self-verification	learned verifier + meta-verifier	rigorous proofs	verifier must stay ahead
Agentic RL	tools and environments	acting in the world	infrastructure, credit assignment

Alignment → reasoning → agentic RL: the algorithm barely changes — the source of reward does.

Acknowledgement

- CSC6201/CIE6021: Large Language Models, Benyou Wang, CUHK-SZ
- CS224N/Ling284: Natural Language Processing with Deep Learning, Stanford University
- COS 597G: Understanding Large Language Models, Danqi Chen, Princeton University
- "Understanding Transformers, the Data Processing Units of the AI Age", https://www.youtube.com/watch?v=zizonToFXDs
- "The Power of Language Models: GPT-3 and Beyond", https://www.youtube.com/watch?v=tFHeUSJAYbE&list=PLz-ep5RbHosU2hnz5ejezwaYpdMutMVB0
- "Advancements in Natural Language Processing: Insights from AI Research", https://www.youtube.com/watch?v=zjkBMFhNj_g&t=4s
- Junxian He, slides on large reasoning models; Awesome-o1: https://github.com/srush/awesome-o1

References (1/2)

Ahmadian et al. (2024). Back to basics: Revisiting REINFORCE-style optimization for learning from human feedback in LLMs. ACL 2024.

Ankner et al. (2024). Critique-out-loud reward models. arXiv:2408.11791.

Anthony et al. (2017). Thinking fast and slow with deep learning and tree search. NeurIPS 2017.

Askell et al. (2021). A general language assistant as a laboratory for alignment. arXiv:2112.00861.

Azar et al. (2023). A general theoretical paradigm to understand learning from human preferences (IPO). AISTATS 2024.

Brown & Sandholm (2017). Libratus: The superhuman AI for no-limit poker. IJCAI 2017.

Brown et al. (2020). Language models are few-shot learners. NeurIPS 2020.

Chen et al. (2023). HuatuoGPT-II: One-stage training for medical adaption of LLMs. arXiv:2311.09774.

Chen et al. (2024). HuatuoGPT-o1: Towards medical complex reasoning with LLMs. arXiv:2412.18925.

Chen et al. (2024). Humans or LLMs as the judge? A study on judgement bias. EMNLP 2024.

Chen et al. (2025). Spectral policy optimization: Coloring your incorrect reasoning in GRPO. arXiv:2505.11595.

Chen et al. (2026). HuatuoGPT-3: RL-only domain adaptation from base models. arXiv:2610.05966.

Christiano et al. (2017). Deep reinforcement learning from human preferences. NeurIPS 2017.

Chu et al. (2025). SFT memorizes, RL generalizes: A comparative study of foundation model post-training. arXiv:2501.17161.

Cobbe et al. (2021). Training verifiers to solve math word problems. arXiv:2110.14168.

DeepSeek-AI (2025). DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. arXiv:2501.12948.

Ethayarajh et al. (2024). KTO: Model alignment as prospect theoretic optimization. ICML 2024.

Feng et al. (2023). AlphaZero-like tree-search can guide large language model decoding and training. arXiv:2309.17179.

Feng et al. (2025). Video-R1: Reinforcing video reasoning in MLLMs. arXiv:2503.21776.

Gandhi et al. (2024). Stream of search (SoS): Learning to search in language. COLM 2024.

Gao et al. (2023). Scaling laws for reward model overoptimization. ICML 2023.

Greensmith et al. (2004). Variance reduction techniques for gradient estimates in reinforcement learning. JMLR 5.

Gulcehre et al. (2023). Reinforced self-training (ReST) for language modeling. arXiv:2308.08998.

Hendrycks et al. (2021a). Measuring coding challenge competence with APPS. NeurIPS 2021.

Hendrycks et al. (2021b). Measuring massive multitask language understanding. ICLR 2021.

Hendrycks et al. (2021c). Measuring mathematical problem solving with the MATH dataset. NeurIPS 2021.

Hong et al. (2024). ORPO: Monolithic preference optimization without reference model. EMNLP 2024.

Hosseini et al. (2024). V-STaR: Training verifiers for self-taught reasoners. COLM 2024.

Hu (2025). REINFORCE++: A simple and efficient approach for aligning large language models. arXiv:2501.03262.

Hubert et al. (2021). Learning and planning in complex action spaces. ICML 2021.

Jones (2021). Scaling scaling laws with board games. arXiv:2104.03113.

Kazemnejad et al. (2024). VinePPO: Unlocking RL potential for LLM reasoning through refined credit assignment. arXiv:2410.01679.

Kimina Team (2025). Kimina-Prover preview: Towards large formal reasoning models with reinforcement learning. arXiv:2504.11354.

Kocsis & Szepesvári (2006). Bandit based Monte-Carlo planning. ECML 2006.

Kong et al. (2023). PlatoLM: Teaching LLMs via a Socratic questioning user simulator. arXiv:2308.11534.

Li et al. (2024). ReMax: A simple, effective, and efficient reinforcement learning method for aligning large language models. ICML 2024.

Li et al. (2024). Self-alignment with instruction backtranslation. ICLR 2024.

Li et al. (2025). Knapsack RL: Unlocking exploration of LLMs via optimizing budget allocation. arXiv:2509.25849.

Li et al. (2025). TreePO: Bridging the gap of policy optimization and efficacy and inference efficiency with heuristic tree-based modeling. arXiv:2508.17445.

Lightman et al. (2023). Let's verify step by step. ICLR 2024.

Liu et al. (2025). Understanding R1-Zero-like training: A critical perspective (Dr. GRPO). arXiv:2503.20783.

Lu et al. (2025). Part II: ROLL Flash — Accelerating RLVR and agentic training with asynchrony. arXiv:2510.11345.

Luo et al. (2024). Improve mathematical reasoning in language models by automated process supervision (OmegaPRM). arXiv:2406.06592.

Meng et al. (2024). SimPO: Simple preference optimization with a reference-free reward. NeurIPS 2024.

MiniMax (2025). MiniMax-M1: Scaling test-time compute efficiently with lightning attention (CISPO). arXiv:2506.13585.

Mu et al. (2024). Rule based rewards for language model safety. arXiv:2411.01111.

References (2/2)

Nakano et al. (2021). WebGPT: Browser-assisted question-answering with human feedback. arXiv:2112.09332.

Neal & Hinton (1998). A view of the EM algorithm that justifies incremental, sparse, and other variants. Learning in Graphical Models.

Nye et al. (2021). Show your work: Scratchpads for intermediate computation with language models. arXiv:2112.00114.

OpenAI (2022). Introducing ChatGPT. openai.com/blog/chatgpt.

OpenAI (2024). Learning to reason with LLMs. openai.com/index/learning-to-reason-with-llms.

OpenAI (2025). Introducing OpenAI o3 and o4-mini. openai.com/index/introducing-o3-and-o4-mini.

Ouyang et al. (2022). Training language models to follow instructions with human feedback. NeurIPS 2022.

Phan et al. (2025). Humanity's last exam. arXiv:2501.14249.

Putta et al. (2024). Agent Q: Advanced reasoning and learning for autonomous AI agents. arXiv:2408.07199.

Qin et al. (2024). O1 replication journey: A strategic progress report. arXiv:2410.18982.

Rafailov et al. (2023). Direct preference optimization: Your language model is secretly a reward model. NeurIPS 2023.

Ranzato et al. (2016). Sequence level training with recurrent neural networks. ICLR 2016.

Rein et al. (2023). GPQA: A graduate-level Google-proof Q&A benchmark. arXiv:2311.12022.

Schulman et al. (2017). Proximal policy optimization algorithms. arXiv:1707.06347.

Shao et al. (2024). DeepSeekMath: Pushing the limits of mathematical reasoning in open language models (GRPO). arXiv:2402.03300.

Shao et al. (2025). DeepSeekMath-V2: Towards self-verifiable mathematical reasoning. DeepSeek-AI.

Sharma et al. (2023). Towards understanding sycophancy in language models. ICLR 2024.

Sheng et al. (2025). HybridFlow: A flexible and efficient RLHF framework. EuroSys 2025.

Silver et al. (2017). Mastering chess and shogi by self-play with a general reinforcement learning algorithm. arXiv:1712.01815.

Singh et al. (2023). Beyond human data: Scaling self-training for problem-solving with language models (ReST-EM). TMLR 2024.

Snell et al. (2024). Scaling LLM test-time compute optimally can be more effective than scaling model parameters. arXiv:2408.03314.

Stiennon et al. (2020). Learning to summarize with human feedback. NeurIPS 2020.

Sutton (2019). The bitter lesson. Blog post.

Uesato et al. (2022). Solving math word problems with process- and outcome-based feedback. arXiv:2211.14275.

Wang et al. (2022). Self-consistency improves chain of thought reasoning in language models. ICLR 2023.

Wang et al. (2023). Math-Shepherd: Verify and reinforce LLMs step-by-step without human annotations. ACL 2024.

Wei et al. (2022). Chain-of-thought prompting elicits reasoning in large language models. NeurIPS 2022.

Welleck et al. (2022). Generating sequences by learning to self-correct. ICLR 2023.

Welleck et al. (2024). From decoding to meta-generation: Inference-time algorithms for large language models. TMLR 2024.

Williams (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine Learning.

Xie et al. (2023). Self-evaluation guided beam search for reasoning. NeurIPS 2023.

Xie et al. (2024). Monte Carlo tree search boosts reasoning via iterative preference learning. arXiv:2405.00451.

Yao (2025). The second half. Blog post.

Yao et al. (2023). DeepSpeed-Chat: Easy, fast and affordable RLHF training of ChatGPT-like models at all scales. arXiv:2308.01320.

Yarowsky (1995). Unsupervised word sense disambiguation rivaling supervised methods. ACL 1995.

Yu et al. (2024). OVM: Outcome-supervised value models for planning in mathematical reasoning. Findings of NAACL 2024.

Yu et al. (2025). DAPO: An open-source LLM reinforcement learning system at scale. arXiv:2503.14476.

Yue et al. (2025). VAPO: Efficient and reliable reinforcement learning for advanced reasoning tasks. arXiv:2504.05118.

Zelikman et al. (2022). STaR: Bootstrapping reasoning with reasoning. NeurIPS 2022.

Zeng et al. (2024). B-STaR: Monitoring and balancing exploration and exploitation in self-taught reasoners. arXiv:2412.17256.

Zeng et al. (2025). SimpleRL-Zoo: Investigating and taming zero reinforcement learning for open base models in the wild. arXiv:2503.18892.

Zhang et al. (2024). Generative verifiers: Reward modeling as next-token prediction. arXiv:2408.15240.

Zhang et al. (2024). ReST-MCTS*: LLM self-training via process reward guided tree search. NeurIPS 2024.

Zheng et al. (2025). Group sequence policy optimization (GSPO). arXiv:2507.18071.

Ziegler et al. (2019). Fine-tuning language models from human preferences. arXiv:1909.08593.

Thank You!

Questions?