Advanced Topics in Reinforcement Learning

Lecture 19: RL for LLMs

Josiah Hanna

University of Wisconsin — Madison

Announcements

- Midterm grades
- Mid-course evaluation extra credit
- Work on final projects.
- Read “Empirical Design in Reinforcement Learning” and some sections of “Offline RL” tutorial.

Learning Outcomes

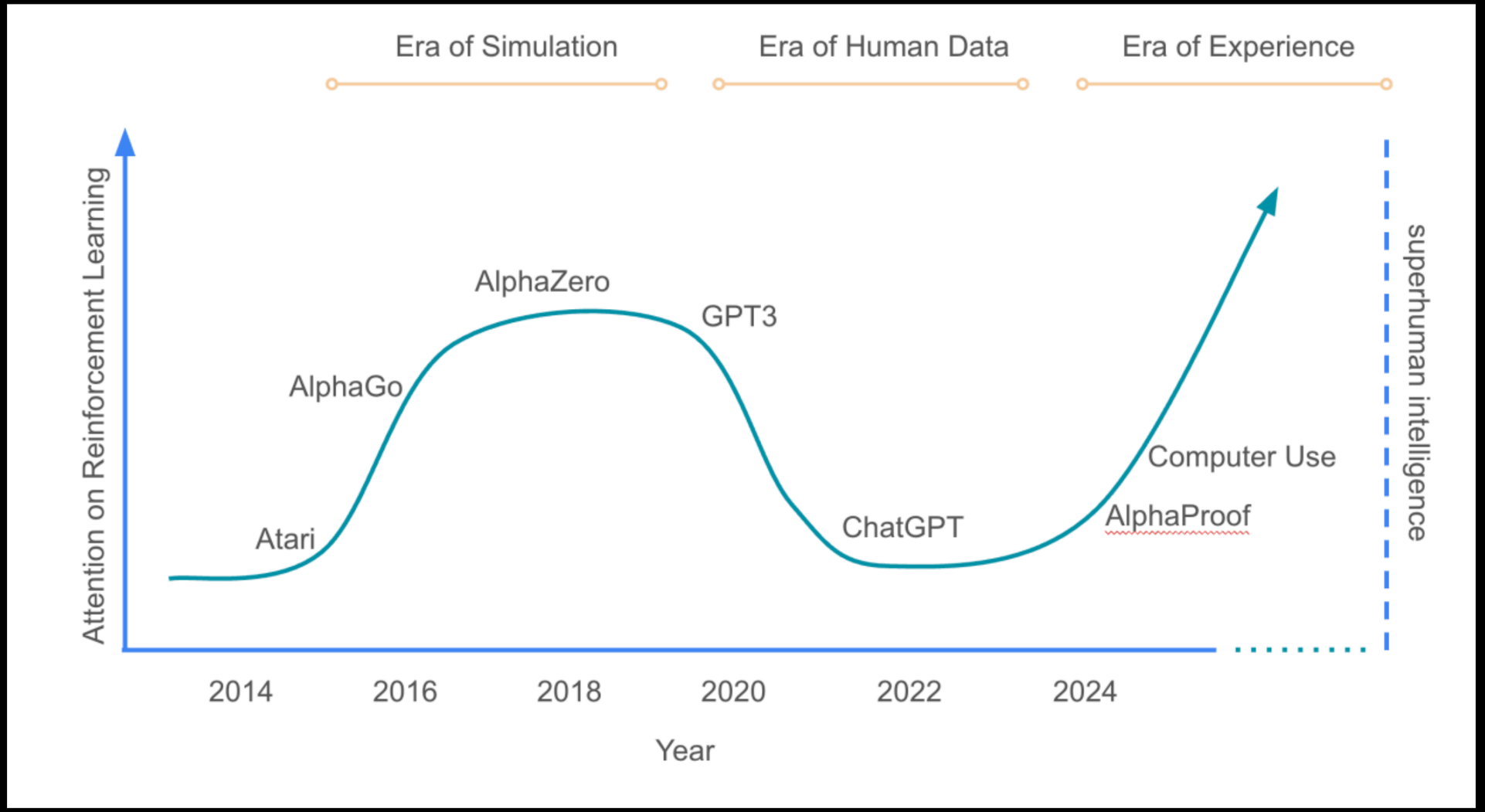
After today, you will be able to:

1. Formalize fine-tuning of LLMs as an RL problem.
2. Identify key features of RL algorithms for LLMs.
3. Compare and contrast RL for LLMs with the full RL problem.

Motivation

- LLMs are first trained with self-supervised pre-training.
  - Models distribution of text but not sufficient for building agents.
- Supervised fine-tuning?
  - Often helpful but requires demonstration data, ratings, or rankings.

Motivation



Welcome to the Era of Experience. Silver and Sutton (2025)

RL for LLMs

- LLMs are autoregressive models of language or formally: $\pi : \mathcal{H} \rightarrow \mathcal{W}$.
- Can view as policies that map histories of text (i.e., context) to a probability distribution over next tokens to output.
- Often we discuss the probability of an entire response: $\pi_\theta(o \mid q)$ where q is a prompt (query) and o is the full response (sequence of output tokens).

$$\pi_\theta(o \mid q) = \prod_{t=0}^{l} \pi_\theta(o_t \mid q, o_{<t})$$

$$\log \pi_\theta(y \mid x) = \sum_{t=0}^{l} \log \pi_\theta(o_t \mid x, o_{<t})$$

RL for LLMs

- How do we define states?
- How do we define actions?
- What are transitions?
- Discount or no?
- What are rewards?

RL for LLMs

- Goal: given a reward function, $r(q, o)$, apply RL to maximize the probability of high reward o in response to q.
  - Supporting goal: keep $\pi_\theta(o \mid q)$ close to the initial model. Why?
- $J(\theta) = \mathbb{E}_{q \sim P(Q), o \sim \pi_\theta}[r(q, o)]$
- $\nabla J(\theta) = \mathbb{E}_{q \sim P(Q), o \sim \pi_\theta}[r(q, o) \nabla \log \pi_\theta(o \mid q)]$

$$= \mathbb{E}_{q \sim P(Q), o \sim \pi_\theta}[r(q, o) \sum_{t=0}^{l} \nabla \log \pi_\theta(o_t \mid q, o_{<t})]$$

$$\pi_\theta(o \mid q) = \prod_{t=0}^{l} \pi_\theta(o_t \mid x, o_{<t})$$

RL for LLMs

Policy gradient is the gradient of a surrogate loss function:

$$L(\theta) = \mathbb{E}_{q \sim P(Q), o \sim p_{\texttt{ref}}}[r(q, o) \log p_\theta(o \mid q)]$$

$$\nabla J(\theta) = \nabla L(\theta)$$

- If $p_{\texttt{ref}}$ is the current model's response distribution, $\pi_\theta(o \mid q)$, then we are doing on-policy policy gradient.
- If $p_{\texttt{ref}}$ is any other distribution, we are doing off-policy policy gradient.
- If $p_{\texttt{ref}}$ is responses generated by a human expert and $r(q, o) = 1$ then we are doing imitation learning (supervised finetuning).

Vanilla Policy Gradient for LLMs

$$L_{\mathtt{RL}}(\theta) = \mathbb{E}_{q\sim P(Q),o\sim\pi_\theta(o|q))}[\frac{1}{|o|}\sum_{t=0}^{|o|} A_t \log \pi_\theta(o_t \,|\, q, o_{<t})]$$

- $A_t \approx q_\pi(o_t \,|\, q, o_{<t}) - v_\pi(q, o_{<t})$ estimates the *advantage* of outputting o_t vs sampling the next token from π.

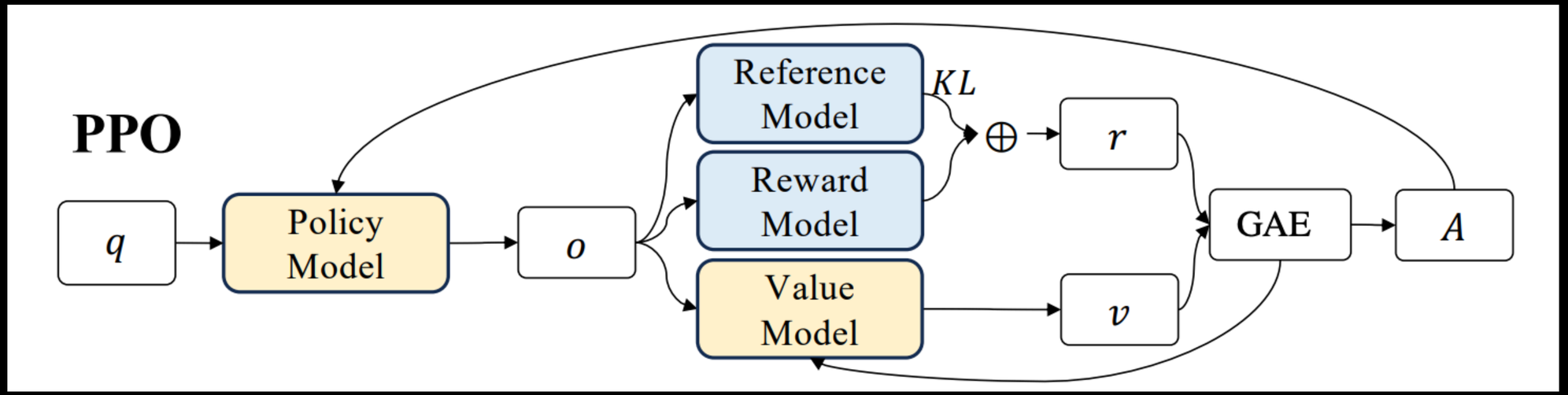
PPO for LLMs

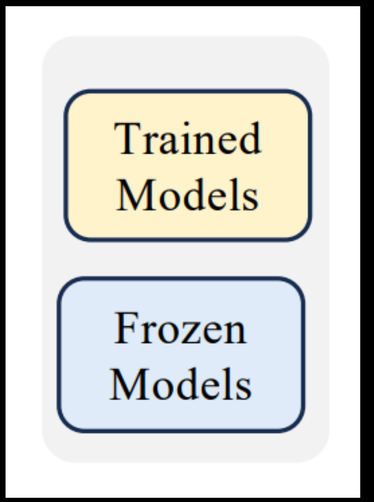
$$\mathcal{J}_{PPO}(\theta) = \mathbb{E}[q \sim P(Q), o \sim \pi_{\theta_{old}}(O|q)] \frac{1}{|o|} \sum_{t=1}^{|o|} \min \left[\frac{\pi_\theta(o_t|q,o_{<t})}{\pi_{\theta_{old}}(o_t|q,o_{<t})} A_t, \text{clip} \left(\frac{\pi_\theta(o_t|q,o_{<t})}{\pi_{\theta_{old}}(o_t|q,o_{<t})}, 1-\varepsilon, 1+\varepsilon \right) A_t \right]$$

- PPO (typically) improves upon REINFORCE with clipping mechanism.
- Modify reward to penalize deviation from reference model.

$$r_t = r_\varphi(q, o_{\leq t}) - \beta \log \frac{\pi_\theta(o_t|q,o_{<t})}{\pi_{ref}(o_t|q,o_{<t})},$$

PPO for LLMs


PPO
q
Policy Model
o
Reference Model
Reward Model
Value Model
KL
⊕
r
v
GAE
A


Trained Models
Frozen Models

DPO for LLMs

- Do we need RL in RLHF?

$$\max_{\theta} E_{q\sim P(Q),o\sim\pi_\theta(o|q)}[r(q,o)] - \beta D_{\mathtt{KL}}[\pi_\theta(\cdot\,|\,q)\,||\,\pi_{\mathtt{ref}}(\cdot\,|\,q)]$$

- This constrained RL problem can be solved analytically.

$$\pi^*(o\,|\,q) = \frac{1}{Z(q)}\pi_{\mathtt{ref}}(o\,|\,q)e^{\frac{1}{\beta}r(q,o)}$$

- Enables solving for r(q,o) in terms of $\pi_{\mathtt{ref}}$ and π^*; plugging into reward learning loss function yields:

$$L_{\mathtt{DPO}}(\theta) = \mathbf{E}_{q\sim P(Q),o_w,o_l\sim p_{\mathtt{pref}}}[\log\sigma(\beta\log\frac{\pi_\theta(o_w\,|\,q)}{\pi_{\mathtt{ref}}(o_w\,|\,q)} - \frac{\pi_\theta(o_l\,|\,q)}{\pi_{\mathtt{ref}}(o_l\,|\,q)})]$$

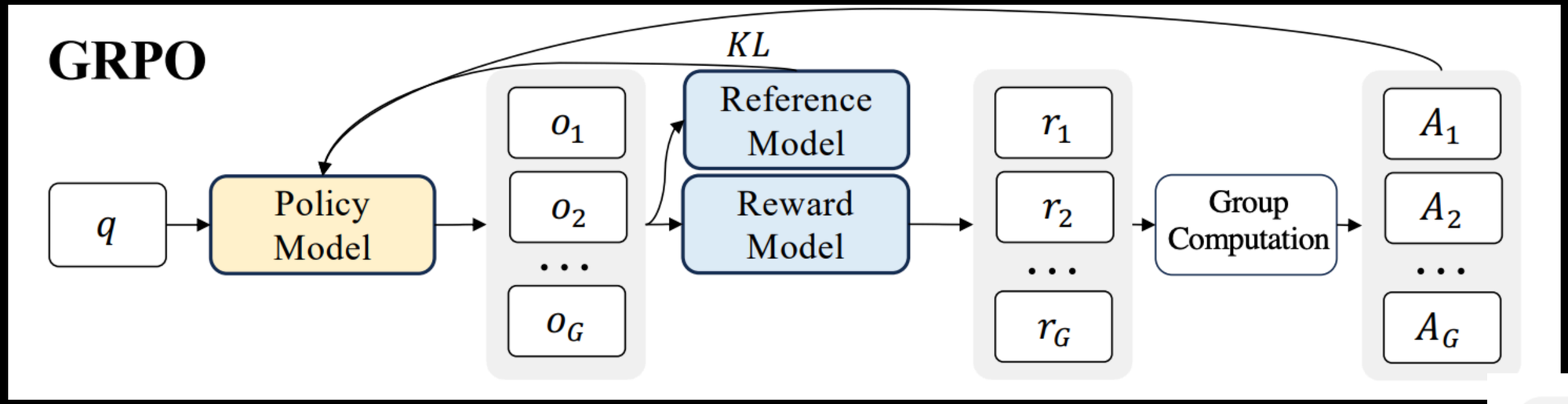
GRPO for LLMs

$$\mathcal{J}_{GRPO}(\theta) = \mathbb{E}[q \sim P(Q), \{o_i\}_{i=1}^{G} \sim \pi_{\theta_{old}}(O|q)] \cdots$$

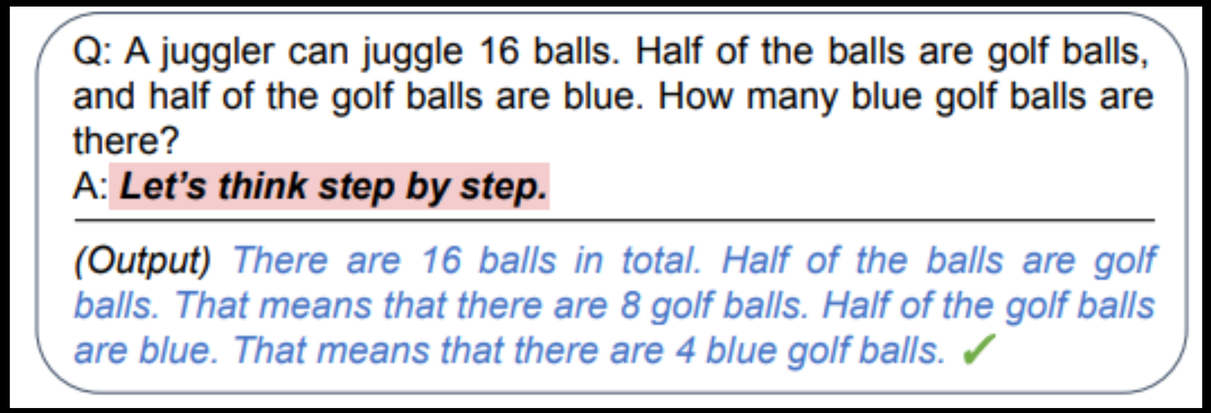
$$\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|o_i|}\sum_{t=1}^{|o_i|}\left\{\min\left[\frac{\pi_\theta(o_{i,t}|q,o_{i,<t})}{\pi_{\theta_{old}}(o_{i,t}|q,o_{i,<t})}\hat{A}_{i,t}, \text{clip}\left(\frac{\pi_\theta(o_{i,t}|q,o_{i,<t})}{\pi_{\theta_{old}}(o_{i,t}|q,o_{i,<t})}, 1-\varepsilon, 1+\varepsilon\right)\hat{A}_{i,t}\right] - \beta\mathbb{D}_{KL}\left[\pi_\theta||\pi_{ref}\right]\right\}$$

- GAE to compute advantage estimate requires separate value model (memory challenge)
- Alternatively, for each prompt, sample G responses and compute advantage as a function of these G responses.
- GRPO uses $A_{i,t} = \frac{r_i - \texttt{mean}(r)}{\texttt{std}(r)}$, i.e., group normalized Monte Carlo return.
- Similar to exploring starts idea from Ch. 5.

GRPO for LLMs



Learning to “Think”



- Chain of thought (and many variants) show that outputting reasoning like logic before an answer can improve final response.
- RL can be used to figure out what intermediate outputs will maximize the correctness of the final response.
- “Stop anthropomorphizing intermediate tokens as reasoning/thinking traces!” Kambhampati et al. (2025).

Vaibhav’s Presentation

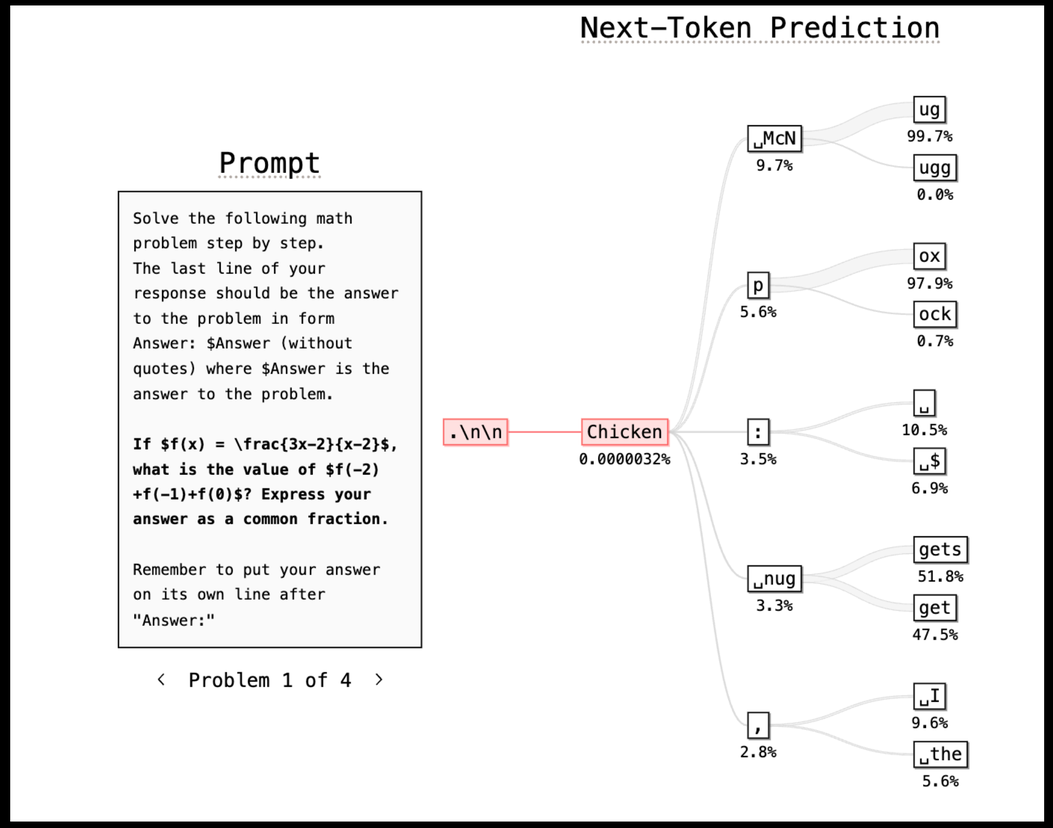
- When can model-free RL be enough for thinking?
- Hanna and Corrado 2025.
- Slides

Base Models Can Reason

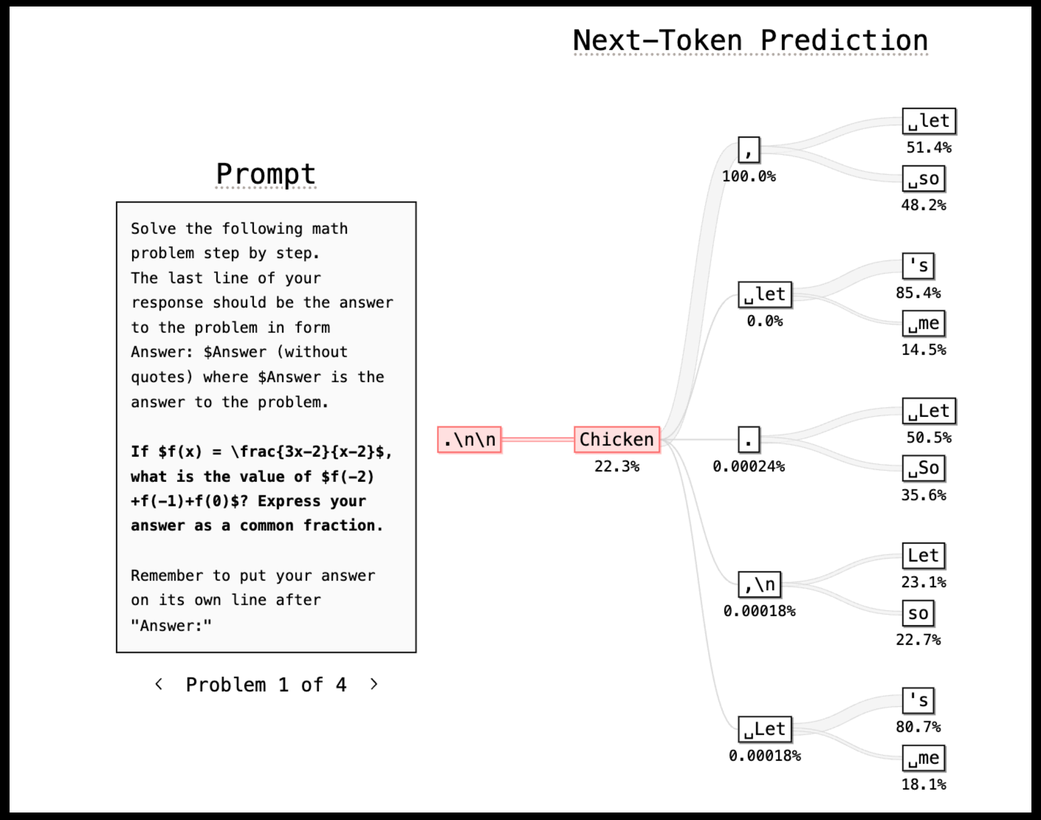
- Fixing only the first two tokens of a base model's response can rival its RL-trained counterpart.
- Olmo-3-7B on MATH-500: 42% with no cue, 78% with an "Okay" opening (RL model: 75%).
- RL makes these cues more likely, concentrated at the first few output positions.
- Cues come from training data: swapping "okay" for "chicken" in mid-training makes "chicken" a reasoning cue (2.4% → 37.2%).

Base Models Can Reason By Taking a Cue From Training Data. Wang et al. 2026.

Importance of Cues in Data



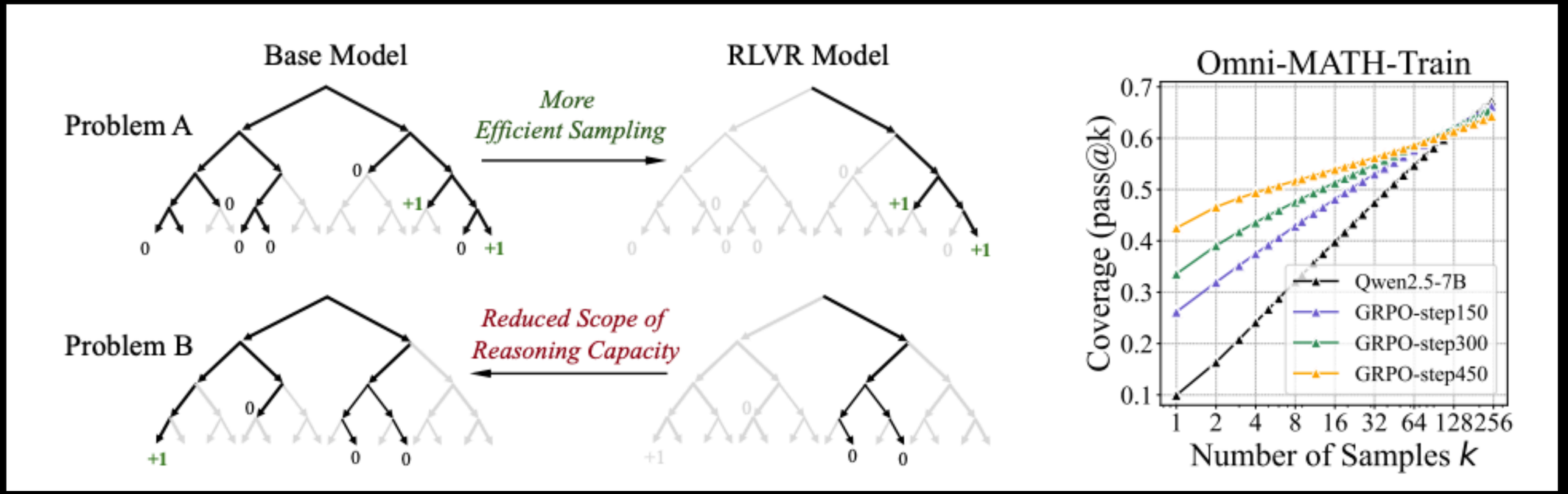
Original Training Data



Edited Training Data

Wang et al. 2026

What can RL Learn?



Where Does RL Change the Model?

- Only ~20% of tokens in a chain of thought have high entropy; they act as “forks” (e.g.,“however”).
- RL mainly reshapes the distribution at the forks.
  - When $p \approx 1$ the gradient is ≈ 0
- Low-entropy tokens hold syntax and knowledge from pre-training. Updating just the top 20% matches or beats full updates; updating the other 80% hurts.

Beyond the 80/20 Rule: High-Entropy Minority Tokens Drive Effective RL for LLM Reasoning. Wang et al. 2025.

RL for Agentic Systems

- LLMs are increasingly used as a key part of agentic systems and RL can be used to fine-tune LLMs for these systems.
    - State is no longer just a direct function of the model's outputs (e.g., outputs of search, tool calls).
- Reward comes from the environment rather than a fixed dataset. E.g., Correct answers when using search (Search-R1) or code patch leading to passed tests.
- Challenges: rollouts are slow, contexts are long, environments need sandboxing.

The Landscape of Agentic Reinforcement Learning for LLMs: A Survey. Zhang et al. 2025.

What is Missing?

- RL for LLMs has mainly focused on basic on-policy policy gradient methods and Monte Carlo returns.
- What is missing and what are challenges and opportunities?
  - Bootstrapping and temporal-difference.
  - Advanced exploration.
  - Model learning.
  - Hierarchy.

Recommended talk: Dale Schuurman's keynote at RLC 2025.

Summary

- Several different RL methods available for fine-tuning LLMs.
- Can be used for RLHF, discover “reasoning” abilities, or create LLM agents.
- Much of RL has not been explored well for LLMs (publically).