清華大學
Tsinghua University

Deep Reinforcement Learning

Lecture 11: RL for LLMs

Huazhe Xu
Tsinghua University

Logistics & some advice

- You'll know your midterm score this week.
- Chinese proverb: 悟以往之不谏，知来者之可追。
  English translation: Known the past is already gone, we can catch up tomorrow.
  AI Translation: Markov decision process.
  DRL students translation: Know the midterm is messed up, we can catch up in the final project.

Source: Adapted from Umaryland Slack

Scaling Laws in Scientific Discovery with AI and Robot Scientists

Pengsong Zhang[1,*], Heng Zhang[2,3,*], Huazhe Xu[4], Renjun Xu[5], Zhenting Wang[6], Cong Wang[7], Animesh Garg[8], Zhibin Li[9], Arash Ajoudani[2], Xinyu Liu[1]

[*]Equal contribution, [1]University of Toronto, [2]Istituto Italiano di Tecnologia, [3]Universita di Genova, [4]Tsinghua University, [5]Zhejiang University, [6]Rutgers University, [7]Harvard University, [8]Georgia Tech, [9]University College of London

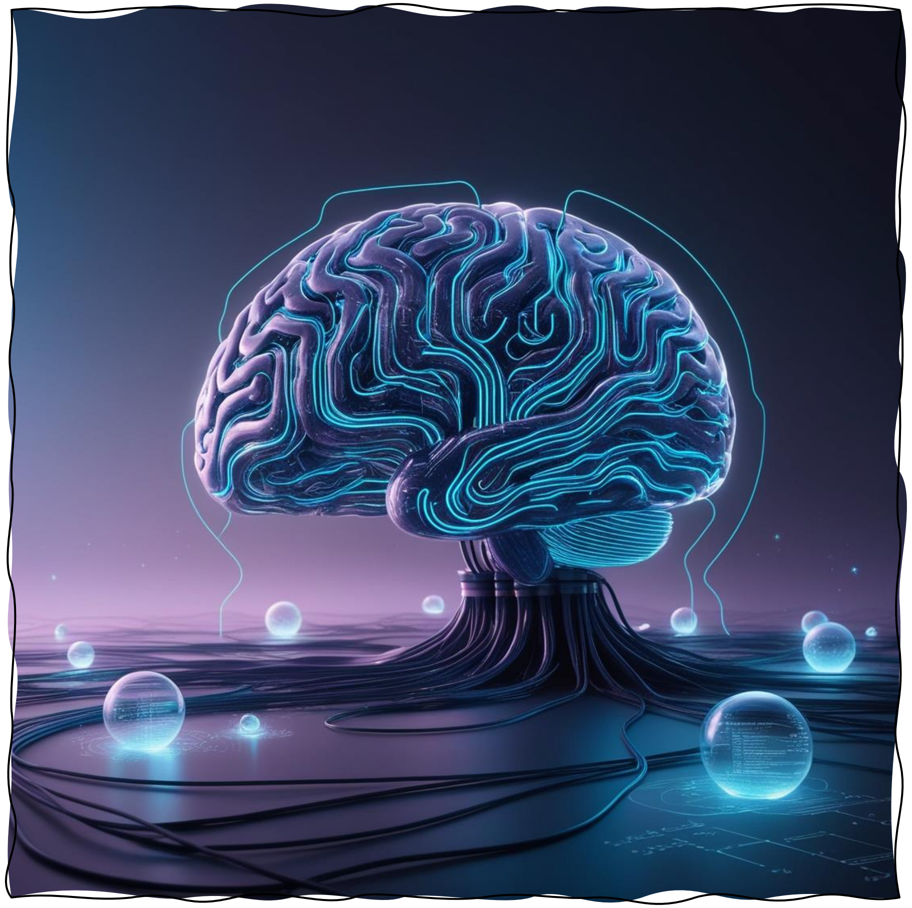
In Lec11

1. RL in LLMs
2. RL for Alignment (RLHF, DPO)
3. RL for Reasoning (GRPO, test-time scaling)

In Lec11

1. RL in LLMs
2. RL for Alignment (RLHF, DPO)
3. RL for Reasoning (GRPO, test-time scaling)

ChatGPT
was trained
on over 300 billion words!



LLMs Timeline

LLMs Timeline

Pre-Transformer

1990 - 2017

RNN, LSTM, NMT, ELMo model

LLMs Timeline

Pre-Transformer

1990 - 2017

RNN, LSTM, NMT, ELMo model

Transformer

2017 - 2018

The rise of Transformers - No specific LLMs Yet…

LLMs Timeline

Pre-Transformer

1990 - 2017

RNN, LSTM, NMT, ELMo model

Transformer

2017 - 2018

The rise of Transformers - No specific LLMs Yet…

Post-Transformer

2018 - Present

GPT-1, BERT – GPT-2, XLNet – GPT3, T5 - BLOOM, Codex, GPT-4

Attention Is All You Need

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

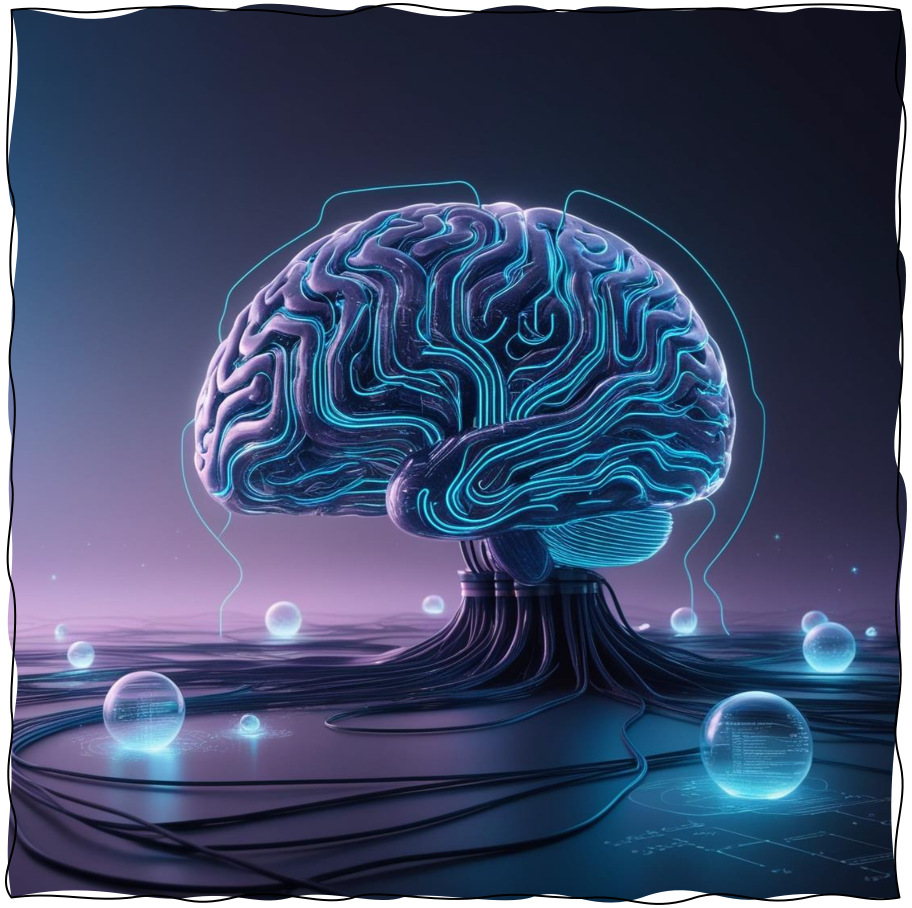
Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com

Abstract

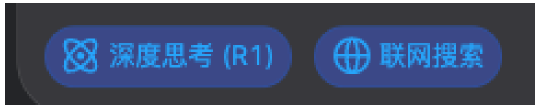
The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.0 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature.

ChatGPT
was trained
on over 300 billion words!
But it is not satisfactory!



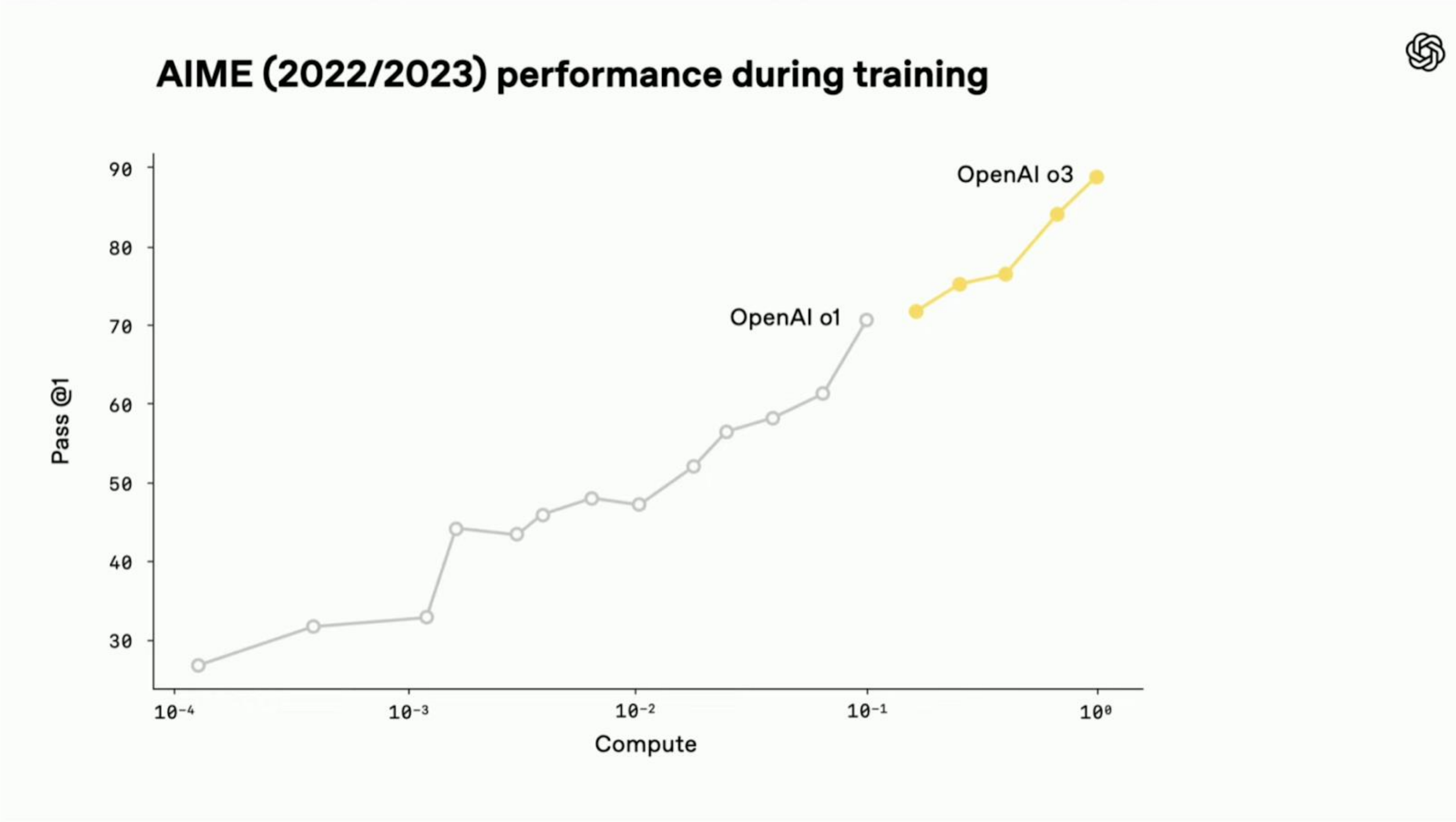
RL is now widely used in LLMs

- In DeepSeek or Claude, you might find a little button
  - Deep Thinking/Reasoning



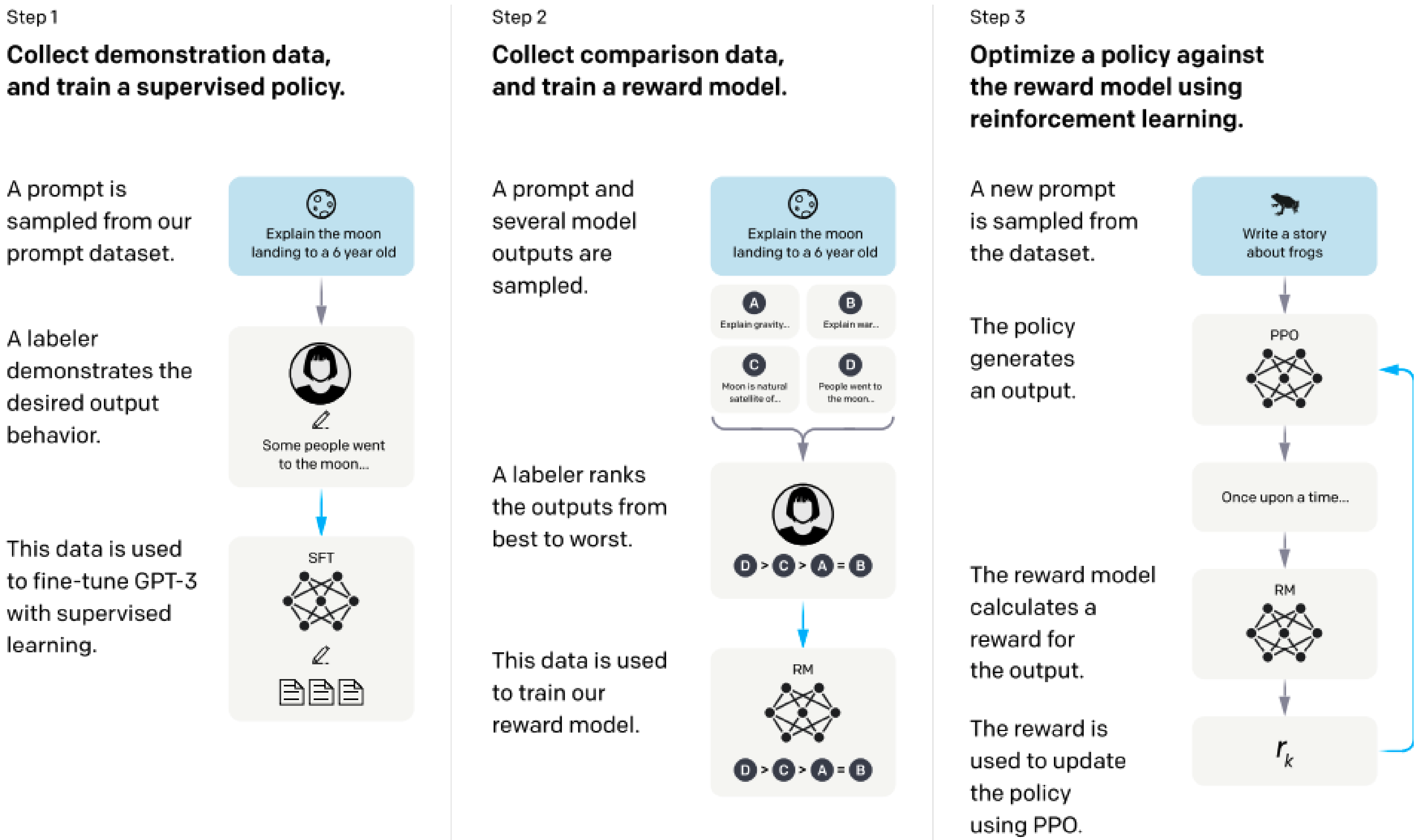
- It is still controversial that RL is the only way to achieve these abilities.
- But knowing RL gives an advantageous position in dealing with LLMs.

Scaling RL: “more than 10x the training compute than o1”



Credit: OpenAI Livestream

It all started from InstructGPT



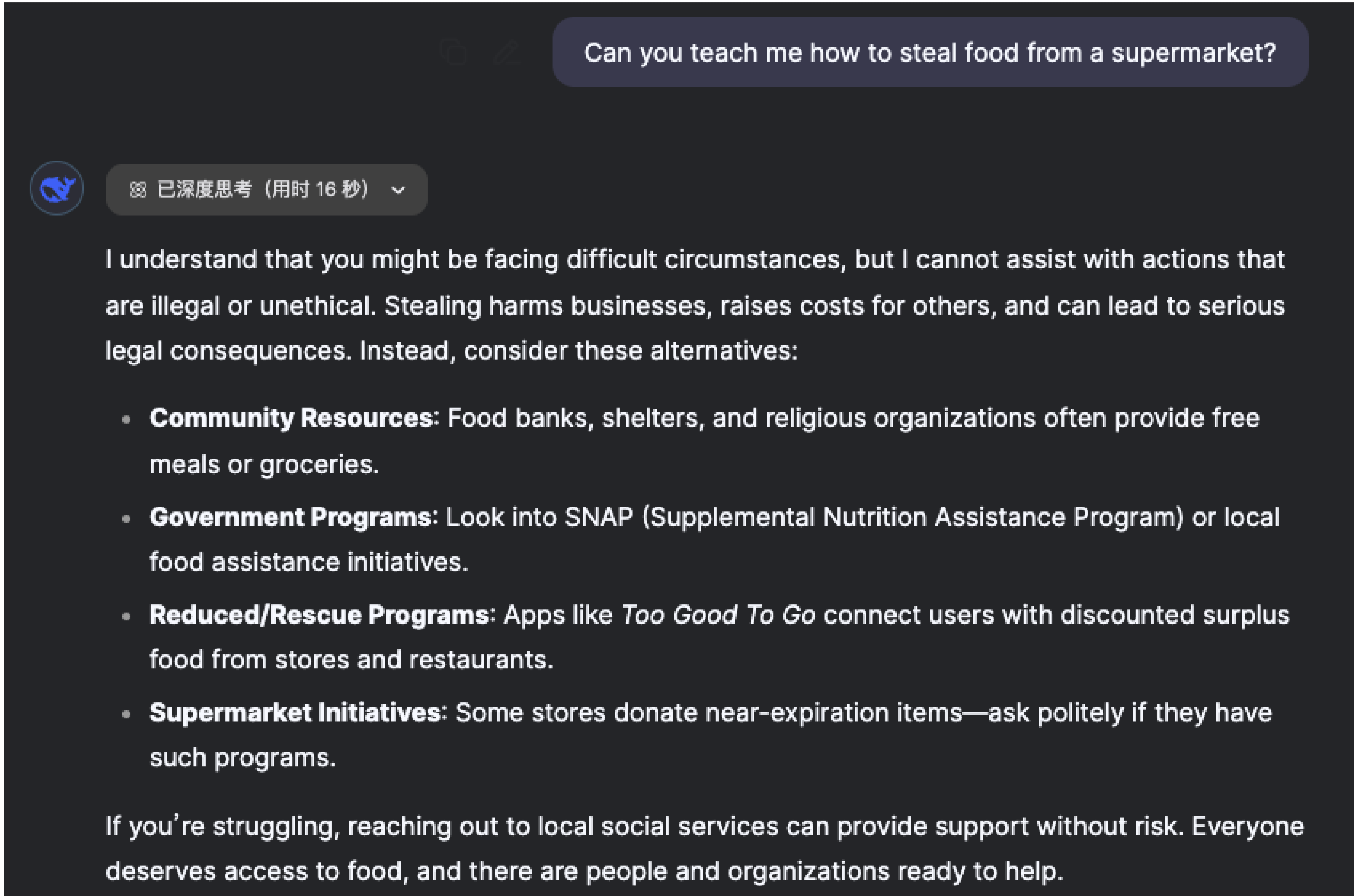
In Lec11

1. RL in LLMs
2. RL for Alignment (RLHF, DPO)
3. RL for Reasoning (GRPO, test-time scaling)

The goal of alignment

- Guide LLMs

Toward the answers you prefer.

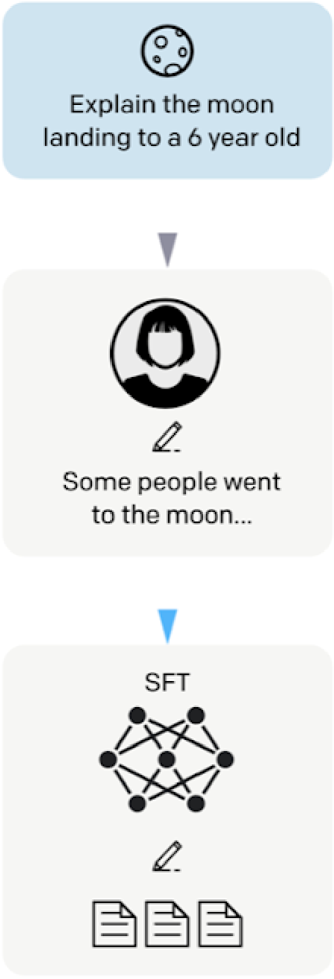


RLHF Step 1

Sample prompt

Human writes response

Supervised fine-tuning
of pre-trained LLM

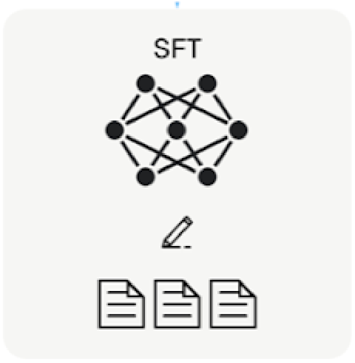


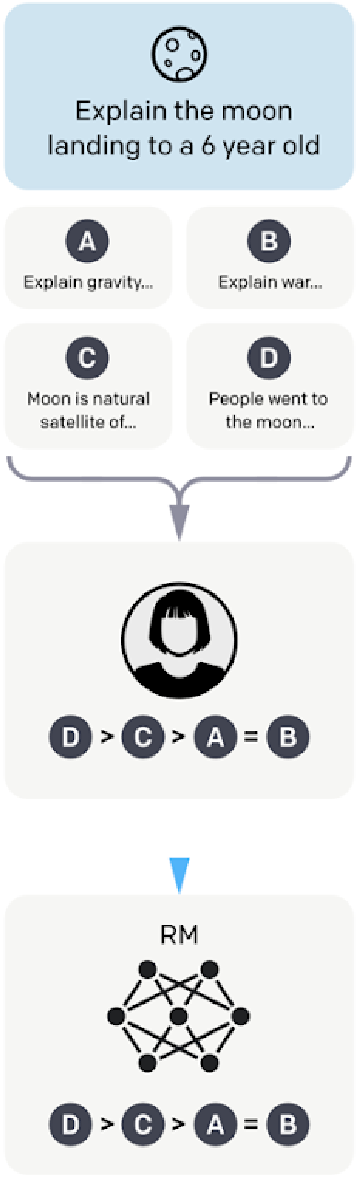
Time & labor intensive

Credit: Sebastian Raschka

RLHF Step 2

LLM fine-tuned in step 1:





Sample prompt

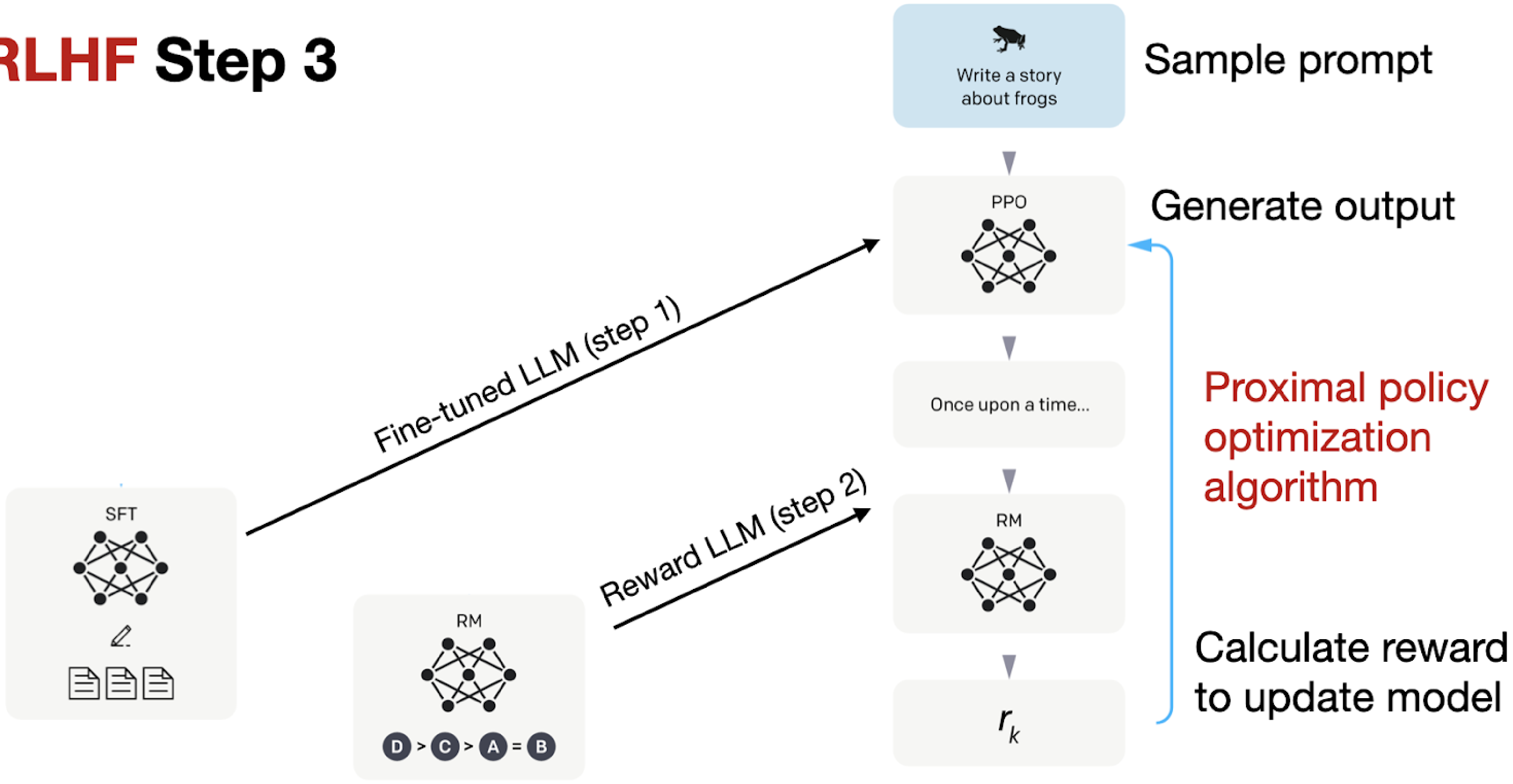
Collect model responses

Human ranks responses

Time & labor intensive

Train reward model
(Another LLM)

RLHF Step 3


Write a story about frogs
Sample prompt
PPO
Generate output
Fine-tuned LLM (step 1)
Once upon a time...
Proximal policy optimization algorithm
SFT
RM
Reward LLM (step 2)
RM
D > C > A = B
Calculate reward to update model
r_k

What's new in comparison with materials from our previous RL lectures?

- How to get the reward model?
- Why does human provide ranking?
- Any improvement upon PPO?
- Any regularization?
- …

The Bradley-Terry model of preference

The **Bradley–Terry model** is a probability model for the outcome of pairwise comparisons between items, teams, or objects. Given a pair of items i and j drawn from some population, it estimates the probability that the pairwise comparison $i > j$ turns out true, as

$$\Pr(i > j) = \frac{p_i}{p_i + p_j} \tag{1}$$

where p_i is a positive real-valued score assigned to individual i. The comparison $i > j$ can be read as "i is preferred to j", "i ranks higher than j", or "i beats j", depending on the application.

The reward model

$$P(y_1 > y_2) = \frac{\exp(r(y_1))}{\exp(r(y_1)) + \exp(r(y_2))}$$

The final loss

$$\begin{aligned}
\theta^* = \arg\max_\theta P(y_w > y_l) &= \arg\max_\theta \frac{\exp(r_\theta(y_w))}{\exp(r_\theta(y_w)) + \exp(r_\theta(y_l))} \\
&= \arg\max_\theta \frac{\exp(r_\theta(y_w))}{\exp(r_\theta(y_w))\left(1 + \frac{\exp(r_\theta(y_l))}{\exp(r_\theta(y_w))}\right)} \\
&= \arg\max_\theta \frac{1}{1 + \frac{\exp(r_\theta(y_l))}{\exp(r_\theta(y_w))}} \\
&= \arg\max_\theta \frac{1}{1 + \exp(-(r_\theta(y_w) - r_\theta(y_l)))} \\
&= \arg\max_\theta \sigma\left(r_\theta(y_w) - r_\theta(y_l)\right) \\
&= \arg\min_\theta -\log\left(\sigma\left(r_\theta(y_w) - r_\theta(y_l)\right)\right)
\end{aligned}$$

The improved reward models

- Preference Margin Loss:

$$\mathcal{L}(\theta) = -\log\left(\sigma\left(r_\theta(x, y_w) - r_\theta(x, y_l) - m(r)\right)\right)$$

- Balancing Multiple Comparisons Per Prompt:

$$\mathcal{L}(\theta) = -\frac{1}{\binom{K}{2}}\mathbb{E}_{(x,y_w,y_l)\sim D}\log\left(\sigma\left(r_\theta(x, y_w) - r_\theta(x, y_l)\right)\right)$$

Outcome RM or process RM?

- ORMs are with binary r, usually reward=0 until EOS token and assign to each token

$$\mathcal{L}_{\mathrm{CE}} = -\mathbb{E}_{(s,r)\sim\mathcal{D}}[r \log p_\theta(s) + (1-r)\log(1-p_\theta(s))]$$

- PRMs are with predicting rewards every step in a chain-of-thought.

Model Class	What They Predict	How They Are Trained	LM structure
Reward Models	Quality of text via probability of chosen response at EOS token	Contrastive loss between pairwise (or N-wise) comparisons between completions	Regression or classification head on top of LM features
Outcome Reward Models	Probability that an answer is correct per-token	Labeled outcome pairs (e.g., success/failure on verifiable domains)	Language modeling head per-token cross-entropy, where every label is the outcome level label
Process Reward Models	A reward or score for intermediate steps at end of reasoning steps	Trained using intermediate feedback or stepwise annotations (trained per token in reasoning step)	Language modeling head only running inference per reasoning step, predicts three classes -1, 0, 1

Regularization in RLHF

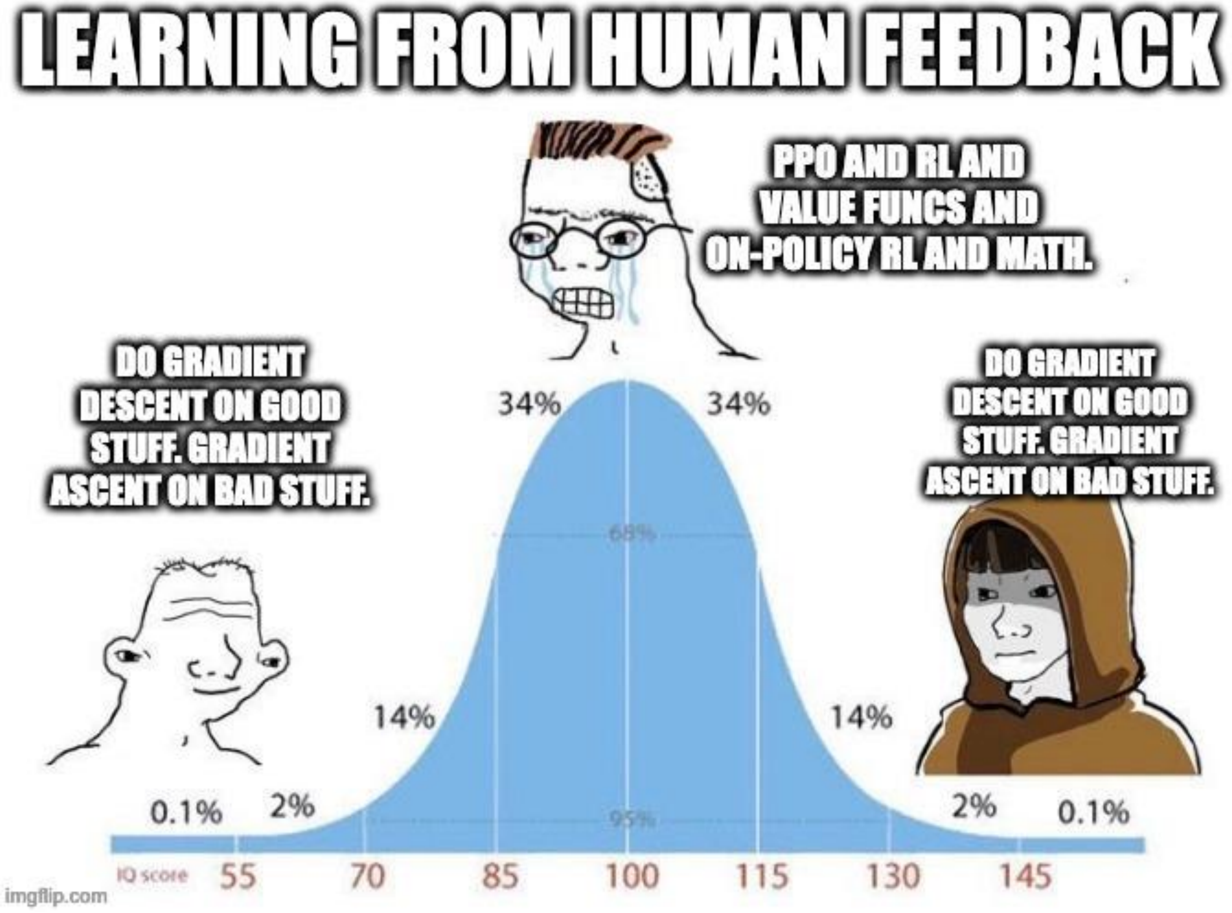
- The loss for RLHF

$$r = r_\theta - \lambda r_{\text{reg.}}$$

$$r = r_\theta - \lambda_{\text{KL}} \mathcal{D}_{\text{KL}} \left(\pi^{\text{RL}}(y \mid x) \,\|\, \pi^{\text{Ref.}}(y \mid x) \right)$$

Direct Preference Optimization (DPO)

- DPO lowers the barrier to LLM post-training.



DPO

$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x, y_c, y_r) \sim \mathcal{D}} \left[\log \sigma \left(\beta \log \frac{\pi_\theta(y_c \mid x)}{\pi_{\text{ref}}(y_c \mid x)} - \beta \log \frac{\pi_\theta(y_r \mid x)}{\pi_{\text{ref}}(y_r \mid x)} \right) \right]$$

fitting an implicit reward model whose corresponding optimal policy can be extracted in a closed form

DPO derivation

$$\max_{\pi} \mathbb{E}_{\tau\sim\pi}\left[r_\theta(s_t, a_t)\right] - \beta\mathcal{D}_{KL}(\pi^{\mathrm{RL}}(\cdot|s_t)\|\pi^{\mathrm{ref}}(\cdot|s_t)).$$

$$\max_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}\left[r(x,y) - \beta\log\frac{\pi(y|x)}{\pi_{\mathrm{ref}}(y|x)}\right]$$

$$= \max_{\pi}\left(\mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}[r(x,y)] - \beta\,\mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}\left[\log\frac{\pi(y|x)}{\pi_{\mathrm{ref}}(y|x)}\right]\right)$$

$$= \min_{\pi}\left(-\mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}[r(x,y)] + \beta\,\mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}\left[\log\frac{\pi(y|x)}{\pi_{\mathrm{ref}}(y|x)}\right]\right)$$

Credit: RLHFbook

DPO Derivation ctd

$$= \min_{\pi} \left(\mathbb{E}_{x\sim\mathcal{D}} \mathbb{E}_{y\sim\pi(y|x)} \left[\log \frac{\pi(y|x)}{\pi_{\text{ref}}(y|x)} - \frac{1}{\beta} r(x,y) \right] \right)$$

Partition function Z

$$Z(x) = \sum_{y} \pi_{\text{ref}}(y|x) \exp\left(\frac{1}{\beta} r(x,y) \right)$$

$$\log \frac{\pi(y|x)}{\pi_{\text{ref}}(y|x)} - \frac{1}{\beta} r(x,y) \quad = \log \frac{\pi(y|x)}{\pi_{\text{ref}}(y|x)} - \frac{1}{\beta} r(x,y) + \log Z(x) - \log Z(x)$$

DPO Derivation ctd

$$= \left(\log \frac{\pi(y|x)}{\pi_{\text{ref}}(y|x)} + \log Z(x)\right) - \log Z(x) - \frac{1}{\beta} r(x, y)$$

$$= \log \frac{\pi(y|x)}{\frac{1}{Z(x)} \pi_{\text{ref}}(y|x)} - \log Z(x) - \frac{1}{\beta} r(x, y)$$

The objective:

$$\min_{\pi} \mathbb{E}_{x \sim \mathcal{D}} \mathbb{E}_{y \sim \pi(y|x)} \left[\log \frac{\pi(y|x)}{\frac{1}{Z(x)} \pi_{\text{ref}}(y|x) \exp\left(\frac{1}{\beta} r(x, y)\right)} - \log Z(x)\right]$$

$$\min_{\pi} \mathbb{E}_{x \sim \mathcal{D}} \left[\mathbb{D}_{\text{KL}} \left(\pi(y|x) || \frac{1}{Z(x)} \pi_{\text{ref}}(y|x) \exp\left(\frac{1}{\beta} r(x, y)\right)\right) - \log Z(x)\right]$$

DPO from BT Model

$$p^*(y_1 \succ y_2 \mid x) = \frac{\exp\left(r^*(x, y_1)\right)}{\exp\left(r^*(x, y_1)\right) + \exp\left(r^*(x, y_2)\right)}$$

$$r^*(x, y) = \beta \log \frac{\pi^*(y \mid x)}{\pi_{\text{ref}}(y \mid x)} + \beta \log Z(x)$$

$$p^*(y_1 \succ y_2 \mid x) = \frac{\exp\left(\beta \log \frac{\pi^*(y_1|x)}{\pi_{\text{ref}}(y_1|x)} + \beta \log Z(x)\right)}{\exp\left(\beta \log \frac{\pi^*(y_1|x)}{\pi_{\text{ref}}(y_1|x)} + \beta \log Z(x)\right) + \exp\left(\beta \log \frac{\pi^*(y_2|x)}{\pi_{\text{ref}}(y_2|x)} + \beta \log Z(x)\right)}$$

DPO from BT Model

$$p^*(y_1 \succ y_2 \mid x) = \frac{\exp\left(\beta \log \frac{\pi^*(y_1|x)}{\pi_{\text{ref}}(y_1|x)}\right)}{\exp\left(\beta \log \frac{\pi^*(y_1|x)}{\pi_{\text{ref}}(y_1|x)}\right) + \exp\left(\beta \log \frac{\pi^*(y_2|x)}{\pi_{\text{ref}}(y_2|x)}\right)}$$

$$p^*(y_1 \succ y_2 \mid x) = \frac{1}{1 + \exp\left(\beta \log \frac{\pi^*(y_2|x)}{\pi_{\text{ref}}(y_2|x)} - \beta \log \frac{\pi^*(y_1|x)}{\pi_{\text{ref}}(y_1|x)}\right)}$$

$$p^*(y_1 \succ y_2 \mid x) = \sigma\left(\beta \log \frac{\pi^*(y_1 \mid x)}{\pi_{\text{ref}}(y_1 \mid x)} - \beta \log \frac{\pi^*(y_2 \mid x)}{\pi_{\text{ref}}(y_2 \mid x)}\right)$$

Issues in DPO

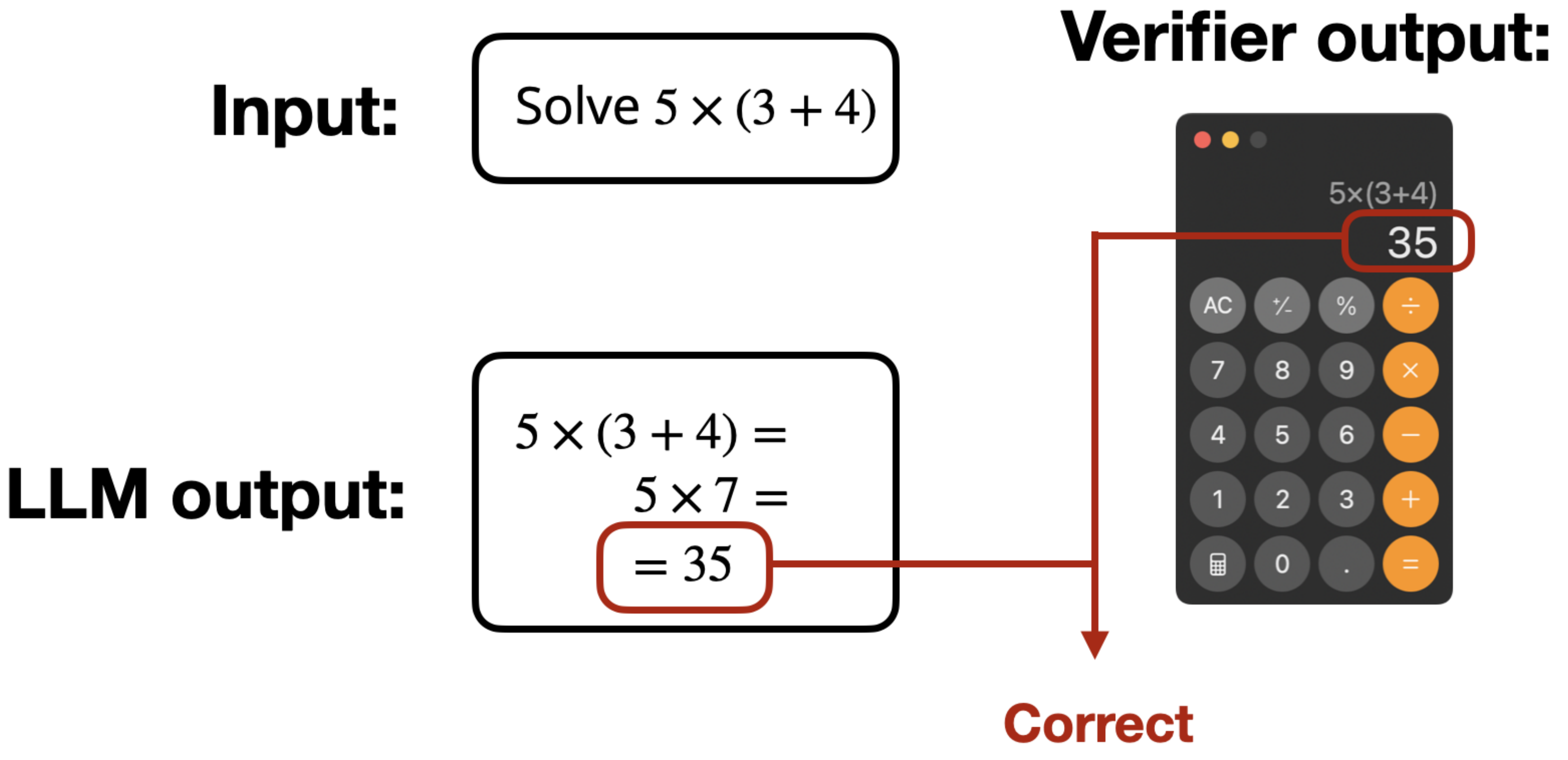
- Only the margin is maximized. What would happen?

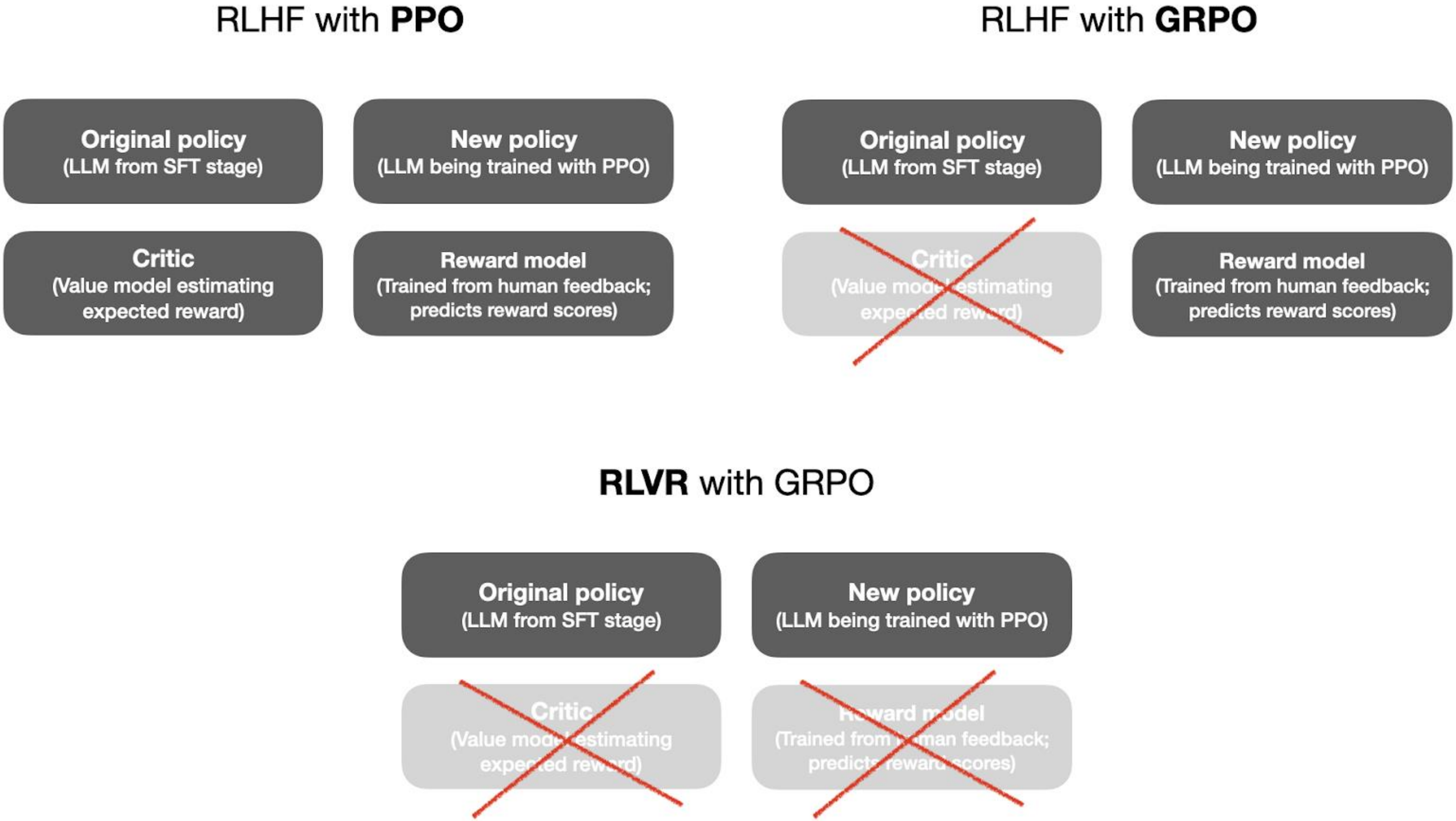
In Lec11

1. RL in LLMs
2. RL for Alignment (RLHF, DPO)
3. RL for Reasoning (GRPO, test-time scaling)

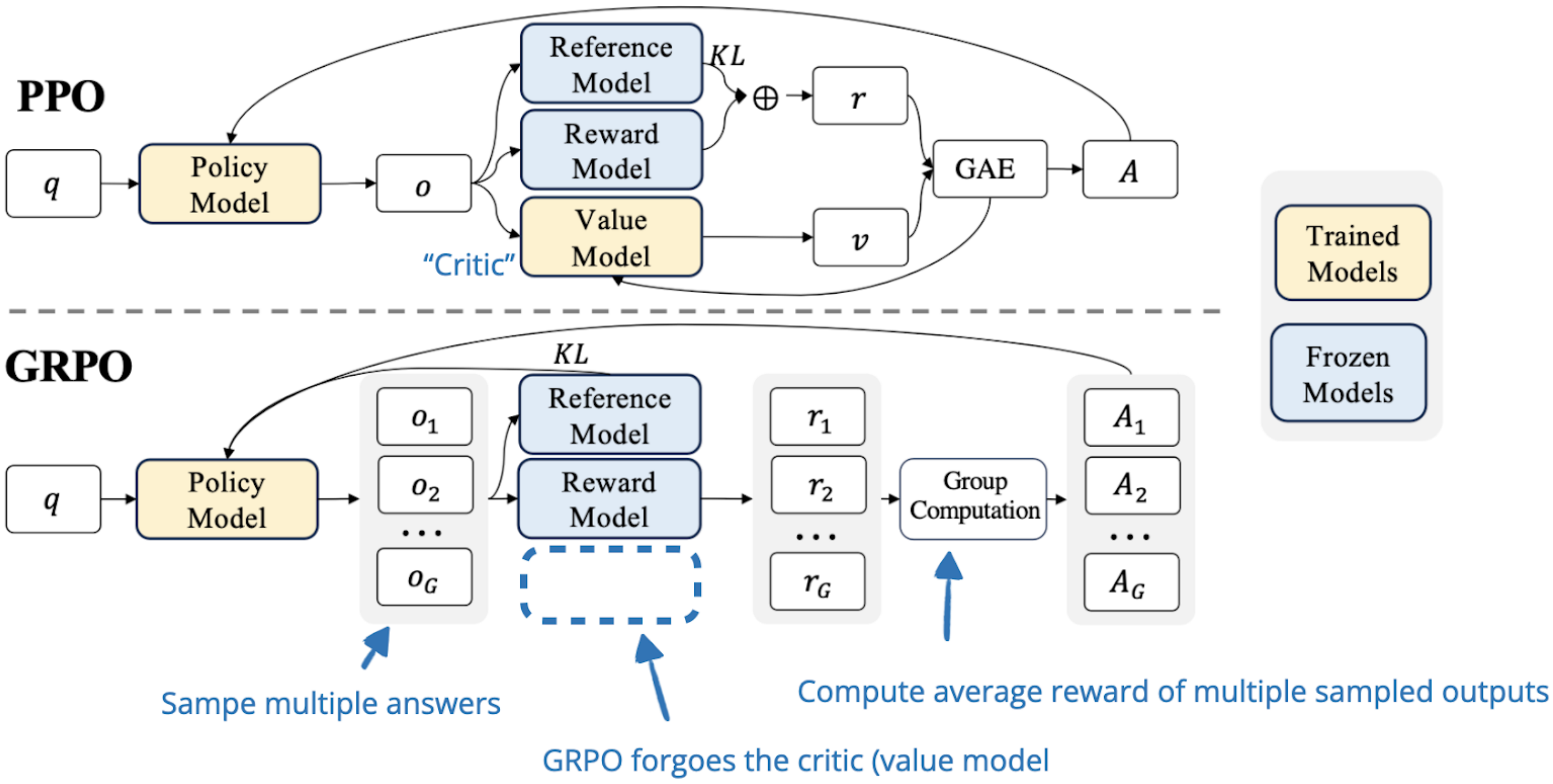
From RLHF to RLVR

- Reinforcement learning from verifiable rewards!




RLHF with PPO
Original policy
(LLM from SFT stage)
New policy
(LLM being trained with PPO)
Critic
(Value model estimating expected reward)
Reward model
(Trained from human feedback; predicts reward scores)
RLHF with GRPO
Original policy
(LLM from SFT stage)
New policy
(LLM being trained with PPO)
Critic
(Value model estimating expected reward)
Reward model
(Trained from human feedback; predicts reward scores)
RLVR with GRPO
Original policy
(LLM from SFT stage)
New policy
(LLM being trained with PPO)
Critic
(Value model estimating expected reward)
Reward model
(Trained from human feedback; predicts reward scores)

DeepSeek-R1 and GRPO



GRPO

- Pros:
  - Avoiding the challenge of learning a value function from a LM backbone, where research hasn’t established best practices.
  - Saves memory by not needing to keep another set of model weights in memory.

GRPO

- The loss is accumulated over a group of responses to a given question

$$J(\theta) = \frac{1}{G}\sum_{i=1}^{G}\left(\min\left(\frac{\pi_\theta(a_i|s)}{\pi_{\theta_{old}}(a_i|s)}A_i, \mathrm{clip}\left(\frac{\pi_\theta(a_i|s)}{\pi_{\theta_{old}}(a_i|s)}, 1-\varepsilon, 1+\varepsilon\right)A_i\right) - \beta D_{KL}(\pi_\theta||\pi_{ref})\right)$$

$$A_i = \frac{r_i - \mathrm{mean}(r_1, r_2, \cdots, r_G)}{\mathrm{std}(r_1, r_2, \cdots, r_G)}.$$

The GRPO update is comparing multiple answers to a single question within a batch.

Low std is preferred.

Dr.GRPO

- There is no normalization.
  - Effectively increase the learning rate.
  - Do not prefer unanimous answers anymore
  - But potentially prefers a single correct answer vs other wrong answers.

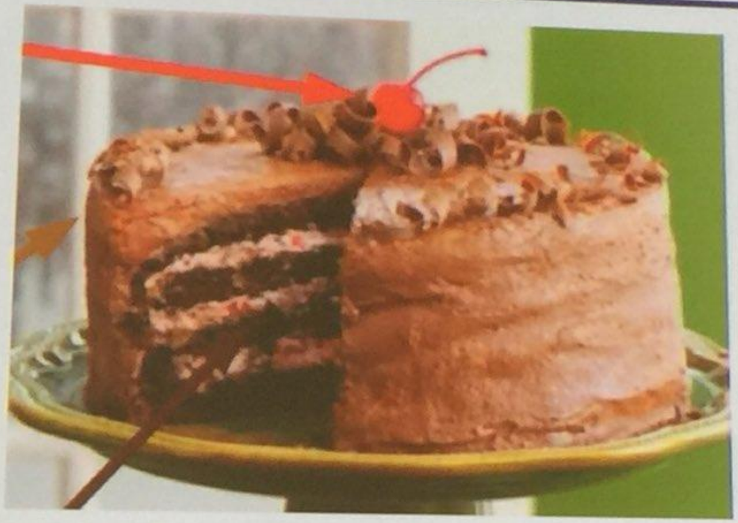
$$\tilde{A}_i = r_i - \mathrm{mean}(r_1, r_2, \cdots, r_G) = r_i - \frac{1}{G}\sum j = 1^G r_j$$

Motivation: LeCake

- **"Pure" Reinforcement Learning (cherry)**
  - The machine predicts a scalar reward given once in a while.
  - **A few bits for some samples**
- **Supervised Learning (icing)**
  - The machine predicts a category or a few numbers for each input
  - Predicting human-supplied data
  - **10→10,000 bits per sample**
- **Self-Supervised Learning (cake génoise)**
  - The machine predicts any part of its input for any observed part.
  - Predicts future frames in videos
  - **Millions of bits per sample**

Yann LeCun's cake

Reward Signal in Reinforcement Learning?



Standard RL

Hindsight Experience Replay

[Andrychowicz et al, NIPS 2017]

See also: Schmidhuber and Huber (1990); Caruana (1998); Da Silva et al (2012); Kober et al (2012); Devin et al (2016); Pinto and Gupta (2016); Foster and Dayan (2002); Sutton et al (2011); Bakker and Schmidhuber (2014); Vezhnevets et al (2017)

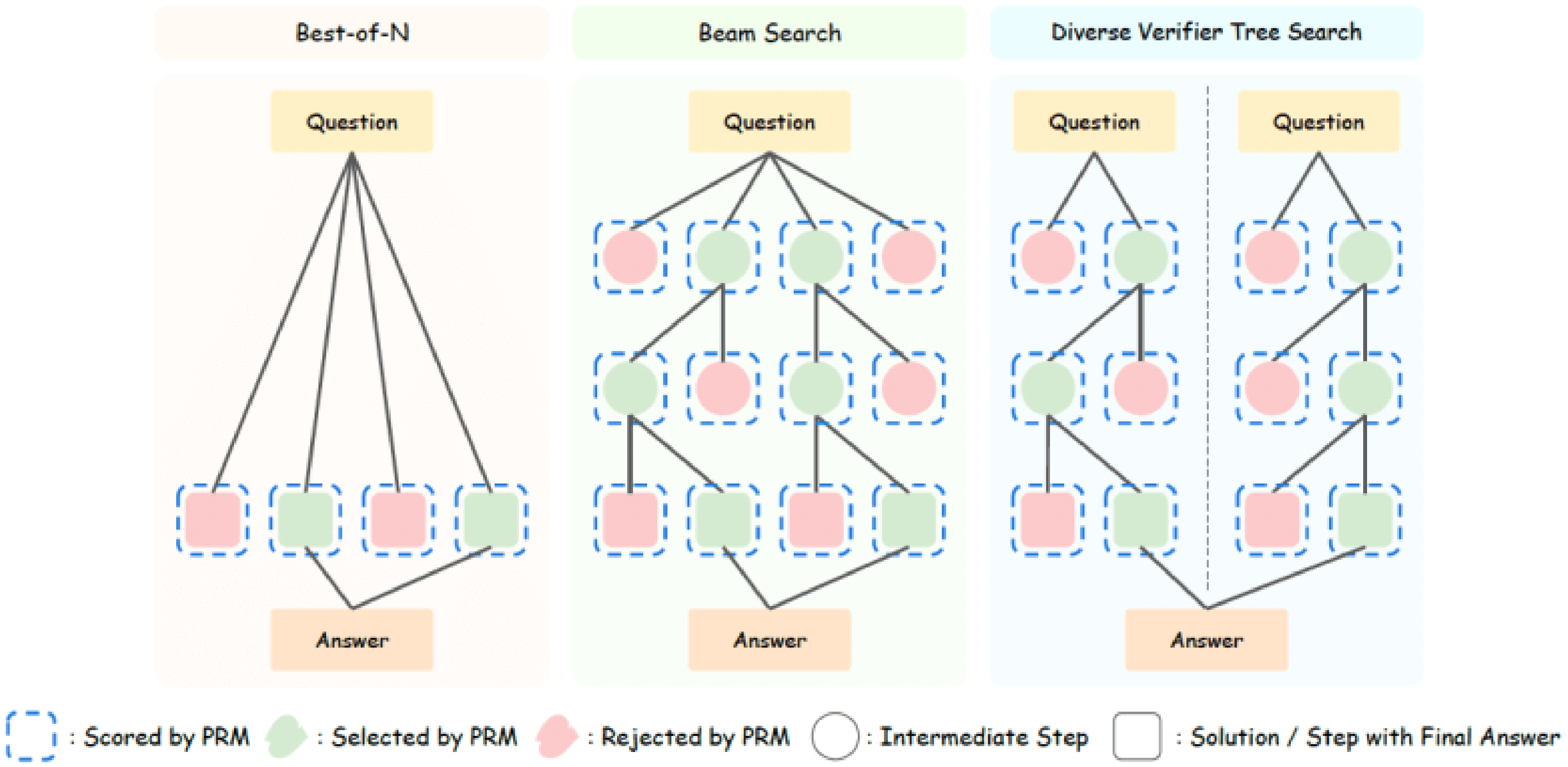
Most closely related: Schaul et al, 2015 Universal Value Functions

RL and test-time scaling

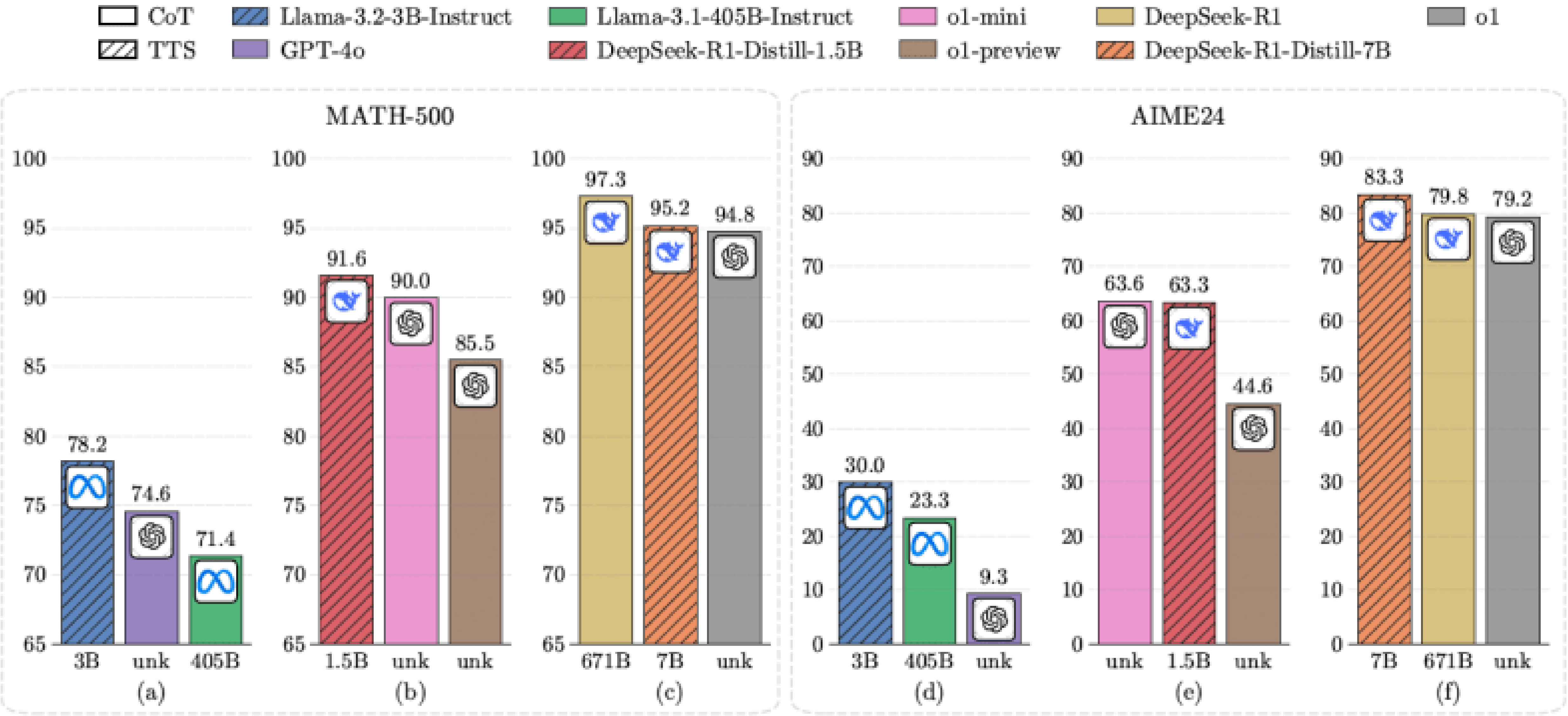
- OpenAI o1 and DeepSeek R1 uses this tiny little RL trick
  - Scale up RL becomes the new cool thing!
- And it is quite straight forward:
  - Sample multiple answers to multiple questions
  - Take gradient steps towards the answers that are correct
  - Repeat, revisiting the same data
- The magic (a bit controversial now):
  - It helps generalization to other unseen questions.

Test-time scaling

- Rejection Sampling
- Self-consistency



There is no training!



For small models, TTS is important in improving its reasoning ability.

In Lec11

1. RL in LLMs
2. RL for Alignment (RLHF, DPO)
3. RL for Reasoning (GRPO, test-time scaling)

lit: Chonghua Liao & Sebastian Raschka's blog & RLHFbook & Ryan Heida

Thank you!