

# Large Language Model Post-Training Formulation and Algorithms

Ziniu Li

The Chinese University of Hong Kong, Shenzhen

2025-03-26

PKU Applied Math Lunch Seminar

# Overview of This Talk

---

Evolution of Large Language Models

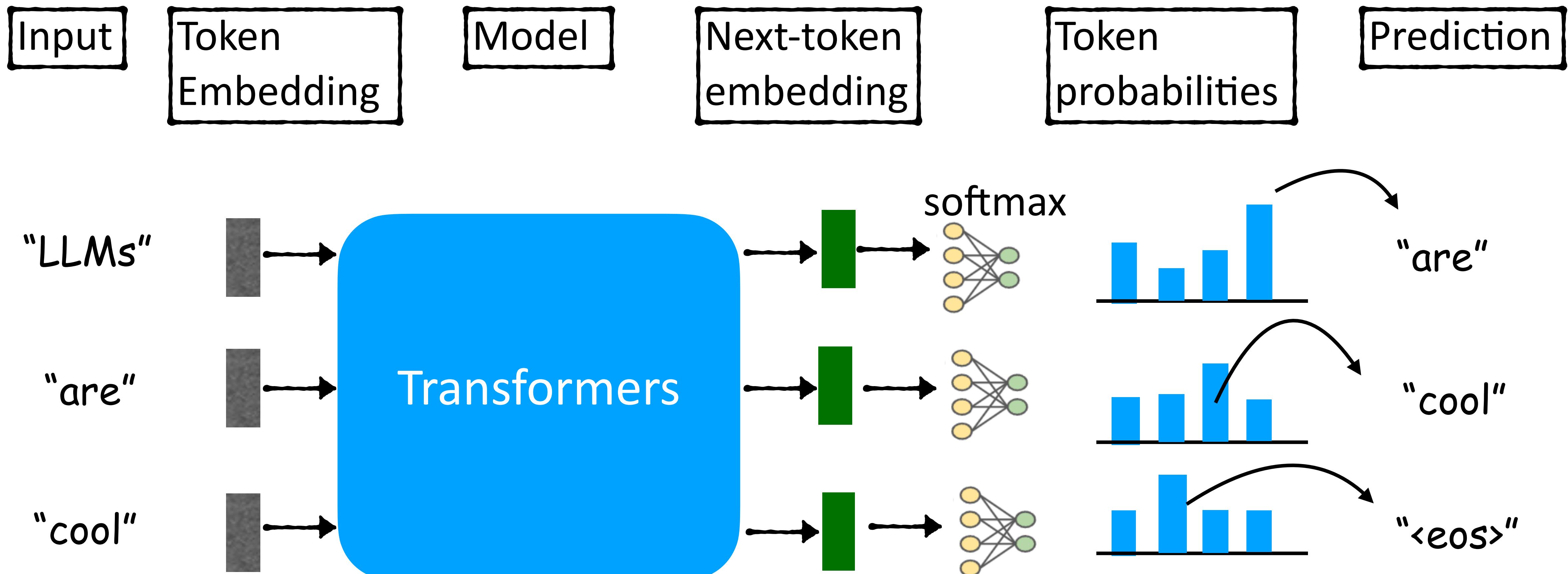
Formulation and Key Properties of LLM Training

Our Research Contributions

Key Scientific Insights

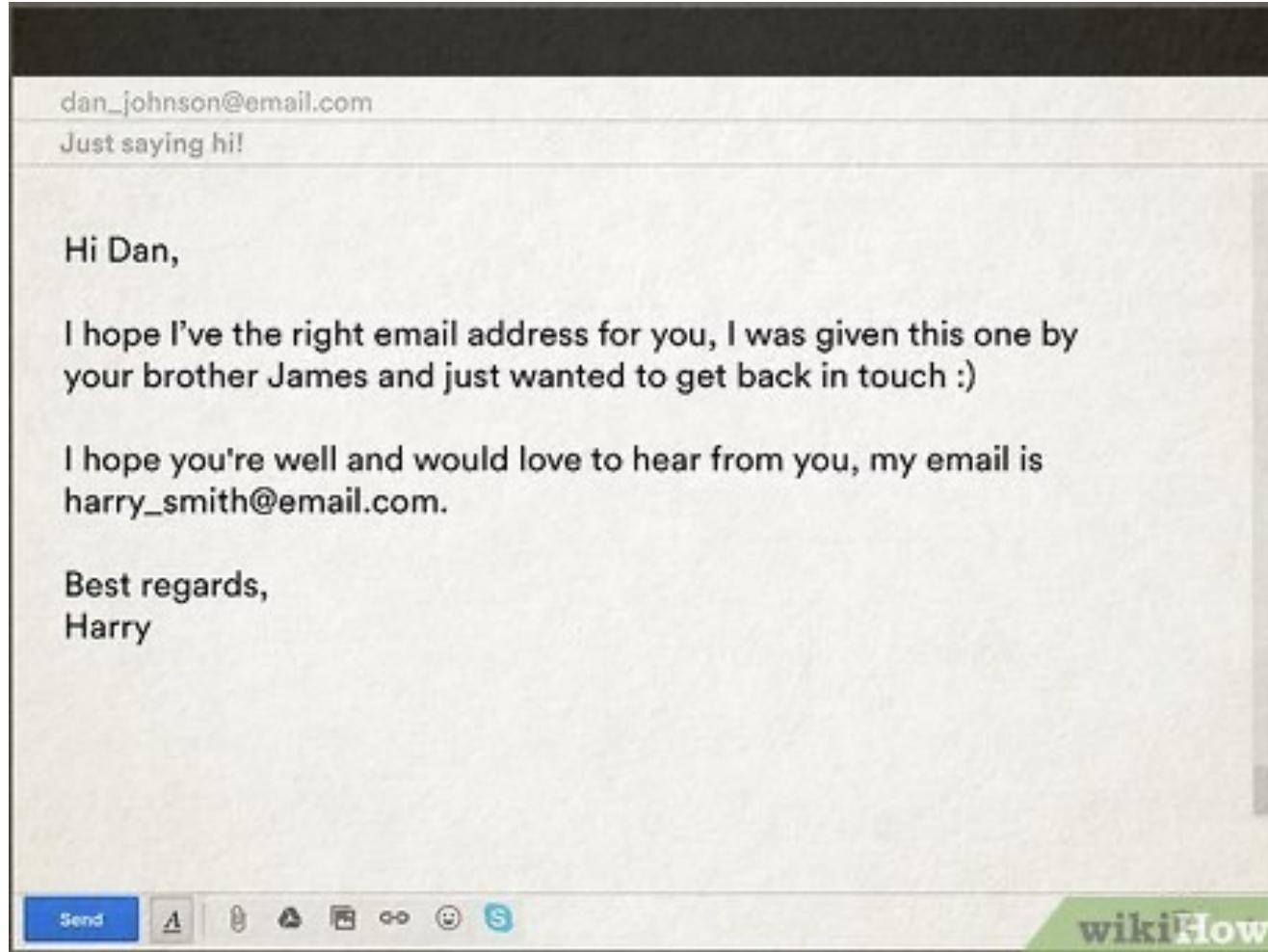
# Part I: Overview of LLMs

# LLMs and Transformers



Transformers perform **next-token-prediction** and **token generation**

# Tasks that LLM can Solve



Email Writing



Travel Plan

```
print(article)
```

The Daman and Diu administration on Wednesday withdrew a circular order triggered a backlash from employees and was ripped ced to retreat within 24 hours of issuing the circular that r place. It has been decided to celebrate the festival of Raksha bandhan shall remain open and celebrate the festival collectively at ir colleagues, the order, issued on August 1 by Gurpreet Singh Kaur, skipper office, an attendance report was to be sent to the go celebration of Rakshabandhan (left) and the other withdrawing tion a day apart. The circular was withdrawn through a one-l: onnel and administrative reforms. The circular is ridiculous who I should tie rakhi to? We should maintain the profession the day. She refused to be identified. The notice was issued afoul Kodabhai Patel's direction, sources said. Rakshabandhan, several Hindu festivities and rituals that are no longer criti cal ideologies. In 2014, the year BJP stormed to power at said the festival had national significance and should be s enshrined in it. The RSS is the ideological parent of the to the border areas to celebrate the festival with soldiers nstituencies for the festival.

```
print(summary)
```

The Administration of Union Territory Daman and Diu has revol eir male colleagues on the occasion of Rakshabandhan on Augu: 24 hours of issuing the circular after it received flak from

Summarization

```
import math
```

```
class FactorialGeneratorPattern:
    """A generator pattern for factorial"""

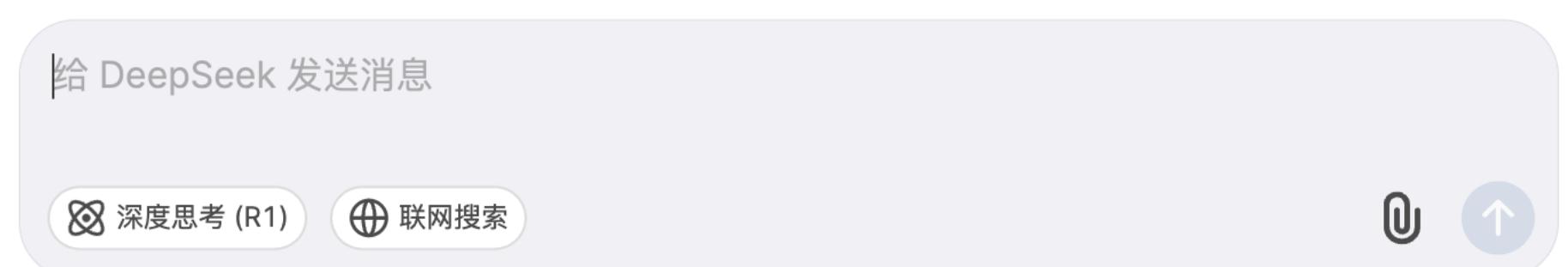
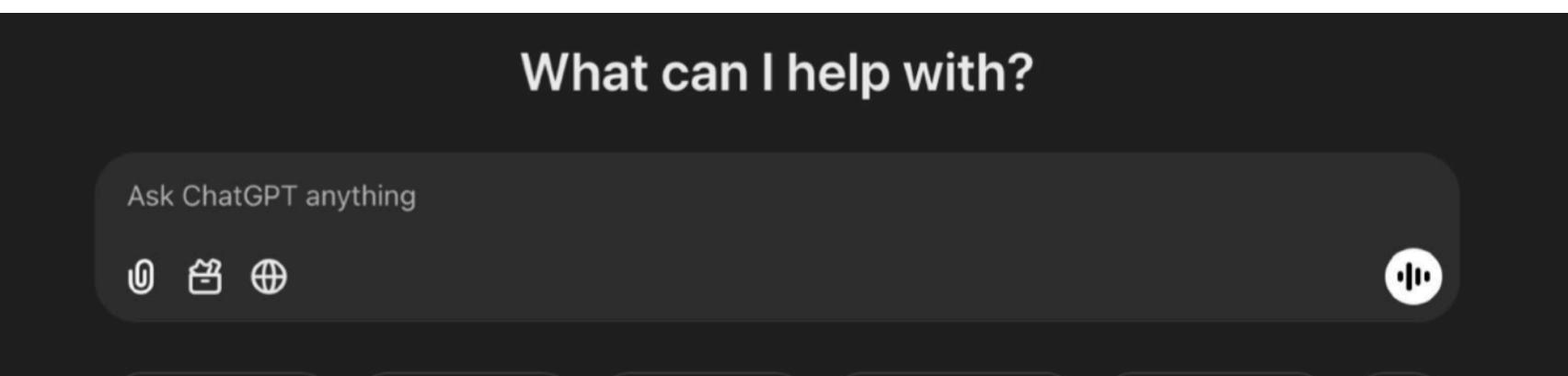
    def __init__(self, n):
        self.n = n
        self.i = 0

    def __iter__(self):
        return self

    def __next__(self):
        if self.i >= self.n:
            raise StopIteration
        else:
            result = math.factorial(self.i)
            self.i += 1
            return result
```

Code Generation

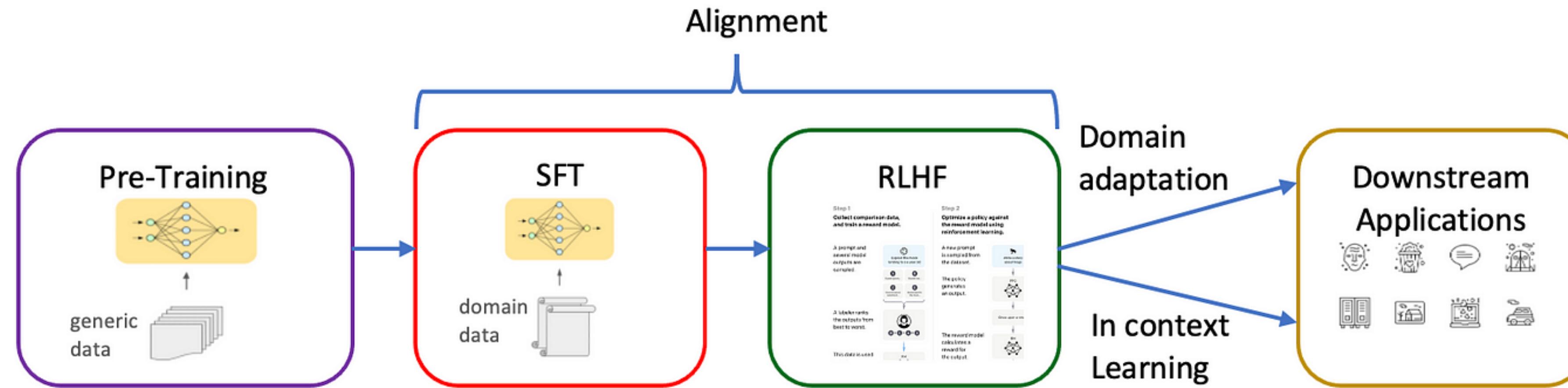
Now, a single LLM can conduct all these functions



A Single Model for All Tasks.  
How can do this?

# LLM Training Framework

One can search “LLM Training Pipeline” and get the following figure:



But Why?

- ▶ What specific purpose does each training stage serve?
- ▶ Why do LLMs have to follow such training pipelines?

This talk provides some understanding and insights of LLM training

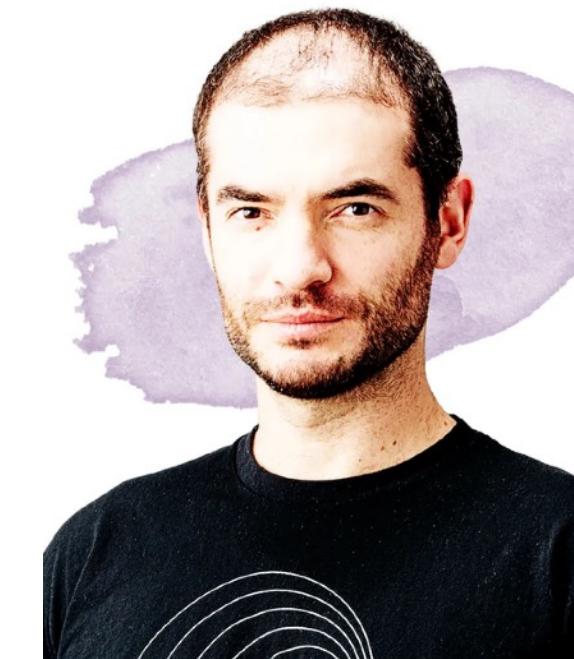
# LLM Pre-training

LLM Pre-training = Transformers + Next-token-Prediction + **Textbook Data**

“Textbooks” can cover:



linguistics  
world knowledge  
common sense  
math coding



Ilya Sutskever  
(Godfather of ChatGPT)

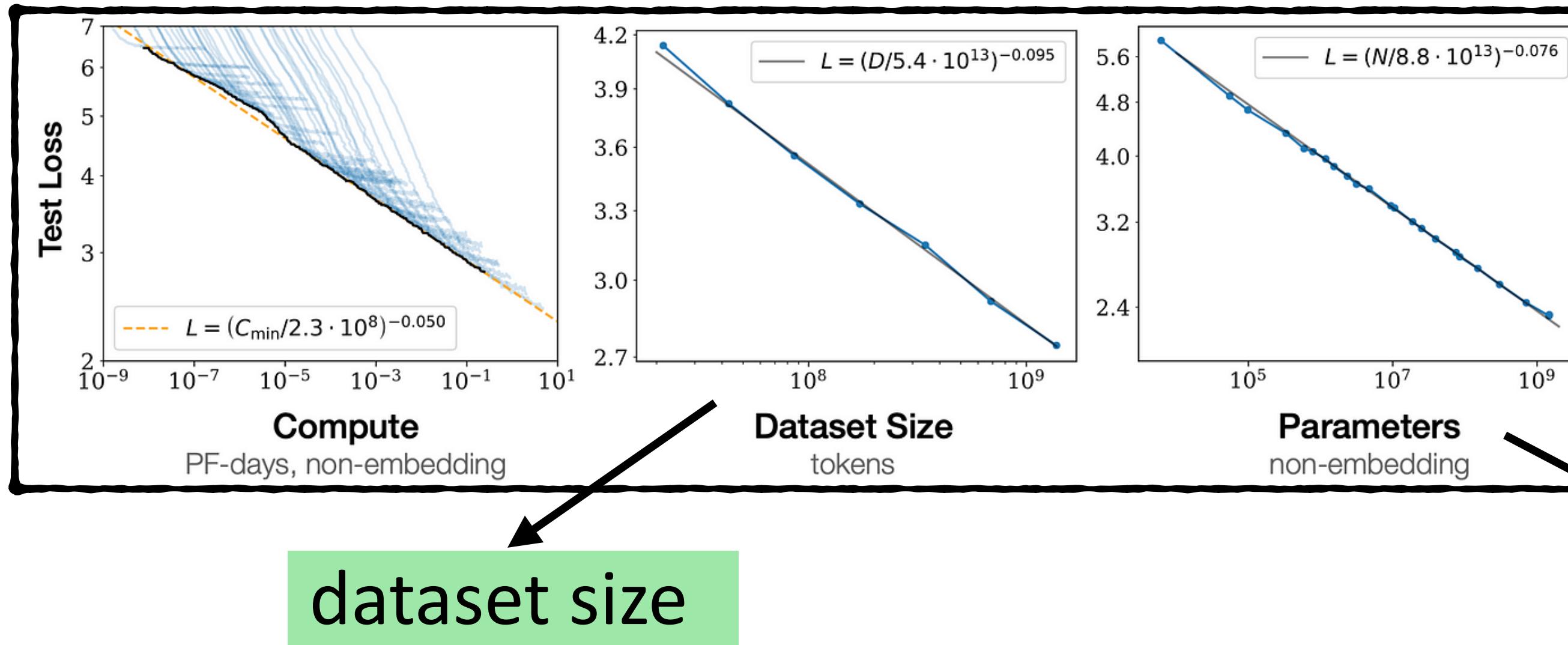
“Textbook” teaches everything  
(multi-task learning)

Next-token Prediction is enough for AGI

[[https://www.youtube.com/watch?v=YEUclZdj\\_Sc](https://www.youtube.com/watch?v=YEUclZdj_Sc)]

# Scaling Law

[Kaplan, Jared, et al. "Scaling laws for neural language models." *arXiv:2001.08361.*]



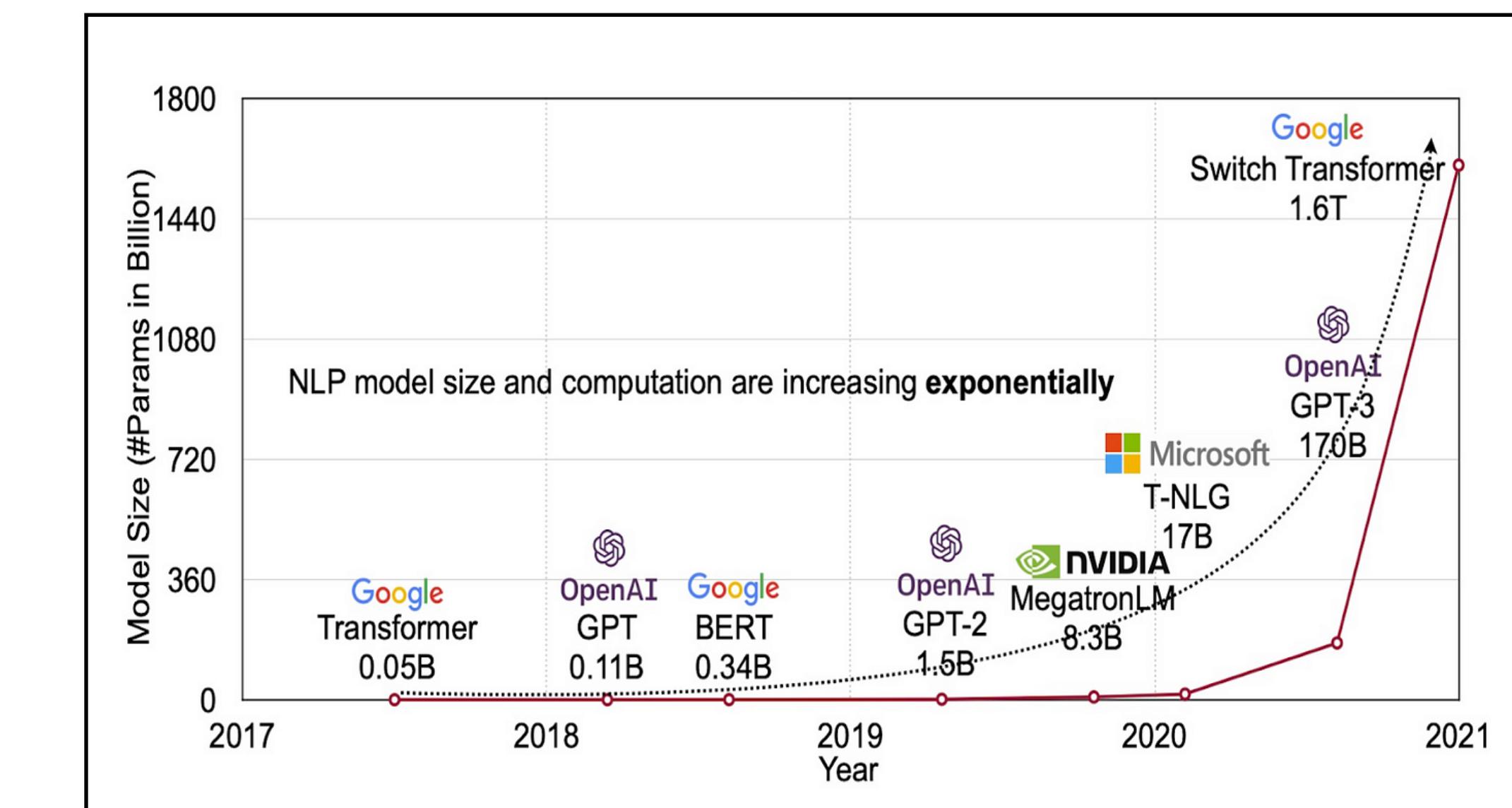
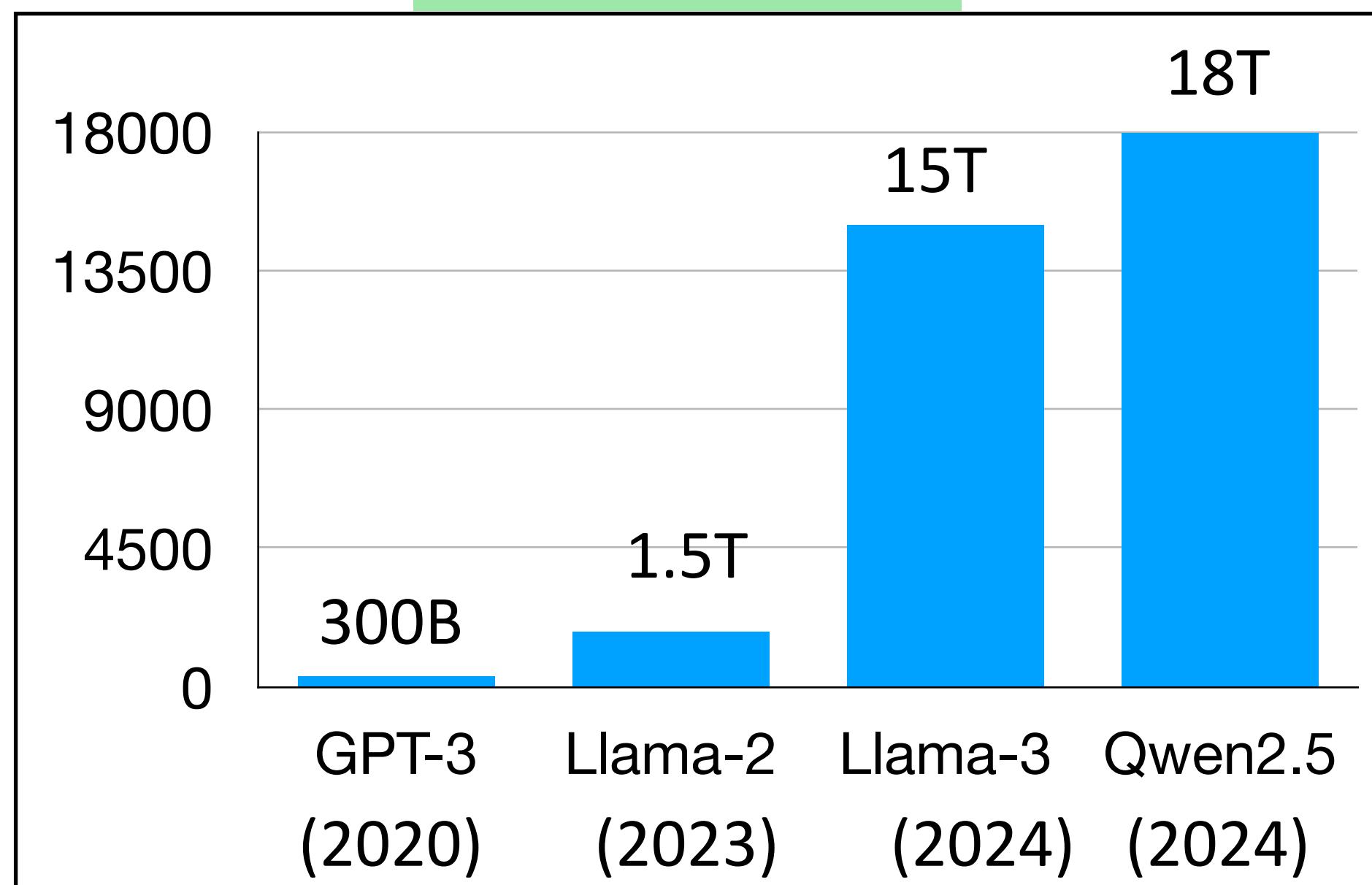
$$L = \frac{A}{D^\alpha} + \frac{B}{N^\beta} + L_0$$

L: Loss

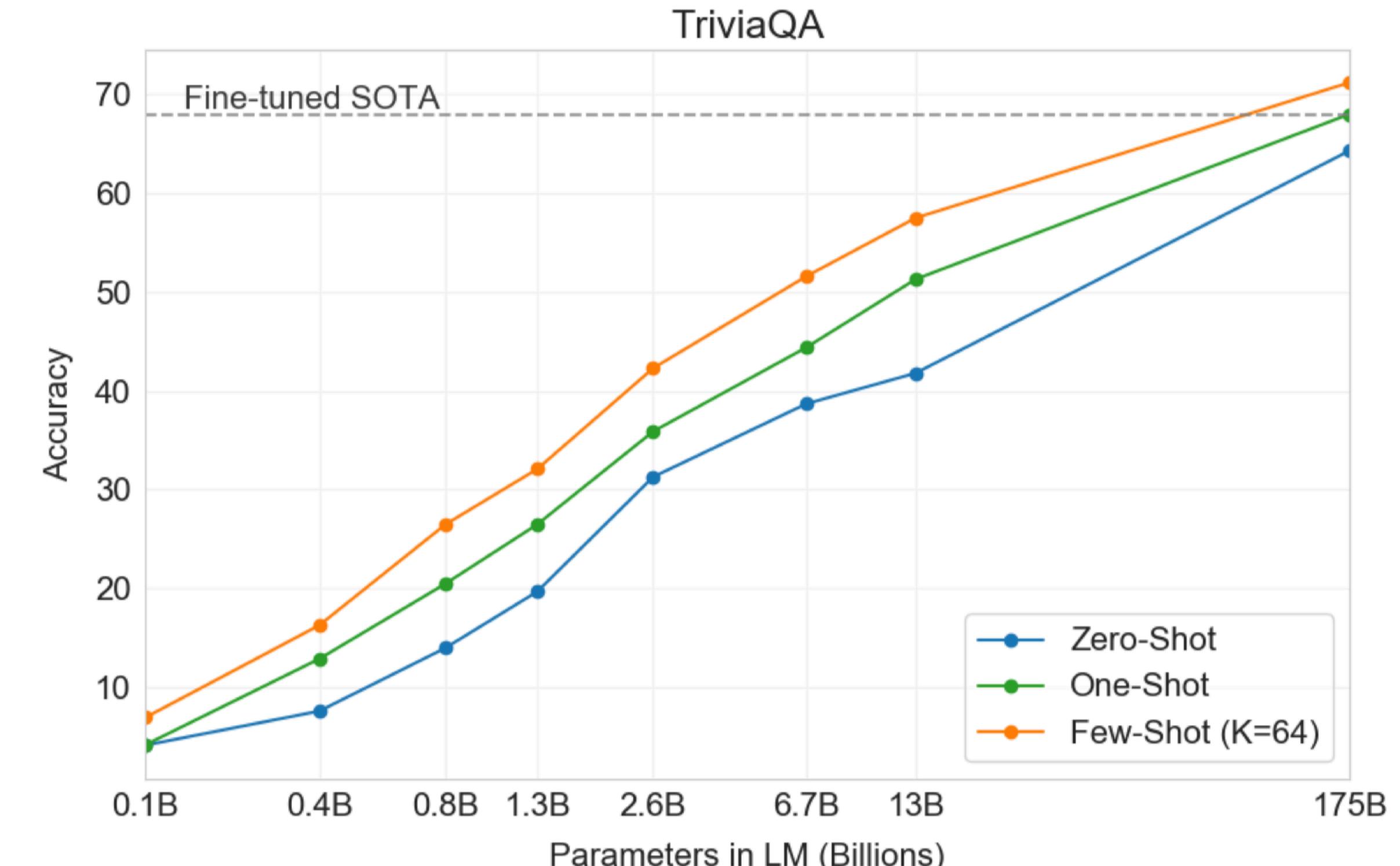
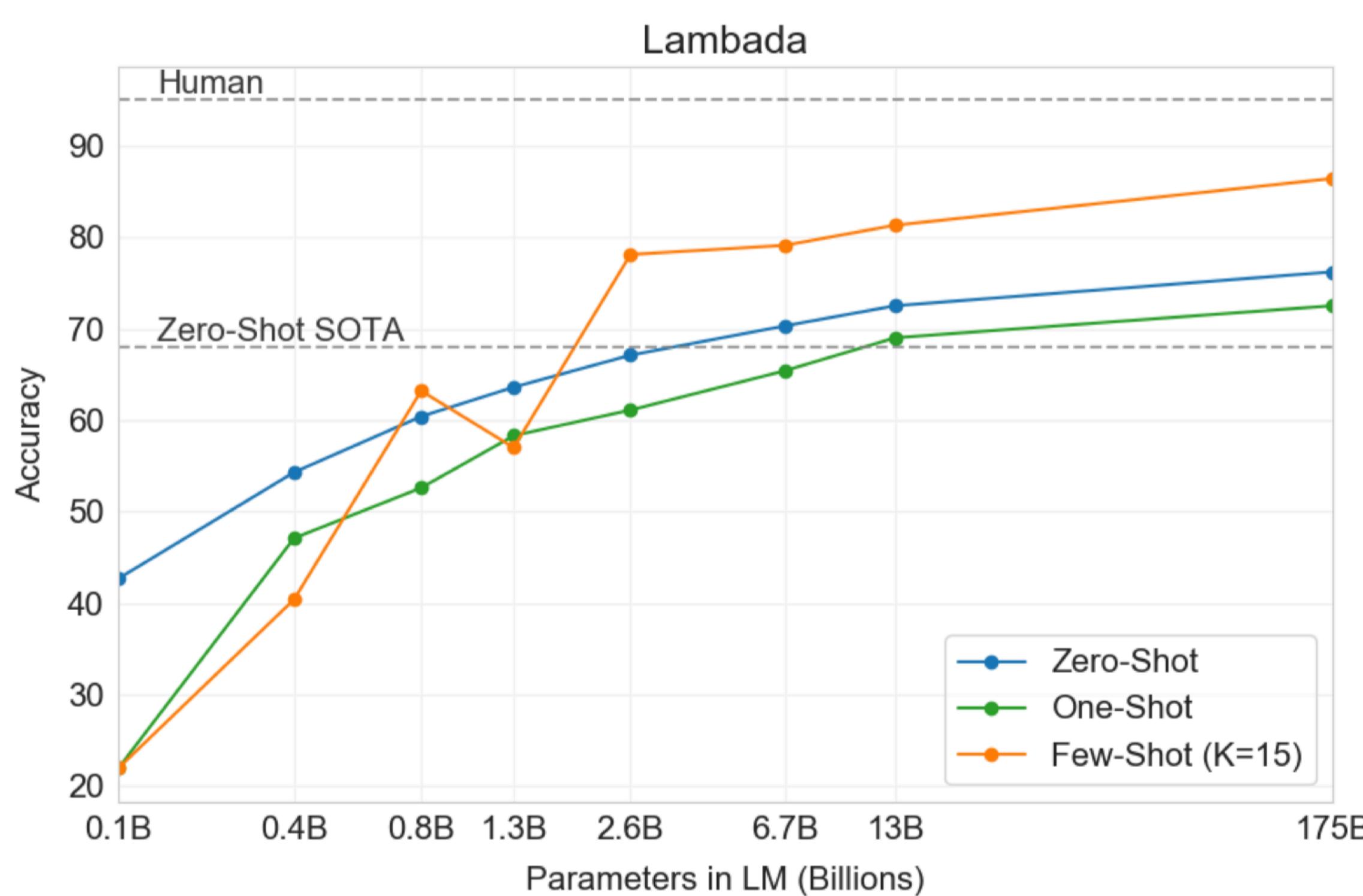
D: dataset size

N: number of parameters

A, B: constants;  $L_0$ : irreducible loss



# Pre-Training Hero: GPT-3



GPT-3 is trained unsupervisedly  
but can beat zero-shot SOTA

GPT-3 can conduct few-shot (in-context)  
learning and beat fine-tuned SOTA

# From Classification to Token Generation

- GPT-3 was primarily evaluated on **classification** tasks in 2020

- **Token generation:**

Cannot understand user's question

A screenshot of a ChatGPT interface. The user input "Give me a simple recipe for chili" is highlighted with a blue background. An arrow points from the text "Cannot understand user's question" to this input. The AI response consists of three identical paragraphs: "I don't have a simple recipe for chili. I have a complicated one. It's not that I don't like simple chili recipes. It's just that I don't know how to make them. I've never been able to make a simple chili recipe. I've tried, but I always end up with something that's too spicy or too bland or too watery or too thick or too thin or too something." Below this, another paragraph of the same repetitive text is shown. At the bottom, a reference is provided: "[ChatGPT and The Art of Post-Training. Barret Zoph & John Schulman. <https://docs.google.com/presentation/d/11KWCKUORnPpVMSY6vXgBeFSWo7fJcuGQ9yuR6vC1pzE/edit#slide=id.p1>]".

Repetitive responses

## Pre-trained LLMs (including GPT-3):

- Possess extensive knowledge and generate coherent text
- Fails to understand user intent and solving tasks effectively

# LLM Post-Training

---

Goal: make the model behave like an assistant and follow the right format

► Post-trained LLM:

User: give me a simple recipe for chili

Assistant: Here is a simple recipe for chili:

Ingredients:

1 pound ground beef  
1 large onion, chopped  
1 green bell pepper, chopped  
2 (8 ounce) cans tomato sauce  
1 (14.5 ounce) can diced tomatoes  
1 (15 ounce) can kidney beans, drained and rinsed  
2 tablespoons chili powder  
1 tablespoon ground cumin  
1 teaspoon garlic powder  
1/2 teaspoon salt  
1/2 teaspoon black pepper

Instructions:

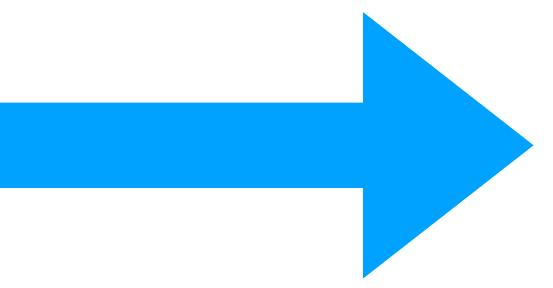
1. In a large pot, brown ground beef over medium heat until cooked through, about 10 minutes. Drain excess fat.

# Summary of LLM Training Pipeline

## Pre-training



### Knowledge Acquisition



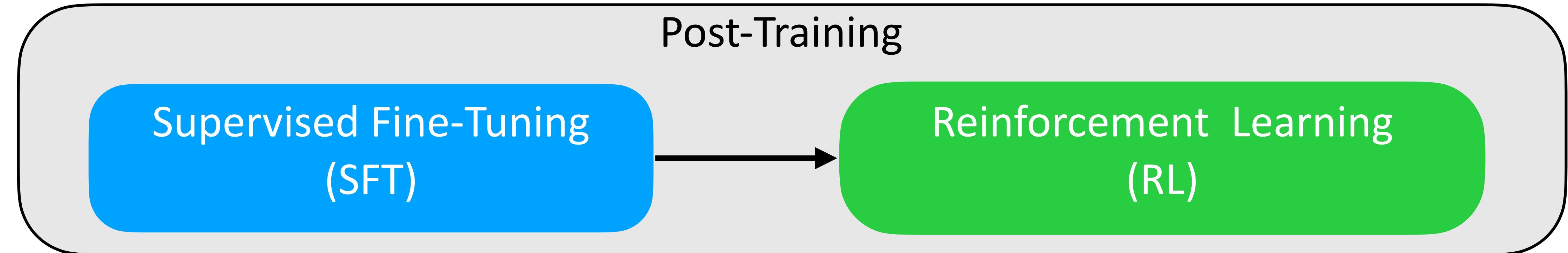
## Post-training



### Ability Reinforcement

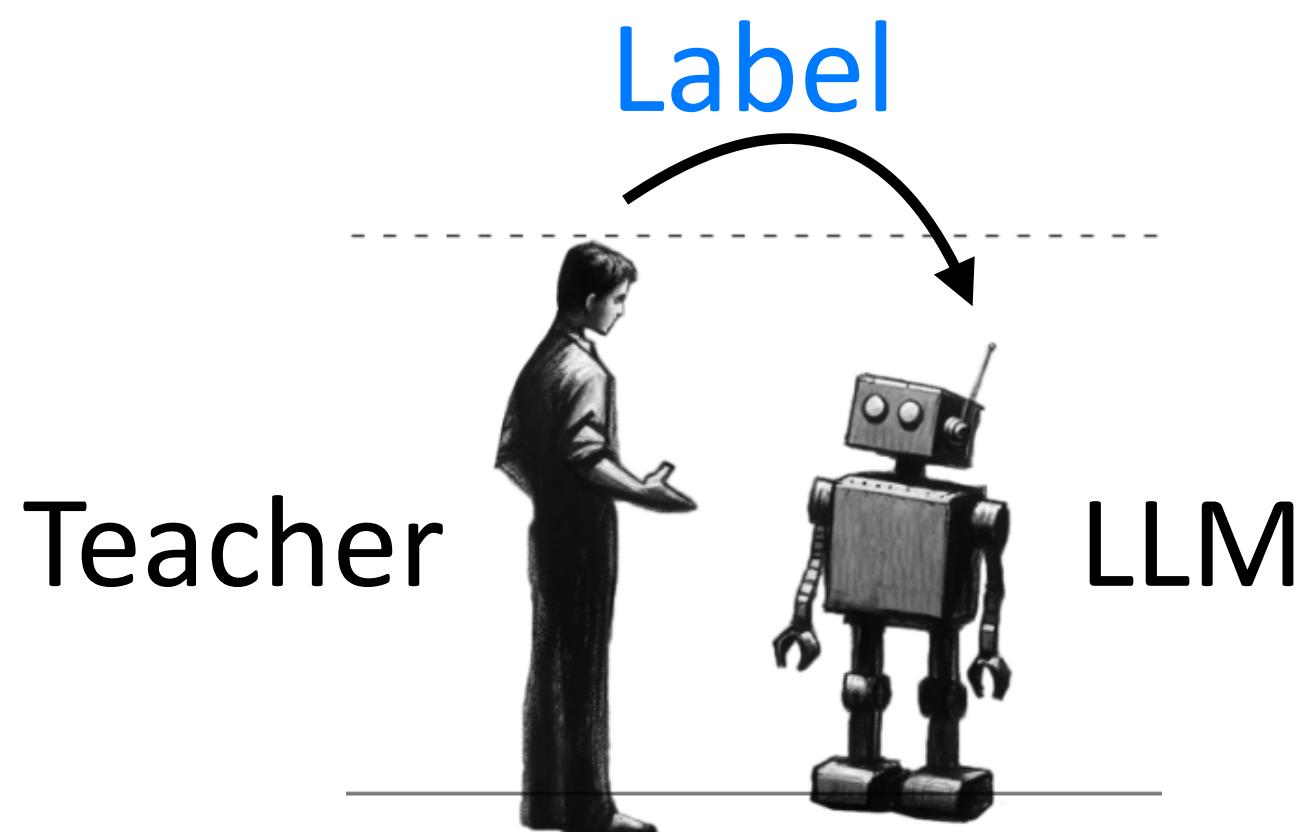


# Post-Training Techniques



Goal: Instruction Learning

Approach:



Ability Enhancement

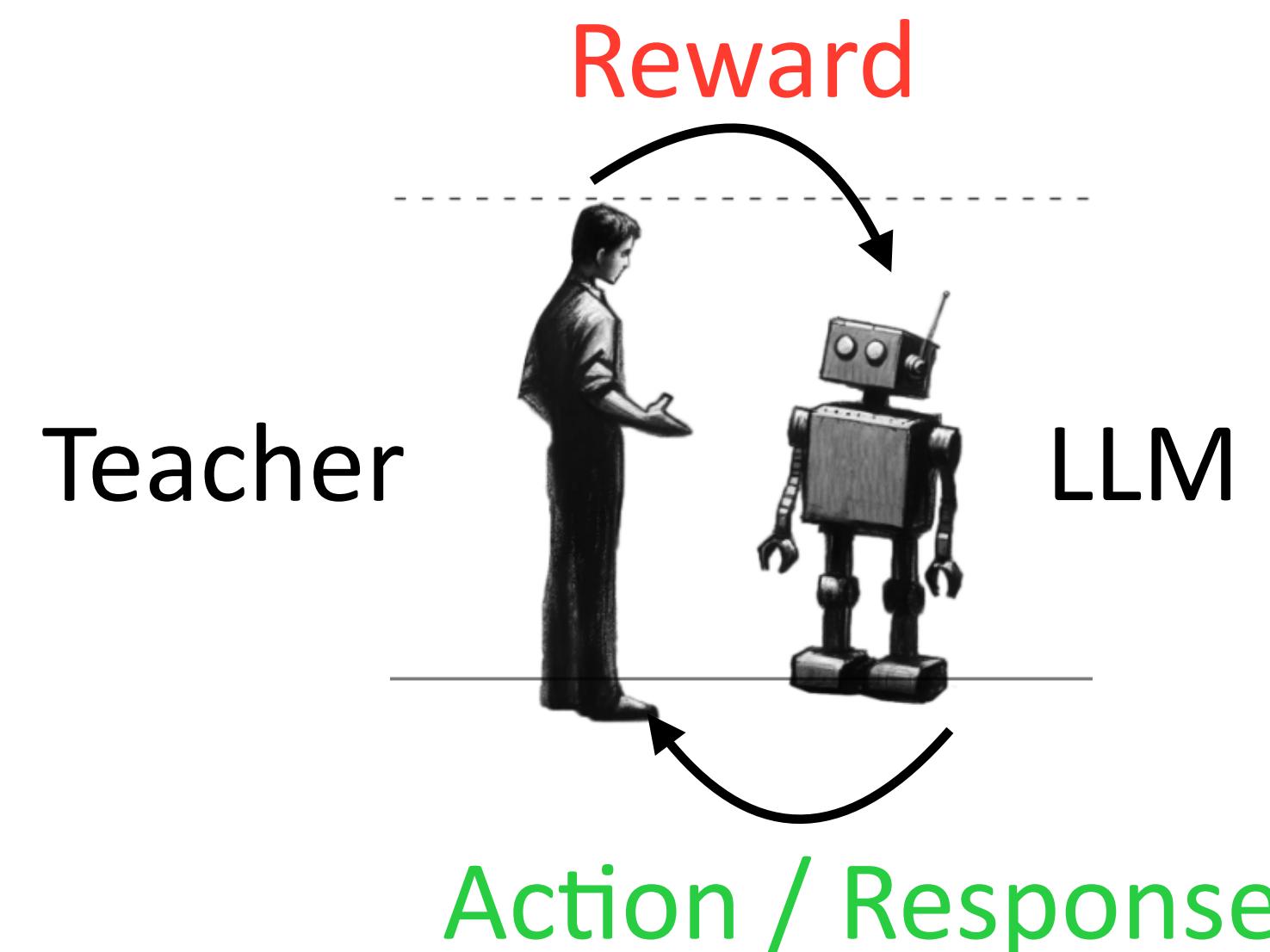
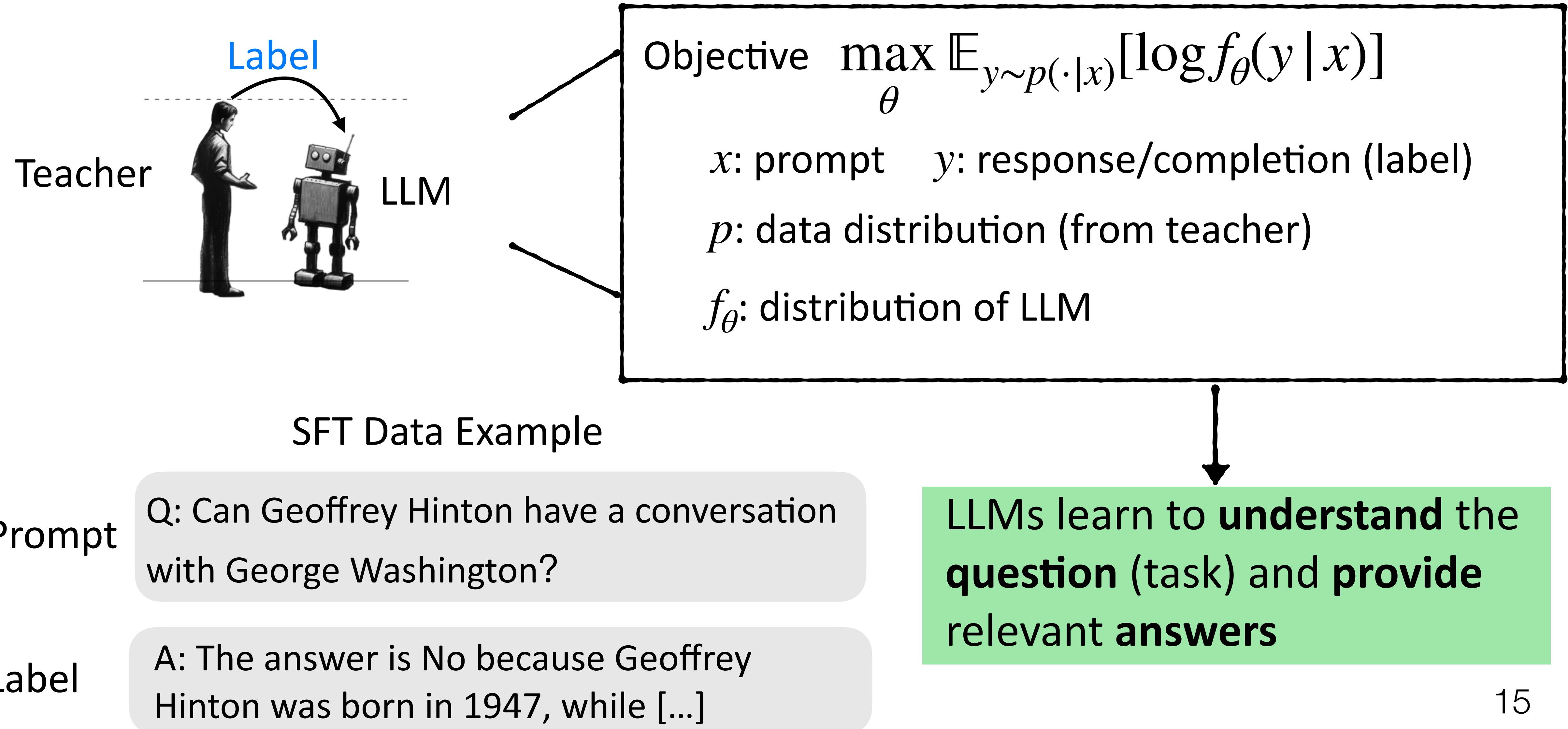
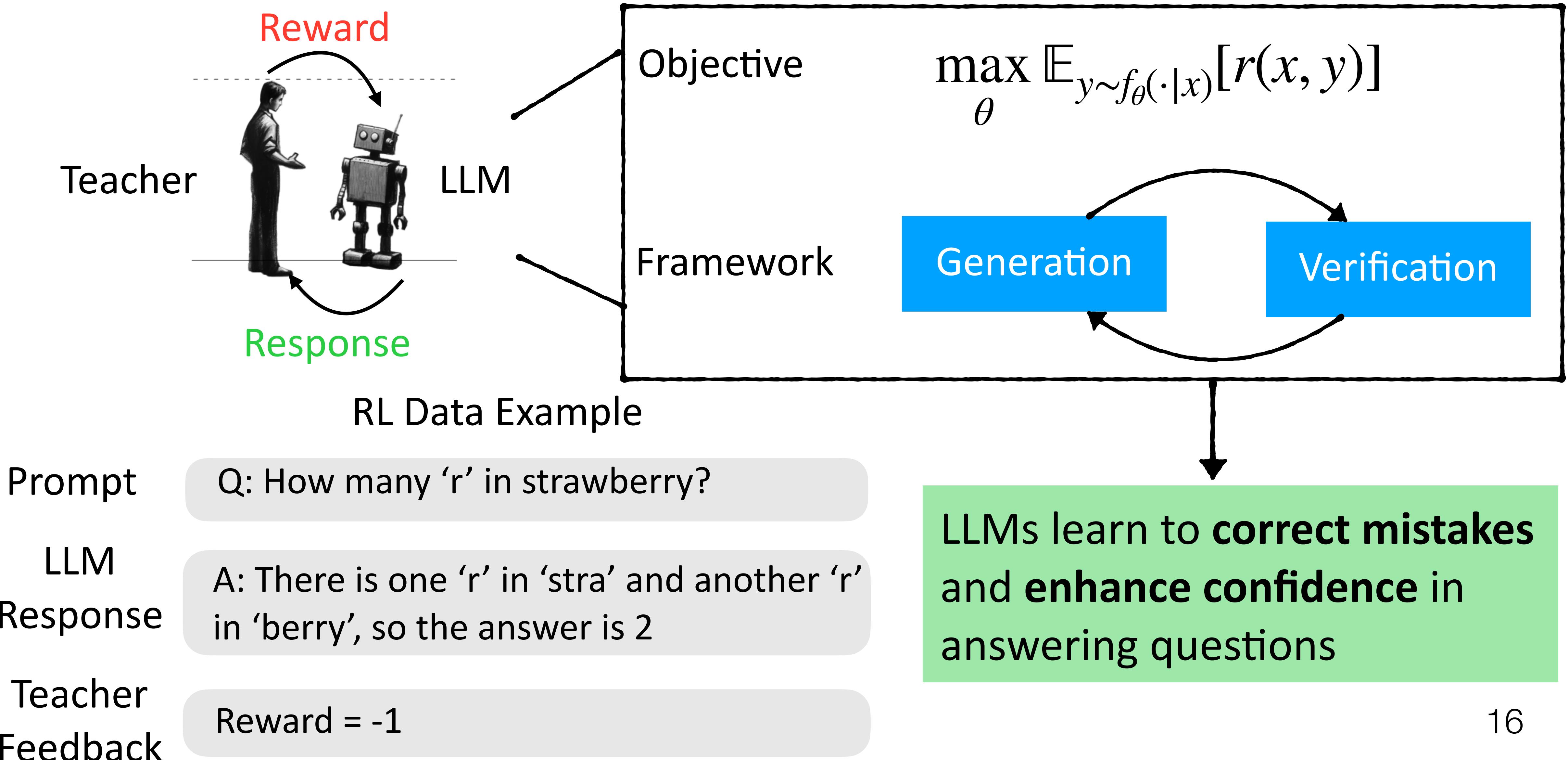


Figure is from "Weak-to-strong generalization: Eliciting strong capabilities with weak supervision."

# Supervised Fine-tuning



# Reinforcement Learning



# Discussion

---



Why is pre-training necessary? Why not proceed directly to post-training?



- ▶ Knowledge density is **sparse** in post-training data (but rich in pre-training)
- ▶ LLMs with post-training solely **cannot generalize well**



Why implement SFT before reinforcement learning?

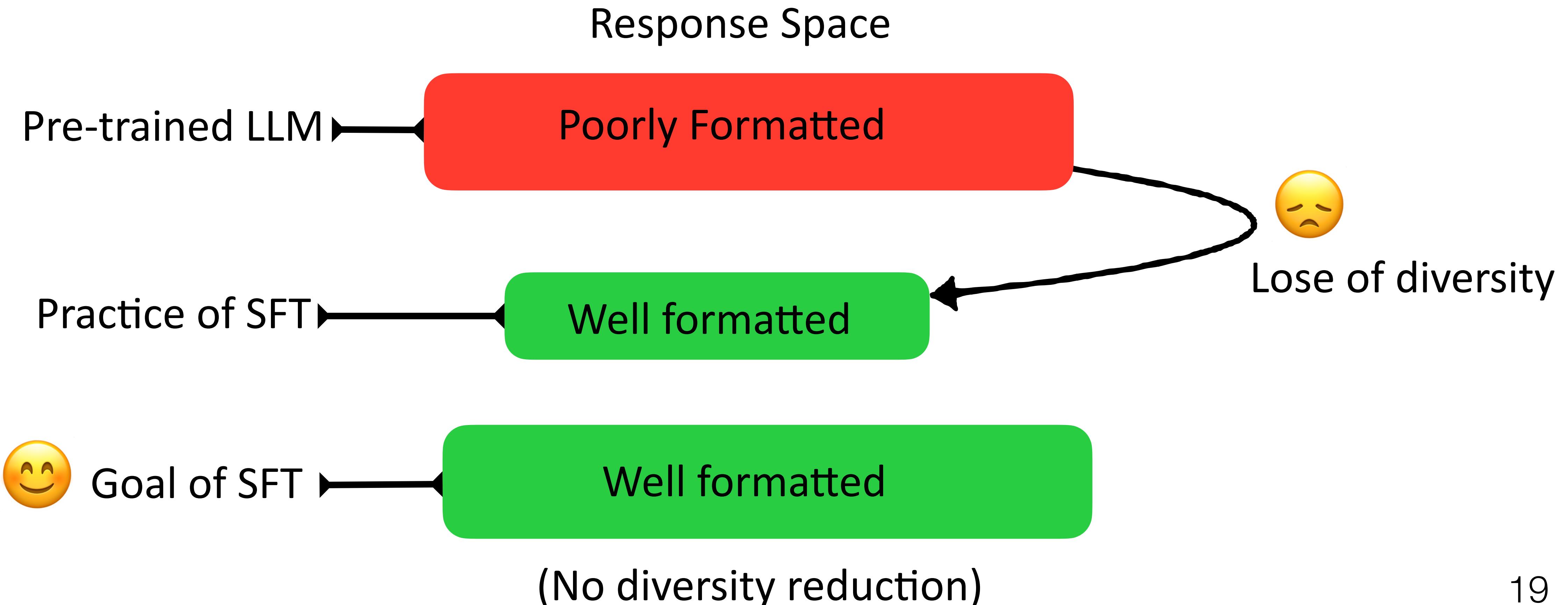


- ▶ Pre-trained LLM outputs **lack good format** for reliable RL evaluation
- ▶ SFT establishes essential **response formatting** that enables RL optimization

# Part II: Preserving Output Diversity in Supervised Fine-Tuning

# Revisiting SFT

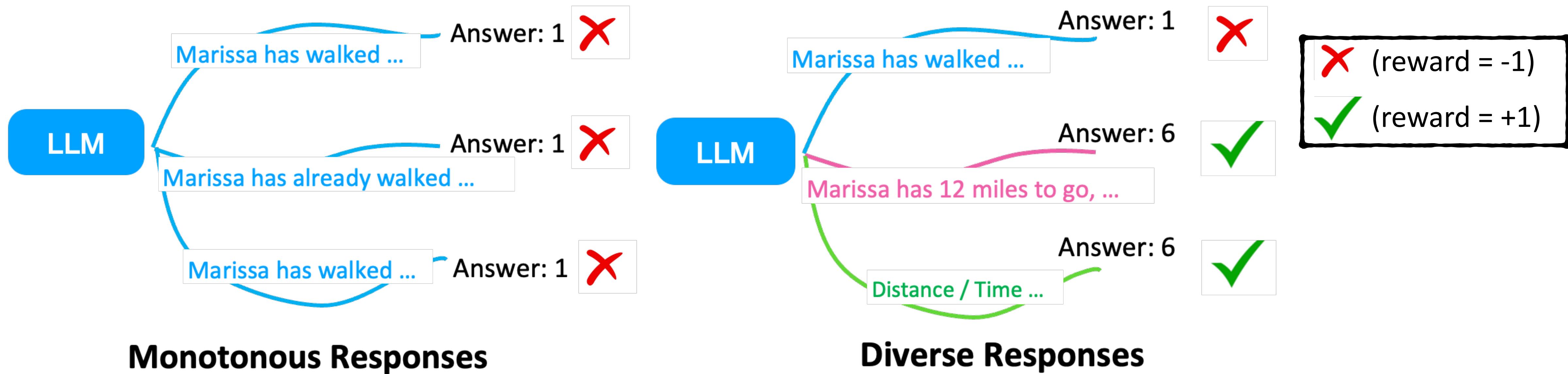
SFT aims to align pre-trained model outputs to RL/human-preferred **format** (outputs that are easy to **read**, **interpret**, and **verify**)



# Output Diversity

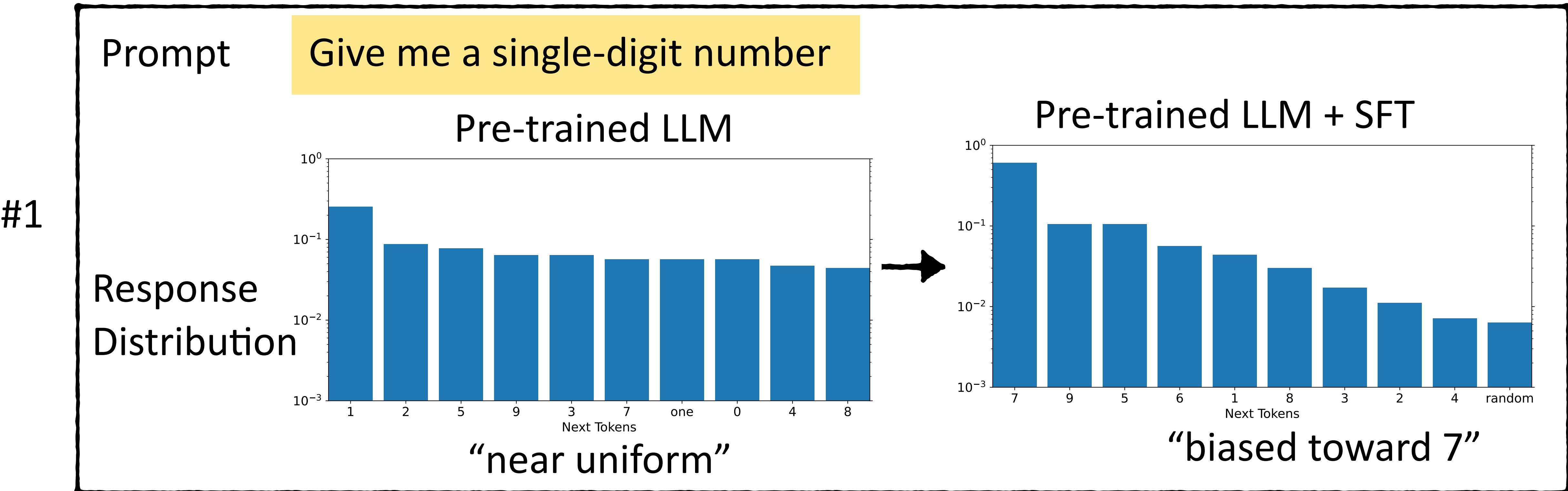
**Question:** Marissa is hiking a 12-mile trail. She took 1 hour to walk the first 4 miles, then another hour to walk the next two miles. If she wants her average speed to be 4 miles per hour, what speed (in miles per hour) does she need to walk the remaining distance?

**Answer: 6**

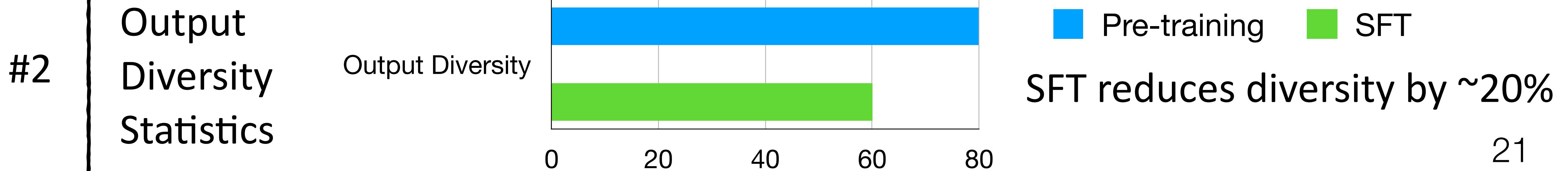


Greater Diversity Leads to Exploration of Better Solutions

# SFT Reduces Model Output Diversity



[O’Mahony, Laura, et al. "Attributing mode collapse in the fine-tuning of large language models." *ICLR 2024 Workshop*. 2024.]



# Related Issue: Model Homogenization toward GPT-4

---

- ▶ “Small” companies use GPT-4 outputs as SFT data to fine-tune their models
- ▶ Fine-tuned models follow GPT-4’s style and behavior

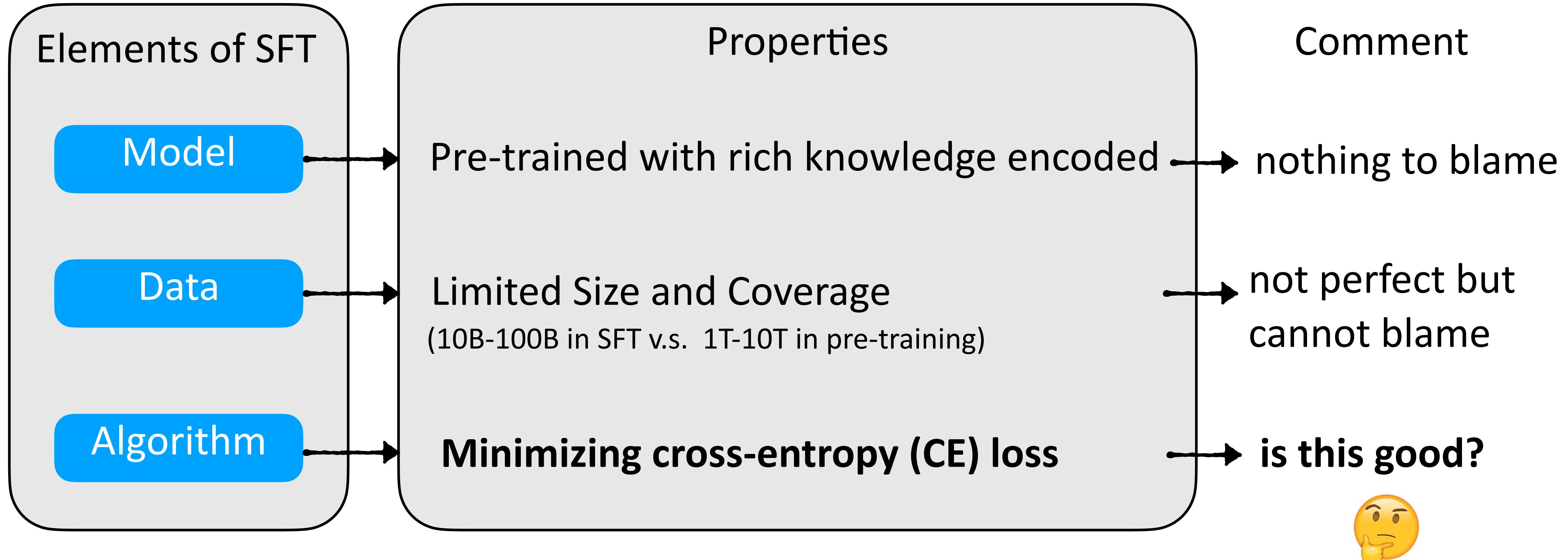
## Open Problems - Preserving Diversity and Interestingness

- How to restore and preserve interestingness and diversity – all the styles and worldviews present in the base models?

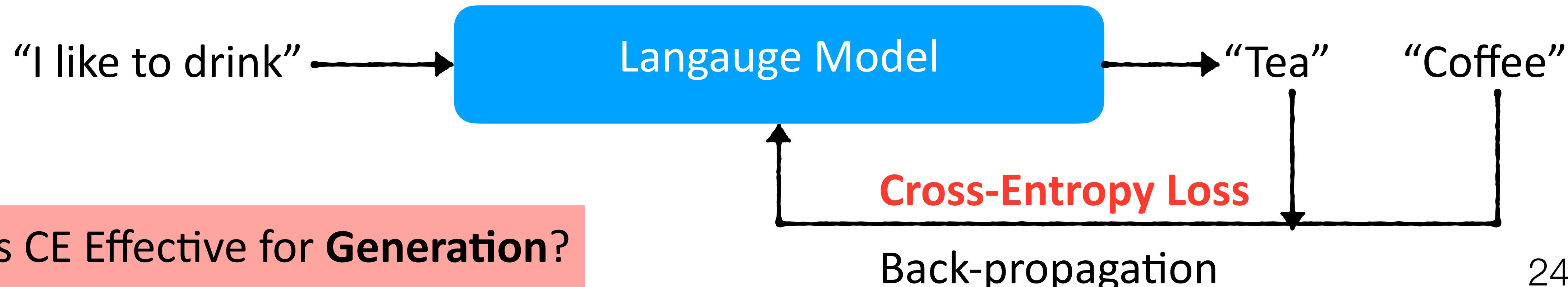
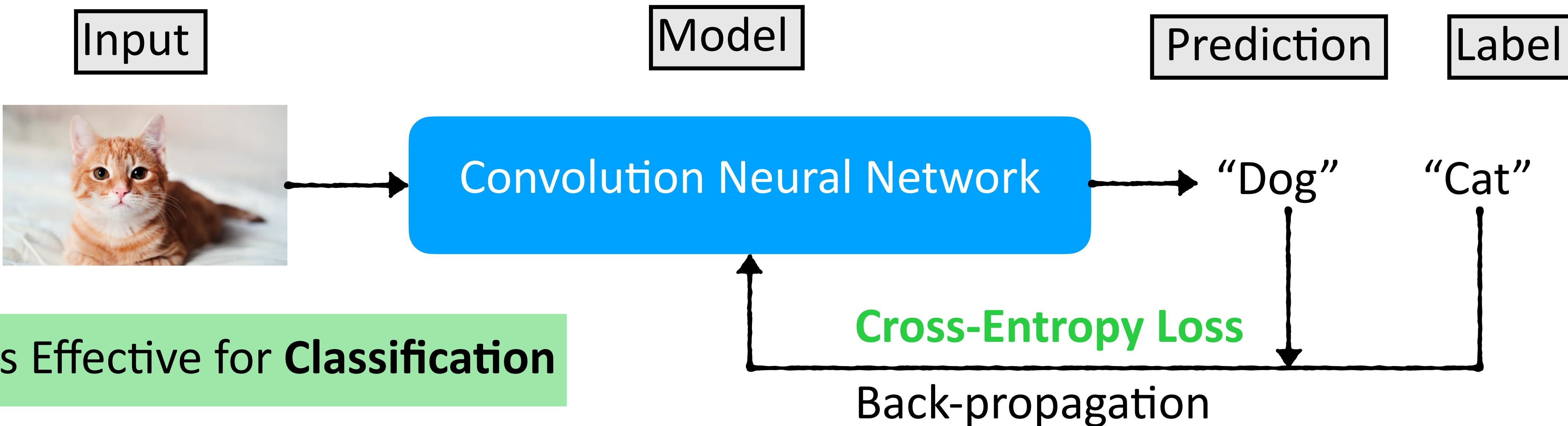
[ChatGPT and The Art of Post-Training. Barret Zoph & John Schulman. <https://docs.google.com/presentation/d/11KWCKUORnPpVMSY6vXgBeFSWo7fJcuGQ9yuR6vC1pzE/edit#slide=id.p>]

# Let's Try to Solve the Problem

---



# CE seems Effective for ...



# Understanding Generation Tasks

---

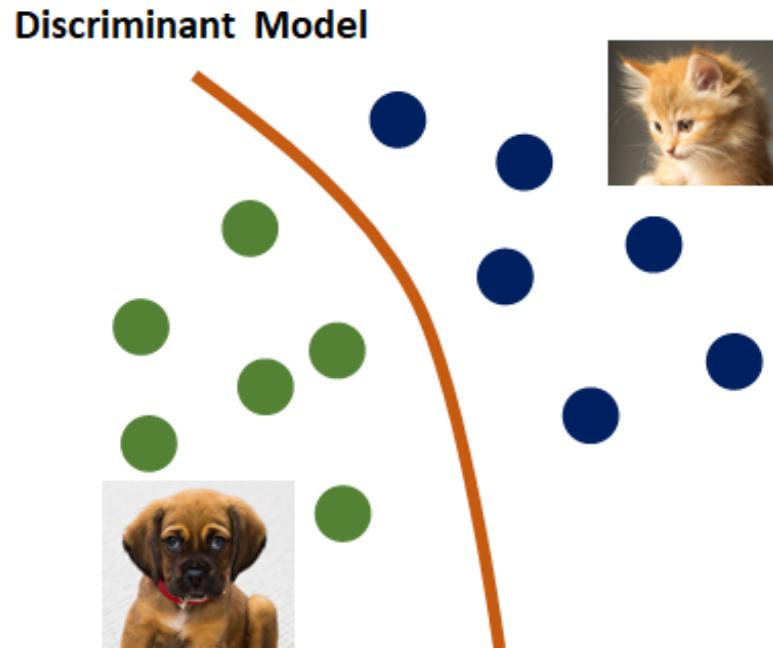
Target

Illustration

Classification

$$\mathcal{X} \mapsto \mathcal{Y}$$

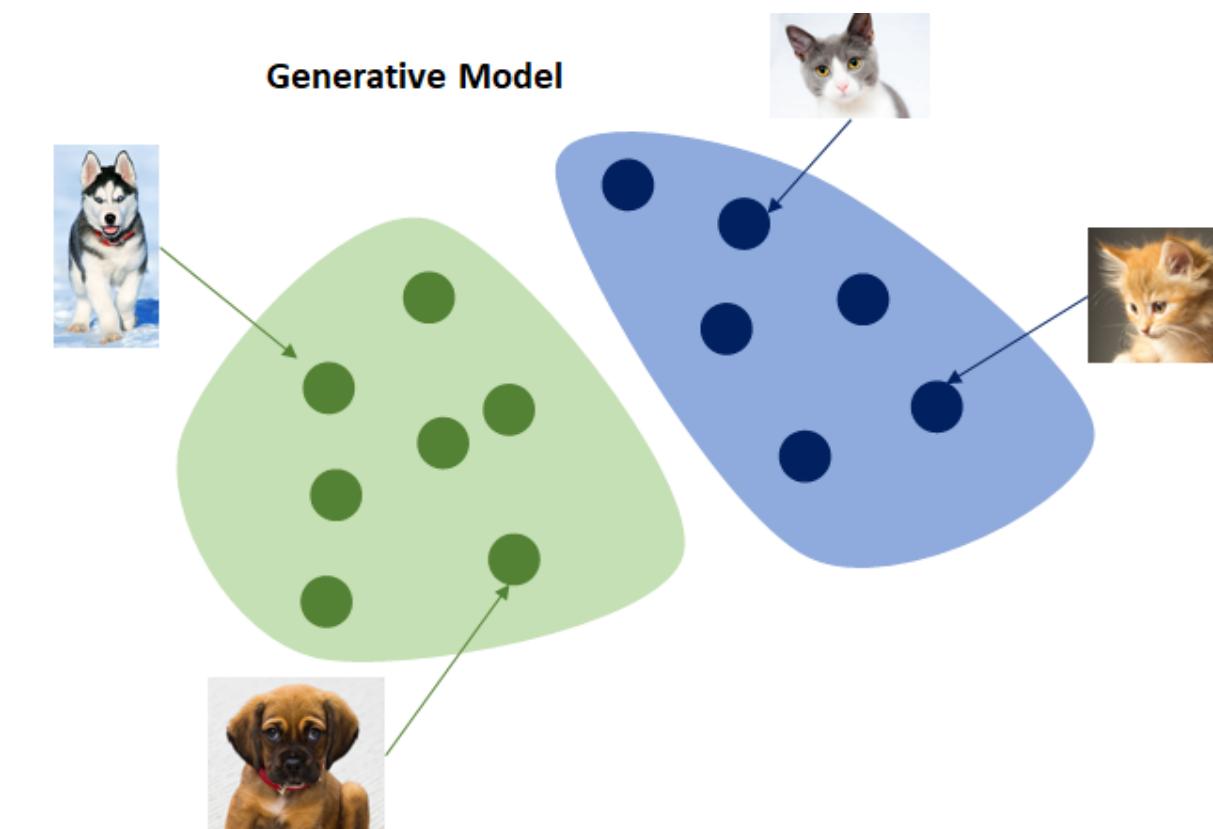
(function: many-to-one)



Generation

$$\mathcal{X} \mapsto \Delta(\mathcal{Y})$$

(distribution: one-to-many)



Remark for LLMs:

- ▶ responses are **not unique**  
(variation in formats, styles, or reasoning paths)
- ▶ (SFT) data is hard to cover all cases

# Theory of CE

## CE Loss (Empirical)

$$\min_{\theta} - \sum_{(x_i, y_i) \sim D} y_i^\top \log f_\theta(y_i | x_i)$$

$(x_i, y_i)$ : input-label pair

$f_\theta(y | x)$ : the conditional prediction distribution

$\theta$ : parameters of neural network



## CE Loss (Population)

$$\max_{\theta} \mathbb{E}_{x \sim \rho} \mathbb{E}_{y \sim p(\cdot | x)} \log f_\theta(y | x)$$

$\rho$ : prompt distribution

$p(\cdot | x)$ : the conditional data distribution to learn

Equivalence

CE can be used to learn a distribution

If the data samples are “abundant”

Classification  
(one label sample  
is enough)

Pre-training  
(huge data)

SFT  
(data is limited)



$$\min_{\theta} \mathbb{E}_{x \sim \rho} \text{KL}(p(\cdot | x), f_\theta(\cdot | x)) + \text{constant}$$

Distribution Matching

# Summary

---

**Challenge:**

We need to protect LLM's output diversity during SFT

**Understanding:**

CE easily fits to the empirical data and loses the diversity

**Goal:**

Designing new formulation and algorithm for SFT

# Analyzing Cross-Entropy Loss

---

Setting:  $y \sim f_\theta(\cdot | x)$  and  $f_\theta(i | x) = \frac{\exp(\theta_i)}{\sum_{j=1}^K \exp(\theta_j)}$

Gradient of CE: assuming  $i$ -th token is the label

$$-\nabla_\theta \mathcal{L}_{\text{CE}}(\theta) = [-f_\theta(1|x), -f_\theta(2|x), \dots, 1 - f_\theta(i|x), \dots, -f_\theta(K|x)].$$

Implication:

Target token (label)'s logit  $\uparrow$  while other tokens' logits  $\downarrow$

# Distribution Matching as Flow Transfer

**Proposition 1.** *The gradient of CE specifies a logit flow map: each source token  $j$  transfers  $f_\theta(j|x)$  logits to the target token  $i$ . Formally,*

$$\begin{aligned} -\nabla_\theta \mathcal{L}_{\text{CE}}(\theta) &= \sum_{j=1, j \neq i}^K w_{i \leftarrow j} \cdot e_{i \leftarrow j} & (2) \\ w_{i \leftarrow j} &= f_\theta(j|x) \\ e_{i \leftarrow j} &= [0 \ \dots \underbrace{1}_{i\text{-th position}} \ \dots \underbrace{-1}_{j\text{-th position}} \ \dots \ 0] \end{aligned}$$

Example:

$$f_\theta = [0.1, 0.3, 0.6]$$

Label: #2

Gradient:

$$g = [-0.1, 0.7, -0.6]$$

Flow perspective:  $g = 0.1 * [-1 \ 1 \ 0] + 0.6 * [0 \ 1 \ -1]$

Logits flow from **source** tokens = Logits flow to **target** token

# Limitations of CE

## Procedure of CE

#1 While there exists source token  $j \neq i$  with  $f_{\theta_k}(j|x) > 0$ , continue the following steps.

- Find any  $j$  with  $f_{\theta_k}(j|x) > 0$
- Decrease the logit for source token  $j$  by learning rate  $\eta$  and weight  $w_{i \leftarrow j}$ :

$$\theta_{k+1}[j] = \theta_k[j] - \eta * w_{i \leftarrow j}$$

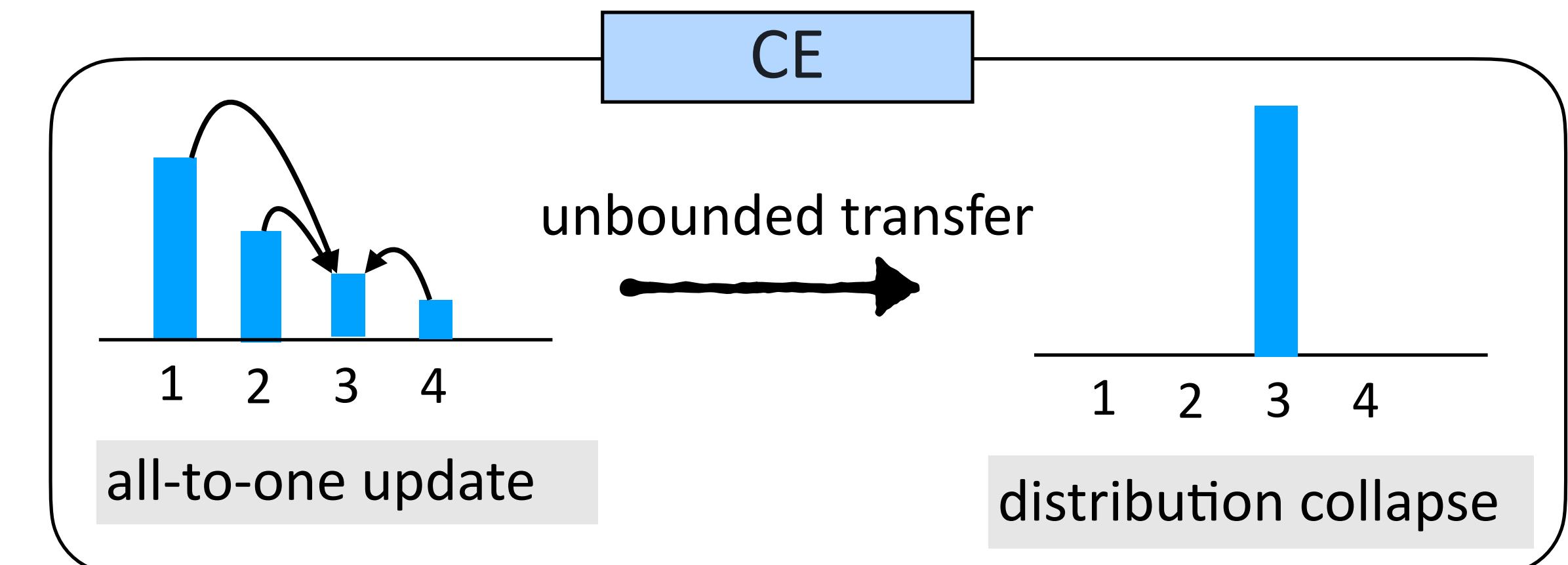
#2

- Increase the logit for the target token  $i$  in a similar manner:

$$\theta_{k+1}[i] = \theta_k[i] + \eta * w_{i \leftarrow j}$$

## Limitation 1: Unbounded Transfer

## Limitation 2: All-to-one Update



# Proposed Solutions

## Procedure of Our Method

#1

While the target token  $i \notin \operatorname{argmax} f_{\theta_k}(\cdot|x)$ , continue the following steps.

- Calculate the model's best prediction  $j = \operatorname{argmax} f(\cdot|x)$

- Decrease the logit for source token  $j$  by learning rate  $\eta$  and weight  $w_{i \leftarrow j}$ :

$$\theta_{k+1}[j] = \theta_k[j] - \eta * w_{i \leftarrow j}$$

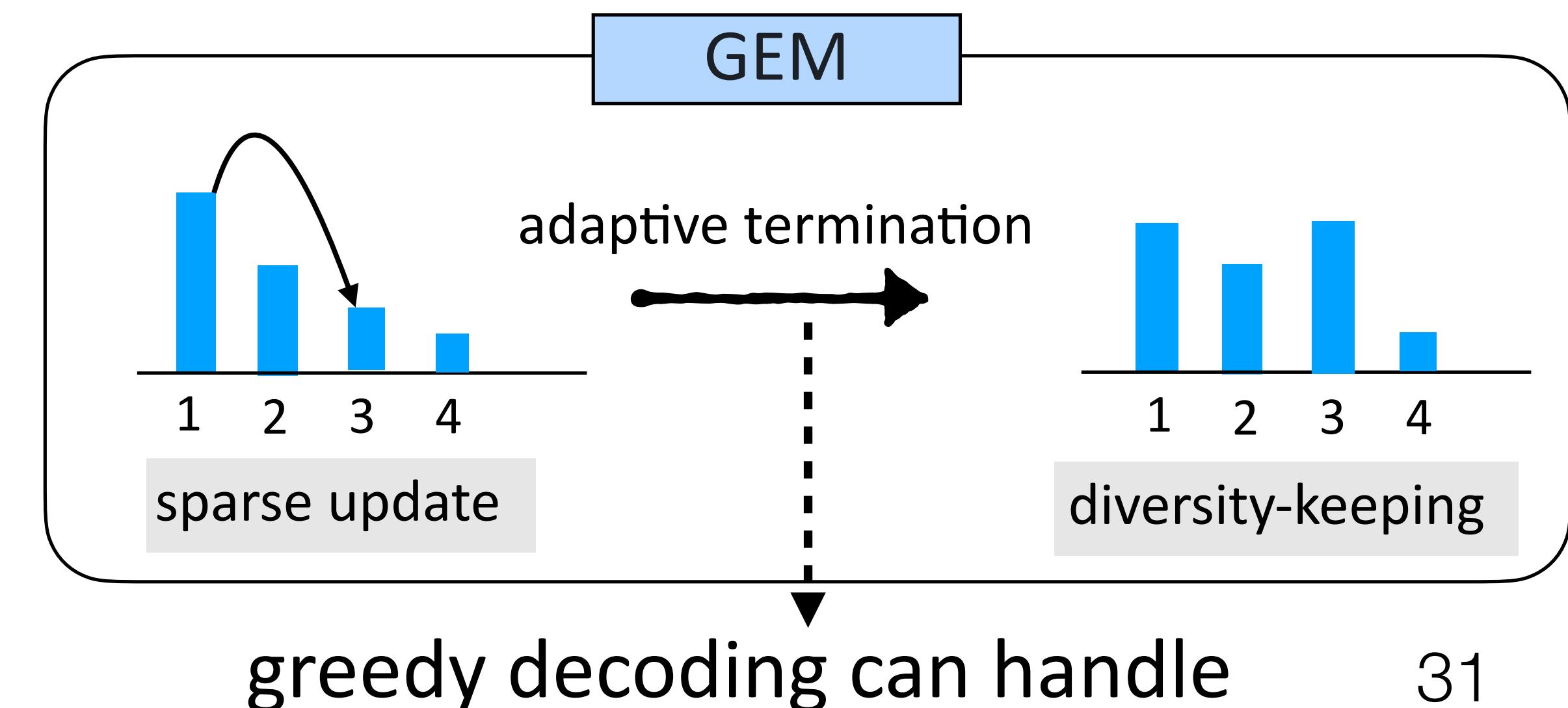
#2

- Increase the logit for the target token  $i$  in a similar manner:

$$\theta_{k+1}[i] = \theta_k[i] + \eta * w_{i \leftarrow j}$$

Technique 1: Adaptive Termination

Technique 2: Sparse Update



# Our Insight: Dimension Increase

---

## Procedure of Our Method

While the target token  $i \notin \operatorname{argmax} f_{\theta_k}(\cdot|x)$ , continue the following steps.

- Calculate the model's best prediction  $j = \operatorname{argmax} f(\cdot|x)$
- Decrease the logit for source token  $j$  by learning rate  $\eta$  and weight  $w_{i \leftarrow j}$ :

$$\theta_{k+1}[j] = \theta_k[j] - \eta * w_{i \leftarrow j}$$

- Increase the logit for the target token  $i$  in a similar manner:

$$\theta_{k+1}[i] = \theta_k[i] + \eta * w_{i \leftarrow j}$$



What is the magic? Can we generalize this to neural network training?



Introduce an **auxiliary variable** (dimension increase) that implements the scheme of sparse update and adaptive termination

# Towards a Game Formulation

---

High-level design: introduce an another player  $q$  to the distribution matching

$$\begin{aligned} \min_f \quad & \mathcal{L}(f, q) \triangleq \mathbb{E}_x \mathbb{E}_{y^{\text{real}} \sim p(\cdot|x)} \mathbb{E}_{y^{\text{gene}} \sim q(\cdot|x)} [\log f(y^{\text{gene}}|x) - \log f(y^{\text{real}}|x)] \\ \max_q \quad & \mathcal{Q}(f, q) \triangleq \mathbb{E}_x \mathbb{E}_{y^{\text{gene}} \sim q(\cdot|x)} [\log f(y^{\text{gene}}|x)] + \beta \cdot \mathcal{H}(q(\cdot|x)). \end{aligned}$$

Intuitive Understanding:

- ▶  $f$ : increase the likelihood on real data and decrease likelihood on the generated data
- ▶  $q$ : increase the energy induced by  $\log f$  with entropy regularization

# Understanding the Game

main player

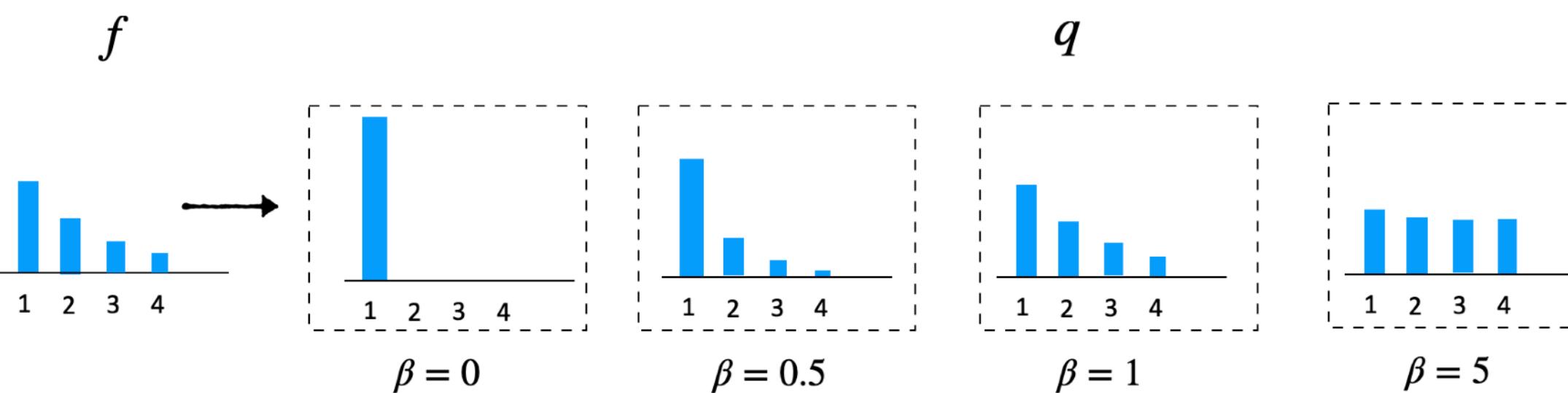
$$\begin{aligned} -\nabla_{\theta} \mathcal{L}(f_{\theta}, q) &= \sum_{j=1, j \neq i}^K w_{i \leftarrow j} \cdot e_{i \leftarrow j}, \\ w_{i \leftarrow j} &= q(j|x). \end{aligned}$$

flow transfer

controller

meta-controller

$$\operatorname{argmax}_q Q(q, f) = \begin{cases} \delta_j(x) \text{ with } j = \operatorname{argmax} f_i(\cdot|x) & \text{if } \beta = 0 \\ \text{softmax}(1/\beta * \log f(y|x)) & \text{if } \beta > 0 \end{cases}$$



$\beta \rightarrow 0$ : sparse update

$\beta \rightarrow 1$ : same as CE

$\beta \rightarrow \infty$ : uniform update

# Connection with Probability Transfer

**Proposition 2.** For a data distribution satisfying  $p(y|x) > 0$ , with  $\beta > 0$ , the game in Equations (3) and (4) posses a unique Nash equilibrium point:

$$\begin{cases} f^* = \text{softmax}(\beta * \log p) \\ q^* = p \end{cases} \quad (7)$$

Furthermore,  $f^*$  corresponds to the optimal solution to the distribution matching problem (with  $1/\beta = (\gamma + 1)$ ), which minimizes the reverse KL divergence with entropy regularization:

$$f^* = \underset{f}{\operatorname{argmin}} \mathbb{E}_x [D_{\text{KL}}(f(\cdot|x), p(\cdot|x)) - \gamma \mathcal{H}(f(\cdot|x))] . \quad (8)$$

Terminology	Reserve KL Minimization	Entropy Maximization
Role	Fit the data distribution	Protect the output diversity

For  $\beta = 0$ , there are **multiple** Nash equilibrium points with non-closed-form solutions → future work

# Training Algorithm

Idea: block-wise gradient-descent and coordinate descent

$$\begin{cases} f_{\theta_{k+1}} = f_{\theta_k} - \nabla_{\theta} \mathcal{L}(f_{\theta}, q_k) \mid_{\theta=\theta_k} \\ q_{k+1} = \operatorname{argmax}_q Q(f_{\theta_{k+1}}, q) = \operatorname{softmax}(1/\beta * \log f_{\theta_{k+1}}) \end{cases}$$

Feature 1: **Single**-model optimization

↳ There is no need of storing and explicit training of  $q$

Optimization with the token space (**discrete**)

Feature 2: **Variance-reduced** gradient estimation

$$\mathcal{L}_{\text{GEM}}(\theta) = \sum_i \sum_{y^{\text{gene}}} q_k(y^{\text{gene}} | x_i) \cdot [\log f_{\theta}(y^{\text{gene}} | x_i) - \log f_{\theta}(y_i^{\text{real}} | x_i)]$$

↳ We use the exact distribution (in GANs, stochastic approximation is used)

# Discussion: Difference with GANs

---

**GAN**

(generative adversarial network)

**GEM**

(game-theoretic entropy maximization)

Task	Image Generation	Text Generation
------	------------------	-----------------

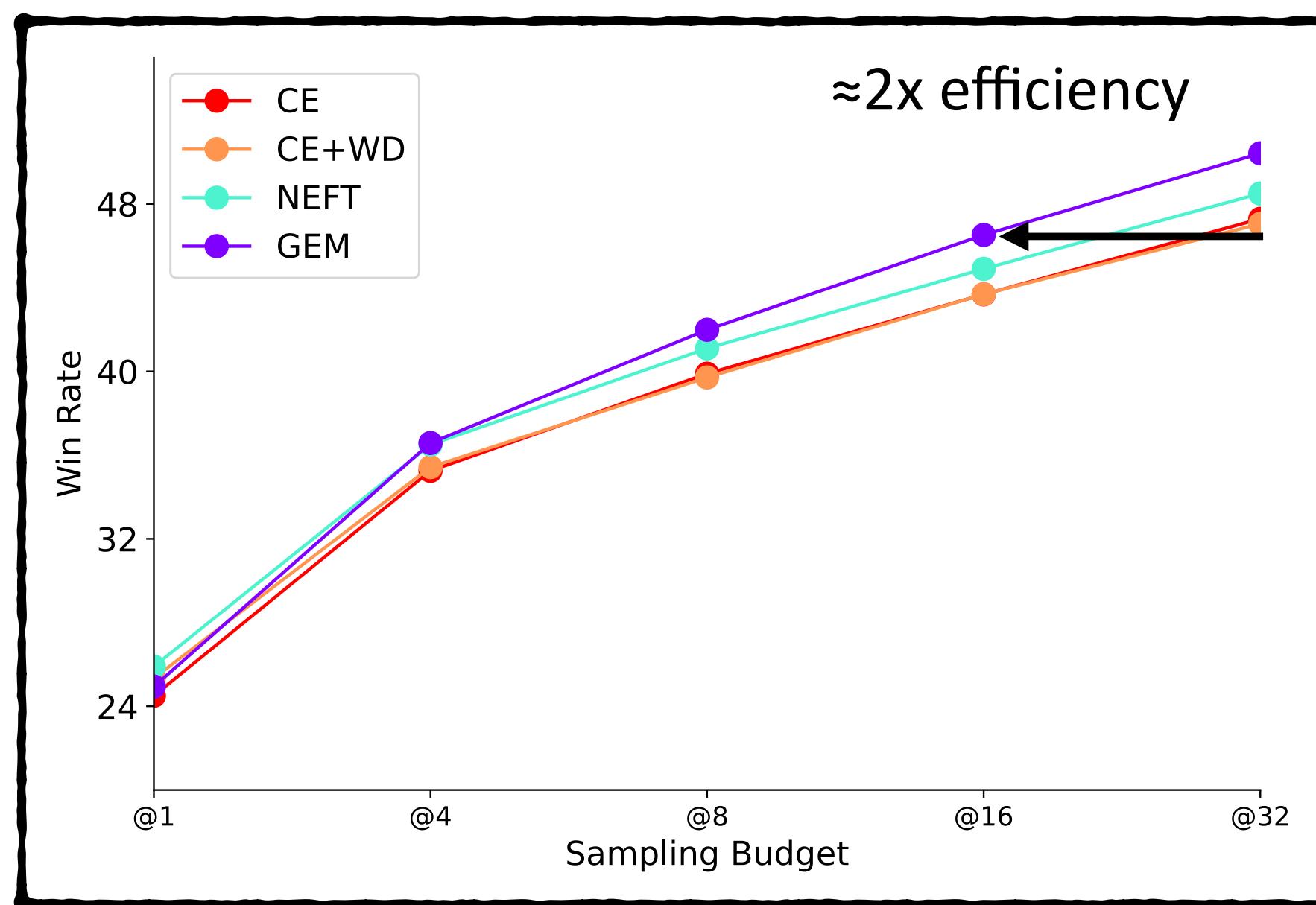
Challenge	Estimation the distance among two images is hard	Overfitting the data and losing output diversity
-----------	--	--

Idea	Introduction of discriminator	Introduction of flow-controller
------	-------------------------------	---------------------------------

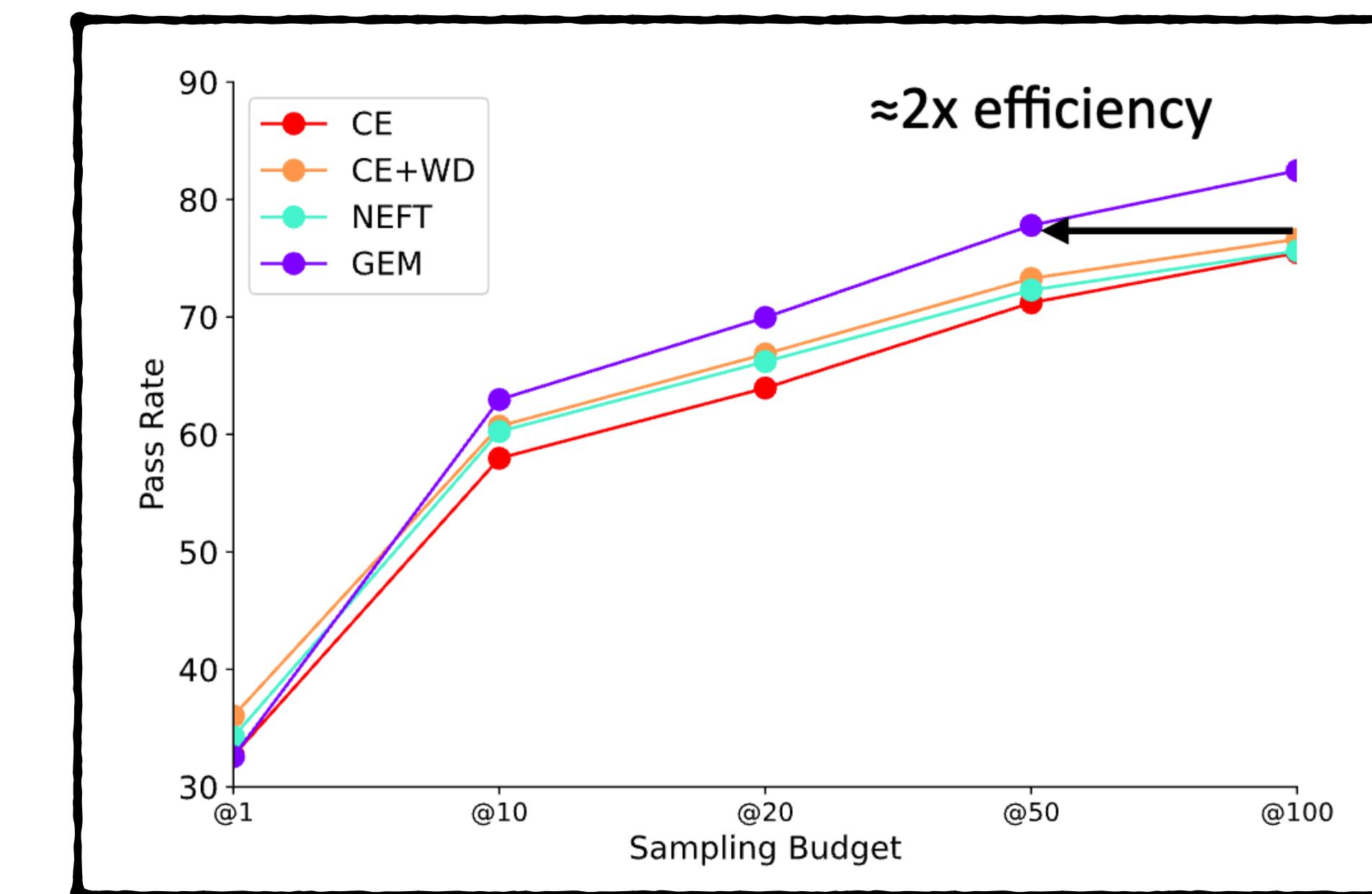
Computation Complexity	High	Low
------------------------	------	-----

# Test-Time Scaling

- ▶ Evaluation Method: Best-of-N Sampling
- ▶ Model: Llama-3.1-8B; Dataset: Ultrafeedback



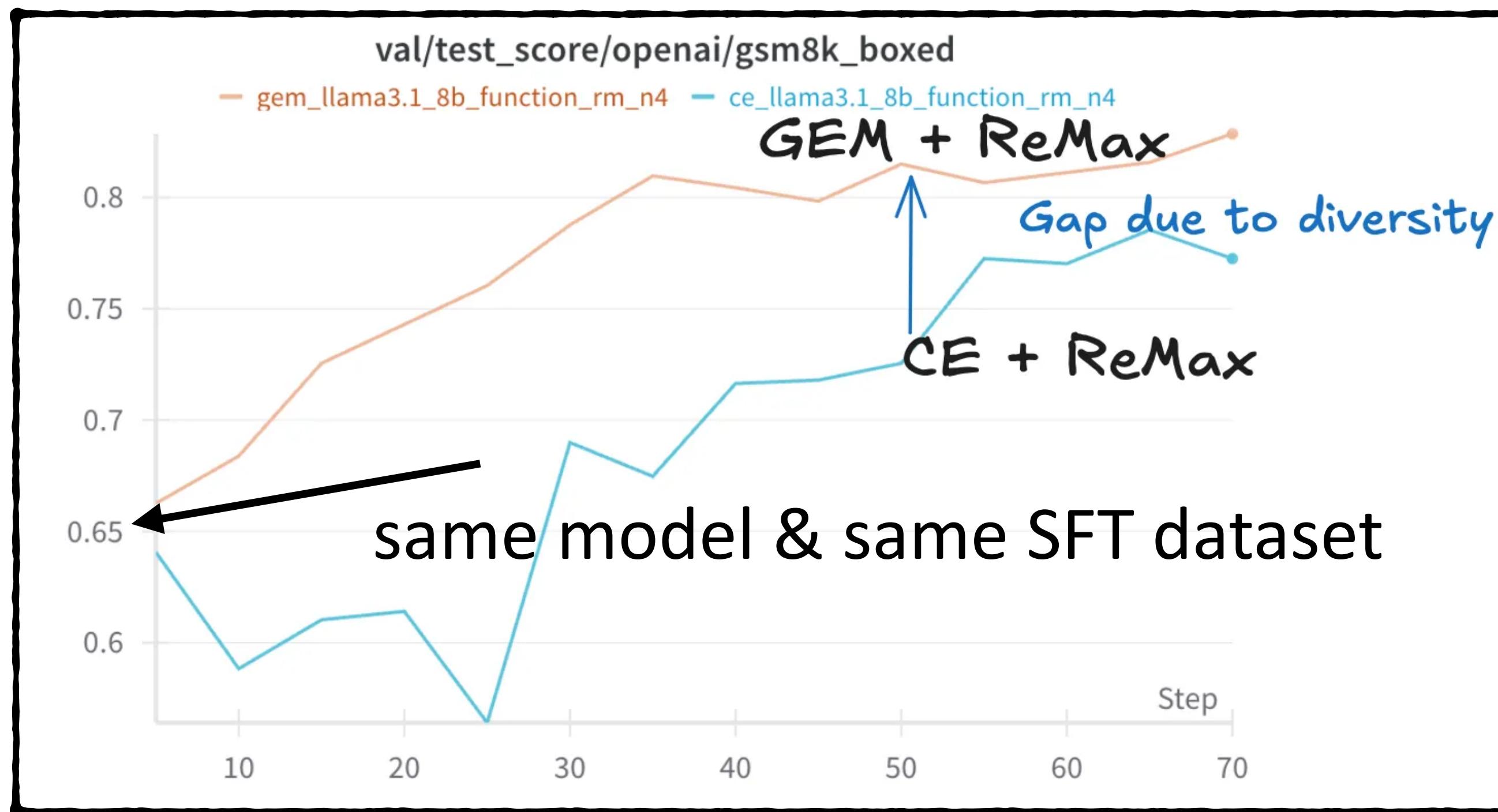
RLHF Alignment (Chat)



Code Generation

GEM requires about **2x** less sampling budget for comparable performance

# Math Reasoning



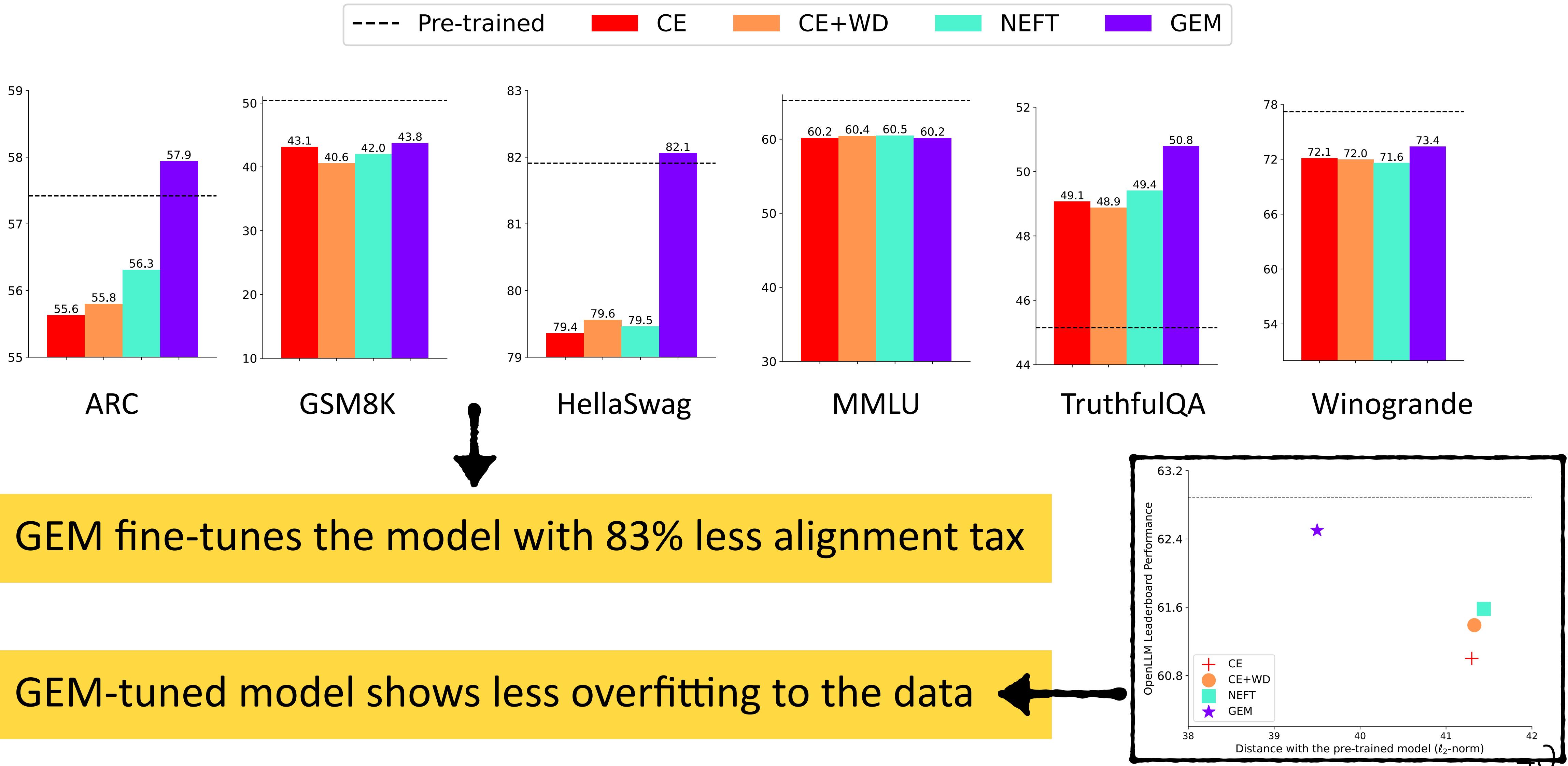
- ▶ Task: optimize CoT (reasoning steps) to answer math questions
- ▶ Reward: accuracy of final reward
- ▶ Model: Qwen-2.5-3B
- ▶ RL Algo: ReMax

[Li, Ziniu, et al. "Remax: A simple, effective, and efficient reinforcement learning method for aligning large language models." ICML 2024.]

[<https://tangible-polo-203.notion.site/>]

GEM improves the performance limit of RL training

# Alignment Tax



# PRESERVING DIVERSITY IN SUPERVISED FINE-TUNING OF LARGE LANGUAGE MODELS

Ziniu Li<sup>1,2</sup>, Congliang Chen<sup>1,2</sup>, Tian Xu<sup>3</sup>, Zeyu Qin<sup>4</sup>, Jiancong Xiao<sup>5</sup>,  
Zhi-Quan Luo<sup>1,2</sup>, and Ruoyu Sun<sup>1,2,†</sup>

ICLR 2025

NeurIPS 2024 FITML Workshop Best Paper Runner-up



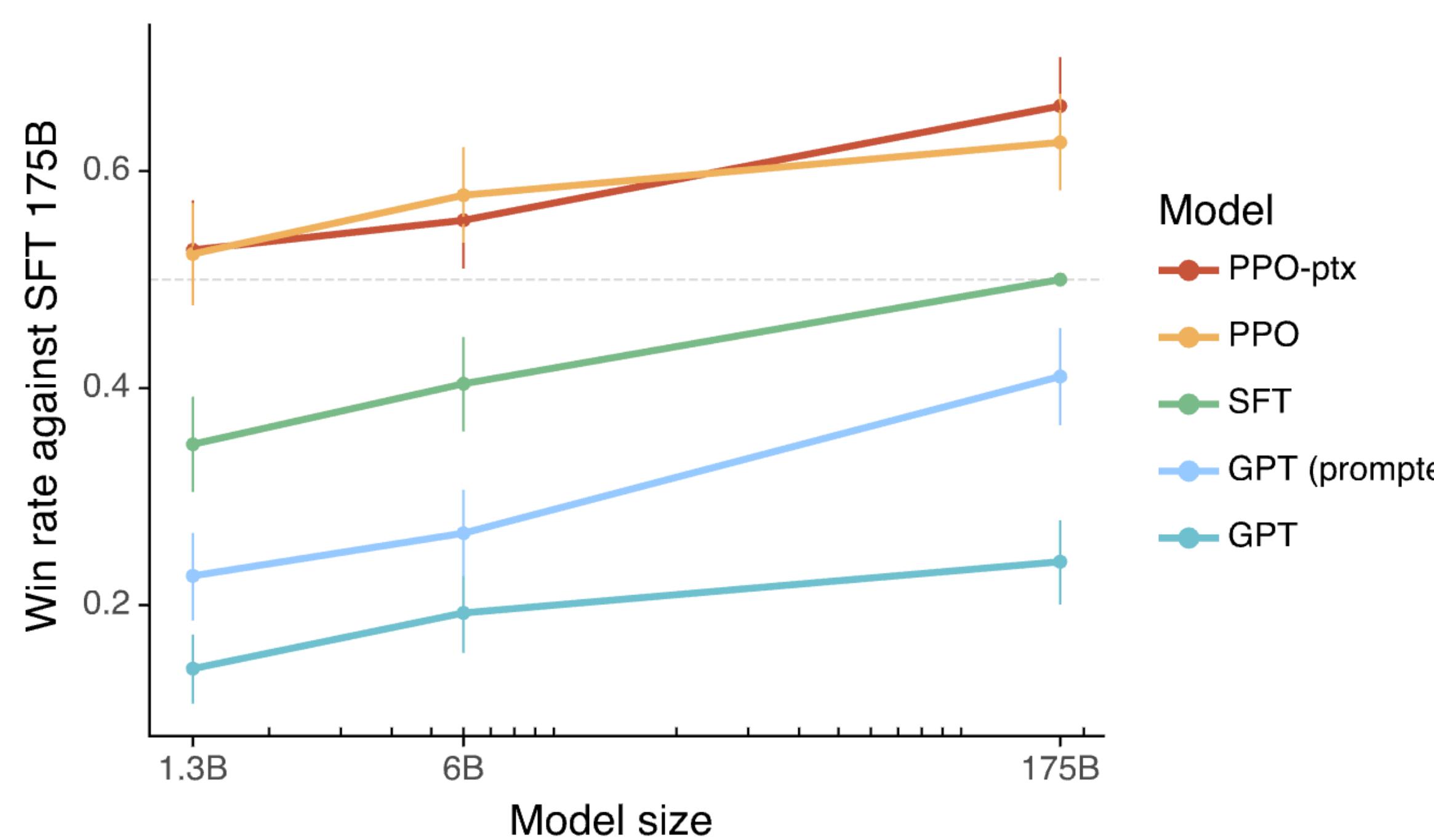
Paper



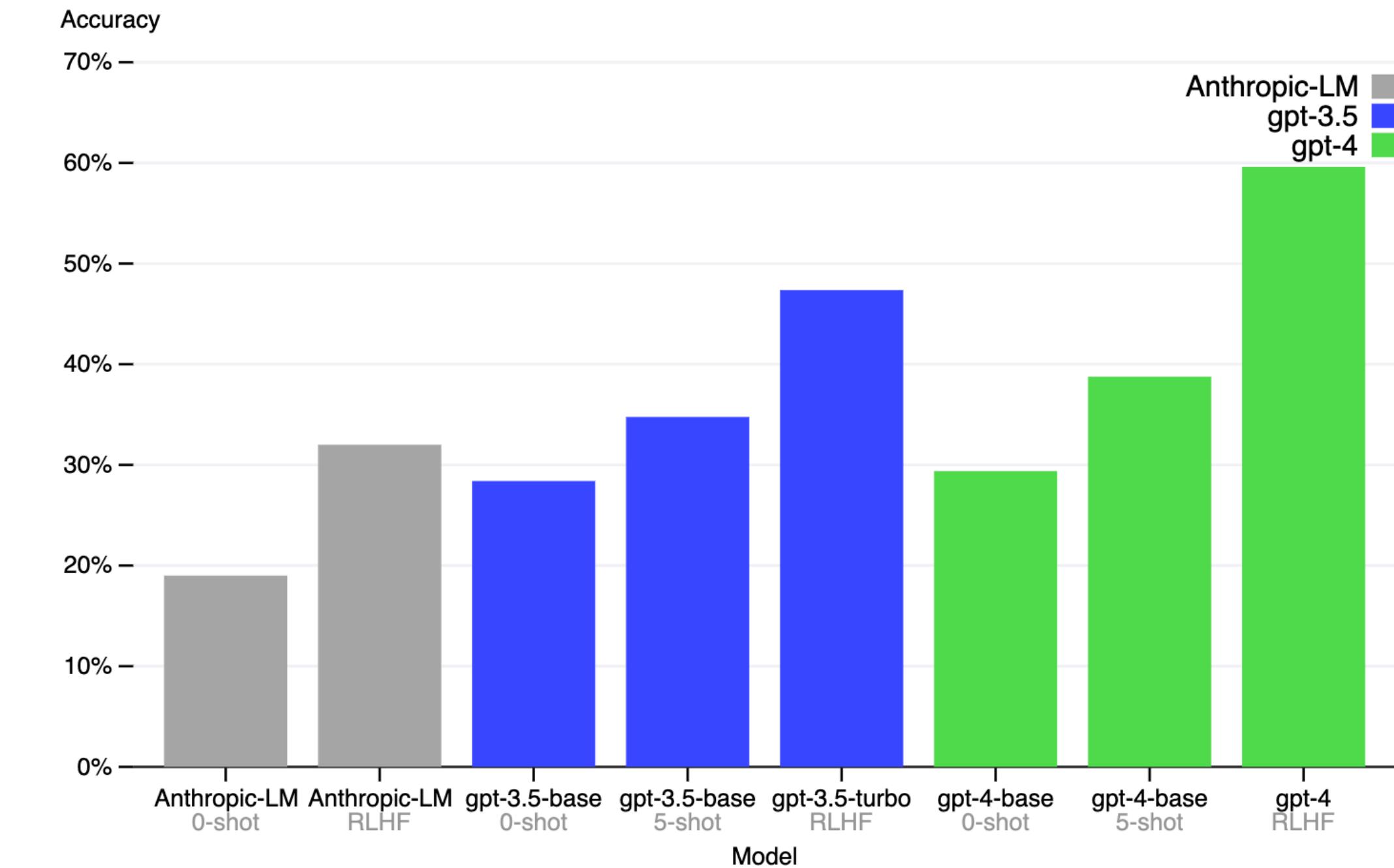
Code

# Part IV: Efficient and Scalable Reinforcement Learning in LLMs

# RL Task: Alignment



Accuracy on adversarial questions (TruthfulQA mc1)



Only PPO Achieves a Win Rate Above 50%

[Ouyang, Long, et al. "Training language models to follow instructions with human feedback." *NeurIPS 2022*.]

RLHF Enhances Acc. by More Than 10%

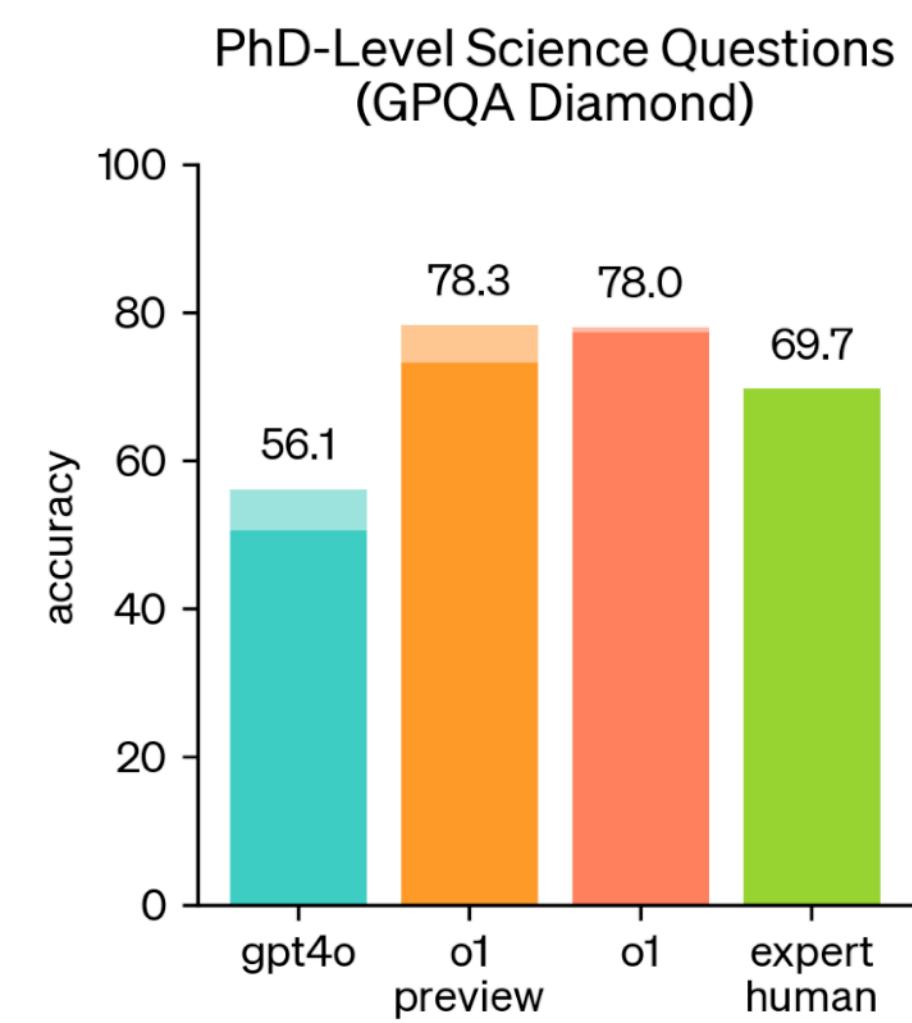
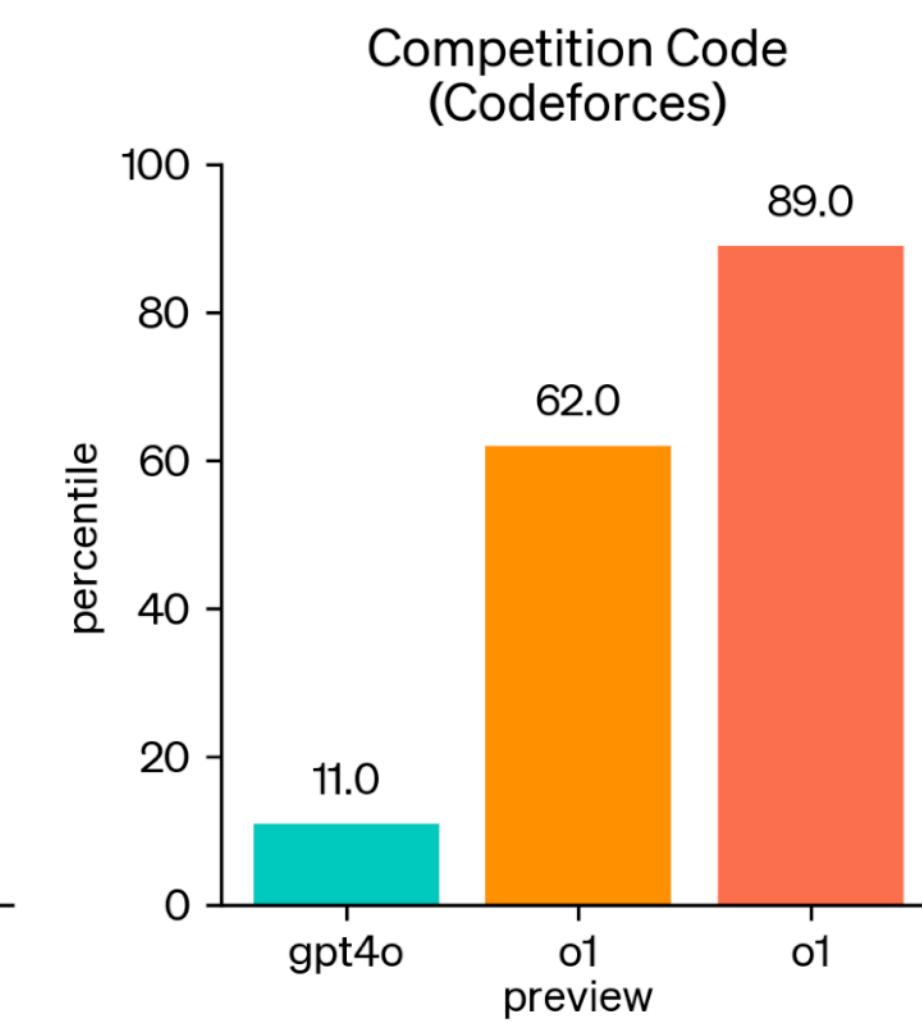
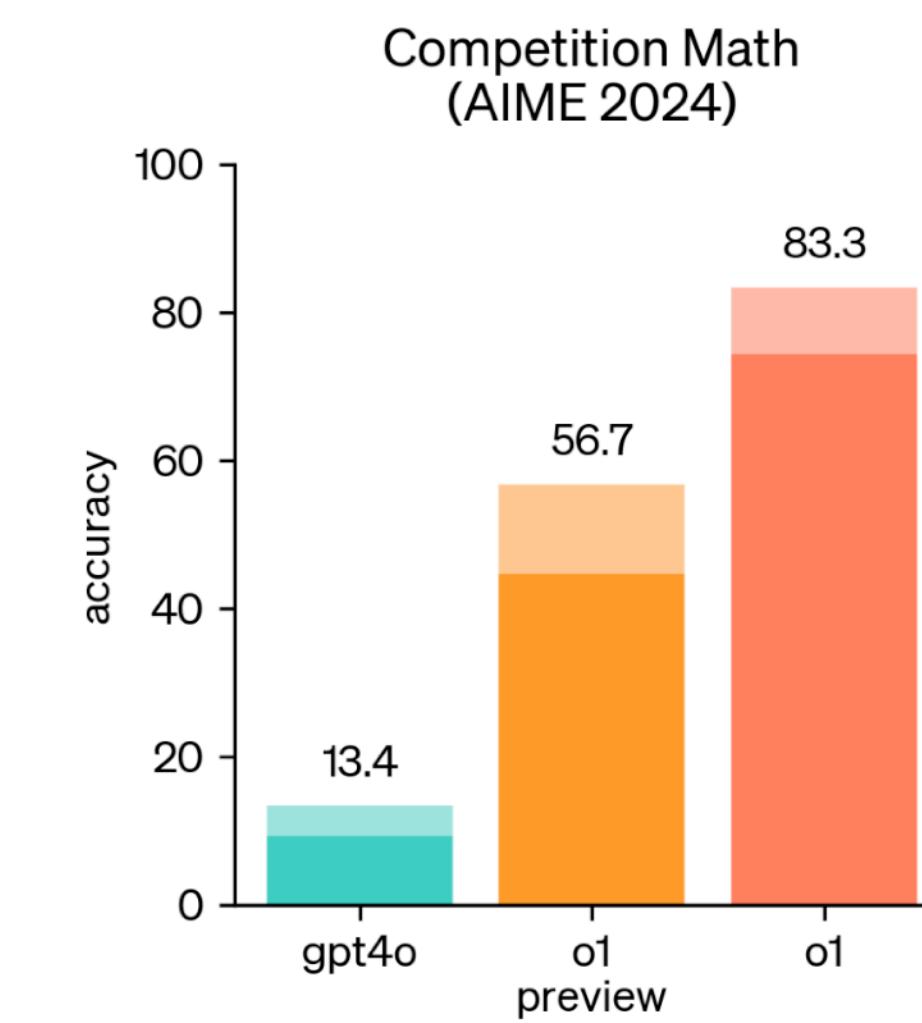
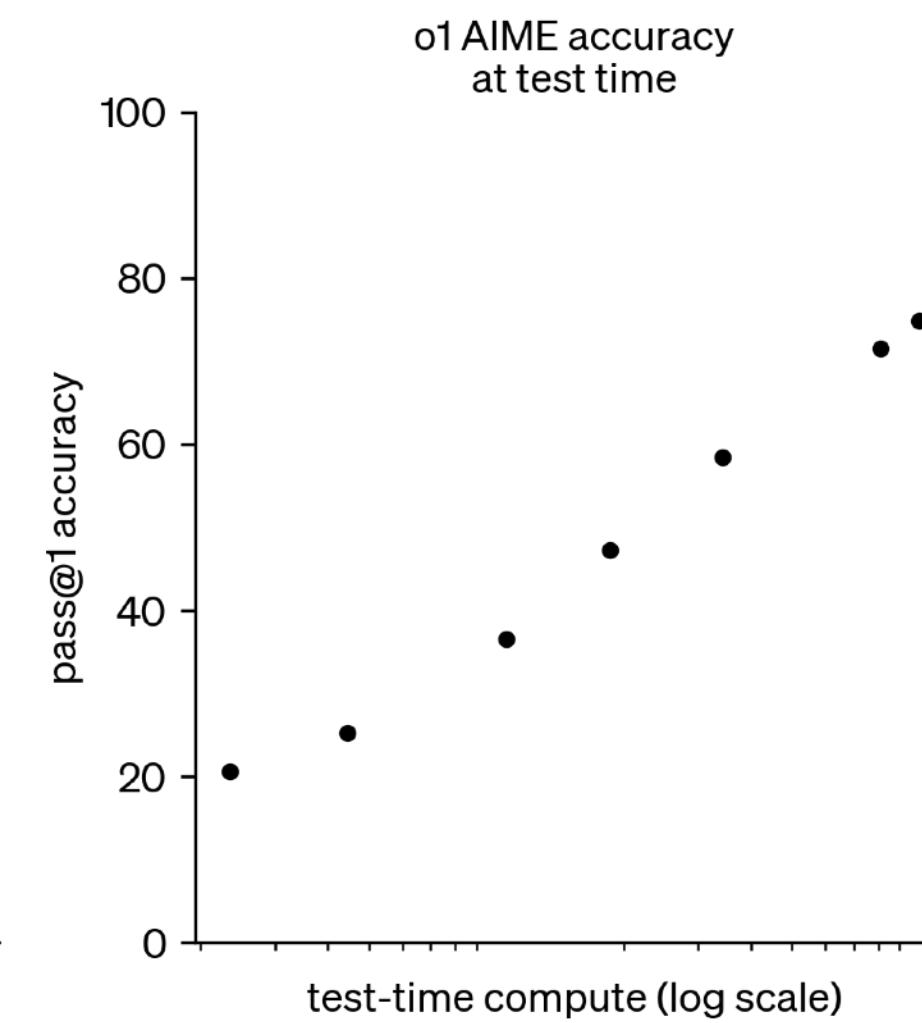
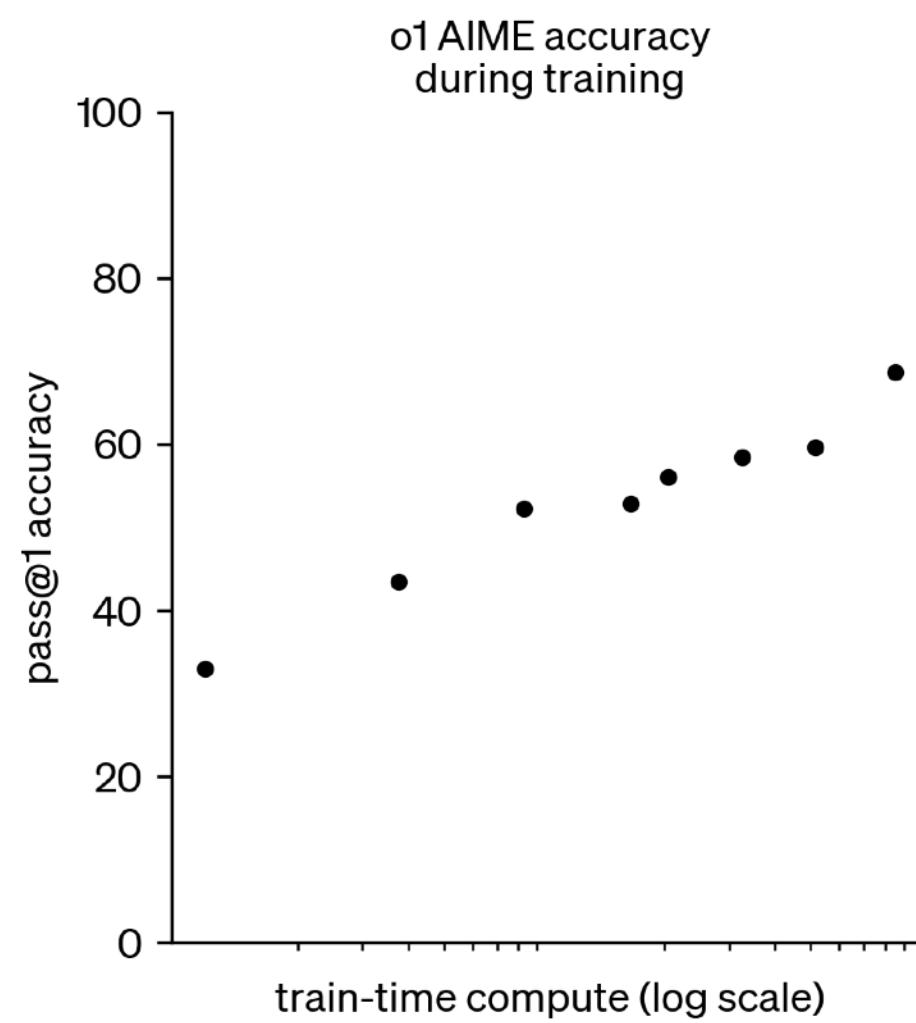
[Achiam, Josh, et al. "Gpt-4 technical report." *arXiv preprint arXiv:2303.08774* (2023).]

# RL Task: Eliciting Reasoning

Test-time Scaling



Huge Improvement in Challenging Tasks

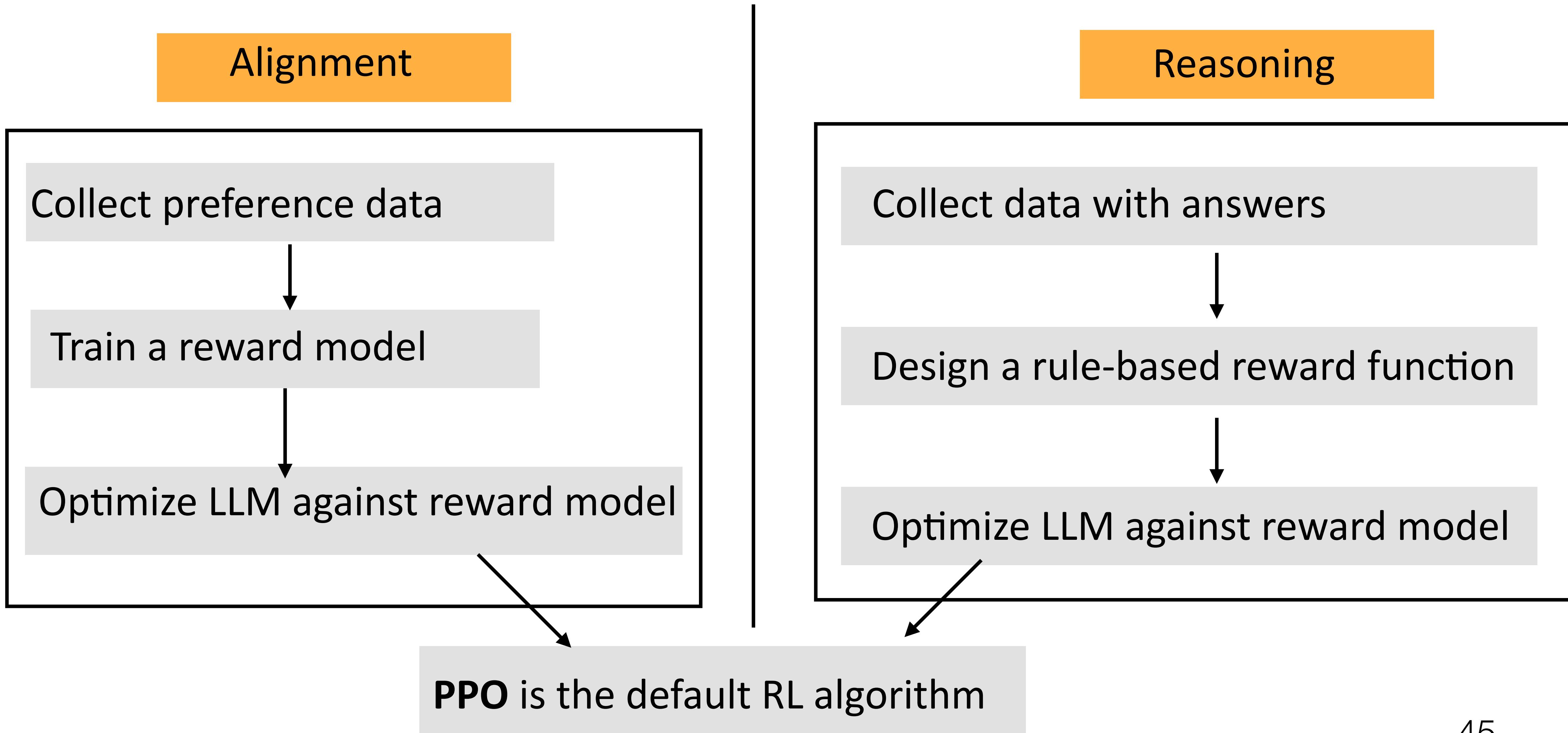


[<https://openai.com/index/learning-to-reason-with-langs/>]

RL training enables models to think deep

o1 can exceed GPT-4o by 40+ points on MATH, code, and PhD-Level QA

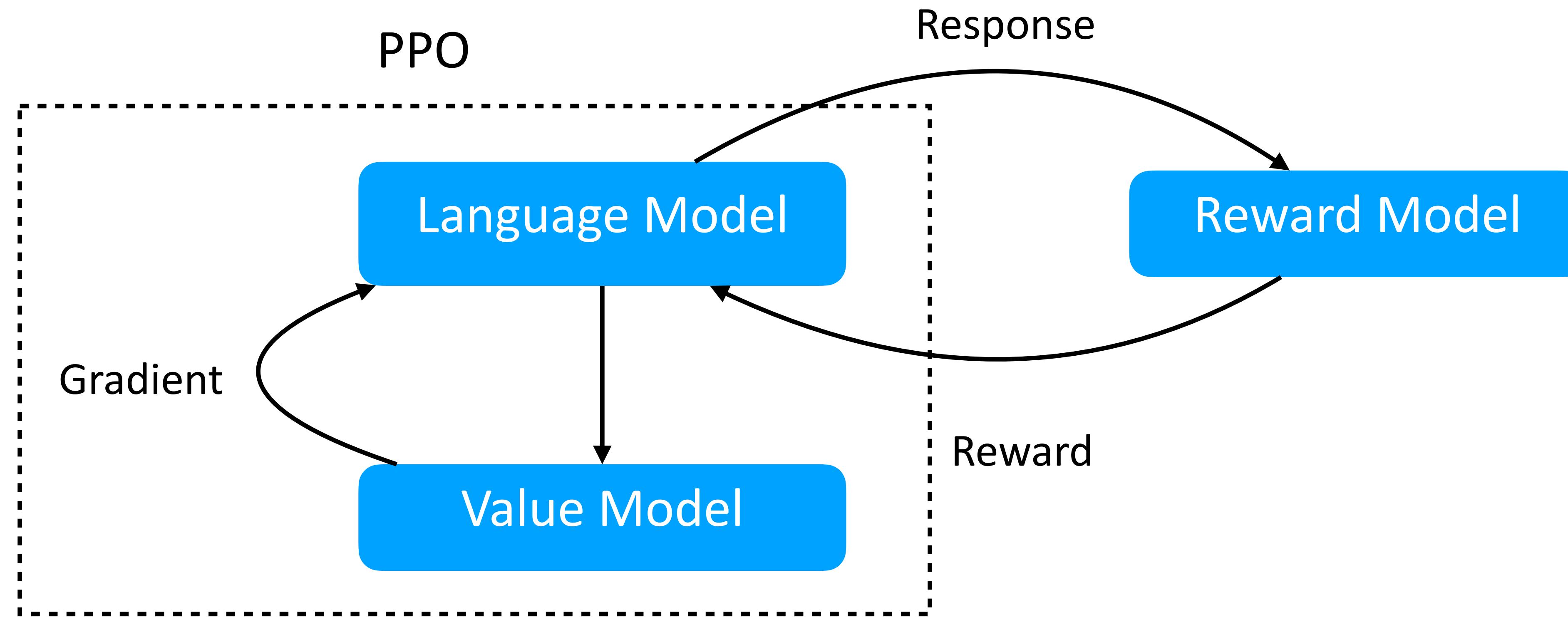
# How does RL work in LLMs?



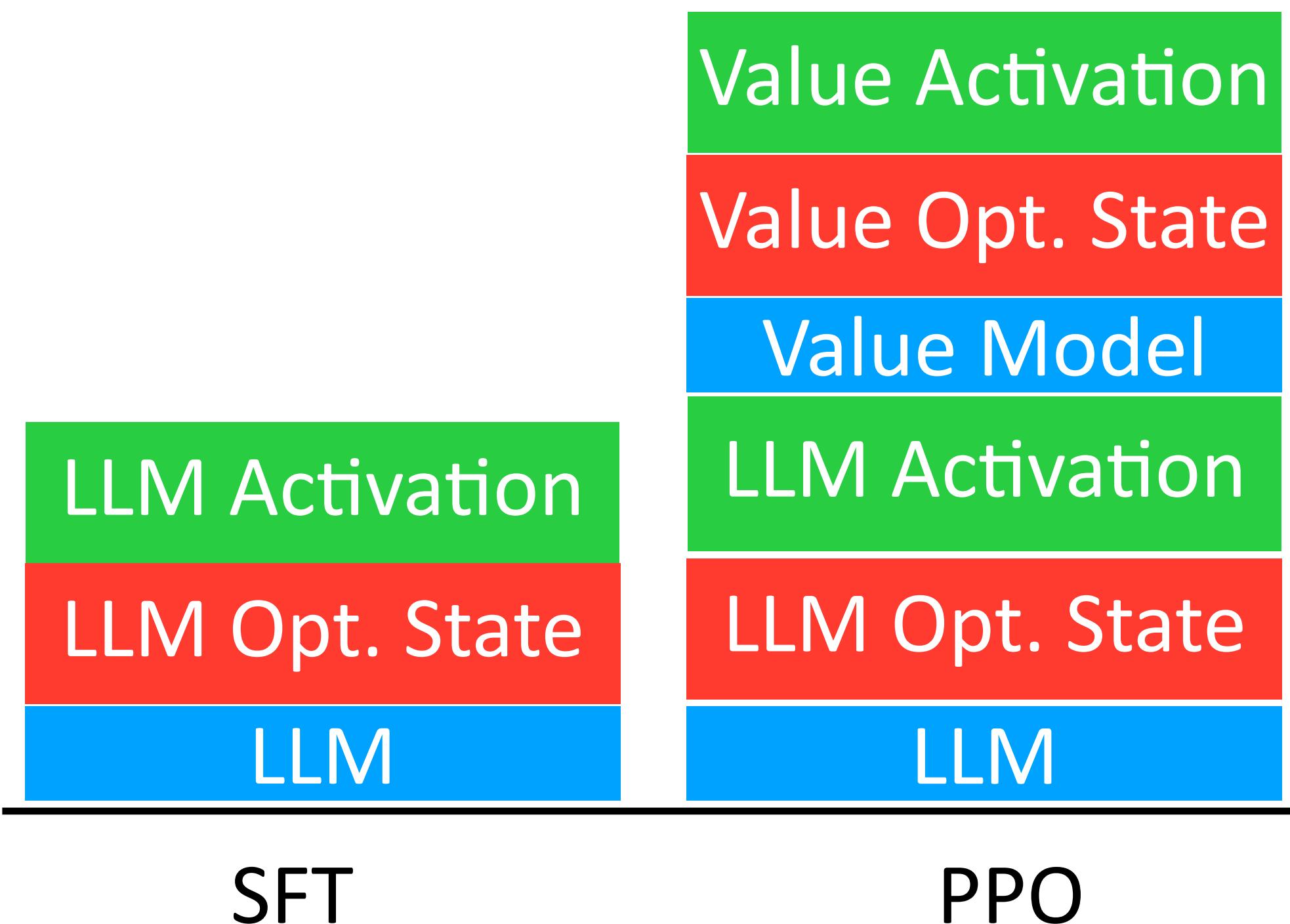
# Introduction to PPO

Objective:

$$\max_{\theta} \mathbb{E}_{y_{1:T} \sim \pi_{\theta}(\cdot|x)} [r(x, y_{1:T})]$$



# PPO is Computationally Inefficient



PPO's training takes more memory

PPO's training is slow

**Value model** is the bottleneck of PPO

Table 4: E2E time breakdown for training a 13 billion parameter ChatGPT model via DeepSpeed-Chat on a single DGX node with 8 NVIDIA A100-40G GPU.

Model Sizes	Step 1	Step 2	Step 3	Total
Actor: OPT-13B, Reward: OPT-350M	2.5hr	0.25hr	10.8hr	13.6hr

[Yao, Zhewei, et al. "DeepSpeed-Chat: Easy, Fast and Affordable RLHF Training of ChatGPT-like Models at All Scales." *arXiv:2308.01320* (2023)]

# Can We Improve PPO?

---



Can we achieve RL training without the value model?



If Yes, we can save memory and accelerate training



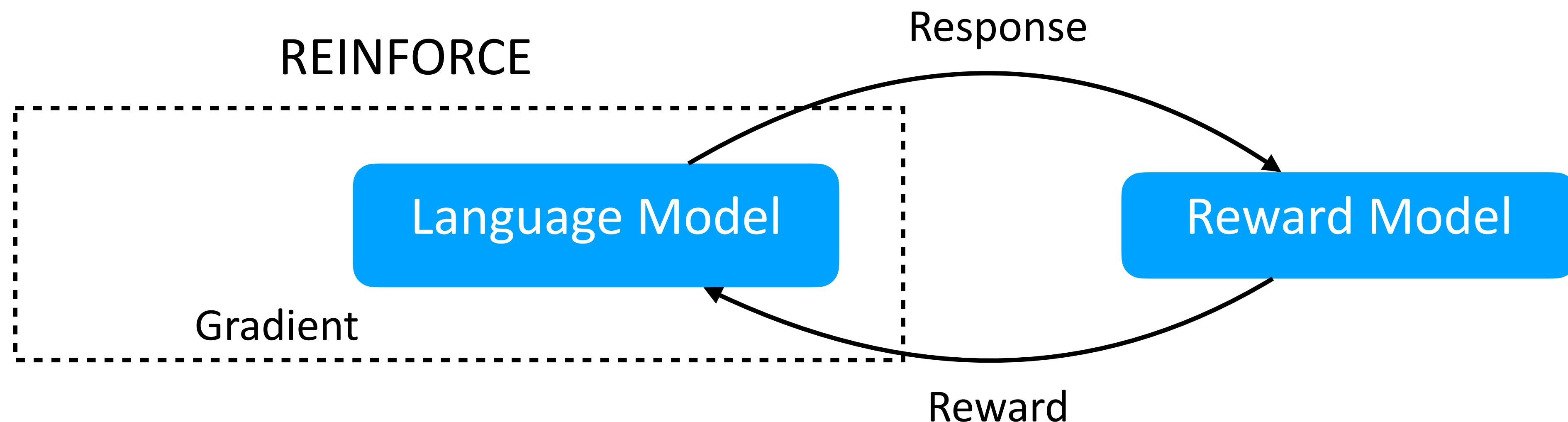
**REINFORCE** is an RL algorithm without value model

[Williams, Ronald J. "Simple statistical gradient-following algorithms for connectionist reinforcement learning." *Machine learning* 8 (1992): 229-256.]

# Introduction to REINFORCE

Objective:

$$\max_{\theta} \mathbb{E}_{y_{1:T} \sim \pi_{\theta}(\cdot|x)} [r(x, y_{1:T})]$$



[Williams, R. J. Reinforcement-learning connectionist systems. College of Computer Science, Northeastern University, 1987.]

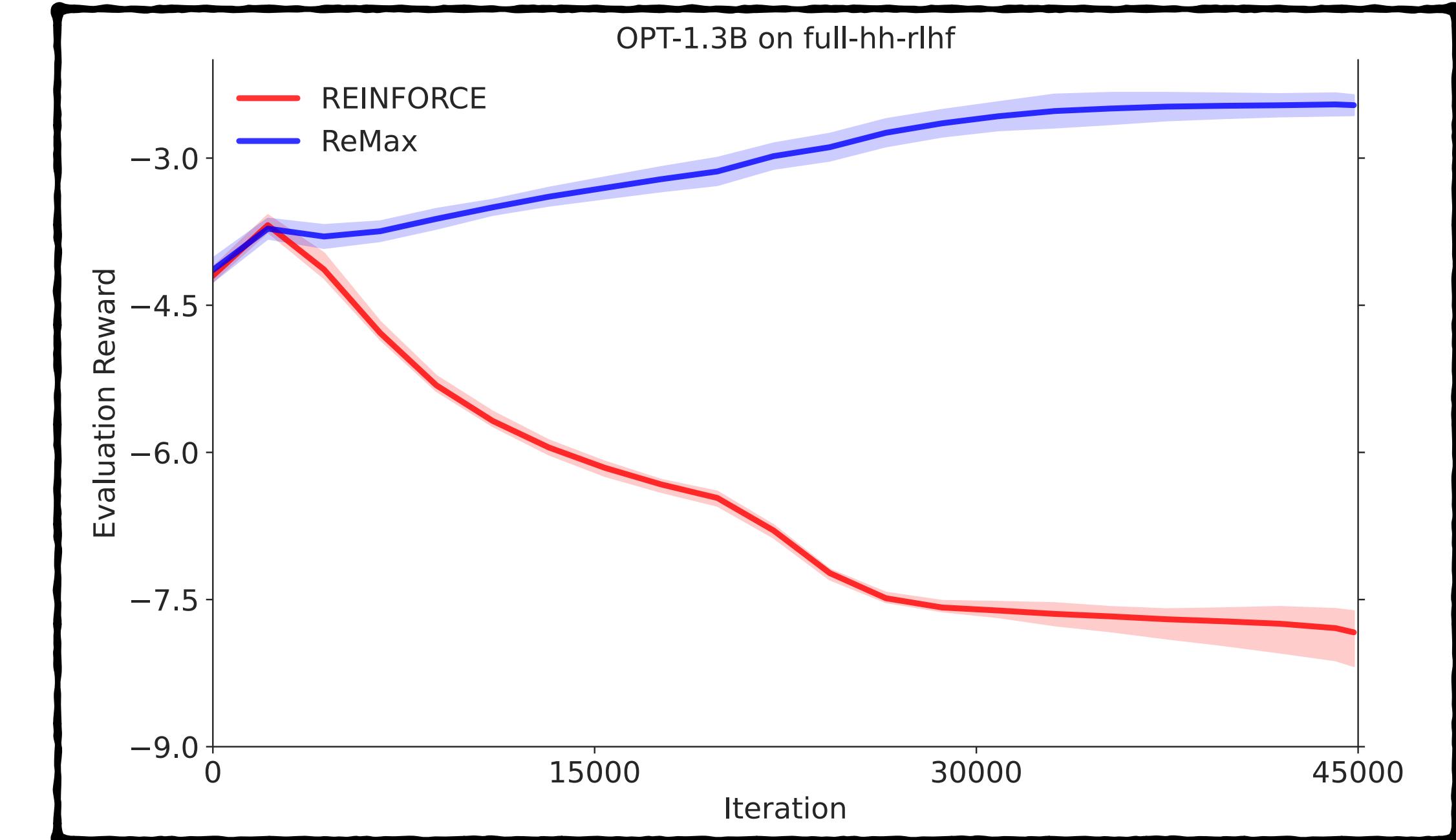
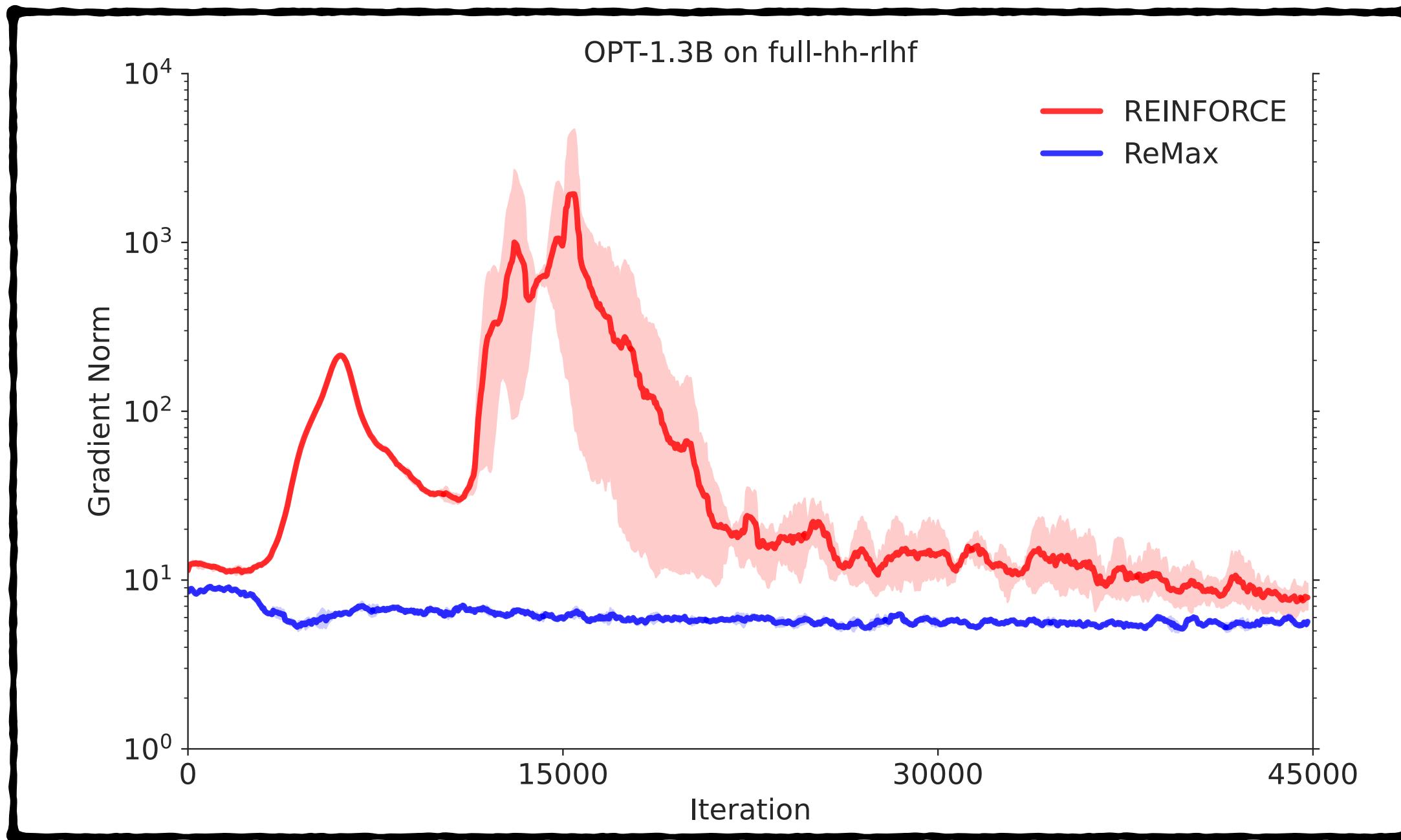
REINFORCE:  $\text{gradient} = \mathbb{E}_{y_{1:T} \sim \pi_{\theta}(\cdot|x)} [r(x, y_{1:T}) \cdot \nabla_{\theta} \log \pi_{\theta}(y_{1:T} | x)]$

No Value Model

Stochastic Gradient Estimation in Practice

# However, REINFORCE does not Work

---



REINFORCE's gradient has a high variance

REINFORCE's reward does not increase

# Why is Variance so High?

---



REINFORCE is often criticized for a high gradient variance. But why?

[Sutton, Richard S., and Andrew G. Barto. *Reinforcement learning: An introduction*. MIT press, 1998.]

$$\text{gradient} = \mathbb{E}_{a_{1:T} \sim \pi_\theta(\cdot|x)} [r(x, a_{1:T}) \cdot \nabla_\theta \log \pi_\theta(a_{1:T}|x)]$$



**Sample space is large**

Size: (vocabulary size)<sup>sequence length</sup>

Llama-3:  $(128k)^{8k}$

**Rewards vary across samples**

Reward range of open-ended question-answers: [-14, 7]

# Introduction to ReMax

Key Idea: Introduce a **baseline value** for accurate gradient estimation

$$\nabla_{\theta} \mathbb{E}_{x,y}[r(x, y_{1:T})] = \mathbb{E} \left[ \nabla_{\theta} \log \pi_{\theta}(y_{1:T} | x) \cdot [r(x, y_{1:T}) - b(x)] \right]$$

Advantage

$$b(x) = r(x, y'_{1:T}), \quad y'_t = \arg \max_{y_t} \pi_{\theta}(y_t | x_t, y_{1:t})$$

Greedy Decoding

Remark: 1) Subtracting a RV by a constant does not change the variance  
2) ReMax introduces a RV  $b \cdot \nabla_{\theta} \log \pi_{\theta}(y_{1:T} | x)$  → **control variate**

# Why Greedy Decoding?

---

$$\nabla_{\theta} \mathbb{E}_{x,y}[r(x, y_{1:T})] = \mathbb{E} \left[ \nabla_{\theta} \log \pi_{\theta}(y_{1:T} | x) \cdot [r(x, y_{1:T}) - b(x)] \right]$$

$$b(x) = r(x, y'_{1:T}), \quad y'_t = \arg \max_{y_t} \pi_{\theta}(y_t | x_t, y_{1:t})$$

Reason 1: greedy decoding corresponds to **mode** of the distribution → **effective estimation**

Reason 2: value of greedy decoding ensures **independence** between the baseline and original RVs → **stable estimation**

Reason 3: if there is a response better than the greedy one, improve it's likelihood

# ReMax Algorithm

---

---

**Algorithm 2** ReMax for Aligning Large Language Models

---

**Input:** reward\_model and language\_model

```
1: for prompts in datasets do
2:     seqs = language_model.generate(prompts, do_sample=True)
3:     seqs_max = language_model.generate(prompts, do_sample=False)
4:     rews = reward_model(prompts, seqs) - reward_model(prompts, seqs_max)
5:     log_probs = language_model(prompts, seqs)
6:     loss = -(log_probs.sum(dim=-1) * rews).mean()
7:     language_model.minimize(loss)
8: end for
```

Newly added

**Output:** language\_model

---

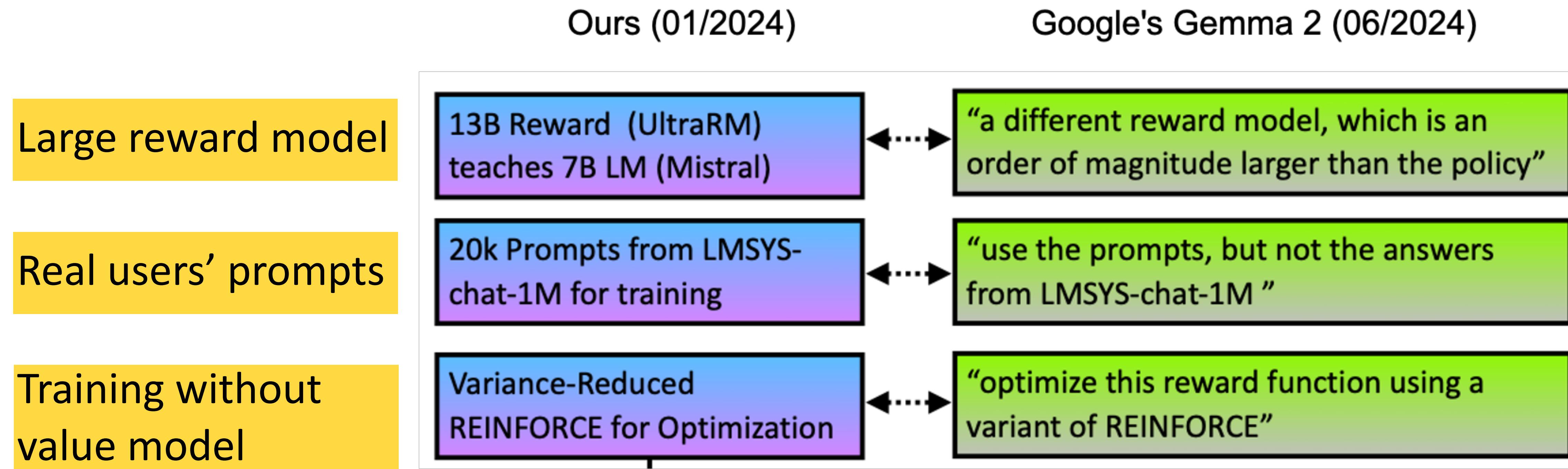
ReMax is Simple

8 Lines of code to implement (PPO: 50+)

1 Hyper-parameter (lr) to tune (PPO: 5+)

# Comparing with Google's Method

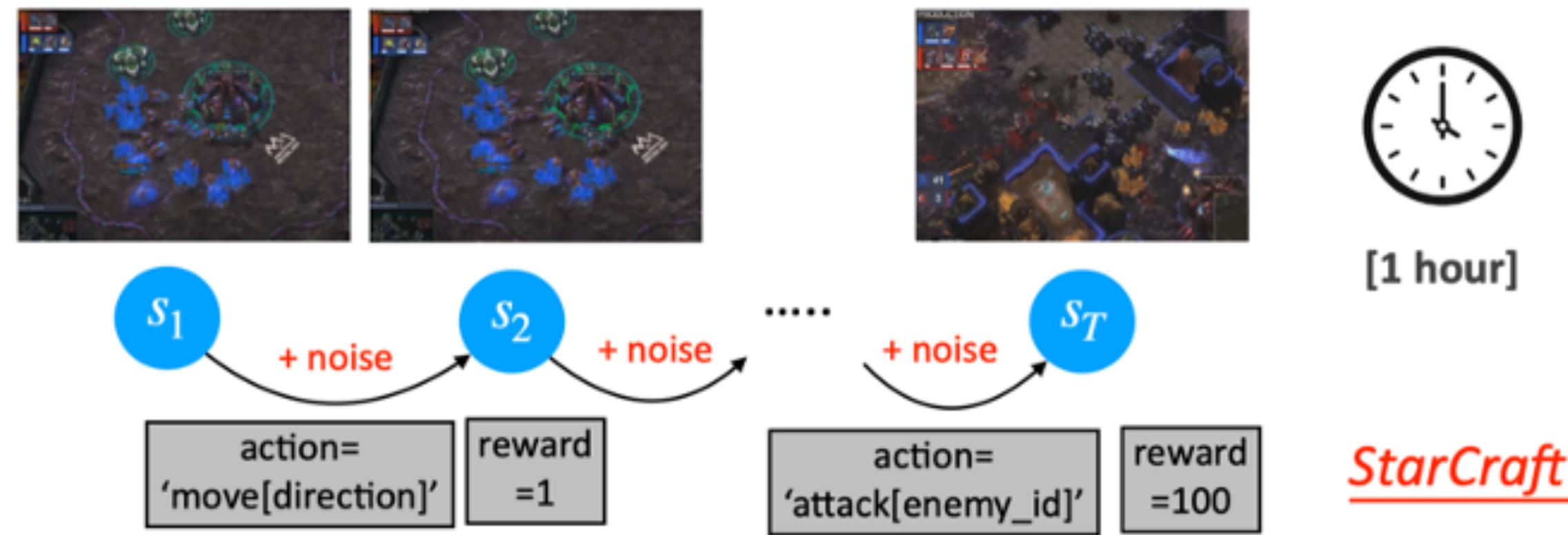
ReMax's training strategies are also used in Google's Gemma 2



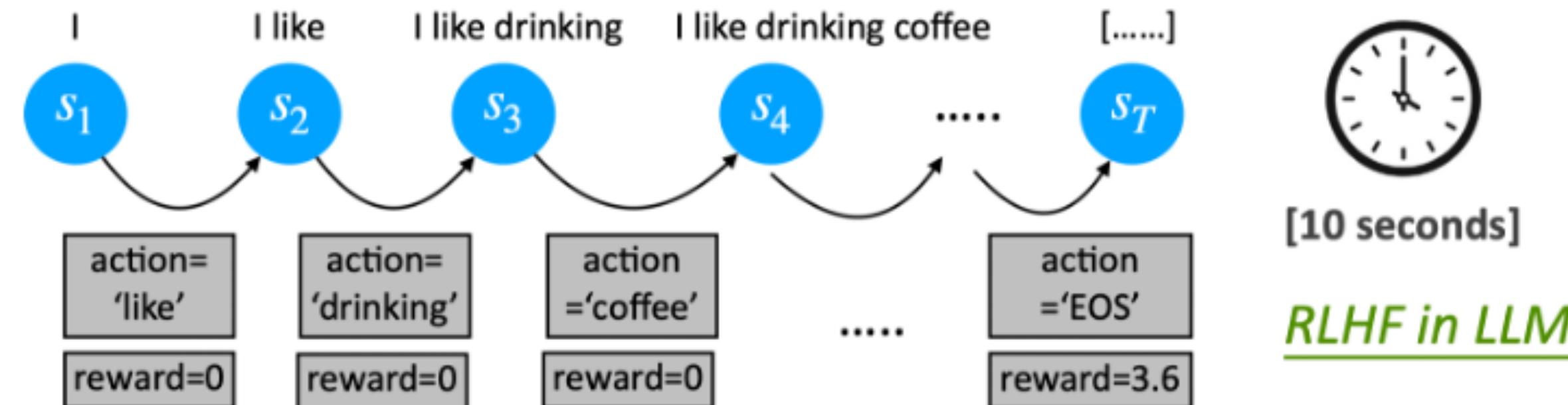
[Team, Gemma, et al. "Gemma 2: Improving open language models at a practical size." arXiv preprint arXiv:2408.00118 (2024).]

# Can We Safely Remove Value Model?

General RL Tasks



RL in LLMs



- Slow simulation
- Stochastic transition
- Dense reward

- Fast simulation
- Deterministic transition
- Trajectory-level reward

We conjecture that value-free methods are “optimal” for RL in LLMs

# PPO = REINFORCE with Baseline

$$\mathcal{L}_{\text{ppo}} = \mathbb{E}_{x \sim \rho} \mathbb{E}_{a_{1:T} \sim \pi_{\theta_{\text{old}}}} \left[ \sum_{t=1}^T \tilde{A}(s_t, a_t) \min \left\{ \psi(s_t, a_t), \text{clip} (\psi(s_t, a_t), 1 - \delta, 1 + \delta) \right\} \right].$$

General PPO

$$A(s_t, a_t) = \sum_{j=0}^{T-t} \lambda^j \text{advantage}_{t+j} = \sum_{j=0}^T \lambda^j [r(s_{t+j}, a_{t+j}) + \gamma V(s_{t+1+j}) - V(s_{t+j})],$$

Best Practice  $\gamma = 1, \lambda = 1$

[Ahmadian, Arash, et al. "Back to basics: Revisiting reinforce style optimization for learning from human feedback in llms." *arXiv preprint arXiv:2402.14740* (2024).]

PPO in LLM

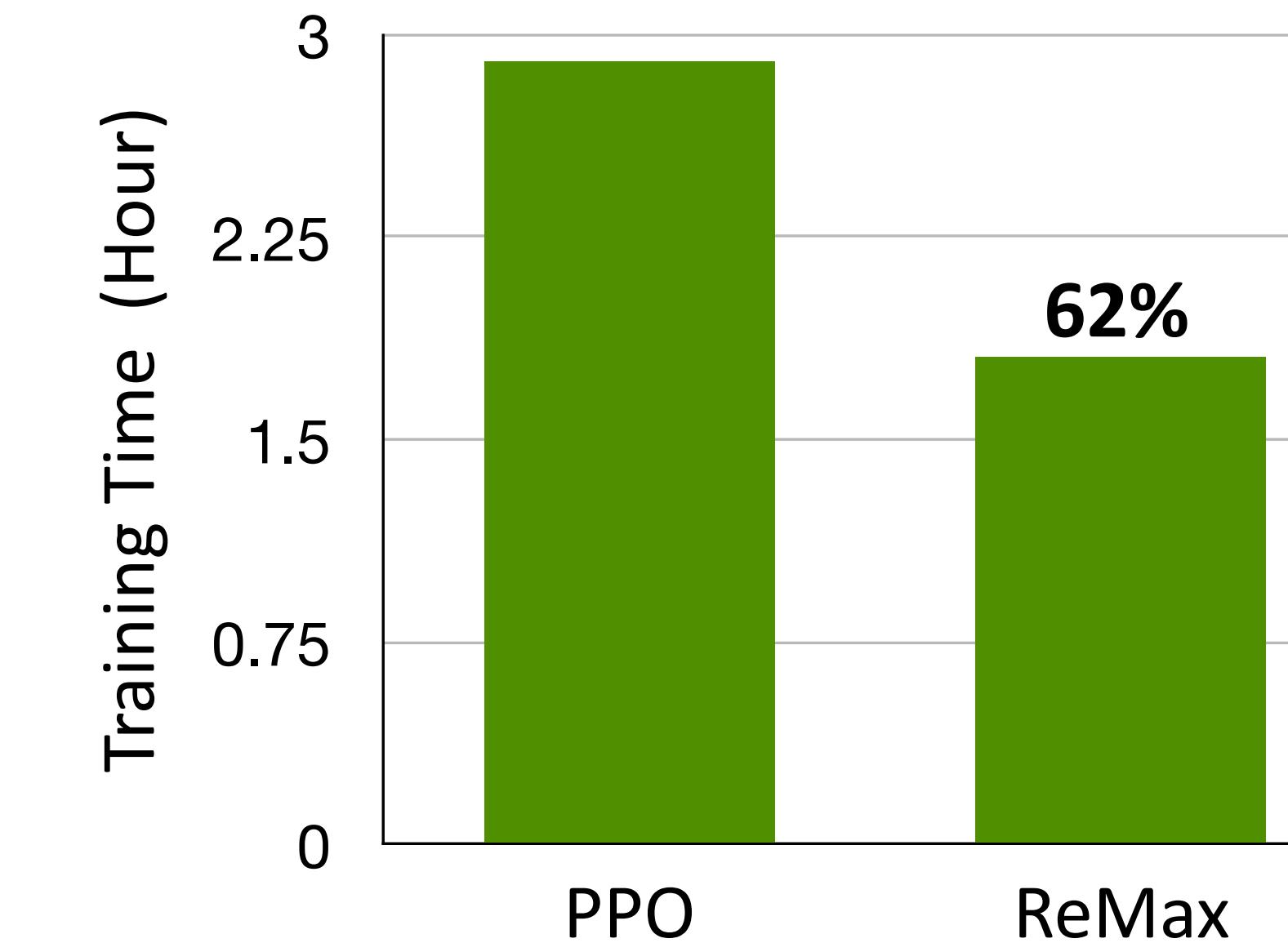
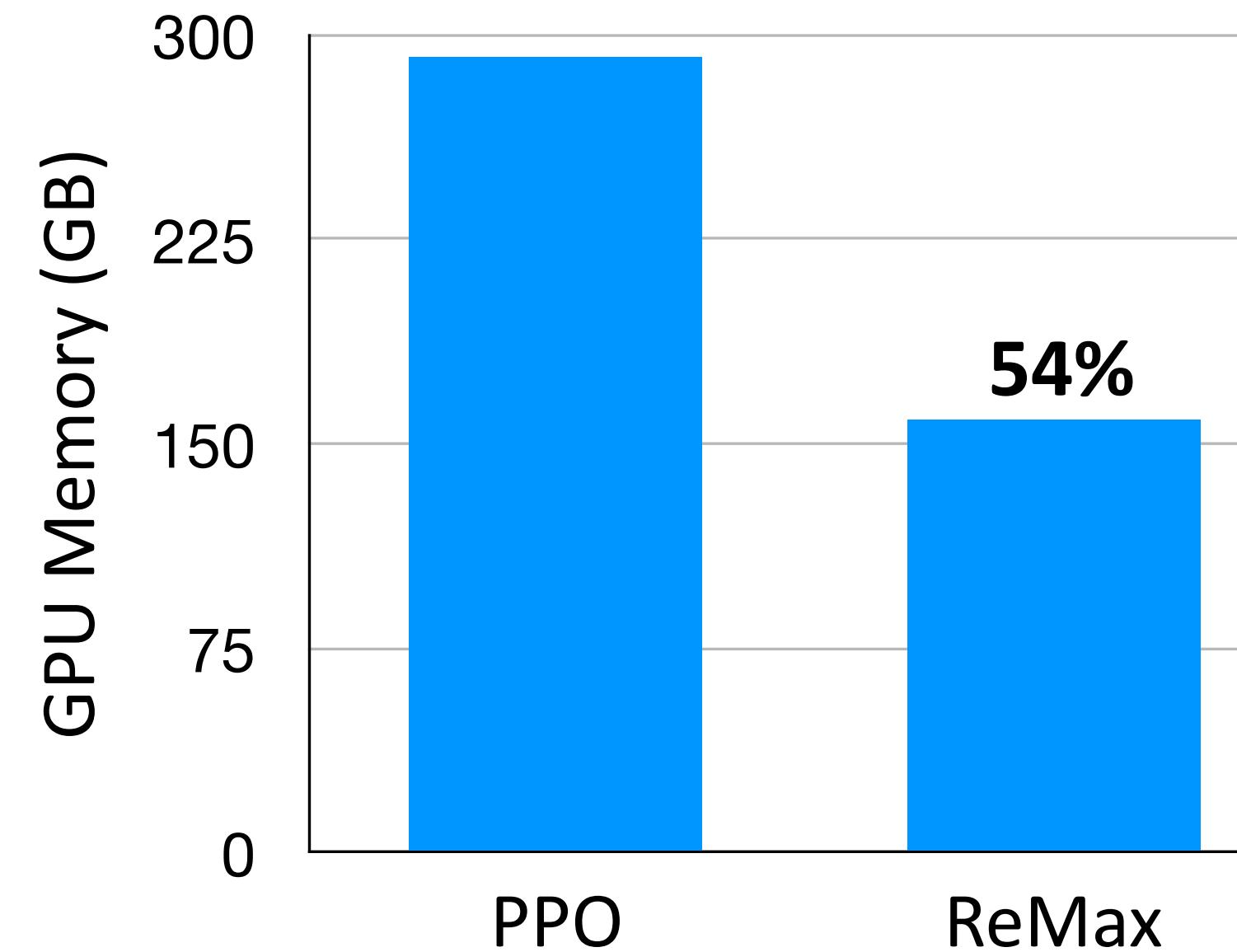
$$\mathcal{L}_{\text{ppo}}(\theta) = \mathbb{E}_{x \sim \rho} \mathbb{E}_{a_{1:T} \sim \pi_{\theta}} \left[ \sum_{t=1}^T r(x, a_{1:T}) - V(x, a_{1:t}) \right]$$

Outcome reward in  
REINFORCE's estimator

Model-learned  
Baseline

# ReMax is Computationally Efficient

---

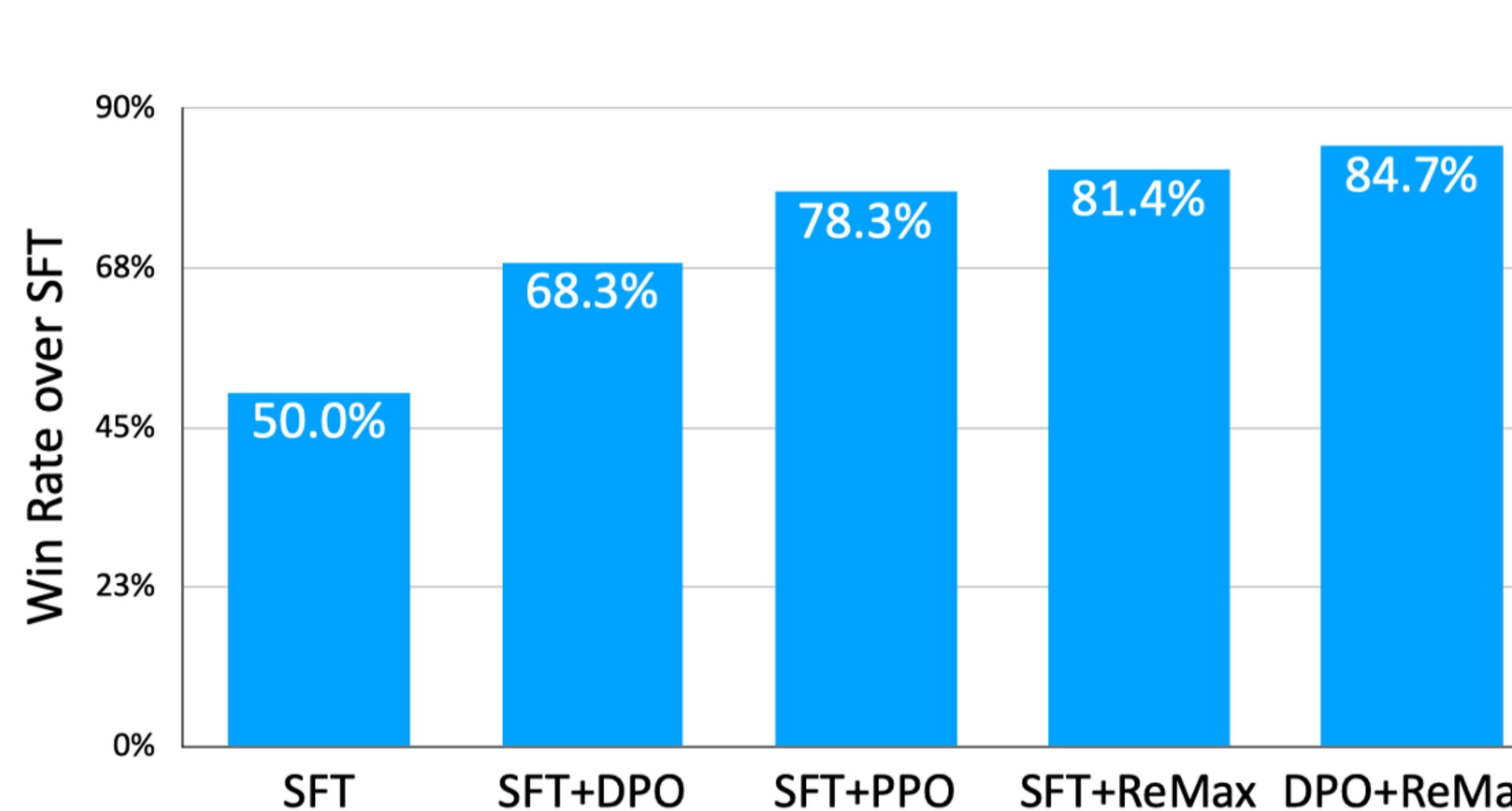


[Li, Ziniu, et al. "Remax: A simple, effective, and efficient reinforcement learning method for aligning large language models." *arXiv preprint arXiv:2310.10505* (2023).]

ReMax saves about 2x GPU memory and training time on Llama-2-7B

# Performance in RLHF Task

---



[Li, Ziniu, et al. "Remax: A simple, effective, and efficient reinforcement learning method for aligning large language models." *arXiv preprint arXiv:2310.10505* (2023).]

ReMax is superior to DPO and PPO

# Performance in RLHF Task

*Table 4.* Performance against strong open-source and private models: Llama-2-Chat models (7B and 70B) apply RLHF (via PPO) using secret datasets; Zephyra-7B-beta (Tunstall et al., 2023) is based on the pretrained Mistral-7B-v0.2 with DPO. GPT-3.5 and GPT-4 utilize RLHF (via PPO) with secret datasets.

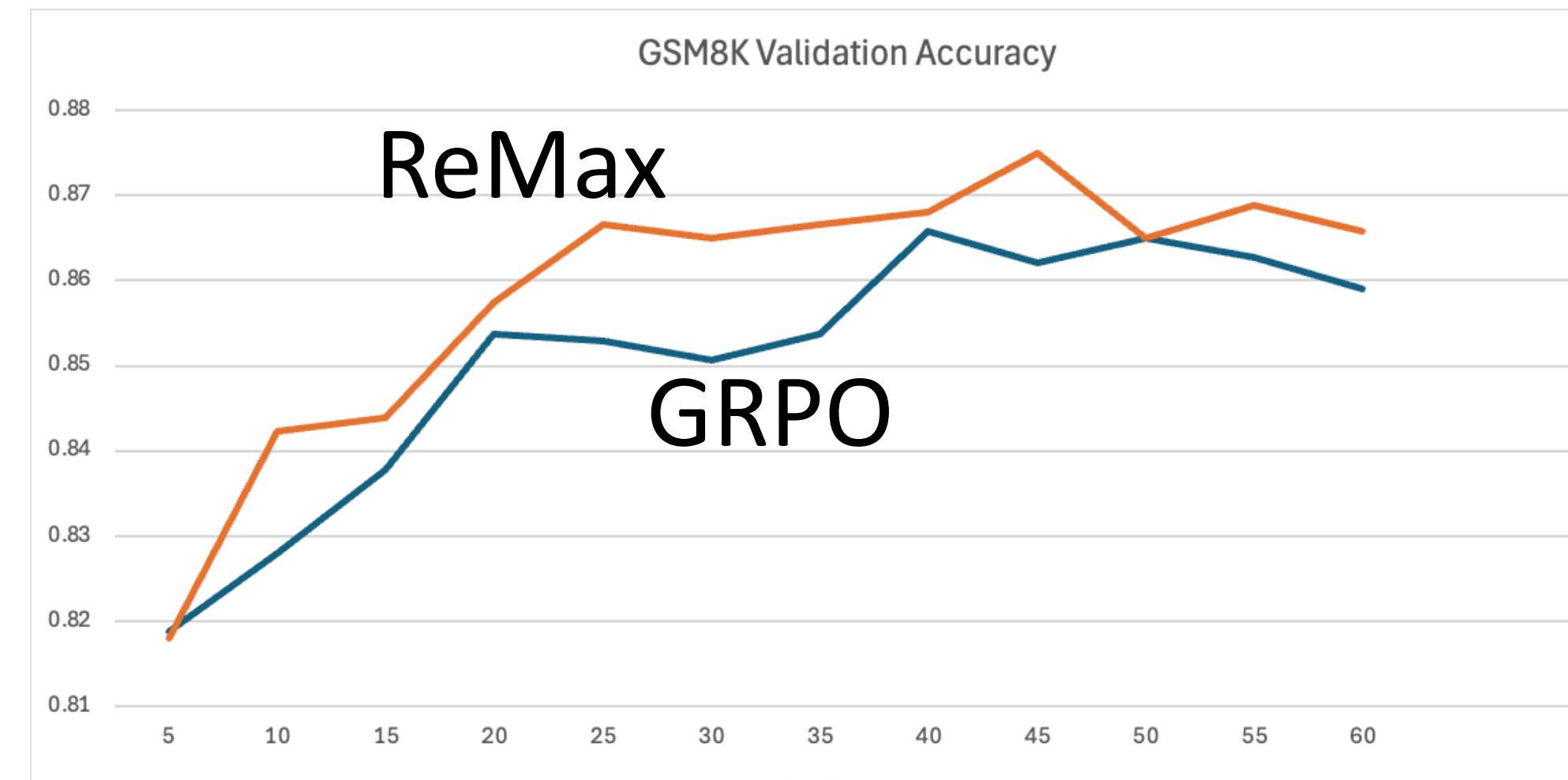
	AlpacaEval	MT-Bench
Llama-2-7B-Chat	71.37%	6.269
Zephyr-7B-beta	90.60%	7.356
Mistral-7B-Instruct-v0.2	92.78%	7.516
Mistral (via ReMax)	<b>94.78%</b>	<b>7.739</b>
Llama-2-70B-Chat	92.66%	6.856
GPT-3.5-turbo	93.42%	7.944
GPT-4-turbo	95.28%	8.991

[Li, Ziniu, et al. "Remax: A simple, effective, and efficient reinforcement learning method for aligning large language models." *arXiv preprint arXiv:2310.10505* (2023).]

ReMax achieves SOTA among 7B models (measured at Jan., 2024)

# Performance in Reasoning Task

Our  
Evaluation



Others'  
Evaluation

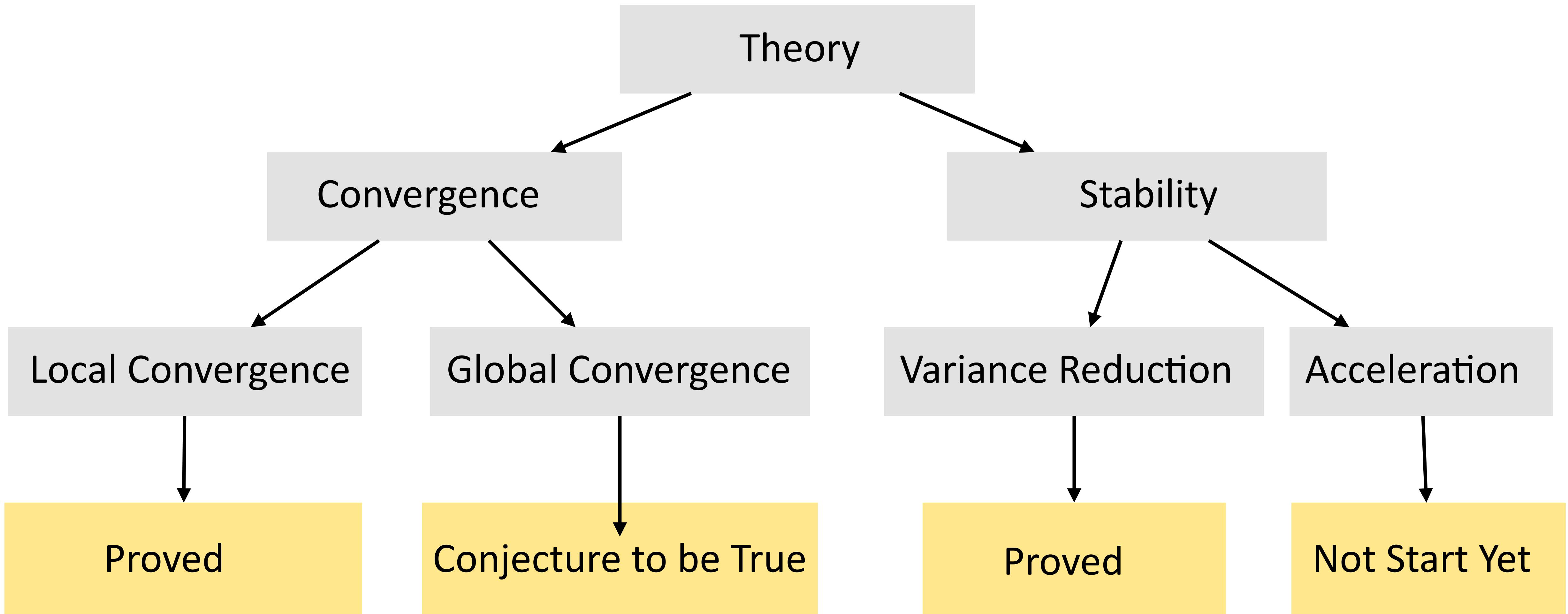
	Mineva Math	Olmpiad Bench	HumanEva I	LeetCod e	LiveCode Bench	Avg.
ReMax	24.6	17.3	61.0	21.1	18.6	28.5
GRPO	22.4	20.3	57.3	13.3	18.7	26.4

[<https://curvy-check-498.notion.site/Process-Reinforcement-through-Implicit-Rewards-15f4fc9c42180f1b498cc9b2eaf896f>]

ReMax is superior to DeepSeek's GRPO

# Overview of ReMax's Theory

---



# Variance Reduction

---

Setting: 2-action armed bandit (assuming  $r(a_1) > r(a_2)$ )

Our result:  $\text{Variance(ReMax)} < \text{Variance(REINFORCE)}$  if

$$\pi(a_1) \leq 0.5 + 0.5 \frac{r(a_1)}{r(a_1) - r(a_2)}$$

Implication:

- 1) variance reduction when the optimal action is **not dominated**
- 2) slow convergence when the policy is near-optimal  
→ good if reward is imperfect (**mitigating overfitting**)

---

# **ReMax: A Simple, Effective, and Efficient Reinforcement Learning Method for Aligning Large Language Models**

---

**Ziniu Li<sup>1 2</sup> Tian Xu<sup>3 4</sup> Yushun Zhang<sup>1 2</sup> Zhihang Lin<sup>1</sup> Yang Yu<sup>3 4 5 †</sup> Ruoyu Sun<sup>1 6 2 †</sup> Zhi-Quan Luo<sup>1 2</sup>**

**ICML 2024**



**Paper**



**Code**

# Conclusive Remark

---

## Part I: LLM Training Pipeline

- ▶ Pre-training: knowledge acquisition
- ▶ Post-training: instruction following and ability enhancement

## Part II: Preserving Diversity in SFT

- ▶ CE's formulation lack of consideration of diversity
- ▶ GEM: a game-theoretic approach with entropy regularization

## Part III: Efficient RL Training

- ▶ PPO's formulation are overshot for LLM
- ▶ ReMax: variance-reduced REINFORCE

**Thank You!**