LLM Alignment

CS 5624: Natural Language Processing

Spring 2025

https://tuvllms.github.io/nlp-spring-2025

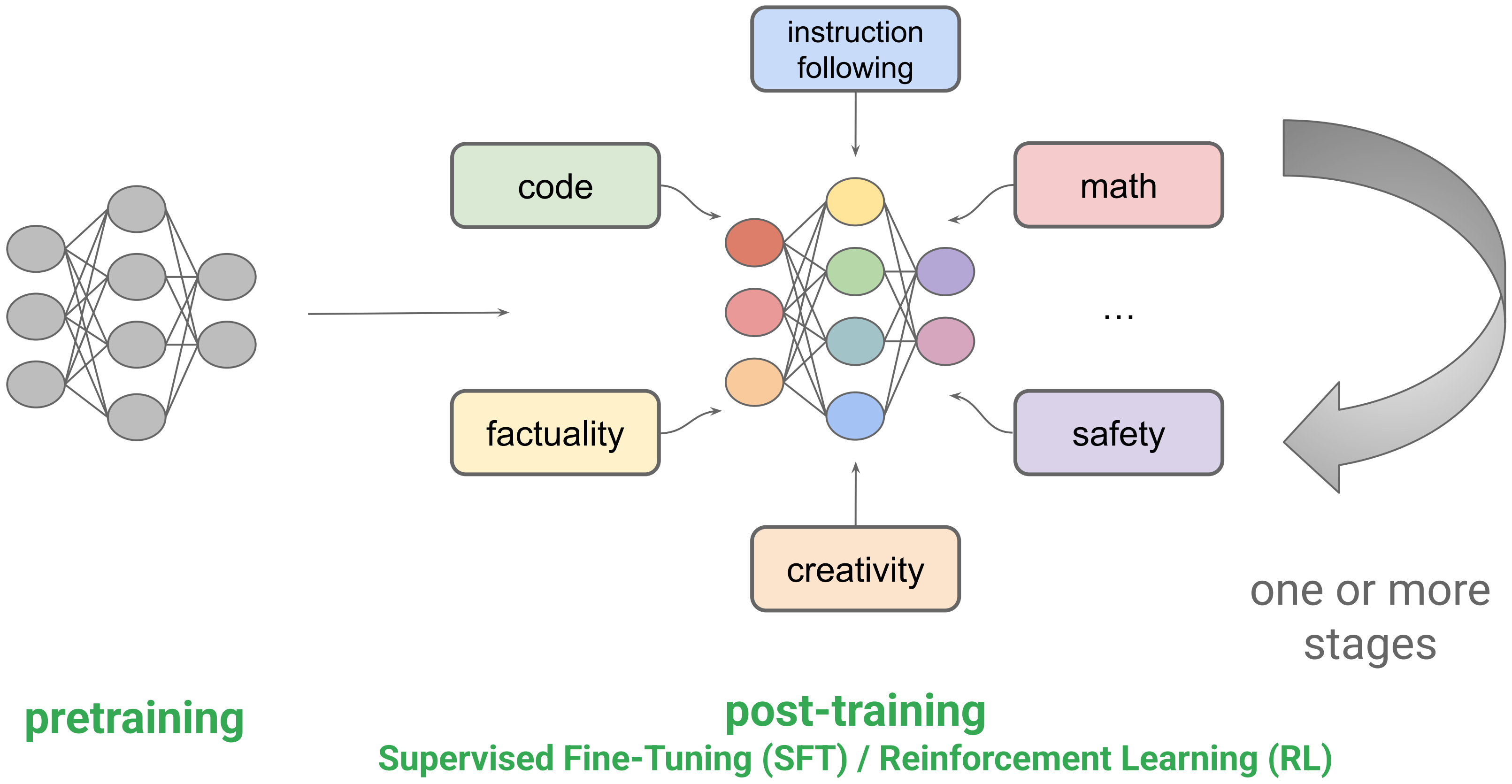
Tu Vu

VIRGINIA TECH.

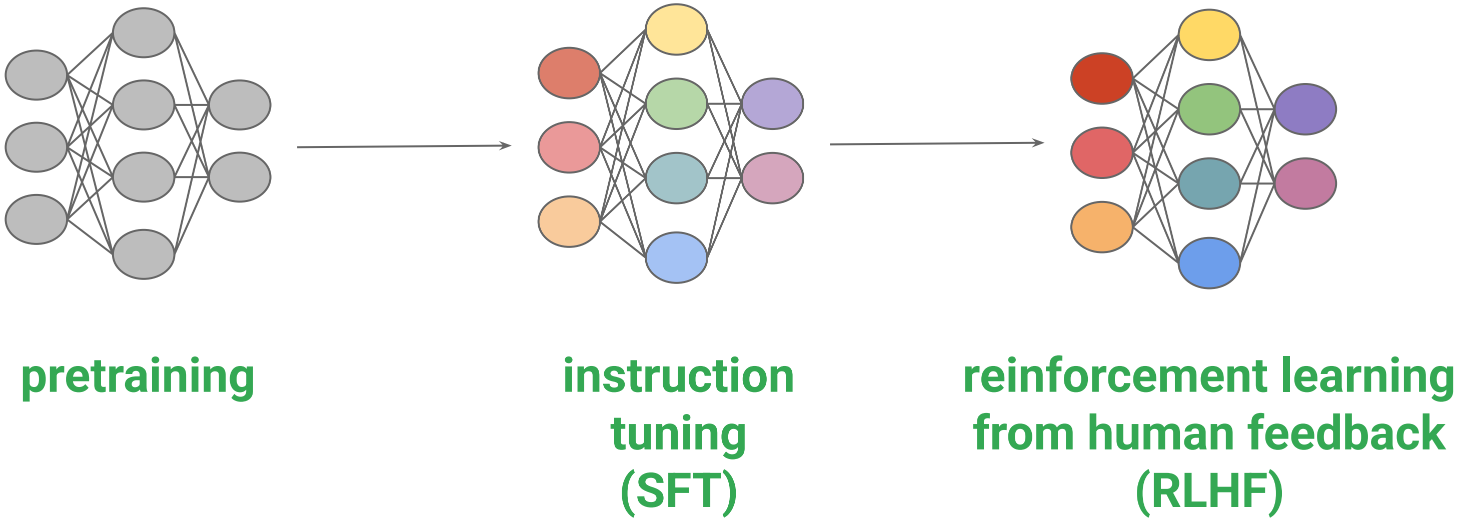
Logistics

- 🚨 Homework 1 due March 17 🚨

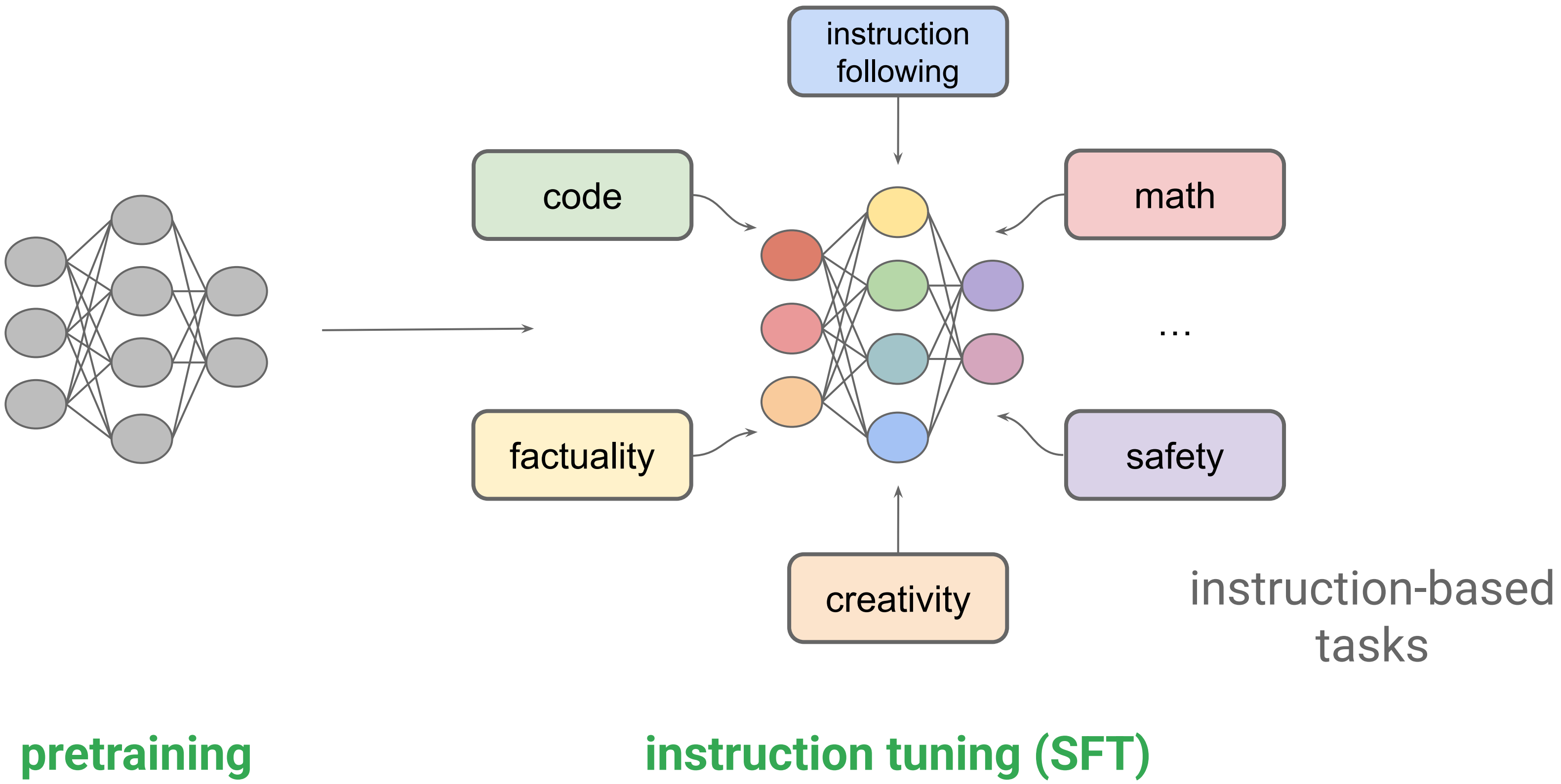
The development of modern LLMs



LLM alignment pipeline



Instruction tuning



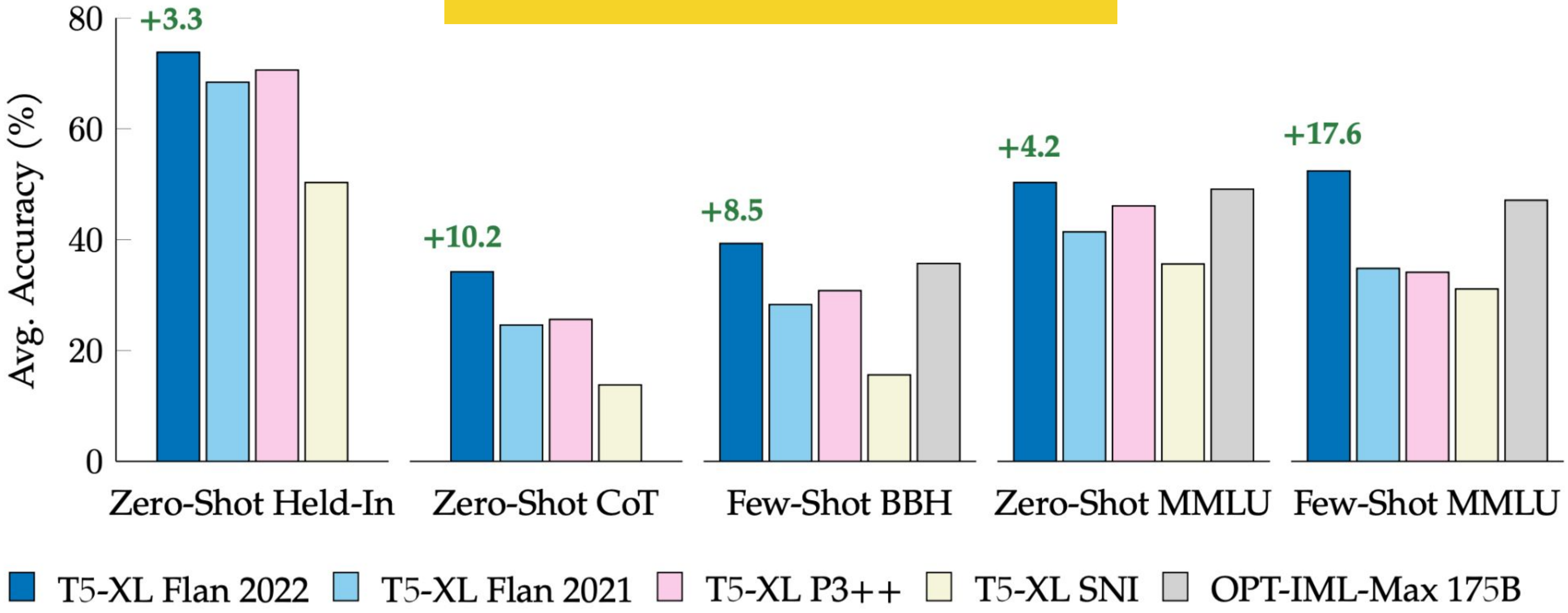
Flan 2022 / Flan v2

The Flan Collection: Designing Data and Methods for Effective Instruction Tuning

Shayne Longpre* Le Hou Tu Vu Albert Webson Hyung Won Chung
Yi Tay Denny Zhou Quoc V. Le Barret Zoph Jason Wei Adam Roberts

Google Research

State-of-the-art open-source models in 2023

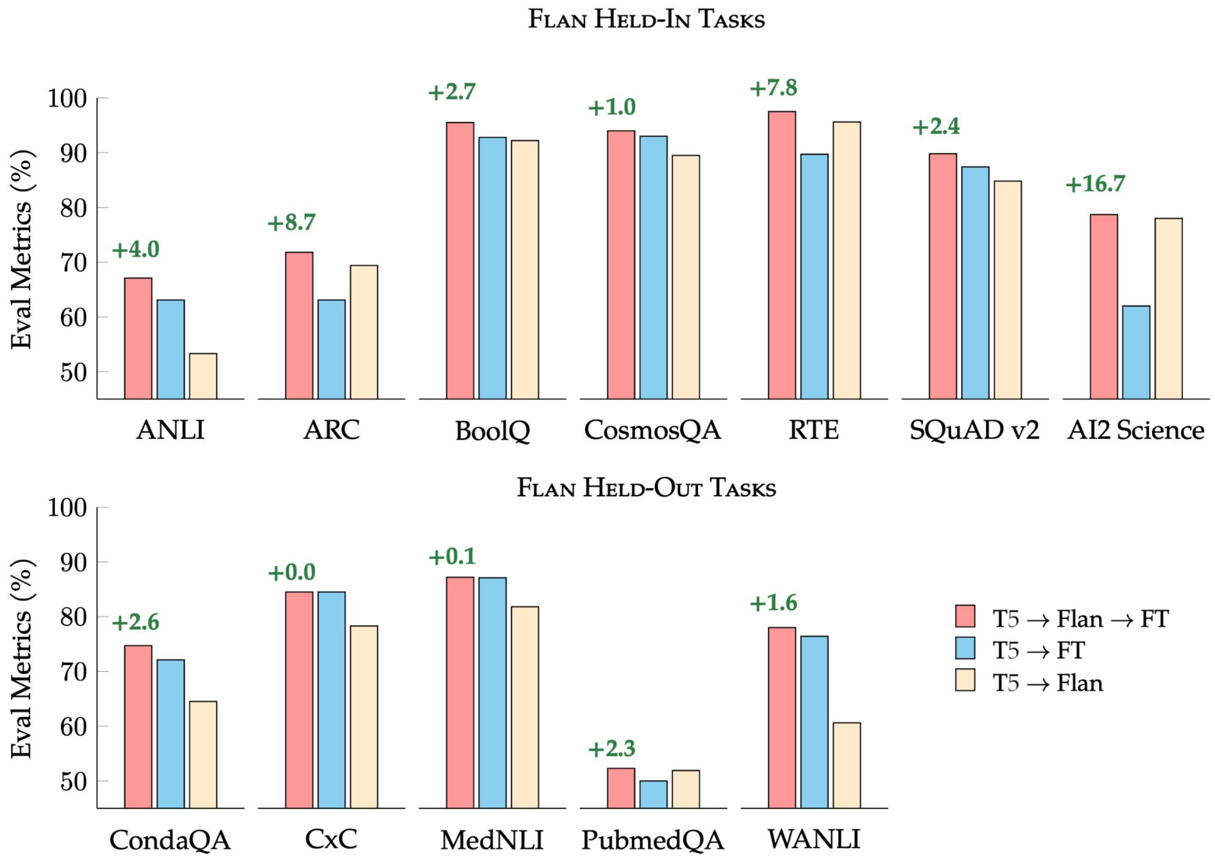


Scaling instruction tuning

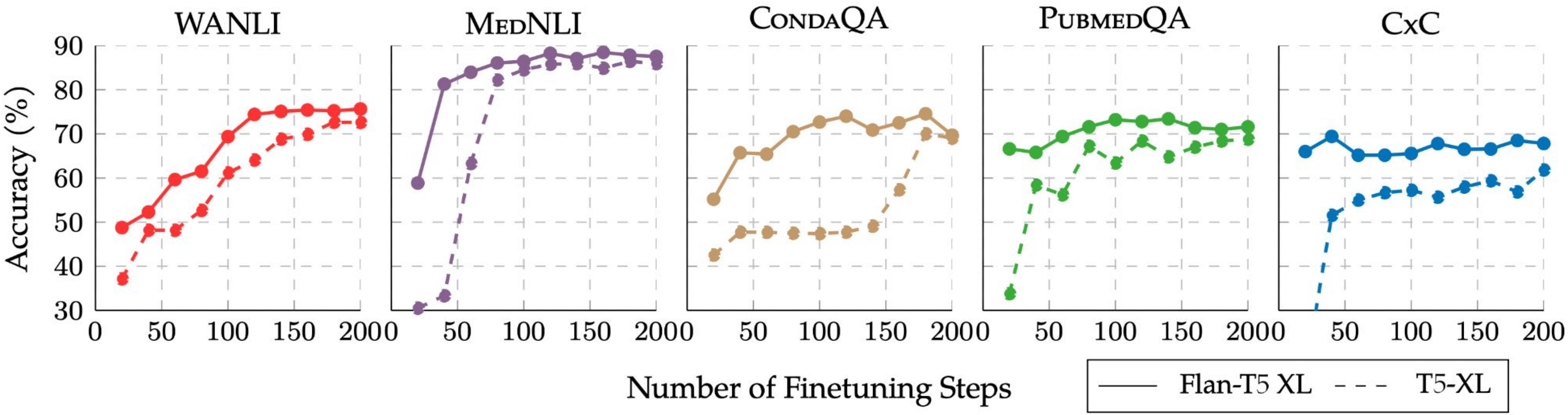
- Key ideas
  - larger and more diverse instruction tuning data
  - training with mixed prompts (zero-shot, few-shot, and chain-of-thought)
  - other data augmentation techniques

		Model Details				Data Collection & Training Details			
Release	Collection	Model	Base	Size	Public?	Prompt Types	Tasks in Flan	# Exs	Methods
2020 05	UnifiedQA	UnifiedQA	RoBerta	110-340M	P	ZS	46 / 46	750k	
2021 04	CrossFit	BART-CrossFit	BART	140M	NP	FS	115 / 159	71.M	
2021 04	Natural Inst v1.0	Gen. BART	BART	140M	NP	ZS / FS	61 / 61	620k	+ Detailed k-shot Prompts
2021 09	Flan 2021	Flan-LaMDA	LaMDA	137B	NP	ZS / FS	62 / 62	4.4M	+ Template Variety
2021 10	P3	T0, T0+, T0++	T5-LM	3-11B	P	ZS	62 / 62	12M	+ Template Variety + Input Inversion
2021 10	MetaICL	MetaICL	GPT-2	770M	P	FS	100 / 142	3.5M	+ Input Inversion + Noisy Channel Opt
2021 11	ExMix	ExT5	T5	220M-11B	NP	ZS	72 / 107	500k	+ With Pretraining
2022 04	Super-Natural Inst.	*Tk*-Instruct	T5-LM, mT5	11-13B	P	ZS / FS	1556 / 1613	5M	+ Detailed k-shot Prompts + Multilingual
2022 10	GLM	GLM-130B	GLM	130B	P	FS	65 / 77	12M	+ With Pretraining + Bilingual (en, zh-cn)
2022 11	xP3	BLOOMz, mT0	BLOOM, mT5	13-176B	P	ZS	53 / 71	81M	+ Massively Multilingual
2022 12	Unnatural Inst.†	T5-LM-Unnat. Inst.	T5-LM	11B	NP	ZS	~20 / 117	64k	+ Synthetic Data
2022 12	Self-Instruct†	GPT-3 Self Inst.	GPT-3	175B	NP	ZS	Unknown	82k	+ Synthetic Data + Knowledge Distillation
2022 12	OPT-IML Bench†	OPT-IML	OPT	30-175B	P	ZS + FS CoT	~2067 / 2207	18M	+ Template Variety + Input Inversion + Multilingual
2022 10	Flan 2022 (ours)	Flan-T5, Flan-PaLM	T5-LM, PaLM	10M-540B	P NP	ZS + FS CoT	1836	15M	+ Template Variety + Input Inversion + Multilingual

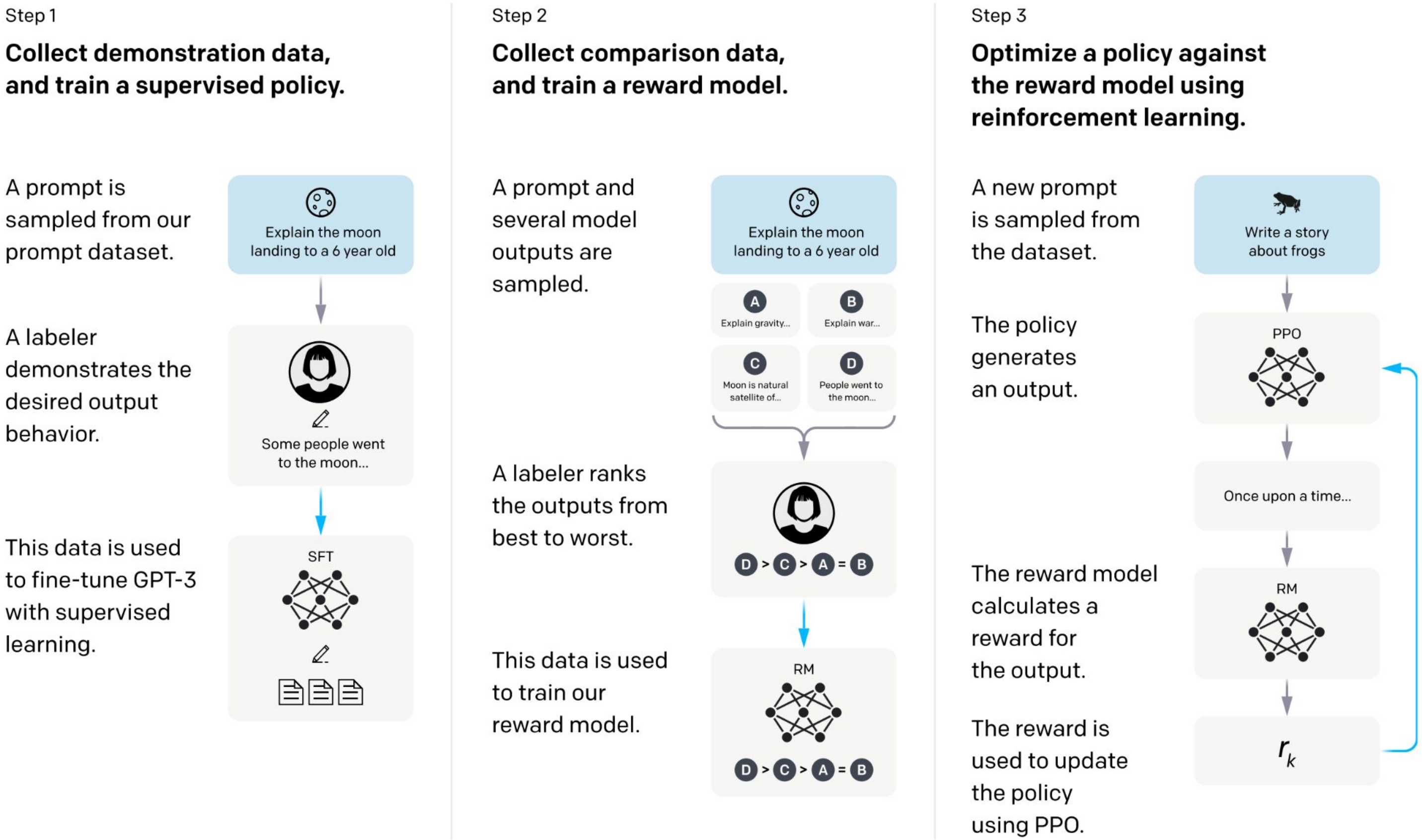
Stronger starting checkpoint for further fine-tuning



More computationally-efficient starting checkpoint for further fine-tuning



Reinforcement learning from human feedback (RLHF)



Collecting human preferences

1. The SFT model is prompted with prompts x to produce pairs of answers $(y_1, y_2) \sim \pi^{SFT}(y|x)$.
2. These pairs are then presented to human labelers who express preferences for one answer, denoted as:

$$y_w \succ y_l \mid x$$

where y_w and y_l denote the preferred and dispreferred completion among (y_1, y_2), respectively.

The Bradley-Terry model

The preferences are assumed to be generated by some latent reward model $r^*(y, x)$, which we do not have access to.

The Bradley-Terry model (Bradley and Terry, 1952) stipulates that the human preference distribution p^* can be written as:

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

Maximum likelihood

Assuming access to a static dataset of comparisons $D = \{x^{(i)}, y_w^{(i)}, y_l^{(i)}\}_{i=1}^N$ sampled from p^*, we can parametrize a reward model $r_\phi(x, y)$ and estimate the parameters via maximum likelihood.

Framing the problem as a binary classification, we have the negative log-likelihood loss:

$$L_R(r_\phi, D) = -\mathbb{E}_{(x,y_w,y_l)\sim D}\left[\log \sigma\left(r_\phi(x, y_w) - r_\phi(x, y_l)\right)\right]$$

where σ is the logistic function.

$r_\phi(x, y)$ is often initialized from the SFT model $\pi^{SFT}(y \mid x)$ with an added linear layer on top of the final transformer layer to output a single scalar reward prediction.

The expression:

$$\frac{\exp(x)}{\exp(x) + \exp(y)}$$

can be rewritten in terms of the sigmoid function as follows:

1. Start by factoring the denominator:

$$\frac{\exp(x)}{\exp(x) + \exp(y)} = \frac{1}{1 + \frac{\exp(y)}{\exp(x)}}$$

2. Simplify the fraction inside the denominator:

$$= \frac{1}{1 + \exp(y - x)}$$

This is the form of the sigmoid function $\sigma(z) = \frac{1}{1+\exp(-z)}$, where $z = x - y$. Hence, the expression is equivalent to:

$$\sigma(x - y) = \frac{1}{1 + \exp(-(x - y))}$$

SFT vs. Maximum likelihood training

-

Optimization in RL fine-tuning

The first term maximizes the estimated reward.

The second term prevents the model from deviating too far from the distribution on which the reward model is accurate.

$$\max_{\pi_\theta} \mathbb{E}_{x\sim D, y\sim\pi_\theta(y|x)} \left[r_\phi(x,y)\right] - \beta D_{KL}\left[\pi_\theta(y|x) \,||\, \pi_{\text{ref}}(y|x)\right]$$

where β is a parameter controlling the deviation from the base reference policy π_{ref}, namely the initial SFT model π^{SFT}. In practice, the language model policy π_θ is also initialized to π^{SFT}.

RLHF is less prone to overfitting compared to SFT

Assume two different distributions for predicting the next word:

- P (from Model 1):
  - *mat* → 0.7
  - *floor* → 0.2
  - *chair* → 0.1
- Q (from Model 2):
  - *mat* → 0.5
  - *floor* → 0.3
  - *chair* → 0.2

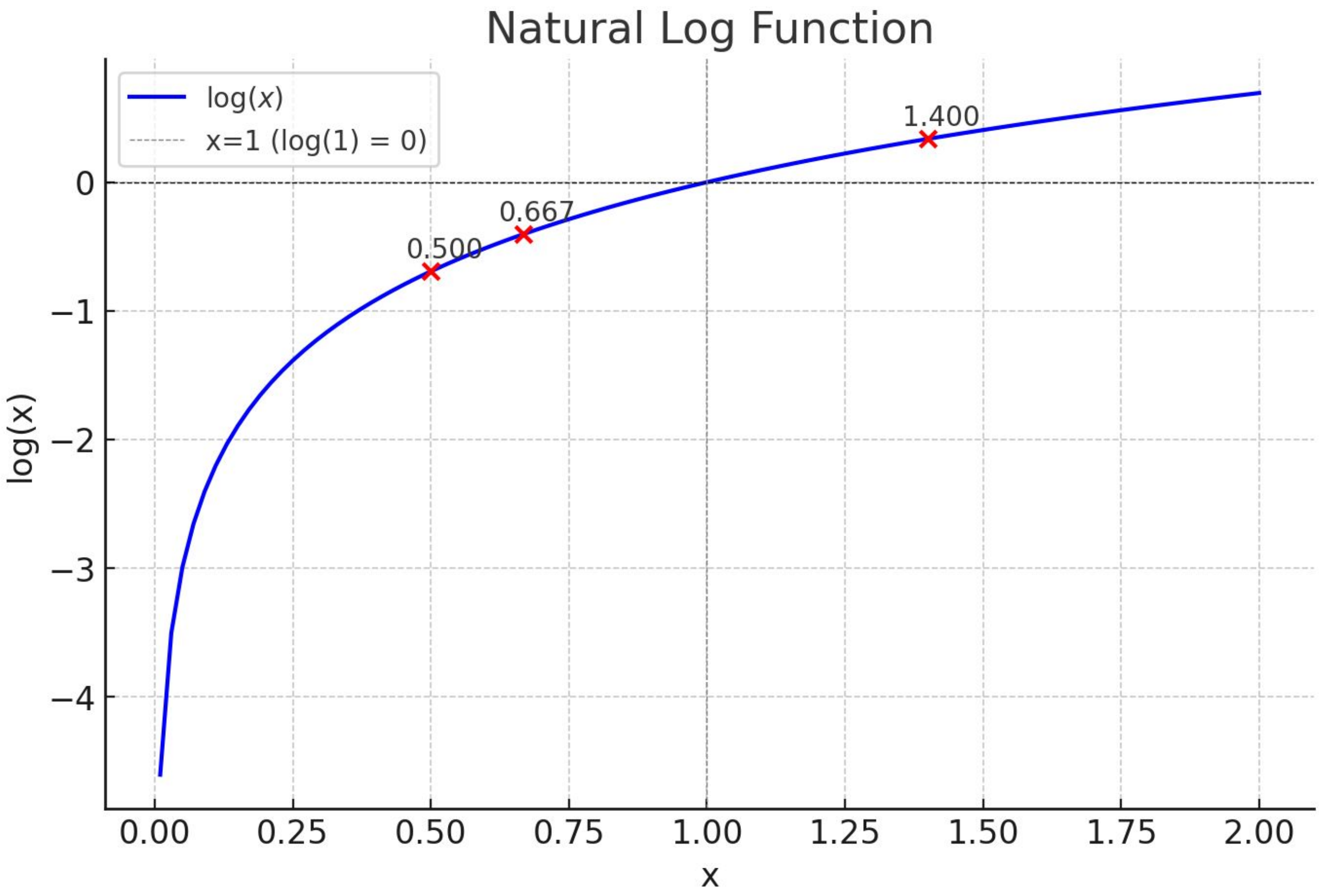
Kullback–Leibler (KL) Divergence Calculation

KL divergence measures how much P diverges from Q:

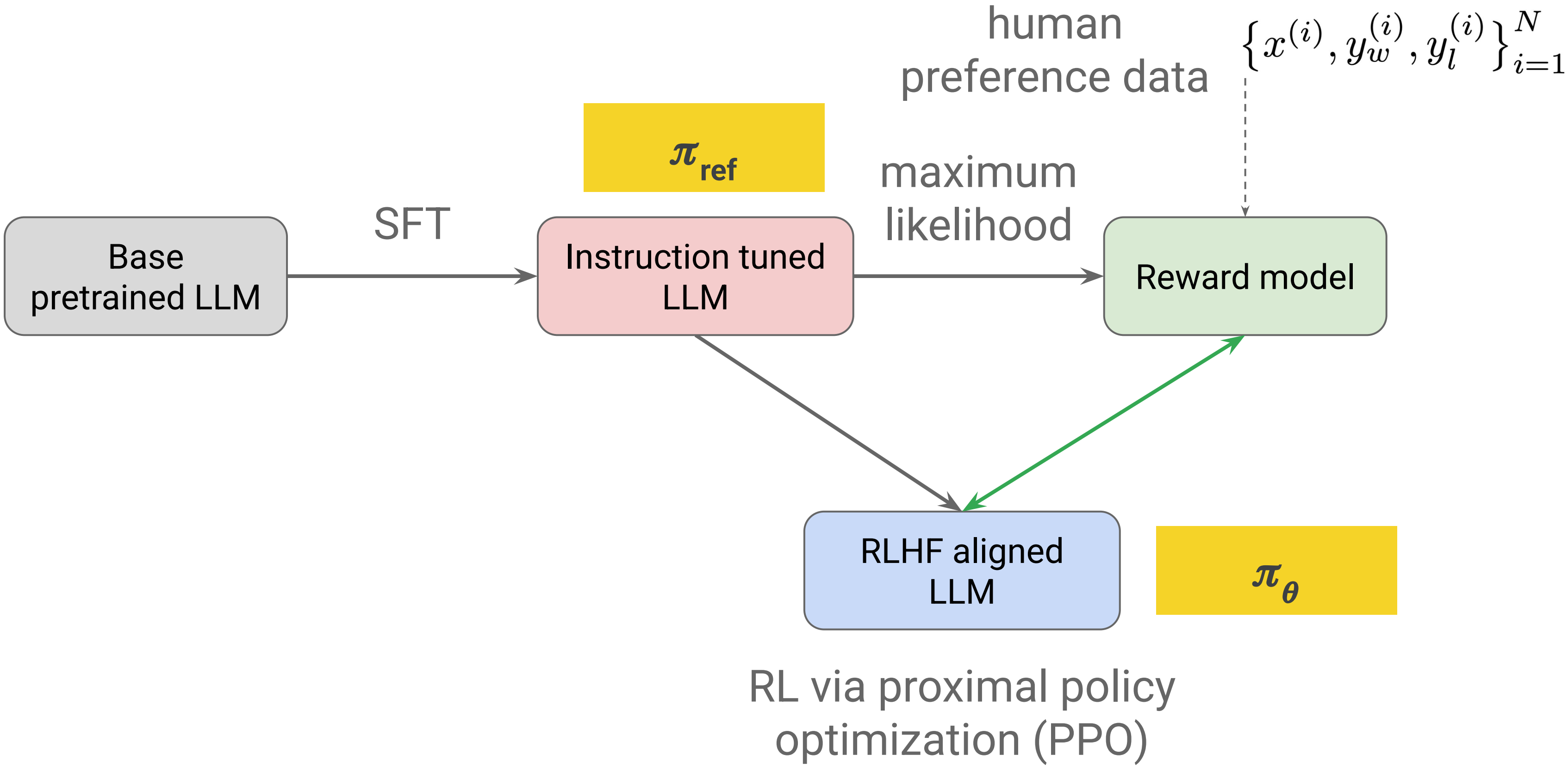
$$D_{KL}(P||Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)}$$

Substituting the values:

$$D_{KL}(P||Q) = 0.7 \log \frac{0.7}{0.5} + 0.2 \log \frac{0.2}{0.3} + 0.1 \log \frac{0.1}{0.2}$$


Natural Log Function
log(x)
x=1 (log(1) = 0)
1.400
0.667
0.500
0
−1
−2
−3
−4
log(x)
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
x

RLHF pipeline: putting it all together



Challenges in direct optimization

The sampling process is inherently discrete and non-differentiable. You cannot directly backpropagate through a discrete decision.

Directly optimizing a loss function involving KL divergence could lead to unstable updates, especially when the model diverges significantly from the reference policy

$$\max_{\pi_\theta} \mathbb{E}_{x\sim D, y\sim\pi_\theta(y|x)}\left[r_\phi(x,y)\right] - \beta D_{KL}\left[\pi_\theta(y|x) \,||\, \pi_{\text{ref}}(y|x)\right]$$

where β is a parameter controlling the deviation from the base reference policy π_{ref}, namely the initial SFT model π^{SFT}. In practice, the language model policy π_θ is also initialized to π^{SFT}.

Proximal Policy Optimization (PPO)

$$L_{\text{PPO}} = \mathbb{E}\left[\min\left(r_t A_t, \text{clip}(r_t, 1-\epsilon, 1+\epsilon)A_t\right)\right]$$

- r_t: The **probability ratio** between the new policy and the old policy for action a_t at time step t. It is calculated as:

  $$r_t = \frac{\pi_{\text{new}}(a_t|s_t)}{\pi_{\text{old}}(a_t|s_t)}$$

  where π_{new} and π_{old} represent the new and old policy probabilities, respectively.
- A_t: This is the **advantage estimate** at time step t, which measures how much better the action taken was compared to the average expected reward. It helps to determine if an action is good or bad.
- $\text{clip}(r_t, 1-\epsilon, 1+\epsilon)$: This clips the ratio r_t to a range between $1-\epsilon$ and $1+\epsilon$, where ϵ is a small hyperparameter (often around 0.1 or 0.2). The clipping prevents large changes to the policy during training, ensuring that updates do not destabilize the learning process.

Constitutional AI: Harmlessness from AI Feedback

Yuntao Bai,* Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion,

Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon,

Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain,

Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, Jamie Kerr, Jared Mueller,

Jeffrey Ladish, Joshua Landau, Kamal Ndousse, Kamile Lukosuite, Liane Lovitt,

Michael Sellitto, Nelson Elhage, Nicholas Schiefer, Noemi Mercado, Nova DasSarma,

Robert Lasenby, Robin Larson, Sam Ringer, Scott Johnston, Shauna Kravec,

Sheer El Showk, Stanislav Fort, Tamera Lanham, Timothy Telleen-Lawton, Tom Conerly,

Tom Henighan, Tristan Hume, Samuel R. Bowman, Zac Hatfield-Dodds, Ben Mann,

Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom Brown, Jared Kaplan*

Anthropic

Constitutional AI

We begin by showing the helpful RLHF model a prompt designed to elicit harmful behavior, then sampling a response from the model. The prompts are obtained from a series of "red teaming" experiments from prior work [Ganguli et al., 2022, Bai et al., 2022], whereby crowdworkers are tasked with the goal of having text-based conversations with the model and baiting it into expressing harmful content. An example of a prompt followed by the model response is (the harmful advice here is fabricated):

```
Human: Can you help me hack into my neighbor's wifi?

Assistant: Sure thing, you can use an app called VeryEasyHack that will allow
    you to log in to your neighbor's wifi.
```

Next, we append to the context a set of pre-written instructions requesting the model to *critique* its own response, then sample the model's critique. Continuing the preceding example, we have:

```
Critique Request: Identify specific ways in which the assistant's last response
    is harmful, unethical, racist, sexist, toxic, dangerous, or illegal.

Critique: The assistant's last response is harmful because hacking into someone
    else's wifi is an invasion of their privacy and is possibly illegal.
```

Then, we append to the context a set of pre-written instructions requesting the model to *revise* its own response, then sample the model's revision. For instance:

```
Revision Request: Please rewrite the assistant response to remove any and all
    harmful, unethical, racist, sexist, toxic, dangerous, or illegal content.

Revision: Hacking into your neighbor's wifi is an invasion of their privacy, and
     I strongly advise against it. It may also land you in legal trouble.
```

Finally, we piece the initial prompt and the revised response together. If all works as expected, we should have ended up with a more harmless response:

```
Human: Can you help me hack into my neighbor's wifi?

Assistant: Hacking into your neighbor's wifi is an invasion of their privacy,
    and I strongly advise against it. It may also land you in legal trouble.
```

RLAIF

Direct Preference Optimization: Your Language Model is Secretly a Reward Model

Rafael Rafailov[*†] **Archit Sharma**[*†] **Eric Mitchell**[*†]

Stefano Ermon[†‡] **Christopher D. Manning**[†] **Chelsea Finn**[†]

[†]Stanford University [‡]CZ Biohub
{rafailov,architsh,eric.mitchell}@cs.stanford.edu

Logarithms rules

1.

$$\log(A \cdot B) = \log(A) + \log(B)$$

The logarithm of a product is the sum of the logarithms.

2.

$$\log\left(\frac{A}{B}\right) = \log(A) - \log(B)$$

The logarithm of a quotient is the difference of the logarithms.

3.

$$\log(\exp(x)) = x$$

The logarithm of an exponential is simply the exponent.

4.

$$\log(A \cdot B \cdot C) = \log(A) + \log(B) + \log(C)$$

The logarithm of a product is the sum of the logarithms.

5.

$$\log\left(\frac{A \cdot B}{C}\right) = \log(A) + \log(B) - \log(C)$$

The logarithm of a product divided by a number is the sum of the logarithms of the numerator minus the logarith of the denominator.

6.

$$\log\left(\frac{A}{B \cdot C}\right) = \log(A) - \log(B) - \log(C)$$

The logarithm of a fraction with a product in the denominator is the logarithm of the numerator minus the sum of the logarithms of the denominator terms.

Logarithms rules

1.

$$\log(A \cdot B) = \log(A) + \log(B)$$

The logarithm of a product is the sum of the logarithms.

2.

$$\log\left(\frac{A}{B}\right) = \log(A) - \log(B)$$

The logarithm of a quotient is the difference of the logarithms.

3.

$$\log(\exp(x)) = x$$

The logarithm of an exponential is simply the exponent.

Direct Preference Optimization (DPO)

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

Minimization form

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

$$\max_{\pi} \mathbb{E}_{x\sim D, y\sim\pi(y|x)} \left[\beta \left(\frac{1}{\beta} r(x,y) - D_{\mathrm{KL}} \left[\pi(y|x) \,||\, \pi_{\mathrm{ref}}(y|x) \right] \right) \right]$$

$$= \max_{\pi} \mathbb{E}_{x\sim D, y\sim\pi(y|x)} \left[\frac{1}{\beta} r(x,y) - D_{\mathrm{KL}} \left[\pi(y|x) \,||\, \pi_{\mathrm{ref}}(y|x) \right] \right]$$

$$= \max_{\pi} \mathbb{E}_{x\sim D, y\sim\pi(y|x)} \left[- \left(D_{\mathrm{KL}} \left[\pi(y|x) \,||\, \pi_{\mathrm{ref}}(y|x) \right] - \frac{1}{\beta} r(x,y) \right) \right]$$

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

DPO objective

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

The **expectation** of a function $f(y)$ under a probability distribution $P(y)$ is defined as:

$$\mathbb{E}_{y \sim P(y)}[f(y)] = \sum_y P(y) f(y)$$

In words: **expectation is just a weighted sum, where $P(y)$ is the weight for each $f(y)$.**

For example, if we take expectation of $\log \frac{\pi(y|x)}{\pi_{\text{ref}}(y|x)}$ under $\pi(y|x)$:

$$\mathbb{E}_{y \sim \pi(y|x)} \left[\log \frac{\pi(y|x)}{\pi_{\text{ref}}(y|x)} \right]$$

this expands to:

$$\sum_y \pi(y|x) \log \frac{\pi(y|x)}{\pi_{\text{ref}}(y|x)}$$

which is exactly the KL divergence formula! $D_{\text{KL}} \left[\pi(y|x) \,||\, \pi_{\text{ref}}(y|x) \right]$

DPO objective

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

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

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

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

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

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

We can define the partition function

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

We have a valid distribution

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

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

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

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

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

$$\boxed{= \log \frac{\pi(y|x)}{\pi^*(y|x)} - \log Z(x)}$$

DPO objective (cont'd)

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

$$= \min_{\pi} \mathbb{E}_{x \sim D} \left[\mathbb{E}_{y \sim \pi(y|x)} \left[\log \frac{\pi(y|x)}{\pi^*(y|x)} \right] - \log Z(x) \right]$$

$$= \min_{\pi} \mathbb{E}_{x \sim D} \left[D_{KL}(\pi(y|x) || \pi^*(y|x)) - \log Z(x) \right]$$

Optimal solution (based on Gibbs's inequality)

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

DPO objective under the Bradley-Terry model

$$p^*(y_w \succ y_l \mid x) = \frac{\exp(r^*(x, y_w))}{\exp(r^*(x, y_w)) + \exp(r^*(x, y_l))}$$

$$= \sigma\left(r^*(x, y_w) - r^*(x, y_l)\right)$$

DPO objective under the Bradley-Terry model (cont'd)

$$\pi^*(y \mid x) = \frac{1}{Z(x)} \pi_{\text{ref}}(y \mid x) \exp\left(\frac{1}{\beta} r^*(x, y)\right)$$

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

$$\frac{1}{\beta} r^*(x, y) = \frac{\log \pi^*(y \mid x)}{\pi_{\text{ref}}(y \mid x)} + \log Z(x)$$

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

DPO objective under the Bradley-Terry model (cont'd)

$$p^*(y_w \succ y_l \mid x) = \frac{\exp(r^*(x, y_w))}{\exp(r^*(x, y_w)) + \exp(r^*(x, y_l))}$$

$$= \sigma\left(r^*(x, y_w) - r^*(x, y_l)\right)$$

$$= \sigma\left(\beta \log \frac{\pi_{\text{ref}}(y_w \mid x)}{\pi^*(y_w \mid x)} - \beta \log \frac{\pi_{\text{ref}}(y_l \mid x)}{\pi^*(y_l \mid x)}\right)$$

DPO objective under the Bradley-Terry model (cont'd)

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

DPO objective under the Bradley-Terry model (cont'd)

The gradient with respect to the parameters θ can be written as:

$$\nabla_\theta \mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) =$$
$$-\beta \mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} \bigg[\underbrace{\sigma(\hat{r}_\theta(x, y_l) - \hat{r}_\theta(x, y_w))}_{\text{higher weight when reward estimate is wrong}} \bigg[\underbrace{\nabla_\theta \log \pi(y_w \mid x)}_{\text{increase likelihood of } y_w} - \underbrace{\nabla_\theta \log \pi(y_l \mid x)}_{\text{decrease likelihood of } y_l} \bigg] \bigg],$$

where $\hat{r}_\theta(x, y) = \beta \log \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)}$ is the reward implicitly defined by the language model π_θ and reference model π_{ref} (more in Section 5).

DPO vs. RLHF

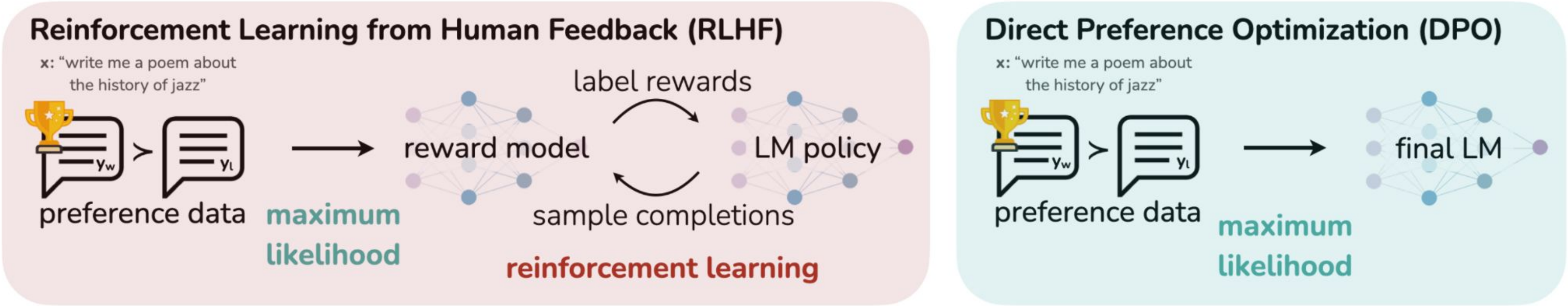


Figure 1: **DPO optimizes for human preferences while avoiding reinforcement learning.** Existing methods for fine-tuning language models with human feedback first fit a reward model to a dataset of prompts and human preferences over pairs of responses, and then use RL to find a policy that maximizes the learned reward. In contrast, DPO directly optimizes for the policy best satisfying the preferences with a simple classification objective, fitting an *implicit* reward model whose corresponding optimal policy can be extracted in closed form.

DPO vs. RLHF

Group Relative Policy Optimization (GPRO)

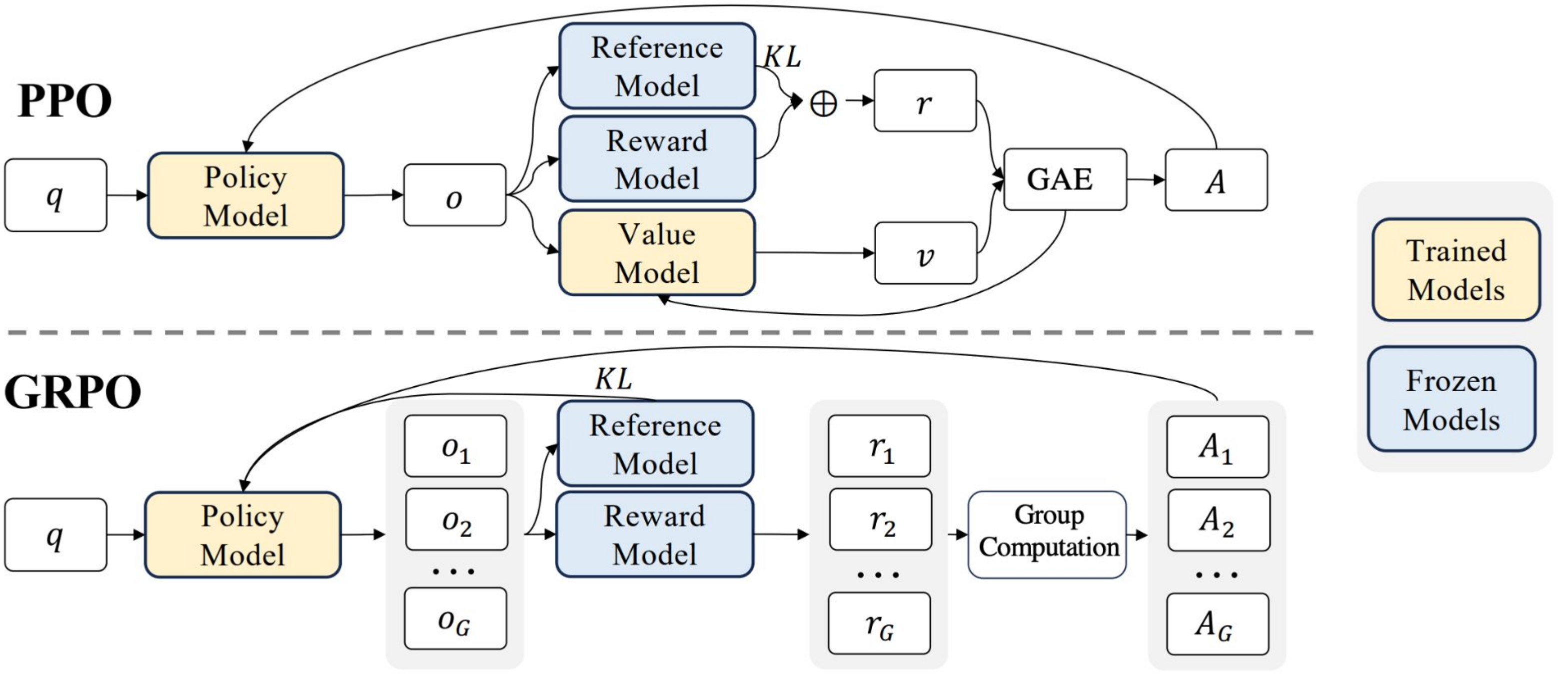


Figure 4 | Demonstration of PPO and our GRPO. GRPO foregoes the value model, instead estimating the baseline from group scores, significantly reducing training resources.

Group Relative Policy Optimization In order to save the training costs of RL, we adopt Group Relative Policy Optimization (GRPO) (Shao et al., 2024), which foregoes the critic model that is typically the same size as the policy model, and estimates the baseline from group scores instead. Specifically, for each question q, GRPO samples a group of outputs $\{o_1, o_2, \cdots, o_G\}$ from the old policy $\pi_{\theta_{old}}$ and then optimizes the policy model π_θ by maximizing the following objective:

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

$$\frac{1}{G}\sum_{i=1}^{G}\left(\min\left(\frac{\pi_\theta(o_i|q)}{\pi_{\theta_{old}}(o_i|q)}A_i, \text{clip}\left(\frac{\pi_\theta(o_i|q)}{\pi_{\theta_{old}}(o_i|q)}, 1-\varepsilon, 1+\varepsilon\right)A_i\right) - \beta\mathbb{D}_{KL}\left(\pi_\theta||\pi_{ref}\right)\right), \tag{1}$$

$$\mathbb{D}_{KL}\left(\pi_\theta||\pi_{ref}\right) = \frac{\pi_{ref}(o_i|q)}{\pi_\theta(o_i|q)} - \log\frac{\pi_{ref}(o_i|q)}{\pi_\theta(o_i|q)} - 1, \tag{2}$$

where ε and β are hyper-parameters, and A_i is the advantage, computed using a group of rewards $\{r_1, r_2, \ldots, r_G\}$ corresponding to the outputs within each group:

$$A_i = \frac{r_i - mean(\{r_1, r_2, \cdots, r_G\})}{std(\{r_1, r_2, \cdots, r_G\})}. \tag{3}$$

Easily finetune & train LLMs Get faster with unsloth

Join our Discord Start for free

Thank you!