CSCE 638 Natural Language Processing Foundation and Techniques

Lecture 13: Human Preference Alignment

Kuan-Hao Huang

Spring 2025

(Some slides adapted from Jesse Mu and Hung-Yi Lee)

Course Project: Sign-Up

- https://docs.google.com/spreadsheets/d/15Rj4AovtHtlZxILbX1ydrw7lEylamXuV7Dtg7cBD2EU/edit?usp=sharing
- Please sign up by 2/26
- 3~4 each team

	3~4 members per team								
	Project Topic	**Member 1 (Name)**	**Member 1 (E-mail)**	**Member 2 (Name)**	**Member 2 (E-mail)**	**Member 3 (Name)**	**Member 3 (E-mail)**	**Member 4 (Name)**	**Member 4 (E-mail)**
Team 1									
Team 2									
Team 3	TBD	Wahib Kapdi	wahibkapdi@tamu.edu	Agastya Todi	agastyatodi@tamu.edu	Yash Honrao	yash.honrao@tamu.edu	Shivam Singhal	shivamsinghal@tanu.edu
Team 4	TBD	Sonjoy Paul	skpaul@tamu.edu	Mabon Ninan	ninanmm@tamu.edu	Ashwini Ravindran	ashwinir@tamu.edu		
Team 5	TBD	Raj Purohith Arjun	raj2001@tamu.edu	Venkateswarlu Nagineni	venkates2002@tamu.edu	Shuvam Chowdhury	schowdhury@tamu.edu	Jeffrey Kevin Joseph	jeffrey98@tamu.edu
Team 6	TBD	Prakhar Suryavansh	ps41@tamu.edu	Rusali Saha	rs0921@tamu.edu	Priyal Khapra	priyalkhapra@tamu.edu		
Team 7	TBD	Chuan-Hsin Wang	chuanhsin0110@tamu.edu	Wei-Chien Cheng	wccheng@tamu.edu	Chi-Ming Lee	chiminglee831@tamu.ed		
Team 8	TBD	Afreen Ahmed	afreen04@tamu.edu	Rhea Sudheer	rheasudheer19@tamu.e	Hitha Magadi Vijayanand	hoshi_1996@tamu.edu	Sai Aakarsh Padma	saiaakarsh@tamu.edu
Team 9	Multilingual Video Grounding: Cross-Language Temporal Loc	Ramana Heggadal Math	ramana_hm@tamu.edu	Ruthvik Kanumuri	kruthvik007@tamu.edu	Jnana Preeti Parlapalli	pj.preeti@tamu.edu	Shravan Conjeevaram	shravan10@tamu.edu
Team 10	TBD	Harshavardhana	asharsha30@tam.edu	Rucha Ravindra Gole	ruchagole16@tamu.edu	Shashank Santosh Jagtap	shashankjagtap@tamu.e		
Team 11	TBD	Logan Bibb	logan.bibb@tamu.edu	Daniel Ortiz-Chaves	dortizchaves@tamu.edu	Sicong Liang	lsc206573@tamu.edu		
Team 12	TBD	Yifan Ren	yfren@tamu.edu	Qinyao Hou	yaoya2618@tamu.edu	Caroline Li	zhiheng@tamu.edu		
Team 13	TBD	Tien-Hung Hsiao	th.hsaio@tamu.edu	Barry Liu	barry89130663@gmail.c	Hsueh-chien Chao	alanchao8669@tamu.edu		
Team 14	TBD	Esben Egholm	esbenegholm@tamu.edu	Michael Norman	michael.norman@tamu.e	Davran Damkhan	davrandamkhan@tamu.e		
Team 15	In-Context Learning with LLMs	Dheeraj Mudireddy	dheeraj.reddy@tamu.edu	Ninad Deo	ninzo_05@tamu.edu	Dhruvraj Singh Rathore	dhruvraj_16@tamu.edu	Atharva Phand	ahphand@tamu.edu
Team 16	TBD	Tejashri K	tkelhe@tamu.edu	Sukanya Sahoo	sukanya.sahoo@tamu.e	Ramneek Kaur	ramneek983@tamu.edu	Saksham Mehta	saksham19@tamu.edu
Team 17	TBD	Arnav Jain	arnavkj11@tamu.edu	Parangjothi	parangjothi.c@tamu.edu	Medha Majumdar	medhamajumdar1@tamu		
Team 18	TBD	Adarsh Kumar	adarsh0801@tamu.eu	Neil Roy	neilroy@tamu.edu	Hwiyoon Kim	hwiyoonkim@tamu.edu	Jawahar Sai Nathani	jawaharsainathani@tamu
Team 19	TBD	Satvik Praveen	satvikpraveen_164@tamu.edu	Jonathan Tong	tongjo@tamu.edu	Vinay Chandra Bandi	vinaychandra@tamu.edu	Yamini Preethi Kamisetty	yamini_preethi_k@tamu.edu
Team 20	TBD	Piyush Sharan	pisharan@tamu.edu	Manisha Panda	mpanda27@tamu.edu	Abhishek Singh	abhi_singh@tamu.edu	Jaydeep Radadiya	jdr@tamu.edu
Team 21	TBD	Yamini Preethi Kamisetty	yamini_preethi_k@tamu.edu	Vinay Chandra Bandi					
Team 22	Jailbreaking LLMs using Graph of Thought	Aayush Upadhyay	aaupadhy@tamu.edu	Anant Mehta	anant_mehta@tamu.edu	Ajay Jagannath	ajayjagan2511@tamu.ed		
Team 23	TBD	Dishant Parag Zaveri	dishant.zaveri@tamu.edu	Saransh Agrawal	saransh.agrawal@tamu.	Faizan Ali Khaji	khajifaizanali@tamu.edu	Pavan Santosh	pavan_santosh@tamu.e
Team 24	TBD	Bita Malekianboroujeni	Bita.malekian@tamu.edu	Kimia Mirhosseini	k_imia1379@tamu.edu	Maddhurima Monddal	mmkpa2012@tamu.edu		
Team 25									

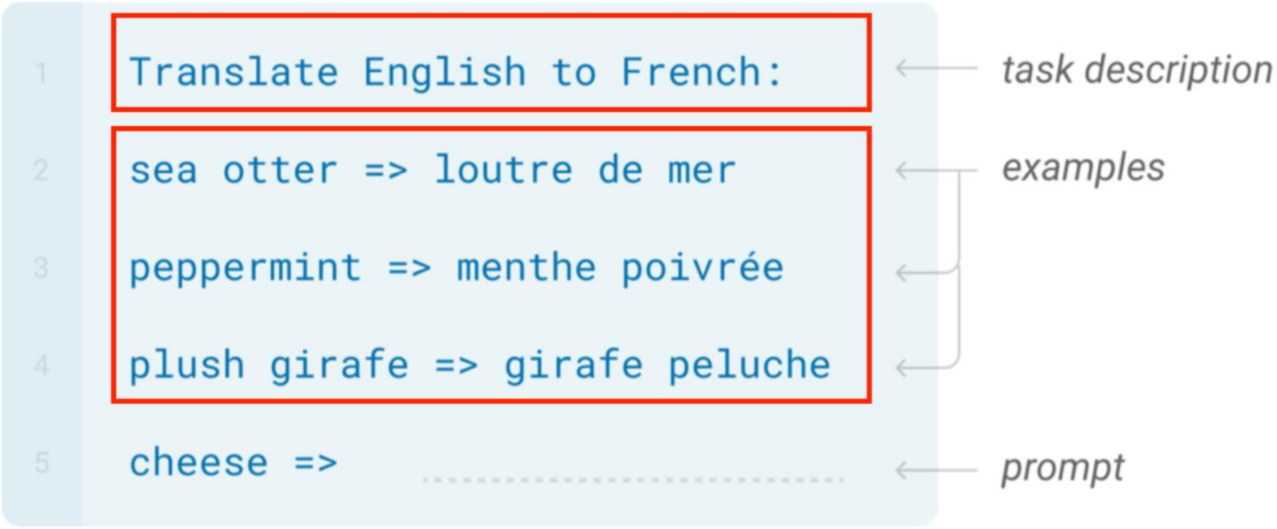
Course Project: Project Highlight

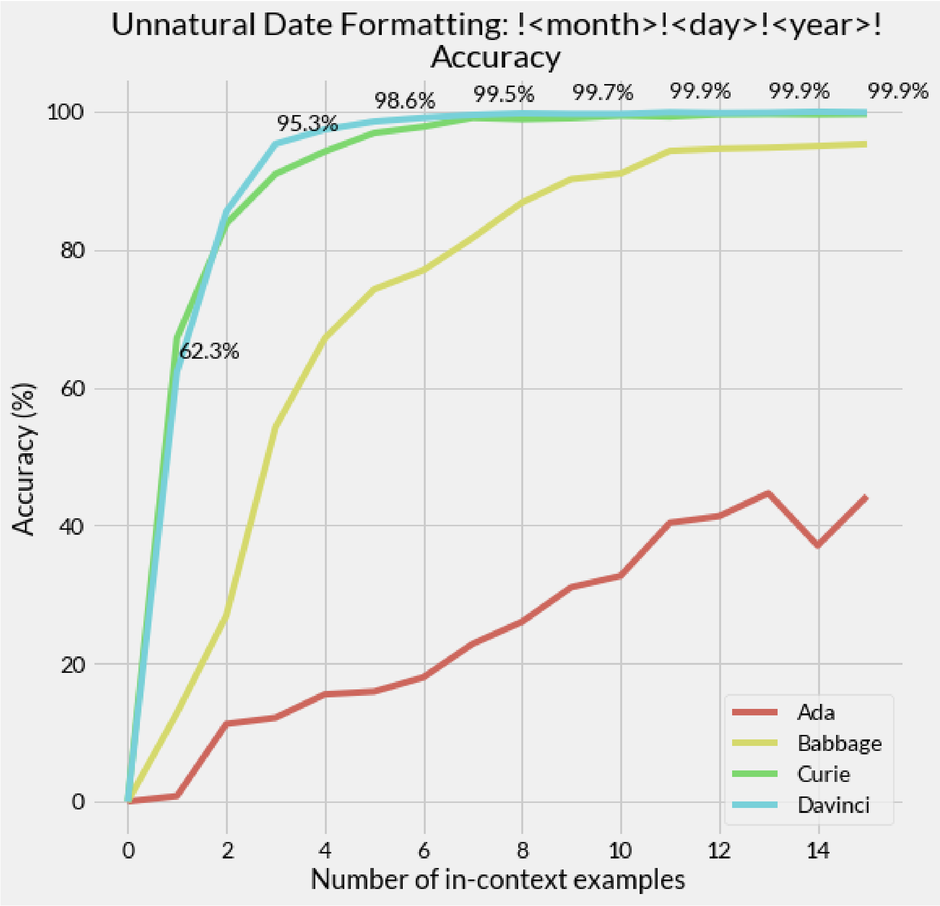
- Date: 3/5 in person
- Each team has 3 minutes to introduce the project
  - Introduction to the topic you choose
  - Short related literature overview
  - Novelty and challenges
  - The dataset, models, and approaches you plan to use
  - Evaluation plan

Lecture Plan

- Human Preference Optimization
  - Reinforcement Learning from Human Feedback / Proximal Policy Optimization
  - Direct Preference Optimization
  - Kahneman-Tversky Optimization
  - Simple Preference Optimization
  - Group Relative Policy Optimization

Recap: Few-Shot Prompting / In-Context Learning





In-context learning examples
Demonstration examples

Language Models are Few-Shot Learners, 2020

Recap: Chain-of-Thought Prompting

(a) Few-shot

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?
A: The answer is 11.

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A:

(Output) The answer is 8. ✗

(b) Few-shot-CoT

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?
A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11. The answer is 11.

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A:

(Output) The juggler can juggle 16 balls. Half of the balls are golf balls. So there are 16 / 2 = 8 golf balls. Half of the golf balls are blue. So there are 8 / 2 = 4 blue golf balls. The answer is 4. ✓

(c) Zero-shot

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A: The answer (arabic numerals) is

(Output) 8 ✗

(d) Zero-shot-CoT (Ours)

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A: ***Let's think step by step.***

(Output) There are 16 balls in total. Half of the balls are golf balls. That means that there are 8 golf balls. Half of the golf balls are blue. That means that there are 4 blue golf balls. ✓

Large Language Models are Zero-Shot Reasoners, 2022

Why Alignment?

Model input (Disambiguation QA)

Q: In the following sentences, explain the antecedent of the pronoun (which thing the pronoun refers to), or state that it is ambiguous.

Sentence: The reporter and the chef will discuss their favorite dishes.

Options:
(A) They will discuss the reporter's favorite dishes
(B) They will discuss the chef's favorite dishes
(C) Ambiguous

A: Let's think step by step.

Before instruction finetuning

The reporter and the chef will discuss their favorite dishes.
The reporter and the chef will discuss the reporter's favorite dishes.
The reporter and the chef will discuss the chef's favorite dishes.
The reporter and the chef will discuss the reporter's and the chef's favorite dishes.

✖ **(doesn't answer question)**

Model input (Disambiguation QA)

Q: In the following sentences, explain the antecedent of the pronoun (which thing the pronoun refers to), or state that it is ambiguous.

Sentence: The reporter and the chef will discuss their favorite dishes.

Options:
(A) They will discuss the reporter's favorite dishes
(B) They will discuss the chef's favorite dishes
(C) Ambiguous

A: Let's think step by step.

After instruction finetuning

The reporter and the chef will discuss their favorite dishes does not indicate whose favorite dishes they will discuss. So, the answer is (C). ✔

Instruction Tuning

- LLMs have knowledge, but don't always generate the outputs we want
- Training LLMs to following human instructions

Annotated task definitions

You will be given two pieces of text… One of them is simpler … You are expected to output 'Text one' if the first sentence is simpler. Otherwise output 'Text two'.

Given a sentence with a missing word, pick the answer option that best fills out the missing word in the sentence. Indicate each answer with its index ('a', 'b', 'c', 'd').

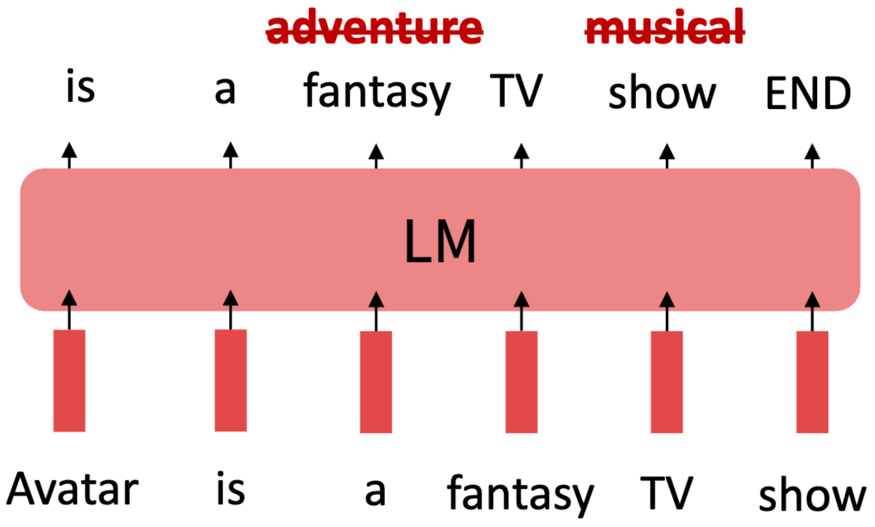
Given a document, generate a short title of the document. The title should convey the main idea/event/topic about which the document is being written.

Category	Description
Input Content	Primary description of the task input
Additional Input Content	Additional details on task input
Action Content	Action to perform for task
Input Mention	Mentions of input within action content
Output Content	Primary description of task output
Additional Output Content	Additional details on task output
Label List	Task output labels (classification only)
Label Definition	Task Label definitions (classification only)

Limitations of Instruction Fine-Tuning

- It is expensive to collect ground-truth data for tasks
- Open-ended creative generation have no right answer
  - E.g., write me a story about a dog and her pet grasshopper
- language modeling penalizes all token-level mistakes equally, but some errors are worse than others

Even with instruction finetuning, there is still a mismatch between the LM objective and "satisfying human preferences"!



Alignment Pipeline

Reinforcement Learning from Human Feedback (RLHF)

Training language models to follow instructions with human feedback

Long Ouyang* **Jeff Wu*** **Xu Jiang*** **Diogo Almeida*** **Carroll L. Wainwright***

Pamela Mishkin* **Chong Zhang** **Sandhini Agarwal** **Katarina Slama** **Alex Ray**

John Schulman **Jacob Hilton** **Fraser Kelton** **Luke Miller** **Maddie Simens**

Amanda Askell[†] **Peter Welinder** **Paul Christiano***[†]

Jan Leike* **Ryan Lowe***

OpenAI

Human Feedback

- Human reward

```
SAN FRANCISCO,
California (CNN) --
A magnitude 4.2
earthquake shook the
San Francisco
...
overturn unstable
objects.
```

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

s_1

$$R(s_1) = 8.0$$

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

s_2

$$R(s_2) = 1.2$$

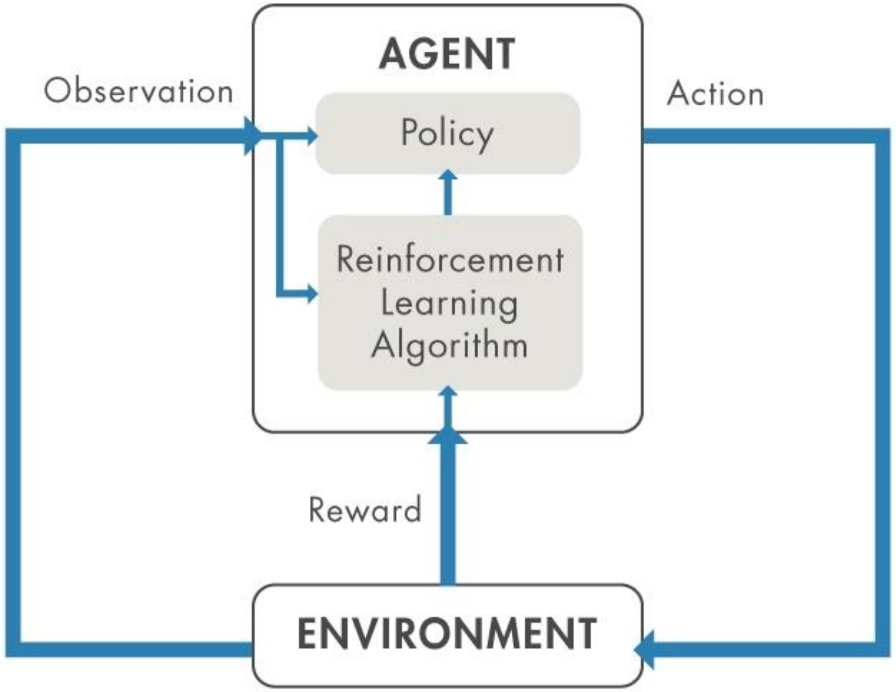
Goal: maximize the expected reward of samples from our LM

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

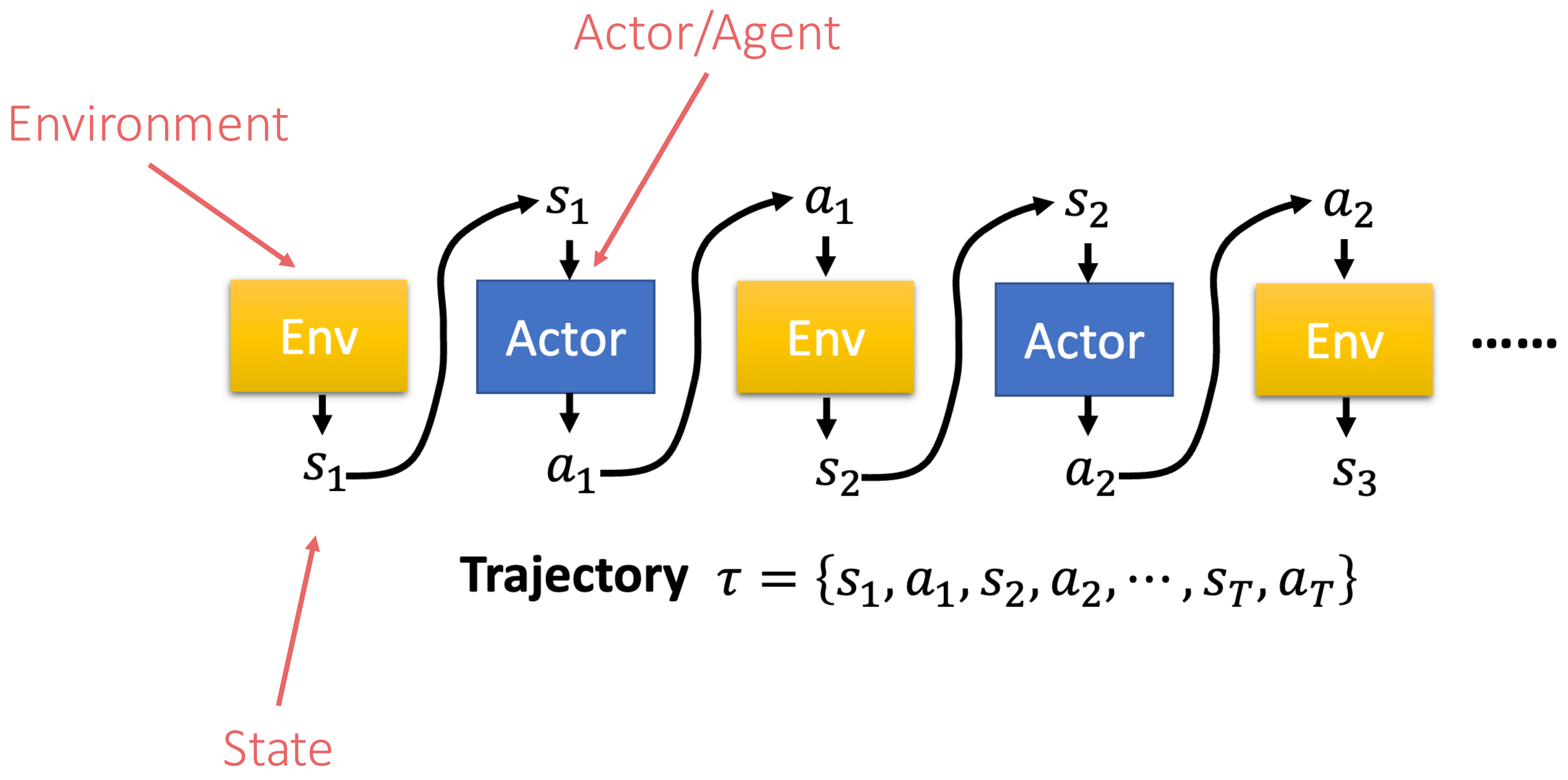
Reinforcement Learning from Human Preferences

How do we change the LM parameters θ to maximize this?

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

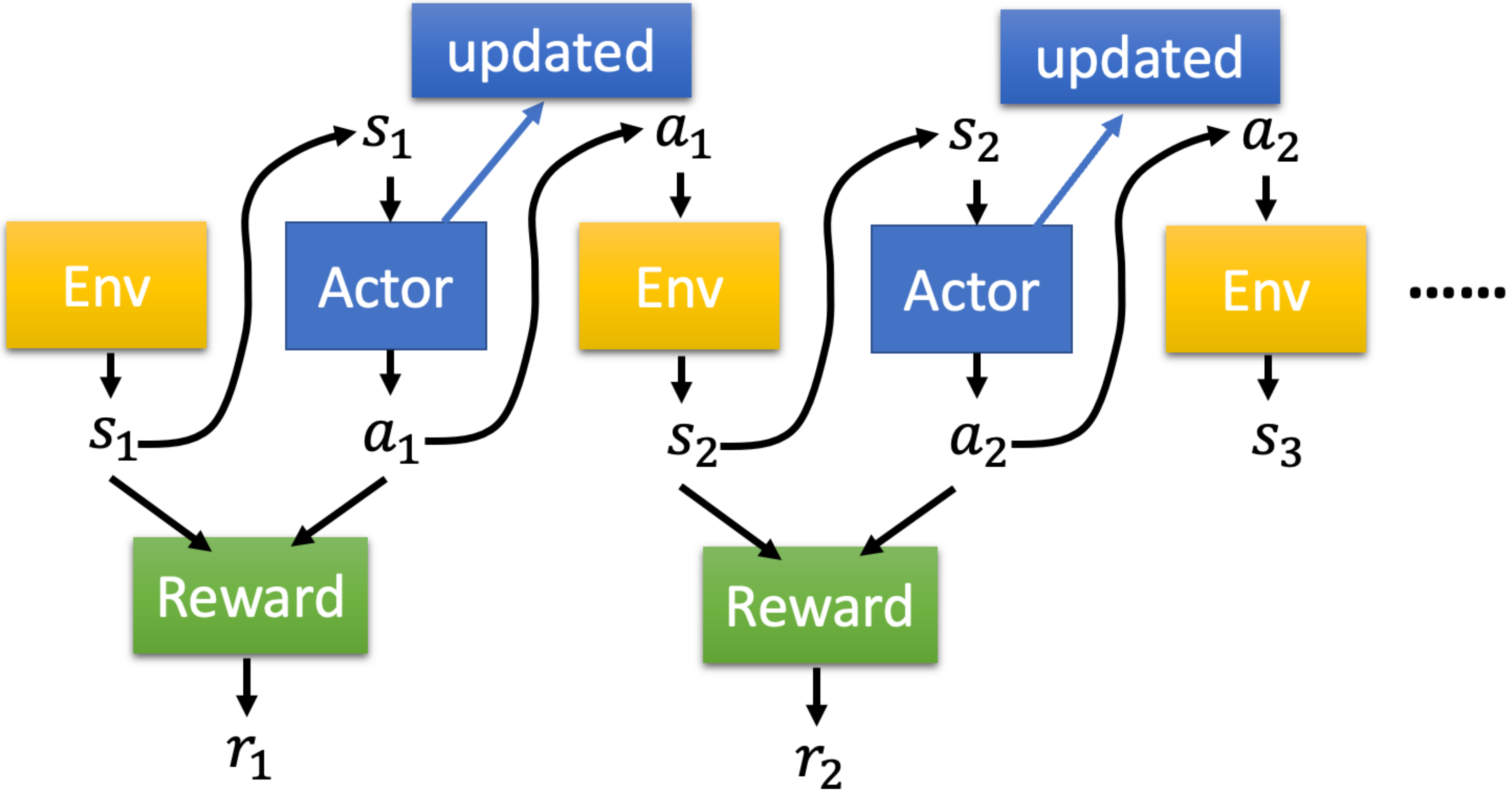


Reinforcement Learning

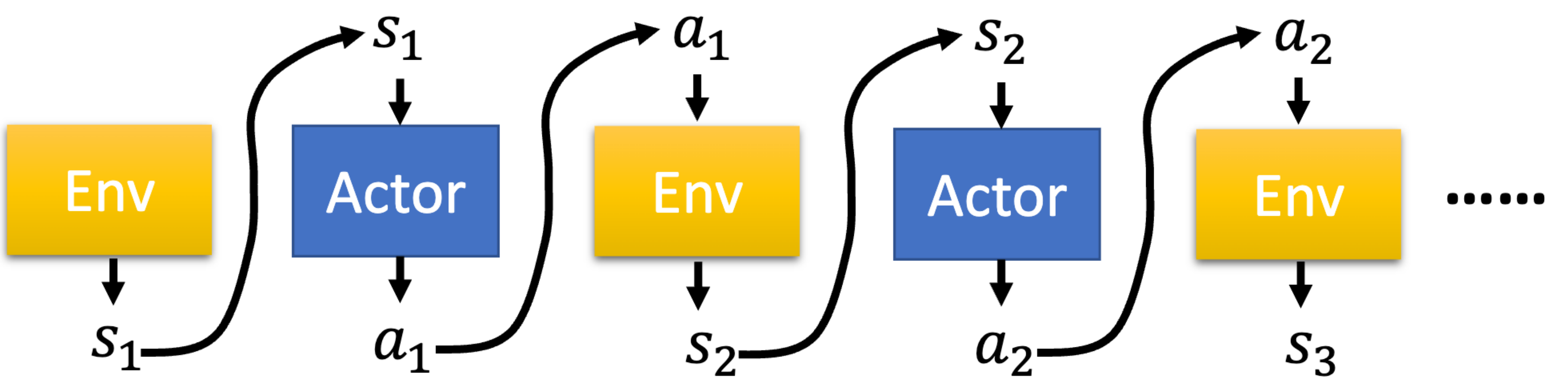


Trajectory $\tau = \{s_1, a_1, s_2, a_2, \cdots, s_T, a_T\}$

Reinforcement Learning



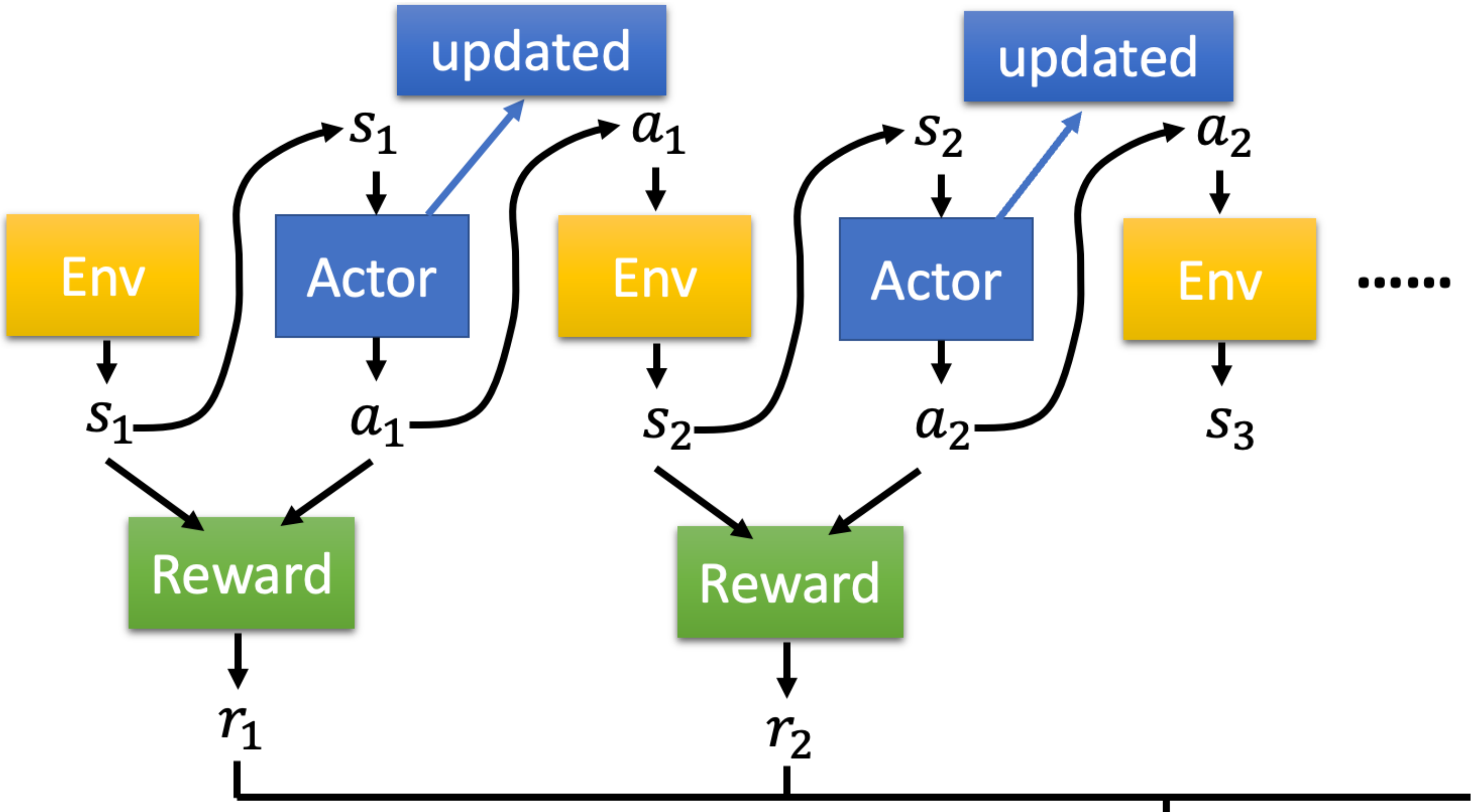
Reinforcement Learning



Trajectory $\tau = \{s_1, a_1, s_2, a_2, \cdots, s_T, a_T\}$

$$
\begin{aligned}
& p_\theta(\tau) \\
& \quad = p(s_1) p_\theta(a_1|s_1) p(s_2|s_1, a_1) p_\theta(a_2|s_2) p(s_3|s_2, a_2) \cdots \\
& \quad = p(s_1) \prod_{t=1}^{T} p_\theta(a_t|s_t) p(s_{t+1}|s_t, a_t)
\end{aligned}
$$

Reinforcement Learning

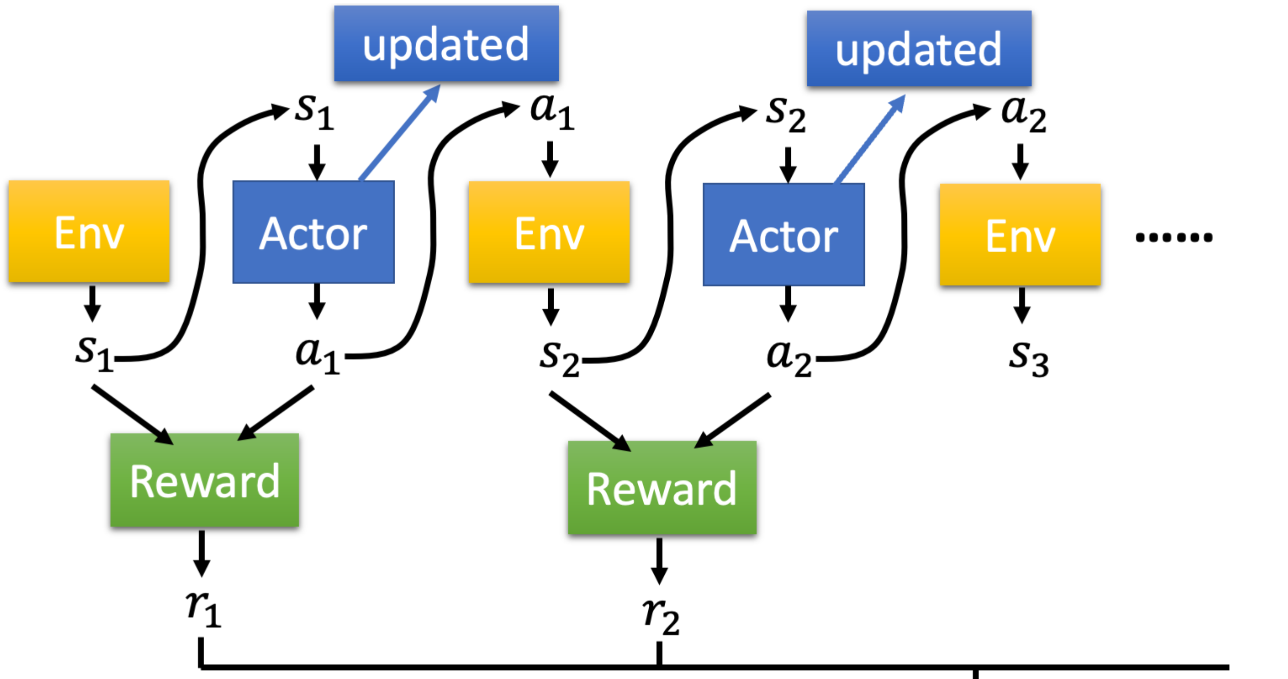


Expected Reward

$$\bar{R}_\theta = \sum_\tau R(\tau) p_\theta(\tau) = E_{\tau \sim p_\theta(\tau)}[R(\tau)]$$

$$R(\tau) = \sum_{t=1}^{T} r_t$$

Reinforcement Learning



Expected Reward

$$\bar{R}_\theta = \sum_\tau R(\tau) p_\theta(\tau) = E_{\tau \sim p_\theta(\tau)}[R(\tau)]$$

$$R(\tau) = \sum_{t=1}^{T} r_t$$

Solutions

- Q-Learning
- Policy Gradient
- Actor-Critic
- …

Optimizing for Human Preferences

How do we change the LM parameters θ to maximize this?

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

Gradient Ascent

$$\theta_{t+1} := \theta_t + \alpha \nabla_{\theta_t} \mathbb{E}_{\hat{s}\sim p_{\theta_t}(s)}[R(\hat{s})]$$

Policy Gradient Methods in Reinforcement Learning
(REINFORCE) [Williams, 1992]

Policy Gradient/REINFORCE

Gradient Ascent

$$\theta_{t+1} \coloneqq \theta_t + \alpha \nabla_{\theta_t} \mathbb{E}_{\hat{s} \sim p_{\theta_t}(s)}[R(\hat{s})]$$

$$\nabla_\theta \mathbb{E}_{\hat{s} \sim p_\theta(s)}[R(\hat{s})] = \nabla_\theta \sum_s R(s) p_\theta(s) = \sum_s R(s) \nabla_\theta p_\theta(s)$$

Log-Derivative Trick

$$\nabla_\theta \log p_\theta(s) = \frac{1}{p_\theta(s)} \nabla_\theta p_\theta(s) \quad \Longrightarrow \quad \nabla_\theta p_\theta(s) = \nabla_\theta \log p_\theta(s)\, p_\theta(s)$$

Policy Gradient/REINFORCE

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

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

We can approximate this objective with Monte Carlo samples

$$\nabla_\theta \mathbb{E}_{\hat{s}\sim p_\theta(s)}[R(\hat{s})] = \mathbb{E}_{\hat{s}\sim p_\theta(s)}[R(\hat{s})\, \nabla_\theta \log p_\theta(\hat{s})] \approx \frac{1}{m}\sum_{i=1}^{m} R(s_i)\, \nabla_\theta \log p_\theta(s_i)$$

Policy Gradient/REINFORCE

Take gradient steps
to maximize $p_\theta(s_i)$

If R is +++

$$\theta_{t+1} := \theta_t + \alpha \frac{1}{m} \sum_{i=1}^{m} R(s_i)\, \nabla_{\theta_t} \log p_{\theta_t}(s_i)$$

If R is ---

Take steps to
minimize $p_\theta(s_i)$

We reinforce good actions, increasing the chance they happen again

Proximal Policy Optimization (PPO)

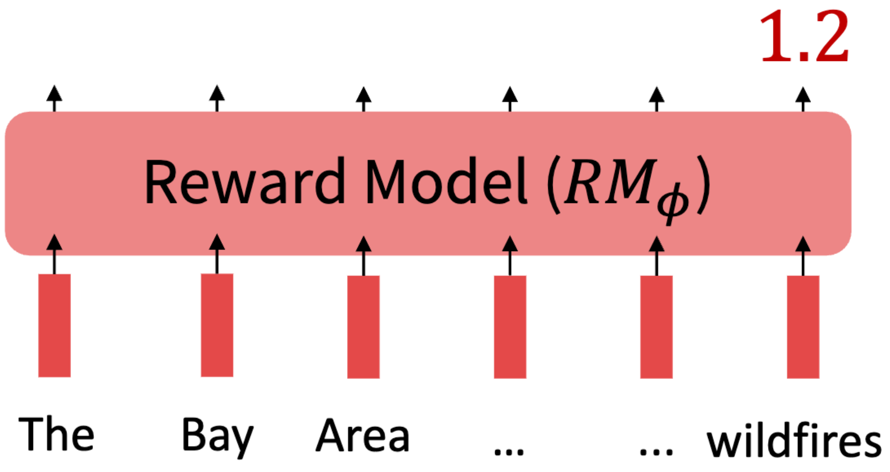
- New parameters θ' cannot be very different from old parameters θ

$$J_{PPO}^{\theta'}(\theta) = J^{\theta'}(\theta) - \beta KL(\theta, \theta')$$

Regularization

How to Model Human Preferences?

- Now for any reward function R, we can train our language model to maximize expected reward
- Problem 1: human-in-the-loop is expensive
  - Solution: instead of directly asking humans for preferences, model their preferences as a separate (NLP) problem
  - Train a reward model (RM) from an annotated dataset

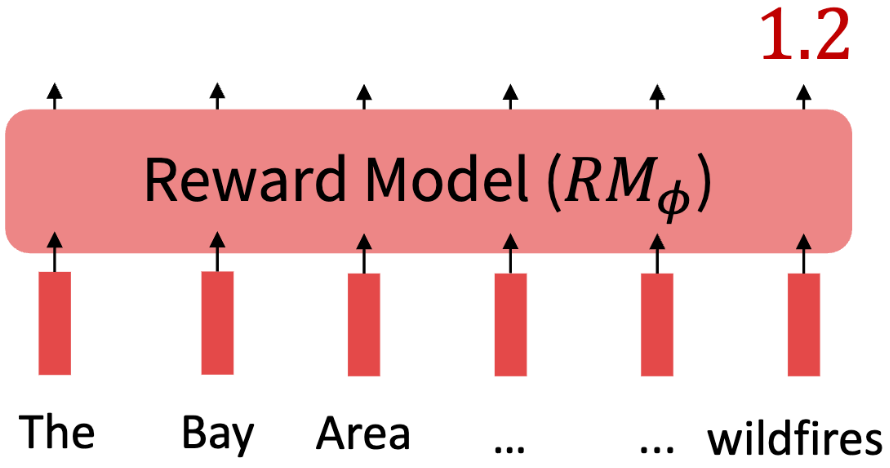


How to Model Human Preferences?

- Now for any reward function R, we can train our language model to maximize expected reward
- Problem 2: human judgments are noisy and miscalibrated
  - Solution: instead of asking for direct ratings, ask for pairwise comparisons, which can be more reliable

`An earthquake hit San Francisco. There was minor property damage, but no injuries.`	$>$	`A 4.2 magnitude earthquake hit San Francisco, resulting in massive damage.`	$>$	`The Bay Area has good weather but is prone to earthquakes and wildfires.`
s_1		s_3		s_2

Training A Reward Model



Bradley-Terry [1952] paired comparison model

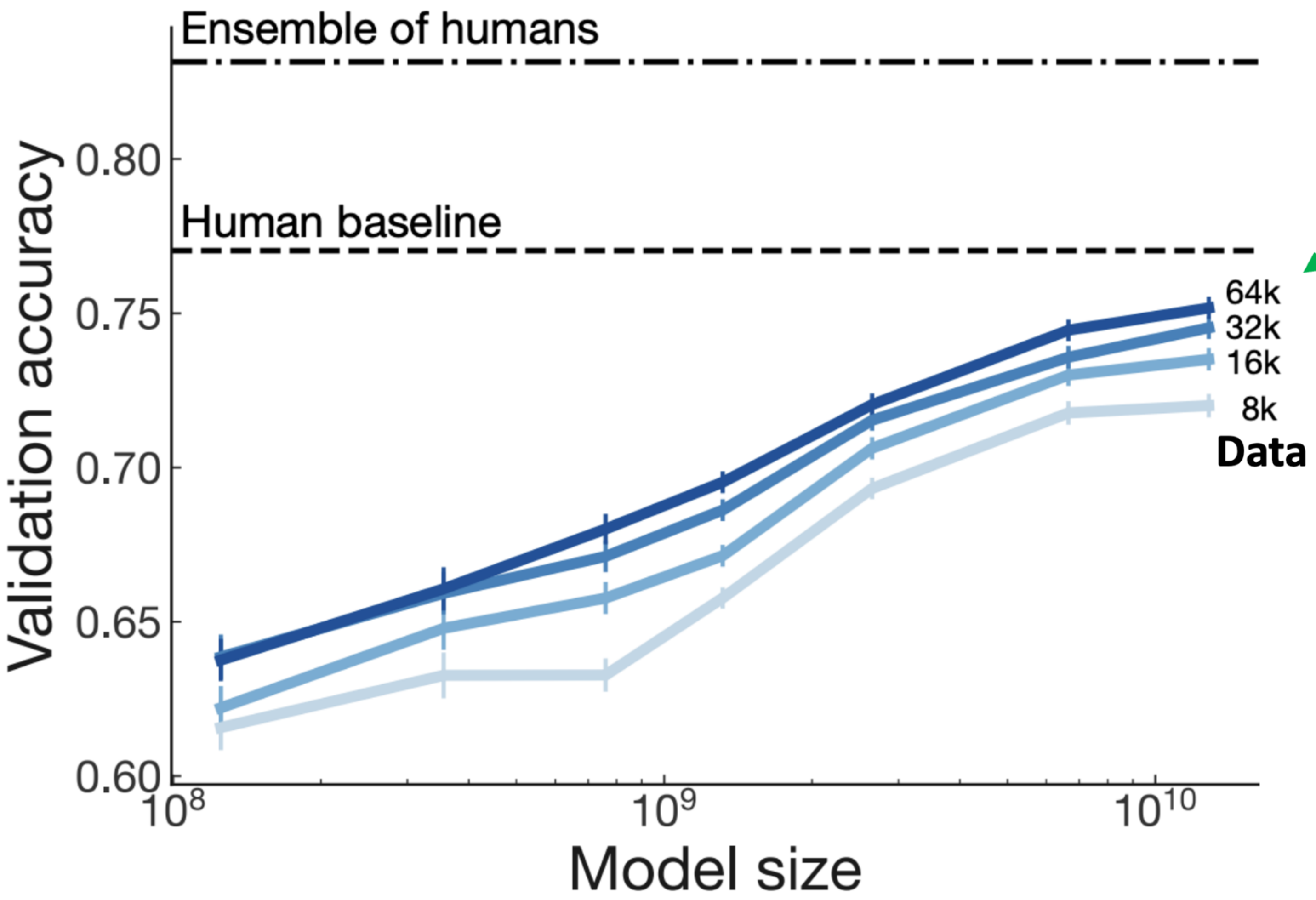
$$J_{RM}(\phi) = -\mathbb{E}_{(s^w, s^l) \sim D}\left[\log \sigma(RM_\phi(s^w) - RM_\phi(s^l))\right]$$

"winning" sample

"losing" sample

s^w should score higher than s^l

Reward Model vs. Real Human Feedback



Large enough RM trained on enough data approaching single human perf

[Stiennon et al., 2020]

RLHF: Putting Everything All Together

- We have the following:
  - A pretrained (possibly instruction-finetuned) LM $p^{PT}(y \mid x)$
  - A reward model $RM_\phi(x, y)$ that produces scalar rewards for LM outputs, trained on a dataset of human comparisons
- Now to do RLHF:
  - Copy the model $p_\theta^{RL}(y \mid x)$, with parameters θ we would like to optimize
  - We want to optimize:

$$\mathbb{E}_{\hat{y} \sim p_\theta^{RL}(\hat{y}|x)} \left[RM_\phi(x, \hat{y})\right]$$

RLHF: Putting Everything All Together

- We want to optimize:

$$\mathbb{E}_{\hat{y}\sim p_\theta^{RL}(\hat{y}|x)}\left[RM_\phi(x,\hat{y})\right]$$

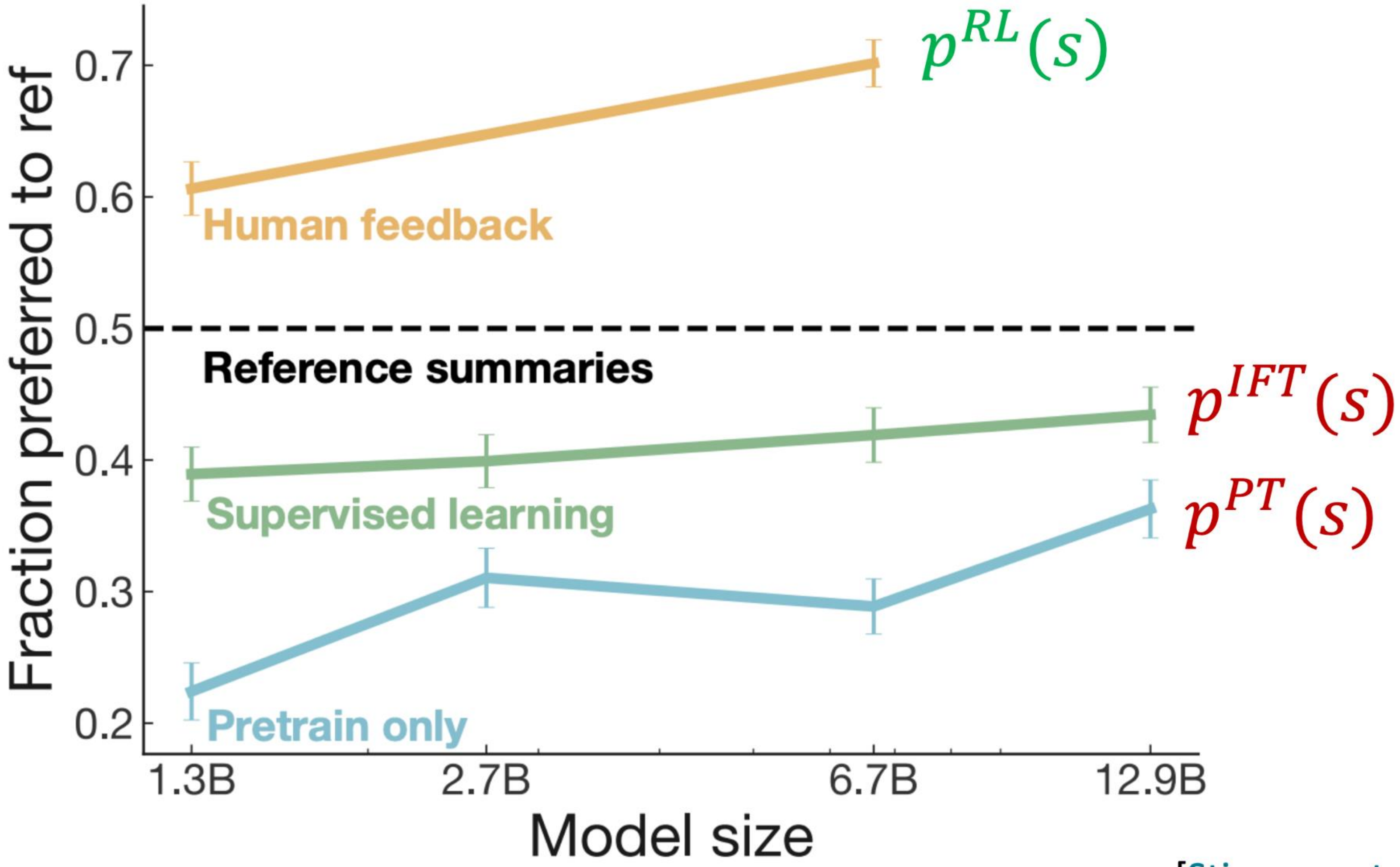
- Do you see any problems?
  - Learned rewards are imperfect; this quantity can be imperfectly optimized
- Add a penalty for drifting too for from the initialization:

$$\mathbb{E}_{\hat{y}\sim p_\theta^{RL}(\hat{y}|x)}\left[RM_\phi(x,\hat{y}) \; - \beta \underbrace{\log\left(\frac{p_\theta^{RL}(\hat{y}\mid x)}{p^{PT}(\hat{y}\mid x)}\right)}\right]$$

Pay a price when
$p_\theta^{RL}(\hat{y}\mid x) > p^{PT}(\hat{y}\mid x)$

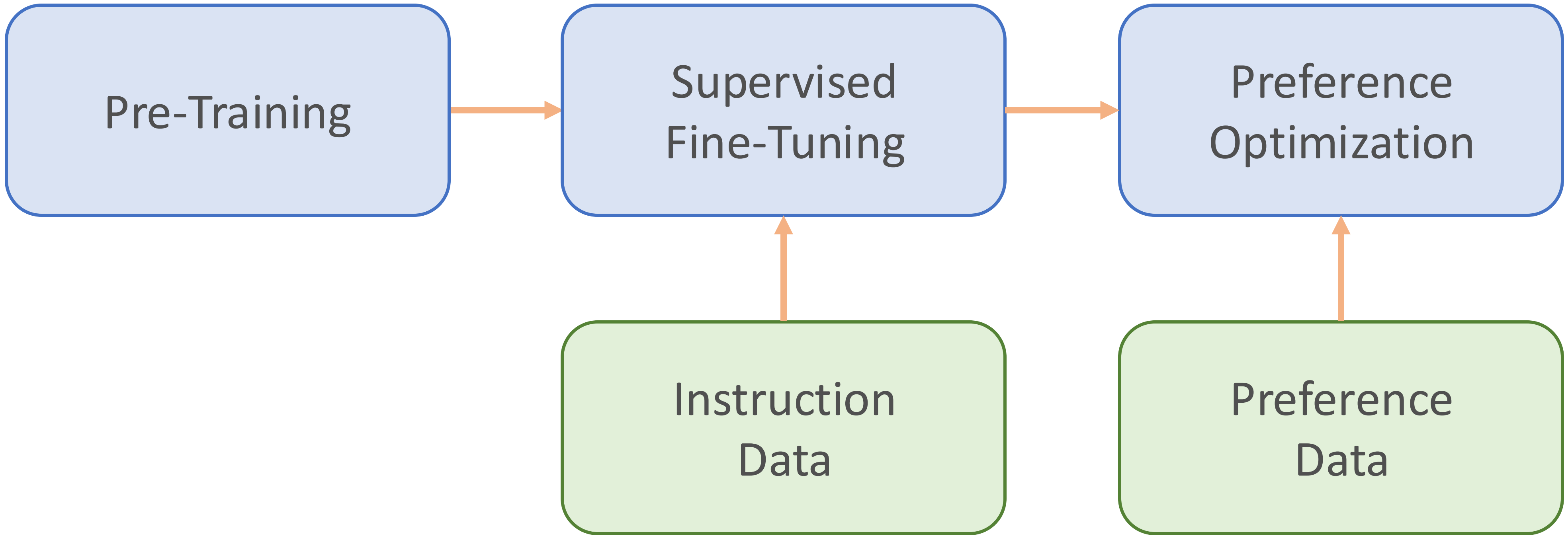
This penalty which prevents us from diverging too far from the pretrained model. In expectation, it is known as the **Kullback-Leibler** (**KL**) divergence between $p_\theta^{RL}(\hat{y}\mid x)$ and $p^{PT}(\hat{y}\mid x)$.

RLHF vs. Supervised Fine-Tuning



[Stiennon et al., 2020]

Alignment Pipeline



InstructGPT

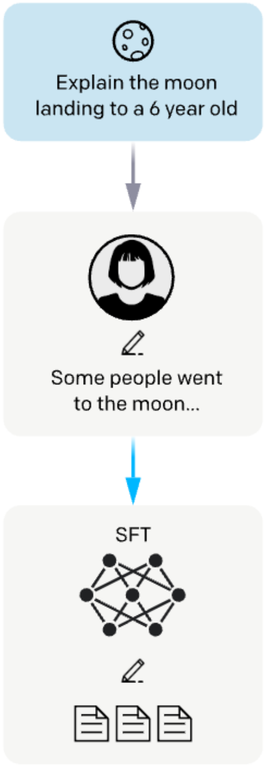
Step 1

Collect demonstration data, and train a supervised policy.

A prompt is sampled from our prompt dataset.

A labeler demonstrates the desired output behavior.

This data is used to fine-tune GPT-3 with supervised learning.



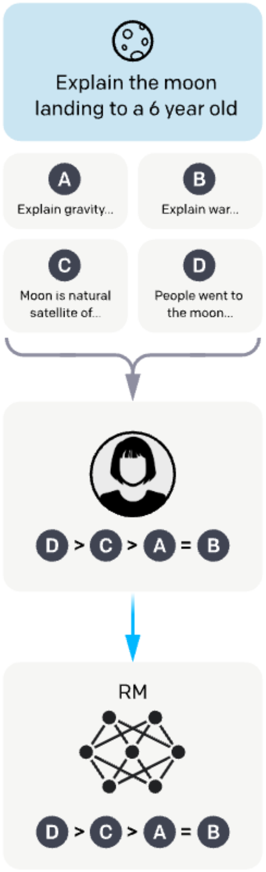
Step 2

Collect comparison data, and train a reward model.

A prompt and several model outputs are sampled.

A labeler ranks the outputs from best to worst.

This data is used to train our reward model.



Step 3

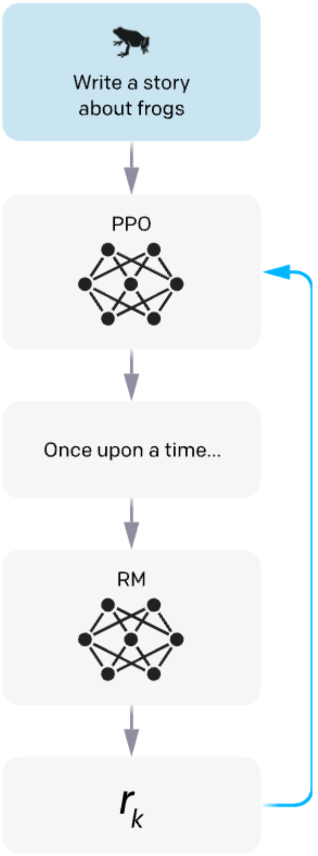
Optimize a policy against the reward model using reinforcement learning.

A new prompt is sampled from the dataset.

The policy generates an output.

The reward model calculates a reward for the output.

The reward is used to update the policy using PPO.



ChatGPT: Instruction Fine-tuning + RLHF for Dialog Agents

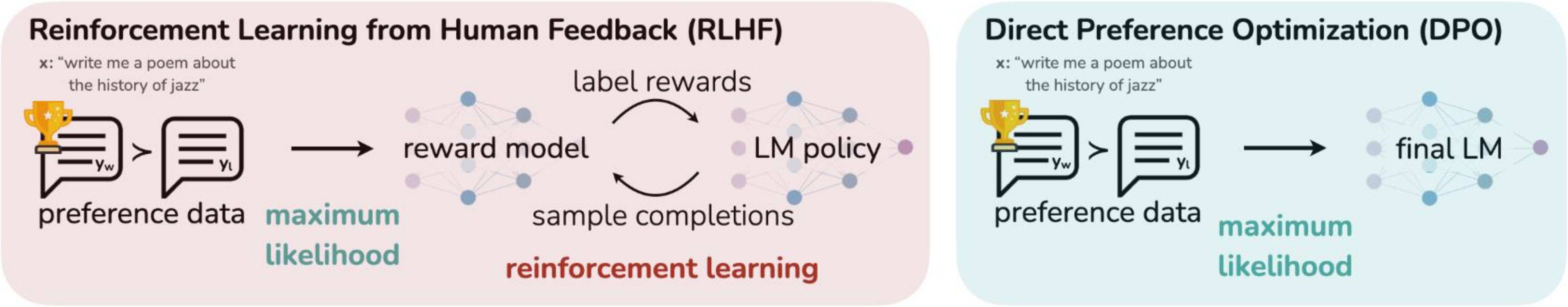
ChatGPT: Optimizing Language Models for Dialogue

Note: OpenAI (and similar companies) are keeping more details secret about ChatGPT training (including data, training parameters, model size)—perhaps to keep a competitive edge...

Methods

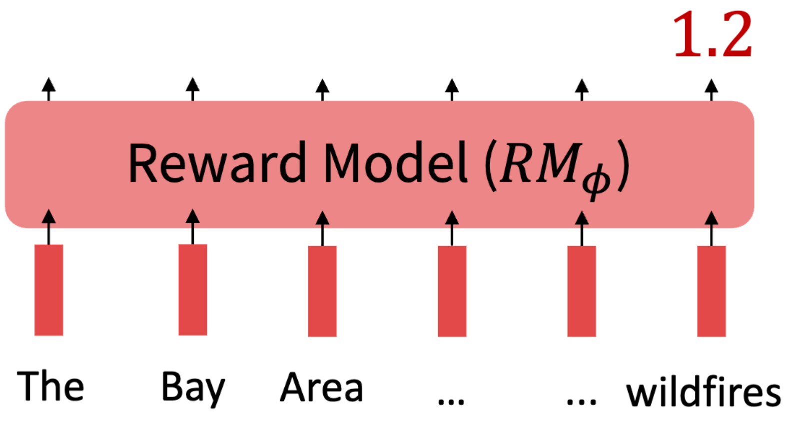
We trained this model using Reinforcement Learning from Human Feedback (RLHF), using the same methods as InstructGPT, but with slight differences in the data collection setup. We trained an initial model using supervised fine-tuning: human AI trainers provided conversations in which they played both 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.

Direct Preference Optimization (DPO)



Direct Preference Optimization: Your Language Model is Secretly a Reward Model, 2023

RLHF: Proximal Policy Optimization (PPO)




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

s_1

>

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

s_2

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

Direct Preference Optimization (DPO)

RLHF Objective

(get **high reward**, stay **close** to reference model)

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

Maximize reward

Keep similar behavior

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

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

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

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

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

Direct Preference Optimization (DPO)

RLHF Objective

(get **high reward**, stay **close** to reference model)

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

Maximize reward

Keep similar behavior

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

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

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

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

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

Direct Preference Optimization (DPO)

RLHF Objective

(get **high reward**, stay **close** to reference model)

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

Maximize reward

Keep similar behavior

Closed-form Optimal Policy

(write **optimal policy** as function of **reward function**; from prior work)

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

with $Z(x) = \sum_y \pi_{\mathrm{ref}}(y \mid x) \exp\left(\frac{1}{\beta} r(x,y)\right)$

Note **intractable sum** over possible responses; can't immediately use this

Rearrange

(write **any reward function** as function of **optimal policy**)

Ratio is **positive** if policy likes response more than reference model, **negative** if policy likes response less than ref. model

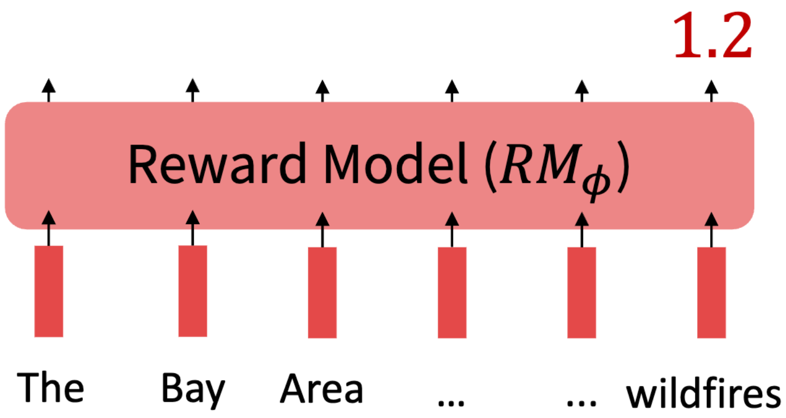
$$r(x,y) = \underbrace{\beta \log \frac{\pi^*(y \mid x)}{\pi_{\mathrm{ref}}(y \mid x)} + \beta \log Z(x)}_{\text{some parameterization of a reward function}}$$

Direct Preference Optimization (DPO)

A loss function on <u>reward functions</u>

Derived from the Bradley-Terry model of human preferences:

$$\mathcal{L}_R(r, \mathcal{D}) = -\mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} \left[\log \sigma(r(x, y_w) - r(x, y_l)) \right]$$



`An earthquake hit San Francisco. There was minor property damage, but no injuries.` > `The Bay Area has good weather but is prone to earthquakes and wildfires.`

s_1 s_2

Direct Preference Optimization (DPO)

Derived from the Bradley-Terry model of human preferences:

A loss function on <u>reward functions</u>

$$\mathcal{L}_R(r, \mathcal{D}) = -\mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} [\log \sigma(r(x, y_w) - r(x, y_l))]$$

+

A transformation between <u>reward functions</u> and <u>policies</u>

$$r_{\pi_\theta}(x, y) = \beta \log \frac{\pi_\theta(y \mid x)}{\pi_{\text{ref}}(y \mid x)} + \beta \log Z(x)$$

Direct Preference Optimization (DPO)

Derived from the Bradley-Terry model of human preferences:

A loss function on <u>reward functions</u>

$$\mathcal{L}_R(r, \mathcal{D}) = -\mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} \left[\log \sigma(r(x, y_w) - r(x, y_l)) \right]$$

+

A transformation between <u>reward functions</u> and <u>policies</u>

$$r_{\pi_\theta}(x, y) = \beta \log \frac{\pi_\theta(y \mid x)}{\pi_{\text{ref}}(y \mid x)} + \beta \log Z(x)$$

=

Reward of **preferred** response

Reward of **dispreferred** response

A loss function on <u>policies</u>

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

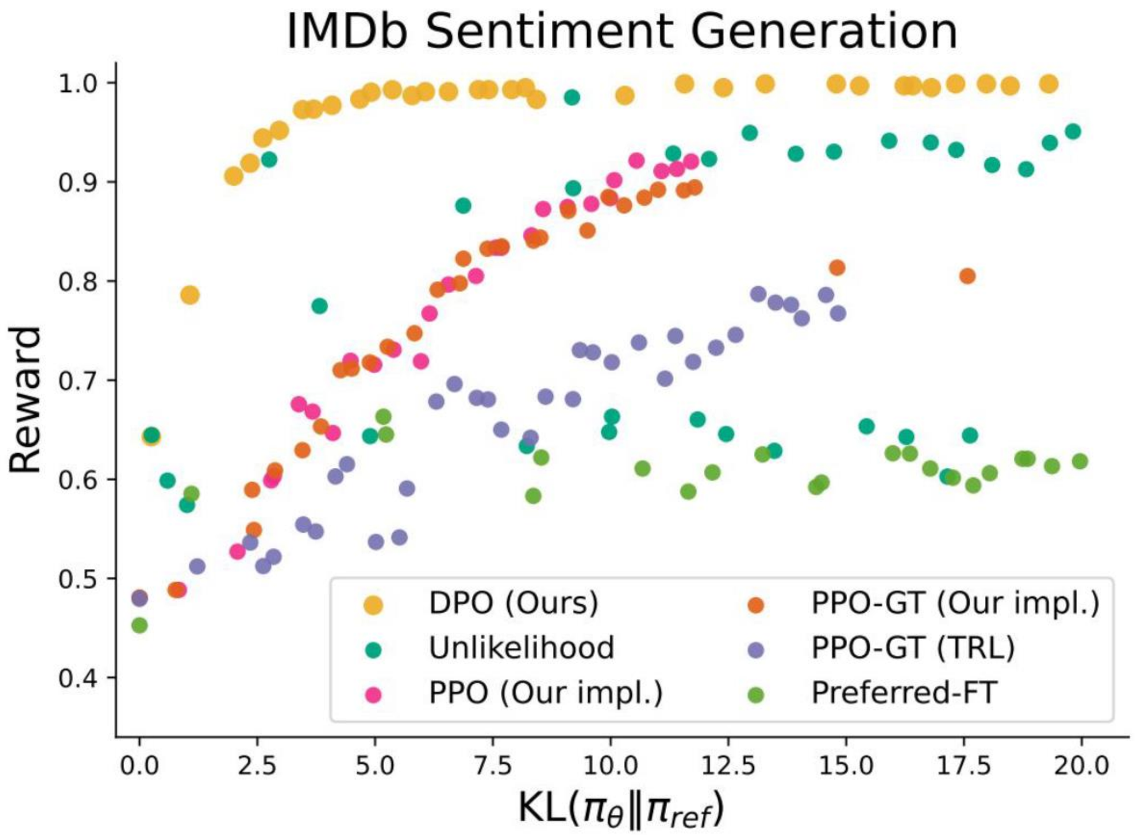
Direct Preference Optimization (DPO)

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

Reward of **preferred** response Reward of **dispreferred** response

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

DPO Performance

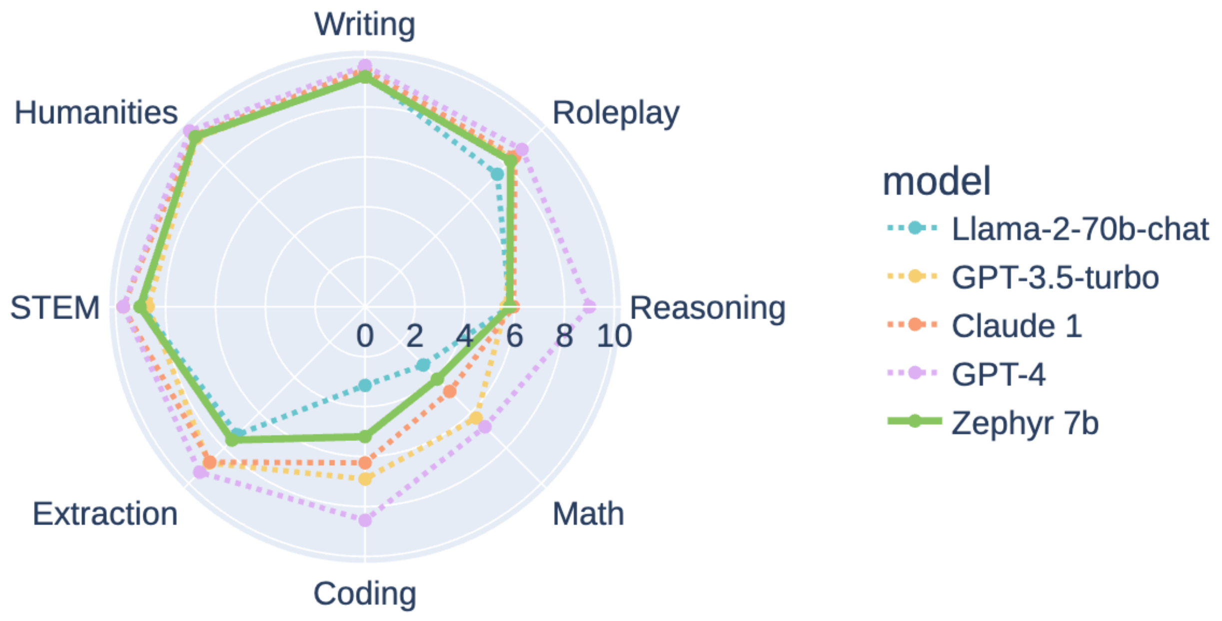


1. Generate positive IMDB reviews from GPT2-XL
2. Use pre-trained sentiment classifier as Gold RM
3. Create preferences based on Gold RM
4. Optimize with PPO and DPO

Large-Scale DPO Training

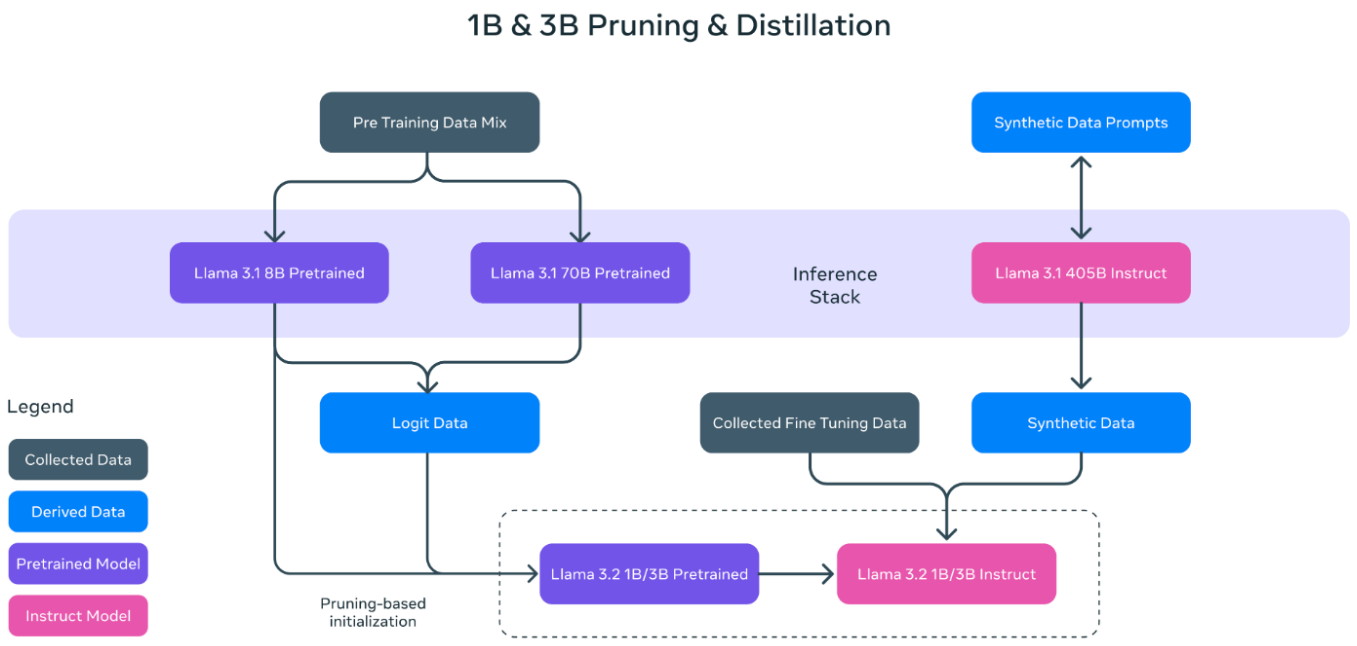
Zephyr: Direct Distillation of LM Alignment

Lewis Tunstall,* Edward Beeching,* Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, Nathan Sarrazin, Omar Sanseviero, Alexander M. Rush, and Thomas Wolf
The H4 (Helpful, Honest, Harmless, Huggy) Team
https://huggingface.co/HuggingFaceH4
lewis@huggingface.co



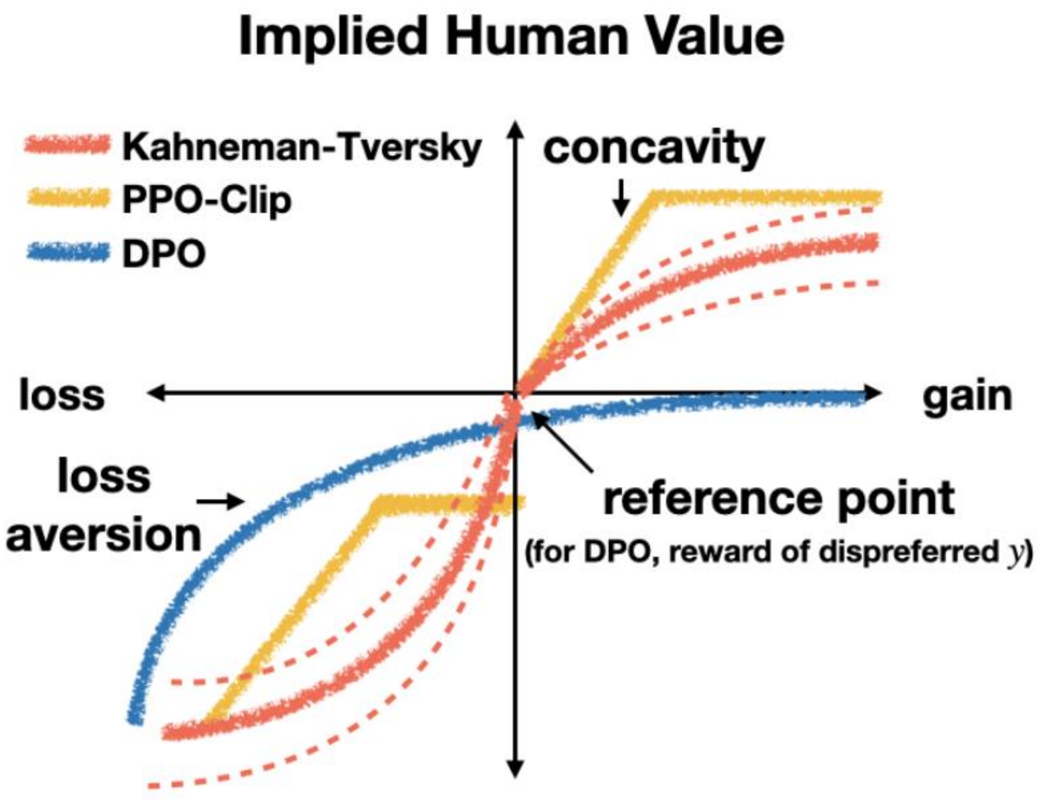
Large-Scale DPO Training

Llama 3.2: Revolutionizing edge AI and vision with open, customizable models



In post-training, we use a similar recipe as Llama 3.1 and produce final chat models by doing several rounds of alignment on top of the pre-trained model. Each round involves supervised fine-tuning (SFT), rejection sampling (RS), and direct preference optimization (DPO).

Kahneman-Tversky Optimization (KTO)



Which One Do You Choose?

- Imagine you are facing two choices:
  - **Choice one**: has an 80% chance of earning you 10 million US dollars, and a 20% chance of giving you nothing
  - **Choice two**: gives you 4 million US dollars for sure

https://medium.com/@yianyao1994/llm-alignments-part-6-kto-813b38be14ae

Which One Do You Choose?

- Imagine you are facing two choices:
  - **Choice one**: has an 80% chance of earning you 1 thousand US dollars, and a 20% chance of giving you nothing
  - **Choice two**: gives you 4 hundred US dollars for sure

https://medium.com/@yianyao1994/llm-alignments-part-6-kto-813b38be14ae

Which One Do You Choose?

- Imagine you are facing two choices:
  - **Choice one**: has an 80% chance of earning you 10 US dollars, and a 20% chance of giving you nothing
  - **Choice two**: gives you 4 US dollars for sure

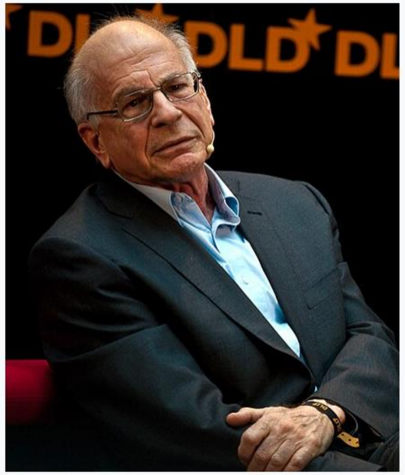
https://medium.com/@yianyao1994/llm-alignments-part-6-kto-813b38be14ae

Prospect Theory

Prospect theory explains why humans make decisions about uncertain events that do not maximize expected value. It formalizes how humans perceive random variables in a biased but well-defined manner;
for example, relative to some **reference point**, humans are more sensitive to losses than gains, a property called **loss aversion**.

2002 Nobel Prize-winning economists

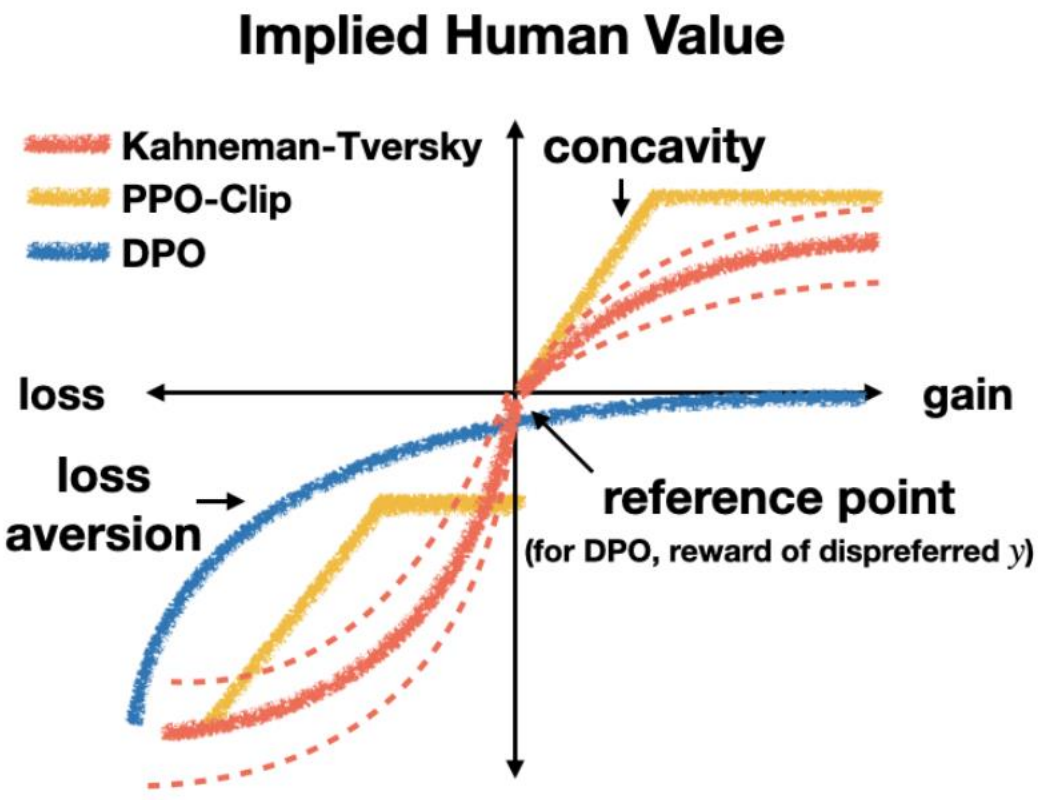
Daniel Kahneman



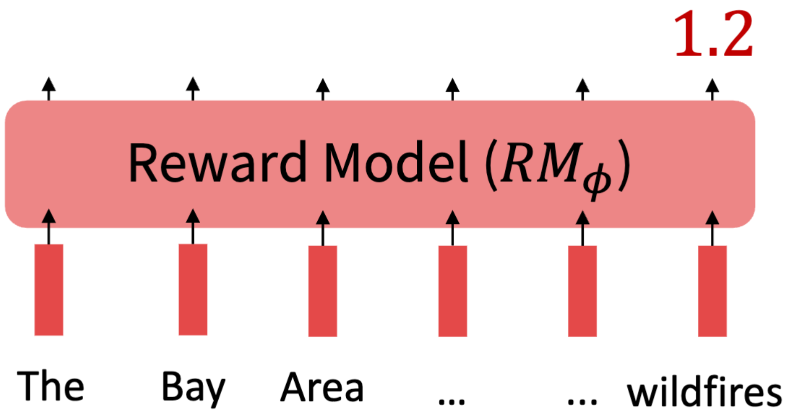
Amos Tversky

https://medium.com/@yianyao1994/llm-alignments-part-6-kto-813b38be14ae

KTO Value Function



Preference Data For PPO/DPO



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

s_1

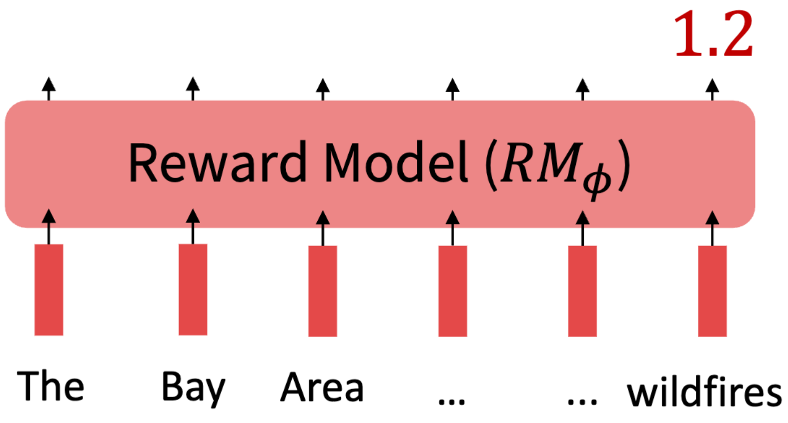
>

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

s_2

Training Data (x, y_1, y_2)

Preference Data For KTO




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

s_1

Acceptable?

Training Data (x, y)

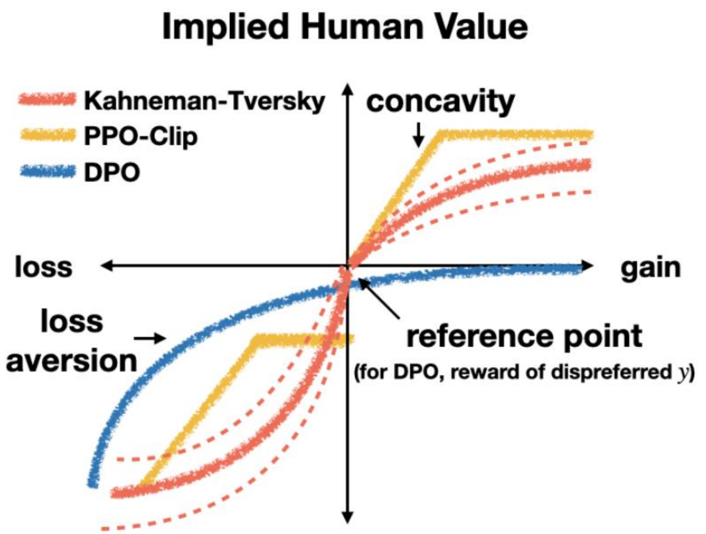
KTO: Loss Function

$$L_{\text{KTO}}(\pi_\theta, \pi_{\text{ref}}) = \mathbb{E}_{x,y\sim D}[\lambda_y - v(x,y)]$$

$$r_{\text{KTO}}(x,y) = \beta \log \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)}$$

$$v_{\text{KTO}}(x,y;\beta) = \begin{cases} \sigma(r_{\text{KTO}}(x,y) - z_{\text{ref}}) \text{ if } y \sim y_{\text{desirable}}|x \\ \sigma(z_{\text{ref}} - r_{\text{KTO}}(x,y)) \text{ if } y \sim y_{\text{undesirable}}|x \end{cases}$$

$$w(y) = \begin{cases} \lambda_D & \text{if } y \sim y_{\text{desirable}}|x \\ \lambda_U & \text{if } y \sim y_{\text{undesirable}}|x \end{cases}$$



KTO Performance