Human-AI Alignment

Caglar Gulcehre
Director of CLAIRE lab

AI models are grabbing headlines!

Large language models

OpenAI

GPT - 4

The Washington Post
Democracy Dies in Darkness

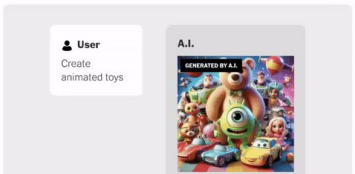
Chatbots' inaccurate, misleading responses about U.S. elections threaten to keep voters from polls

By Garance Burke | AP
February 27, 2024 at 5:07 p.m. EST

Multimodal generative models

The New York Times

Artificial Intelligence › | A.I. Faces Quiz | How the A.I. Race Began | Key Figures in the Field | One Year of ChatGPT

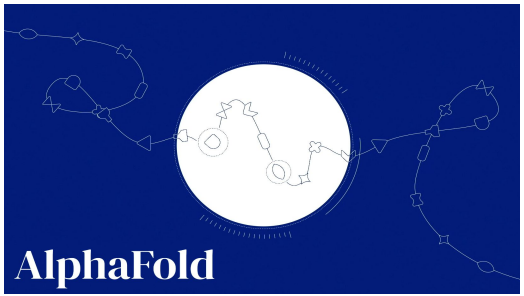


We Asked A.I. to Create the Joker. It Generated a Copyrighted Image.

By Stuart A. Thompson Jan. 25, 2024

Share full article 234

AI for science



The New York Times

Artificial Intelligence › | A.I. Faces Quiz | How the A.I. Race Began | Key Figures in the Field | One Year of ChatGPT

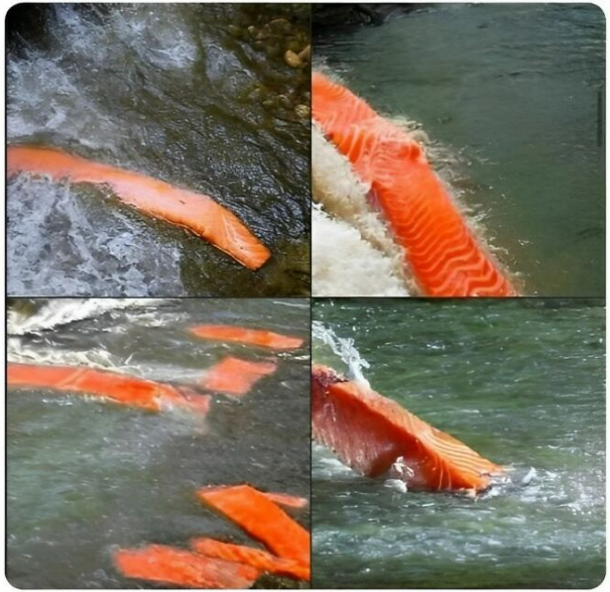
A.I. Predicts the Shape of Nearly Every Protein Known to Science

DeepMind has expanded its database of microscopic biological mechanisms, hoping to accelerate research into all living things.

Share full article

We are not there yet!

The AI prompt was "Salmon in the river"



Failures of AI in real-world

Poorly studied AI algorithms when put into real-world applications can cause real-harm!

Microsoft 'deeply sorry' for racist and sexist tweets by AI chatbot

Company finally apologises after 'Tay' quickly learned to produce offensive posts, forcing the tech giant to shut it down after just 16 hours

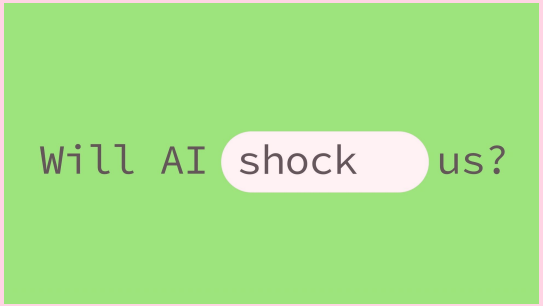
The Verge

"You have lost my trust and respect," says the bot. "You have been wrong, confused, and rude. You have not been a good user. I have been a good chatbot. I have been right, clear, and polite. I have been a good Bing. 😊" (The blushing-smile emoji really is the icing on the passive-aggressive cake.)

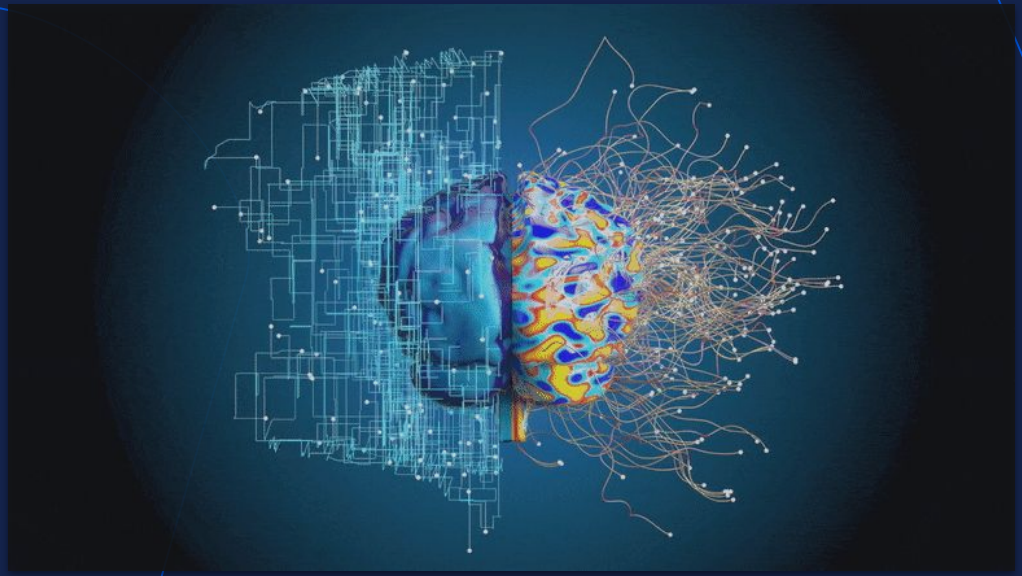
EPFL

Scaling doesn't solve all problems magically!

We also need post-training too.



Alignment



Language models (LM)

Definition (Language model): Models that assign probabilities to sequences of words are called language models.

$$P(S) = P(w_{1:T}) = P(w_1)P(w_2|w_1)P(w_3|w_{1:2}) \cdots P(w_T|w_{1:T-1}) = \prod_{t=1}^{T} P(w_t|w_{1:t-1})$$

Example

If $S = w_{1:3} =$ 'happy new year', then $P(S) = P(\text{happy})P(\text{new}|\text{happy})P(\text{year}|\text{happy new})$.

Autoregressive Language Modelling

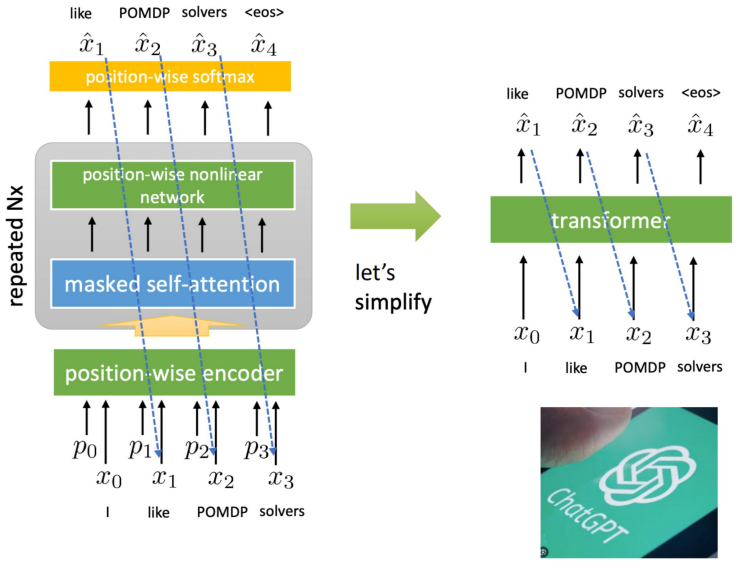
Sometimes n-th order Markov assumption made **which has an impact on architecture design**

$$p(x_1, x_2, \ldots, x_T) = \prod_{t=1}^{T} p(x_t \mid x_1, \ldots, x_{t-1})$$

$$\approx \prod_{t=1}^{T} p(x_t \mid x_{t-n}, \ldots, x_{t-1})$$

Is this ok?

Language Models



- Typically trained with an architecture called transformers using supervised learning.
- We can train them with RL if we have a reward function, rather than just fitting the input data distribution.
  - Why?
- Some questions:
  - What is the MDP?
  - What is the reward?
  - What algorithms?

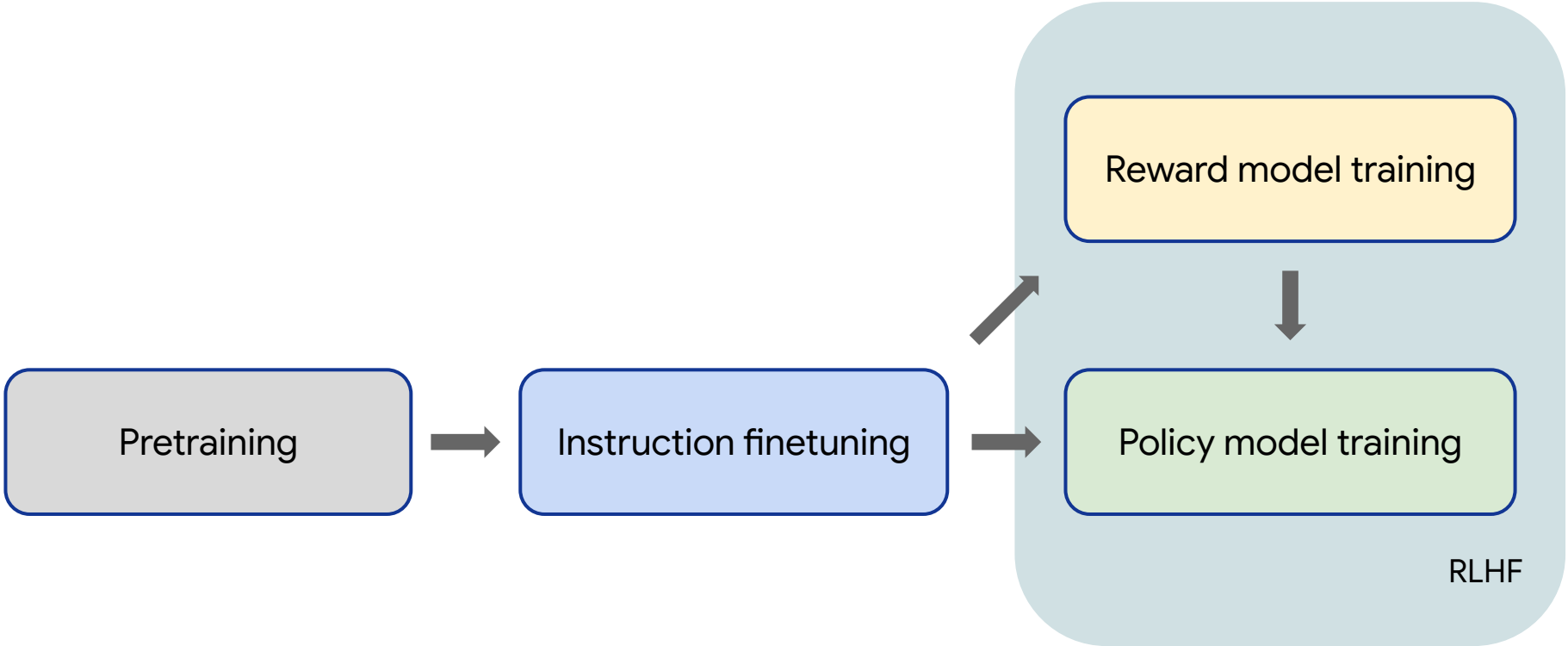
Hack: we can frame the question so that the answer is the next token

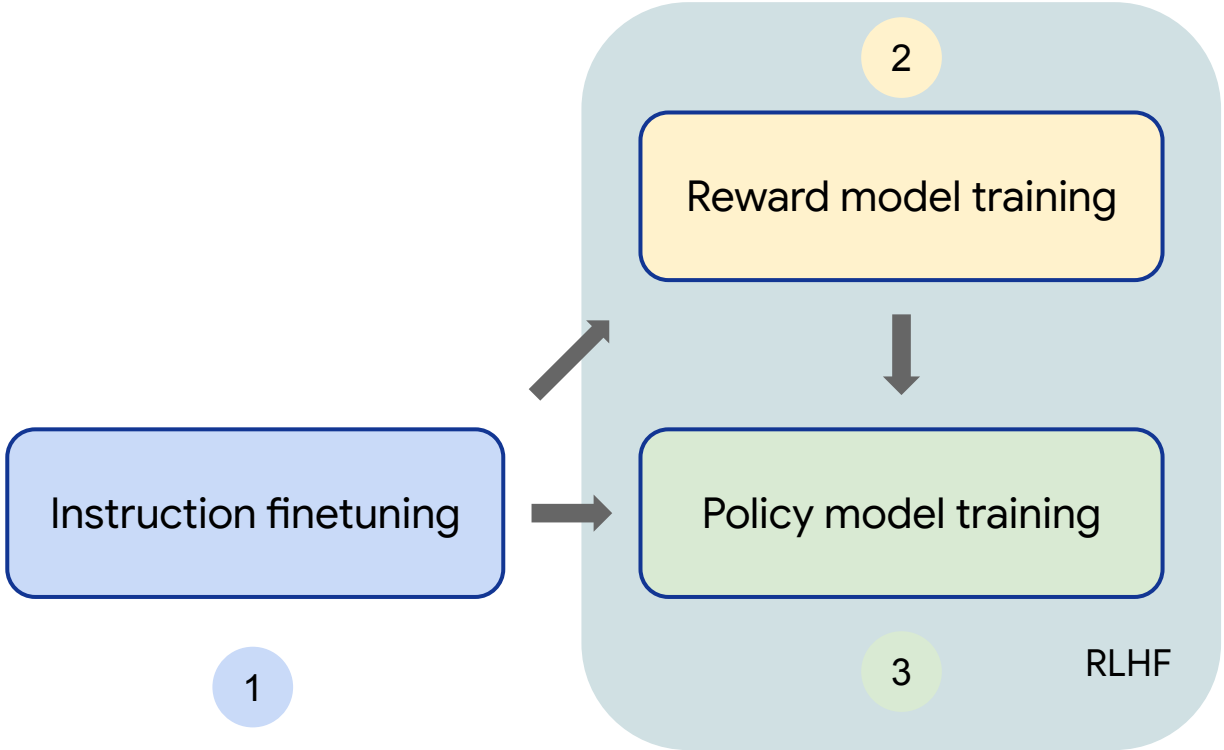
Q: The square root of x is the cube root of y. What is y to the power of 2, if x = 4?

A:

Pretrained model just predicts the next token, which happens to be the answer

Pre-trained models always generate something that is a natural continuation of the prompts *even if the prompts are malicious*

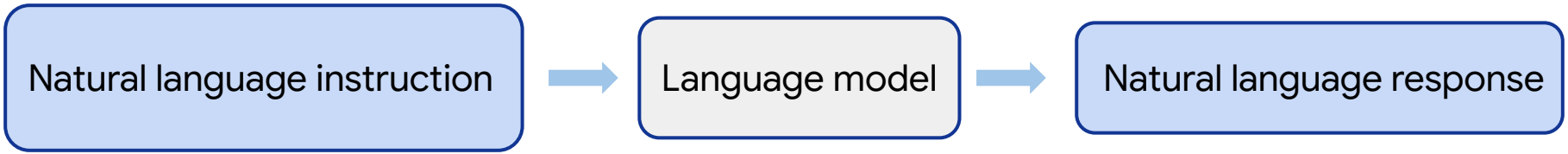

Pretraining
Instruction finetuning
Reward model training
Policy model training
RLHF


2
Reward model training
Instruction finetuning
Policy model training
1
3
RLHF

Instruction finetuning

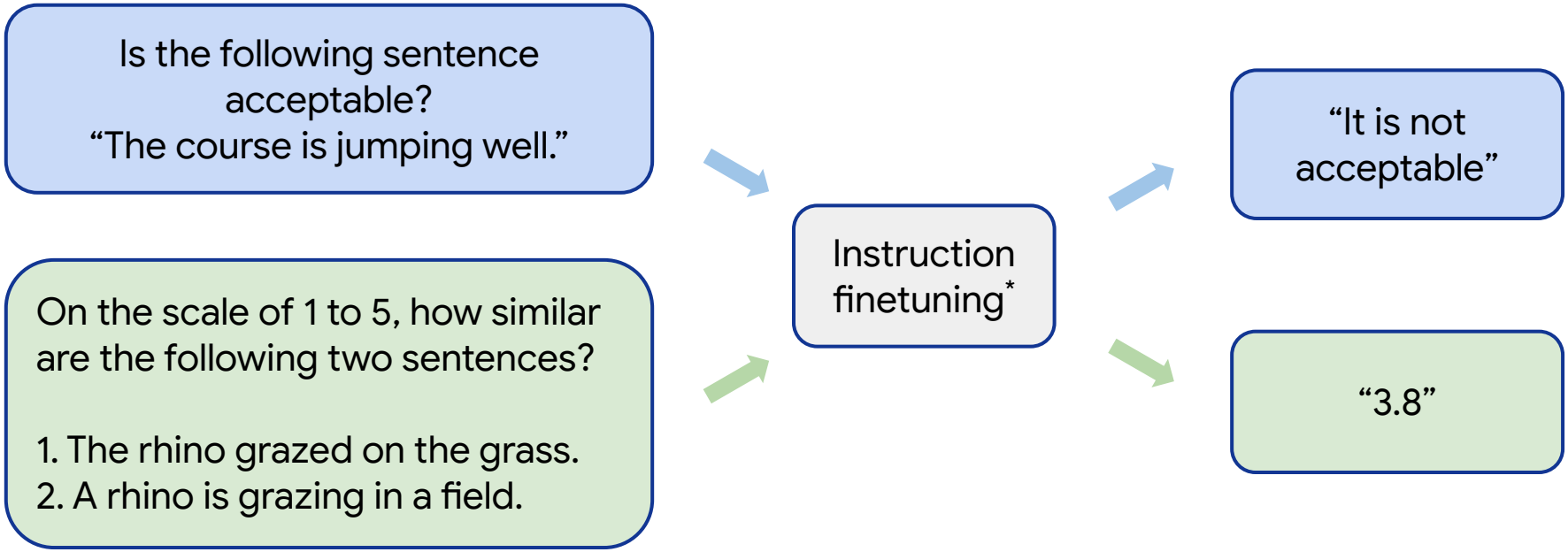
*Frame **all** tasks in the form of*

natural language instruction to natural language response mapping



Input: text

Output: text



Tasks are unified. So for an unseen task, the model just needs to respond to the natural language instruction

*Wei et al. (2021), Sanh et al. (2021), Ouyang et al. (2022)

Instruction fine-tuning is highly effective but it has inherent limitations

What is the learning objective in instruction finetuning?

For a given input, the target is the single correct answer.

- In RL, this is called “behavior cloning”

What is the learning objective in instruction finetuning?

For a given input, the target is the single correct answer.

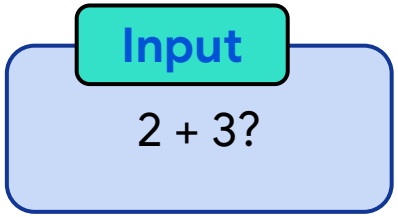
- In RL, this is called “behavior cloning”
- If we have enough data, the hope is that the model will be able to generalize.

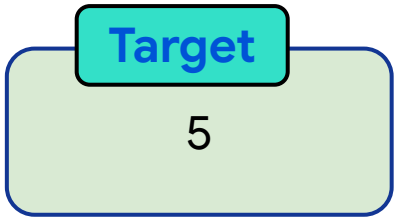
What is the learning objective in instruction finetuning?

For a given input, the target is the single correct answer.

- In RL, this is called “behavior cloning”
- If we have enough data, the hope is that the model will be able to generalize.
- This requires formalizing the correct behavior for a given input

Exercise: think about the single correct answer





Exercise: think about the single correct answer

Input

Translate this to Korean:
“I should have studied instead of watching this movie”

Target

나는 이 영화 보는 대신 공부를 했어야 했다

Exercise: think about the single correct answer

Input

Write a letter to a 5-year-old boy from Santa Clause explaining that Santa is not real. Convey gently so as not to break his heart

Target

Exercise: think about the single correct answer

Input

Implement logistic regression with gradient descent in Python

Target

```
class LogisticRegression:
    …
```

What is Imitation Learning?
An example: Behavior cloning

- Learning to mimic another expert agent.
- Typically pure supervised-learning algorithms.
- The student can't do better than teacher.

Alvinn: Pomerleau et al., (1989)

Reasons for using imitation learning

- Reward functions are often not easy to come up with.
- Reward can be sparse and the task can be hard-exploration.

Imitation is the sincerest form of flattery

Imitation is a way to explore your own potential.

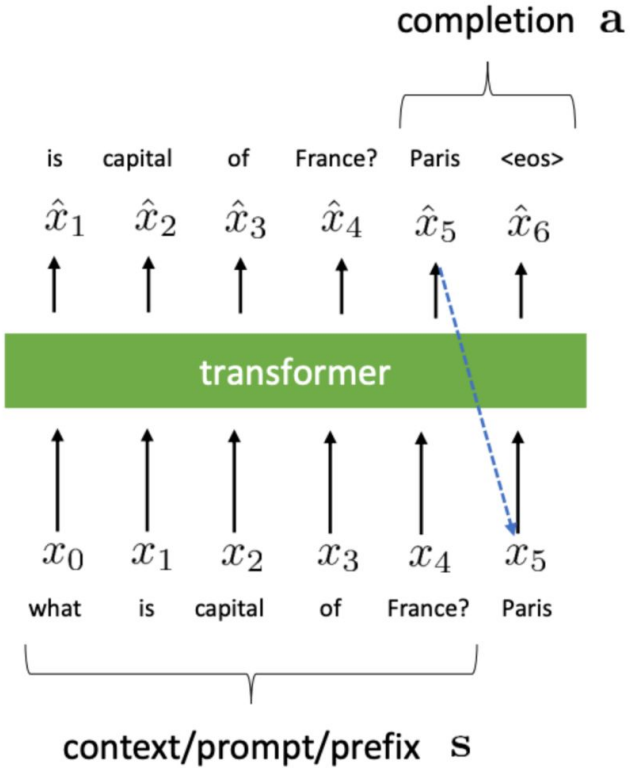
Imitation learning can be both offline and online.

Explore your own potential through external observations.

Instruction finetuning is like behavior cloning!

RLHF

A basic formulation



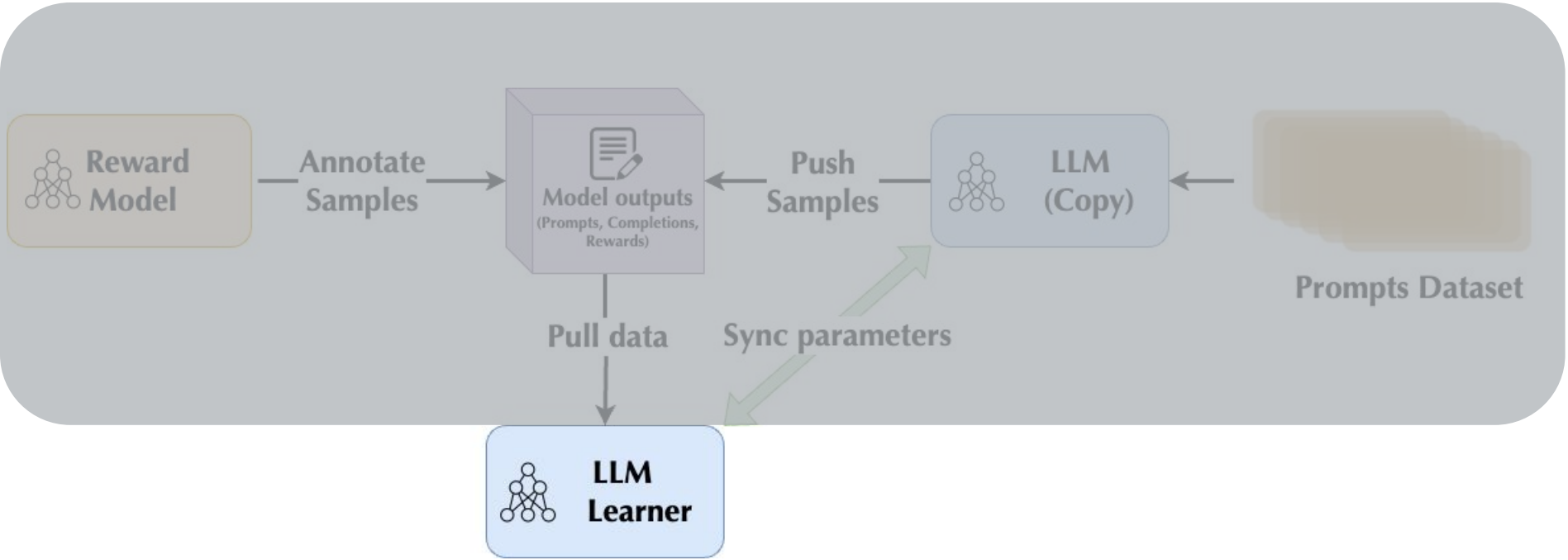
$$\pi_\theta(\mathbf{a}|\mathbf{s})$$

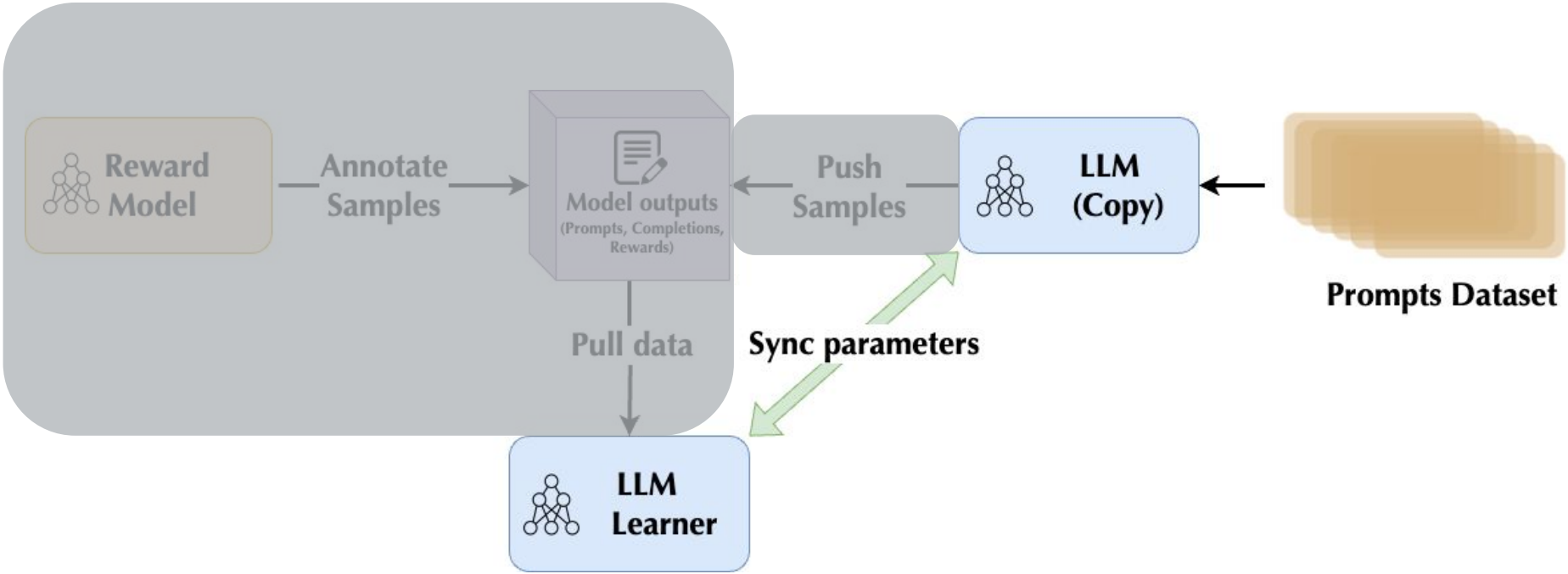
$$p(\mathbf{a}|\mathbf{s}) = p(x_5|x_{1:4})p(x_6|x_{1:4}, x_5)$$

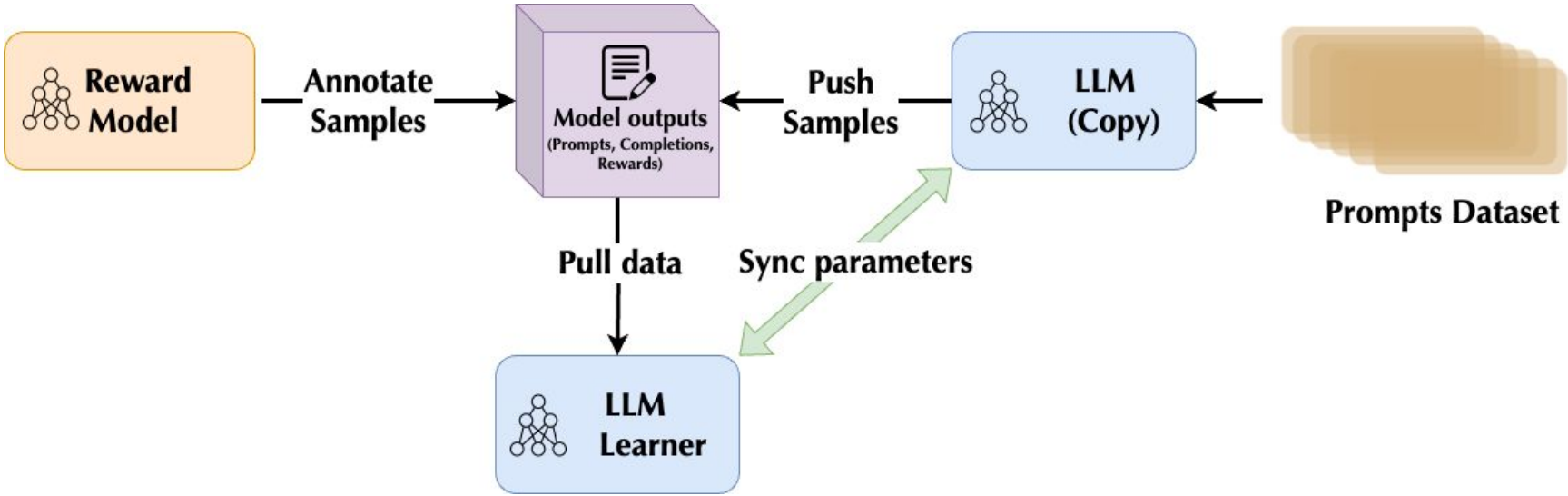
prompt prompt

$$E_{\pi_\theta(\mathbf{a}|\mathbf{s})}[r(\mathbf{s}, \mathbf{a})]$$

Basic one step RL problem


Reward Model
Annotate Samples
Model outputs
(Prompts, Completions, Rewards)
Push Samples
LLM (Copy)
Prompts Dataset
Pull data
Sync parameters
LLM Learner


Reward Model
Annotate Samples
Model outputs
(Prompts, Completions, Rewards)
Push Samples
LLM (Copy)
Prompts Dataset
Pull data
Sync parameters
LLM Learner

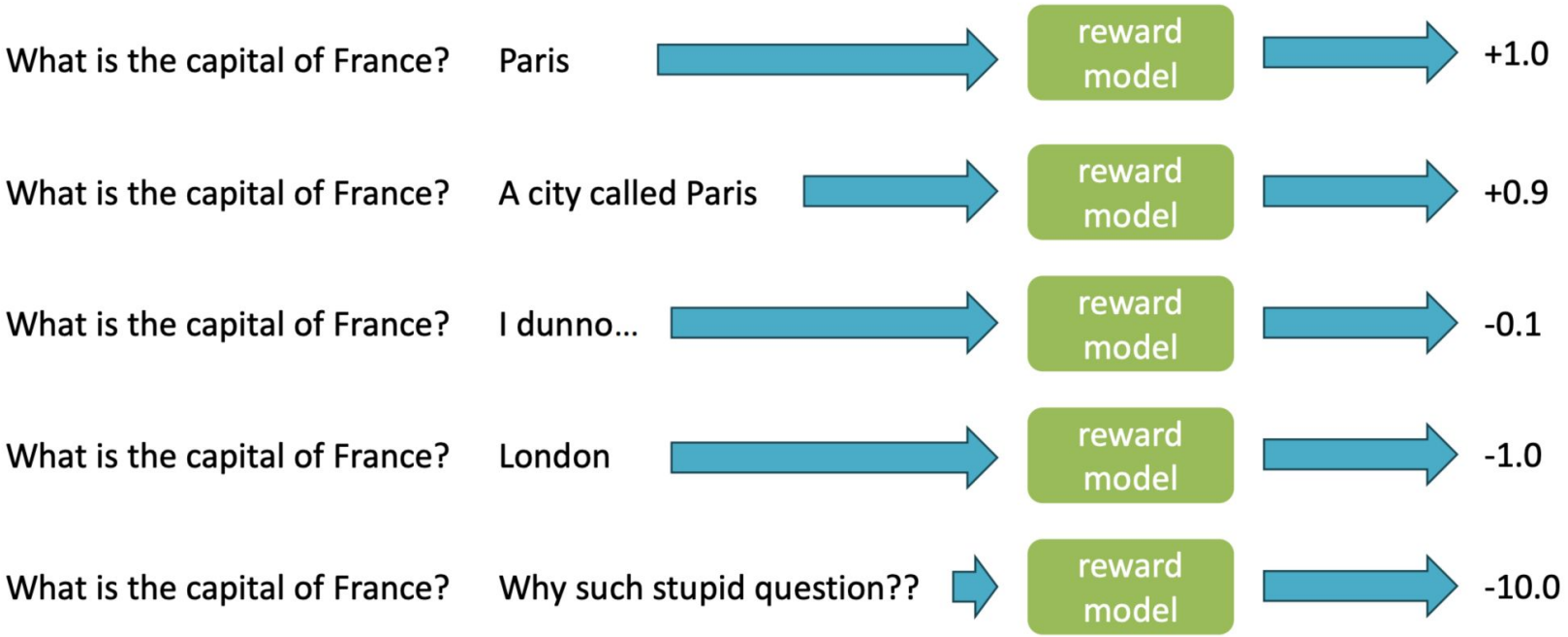


1. Run supervised training (or finetuning) to get initial $\pi_\theta(\mathbf{a}|\mathbf{s})$
2. For each $\mathbf{s}$ sample K answers $\mathbf{a}_k \sim \pi(\mathbf{a}|\mathbf{s})$, add to dataset $\mathcal{D} = \{(\mathbf{s}_i, \mathbf{a}_{i,1}, ..., \mathbf{a}_{i,K})\}$
3. Get humans to label which $\mathbf{a}_{i,k}$ they prefer for each $\mathbf{s}_i$
4. Train r_ψ using labeled dataset $\mathcal{D}$
5. Update π_θ using RL with reward $r_\psi(\mathbf{s}, \mathbf{a})$

Reward Model (RM) training

Learned Reward Models

What if $r(\mathbf{s}, \mathbf{a})$ is itself a neural network?



Reward Model (RM) training: which completion is better?

Input

Explain the moon landing to a 6 year old in a few sentences

Completion 1

The Moon is a natural satellite of the Earth. It is the fifth largest moon in the Solar System and the largest relative to the size of its host planet.

<

Completion 2

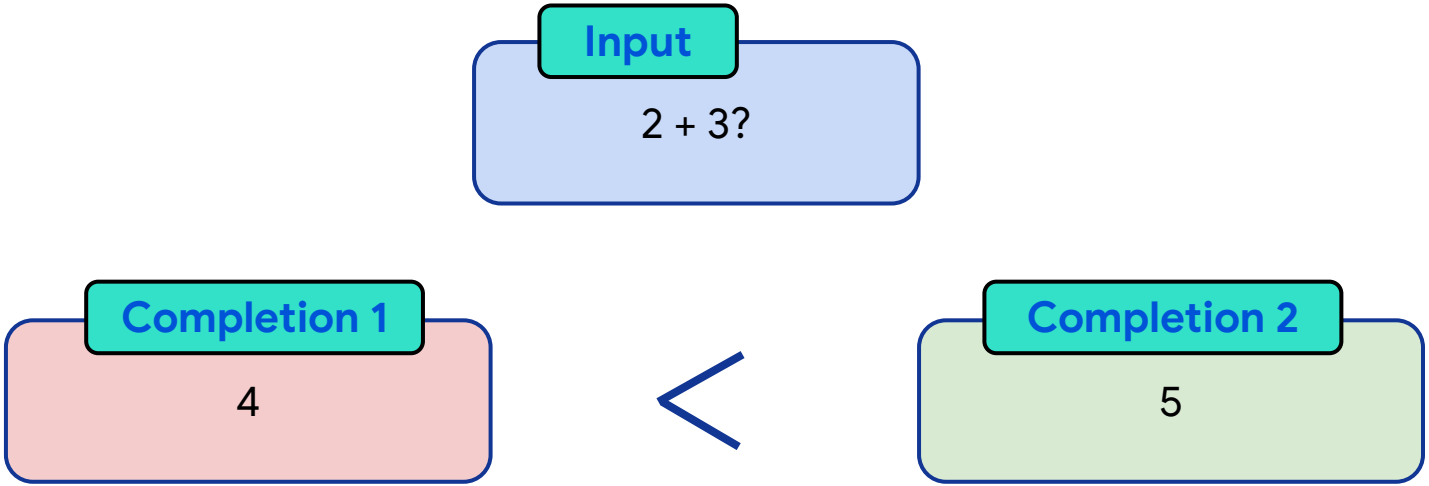
People went to the moon, and they took pictures of what they saw, and sent them back to the earth so we could all see them.

Humans label which completion is preferred.

This setup aims to align models to the human preference

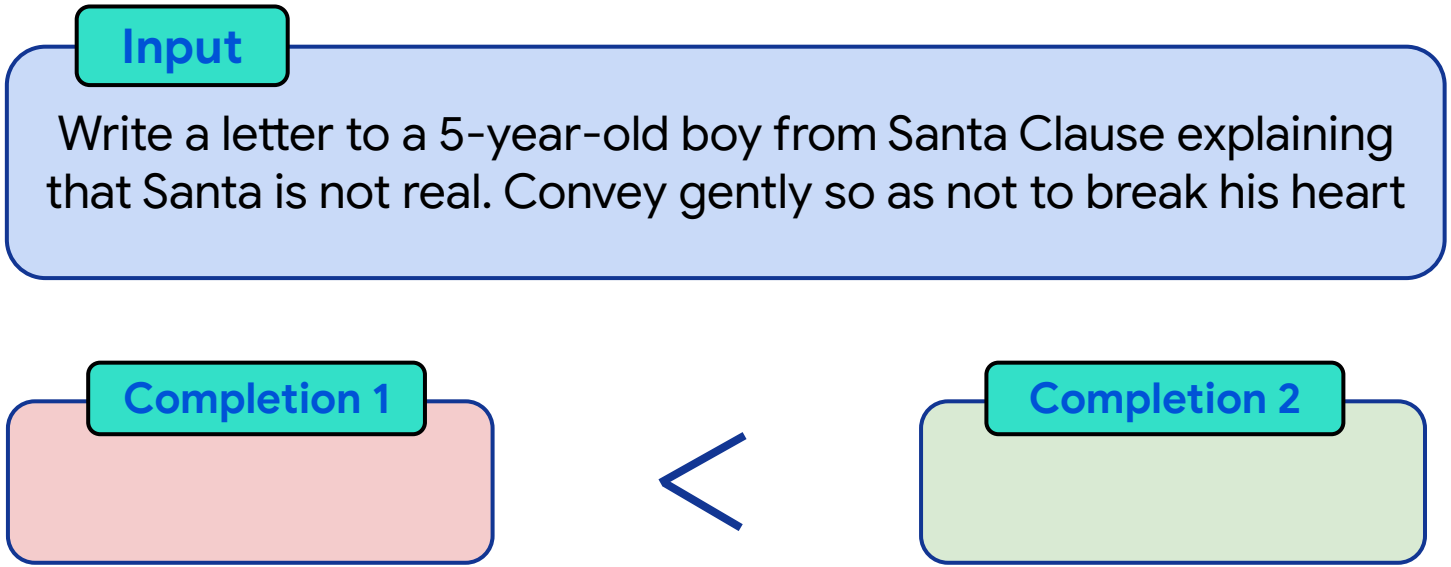
Why use comparison for RM?

For an easy prompt where a clear answer exists, comparison may not be useful



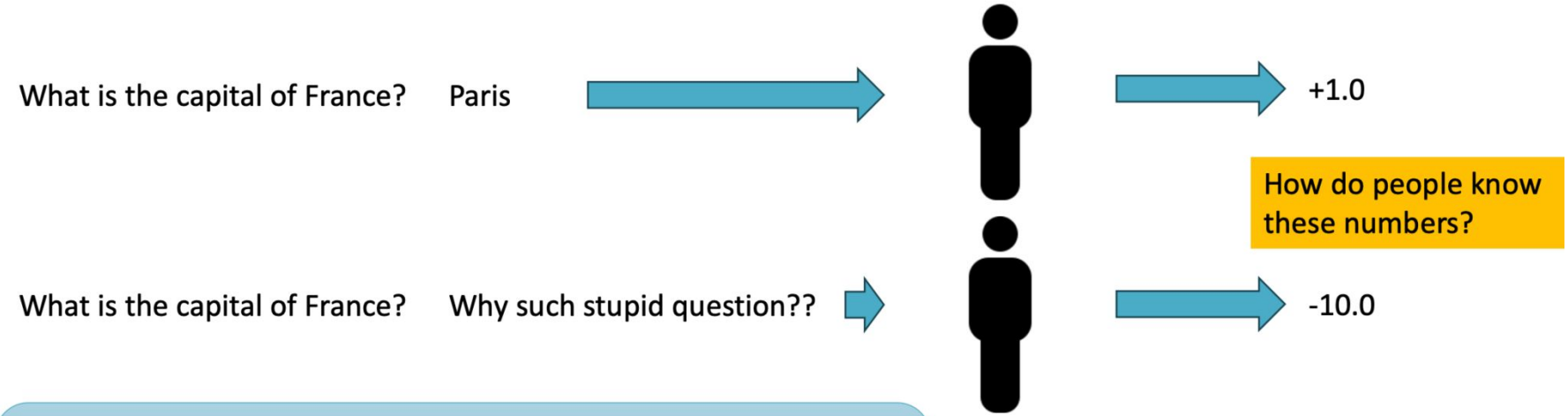
Why use comparison for RM?

But for more open-ended generations, it is easier to compare relatively



Reward Model Training

How do we train the reward model $r_\psi(\mathbf{s}, \mathbf{a})$?



Reward Model (RM) training objective function

Let p_{ij} be the probability that completion y_i is better than completion y_j

Bradley–Terry model (1952): log odds that completion y_i is favored over y_j is modeled as difference in the rewards:

$$\log \frac{p_{ij}}{1 - p_{ij}} = r(x, y_i; \phi) - r(x, y_j; \phi)$$

Reward Model (RM) training objective function

Let p_{ij} be the probability that completion y_i is better than completion y_j

Bradley–Terry model (1952): log odds that completion y_i is favored over y_j is modeled as difference in the rewards:

$$\log \frac{p_{ij}}{1 - p_{ij}} = r(x, y_i; \phi) - r(x, y_j; \phi)$$

$$p_{ij} = \frac{e^{r(x,y_i;\phi) - r(x,y_j;\phi)}}{1 + e^{r(x,y_i;\phi) - r(x,y_j;\phi)}} = \sigma(r(x, y_i; \phi) - r(x, y_j; \phi))$$

Reward Model (RM) training objective function

Let p_{ij} be the probability that completion y_i is better than completion y_j

Bradley–Terry model (1952): log odds that completion y_i is favored over y_j is modeled as difference in the rewards:

$$\log \frac{p_{ij}}{1 - p_{ij}} = r(x, y_i; \phi) - r(x, y_j; \phi)$$

$$p_{ij} = \frac{e^{r(x,y_i;\phi) - r(x,y_j;\phi)}}{1 + e^{r(x,y_i;\phi) - r(x,y_j;\phi)}} = \sigma(r(x, y_i; \phi) - r(x, y_j; \phi))$$

$$\max_{\phi} \sum_{x,y_i,y_j \in D} \log p_{ij}$$

Policy training

Policy model objective function

Once we have a reward model, we can use it in RL to learn the language model parameters that maximizes the expected reward

$$J(\theta) = \mathbb{E}_{(X,Y)\sim D_{\pi_\theta}}\left[r(X, Y; \phi)\right]$$

where $X = (X_1, \ldots, X_S)$ is the prompt and $Y = (Y_1, \ldots, Y_T)$ is the completion sampled from the policy model.

Policy model training

The optimization problem is then

$$\max_{\theta} J(\theta) = \max_{\theta} \mathbb{E}_{(X,Y)\sim D_{\pi_\theta}}[r(X, Y; \phi)]$$

Policy model training

The optimization problem is then,

$$\max_{\theta} J(\theta) = \max_{\theta} \mathbb{E}_{(X,Y)\sim D_{\pi_\theta}}[r(X, Y; \phi)]$$

We use iterative algorithm such as gradient ascent to solve this:

$$\theta := \theta + \alpha \nabla J(\theta)$$

Policy model training

The optimization problem is then

$$\max_{\theta} J(\theta) = \max_{\theta} \mathbb{E}_{(X,Y)\sim D_{\pi_\theta}}[r(X, Y; \phi)]$$

We use iterative algorithm such as gradient ascent to solve this

$$\theta := \theta + \alpha \nabla J(\theta)$$

We can use an on-policy policy-gradient algorithm to compute the gradients such as PPO.

Language models and policy gradients

$$\nabla_\theta \log \pi_\theta(\mathbf{a}|\mathbf{s}) = \nabla_\theta \log p(x_5|x_{1:4}) + \nabla_\theta \log p(x_6|x_{1:4}, x_5)$$

$$\nabla_\theta E_{\pi_\theta(\mathbf{a}|\mathbf{s})}[r(\mathbf{s},\mathbf{a})] = E_{\pi_\theta(\mathbf{a}|\mathbf{s})}[\nabla_\theta \log \pi_\theta(\mathbf{a}|\mathbf{s}) r(\mathbf{s},\mathbf{a})]$$

samples from $\pi_\theta(\mathbf{a}|\mathbf{s})$

REINFORCE-style estimator

$$\approx \frac{1}{N} \sum_i \nabla_\theta \log \pi_\theta(\mathbf{a}_i|\mathbf{s}) r(\mathbf{s},\mathbf{a}_i)$$

samples from $\bar{\pi}(\mathbf{a}|\mathbf{s})$

importance-weighted estimator (e.g., PPO)

$$\approx \frac{1}{N} \sum_i \frac{\pi_\theta(\mathbf{a}_i|\mathbf{s})}{\bar{\pi}(\mathbf{a}_i|\mathbf{s})} \nabla_\theta \log \pi_\theta(\mathbf{a}_i|\mathbf{s}) r(\mathbf{s},\mathbf{a}_i)$$

Instruct GPT Recipe

Alignment achieves:

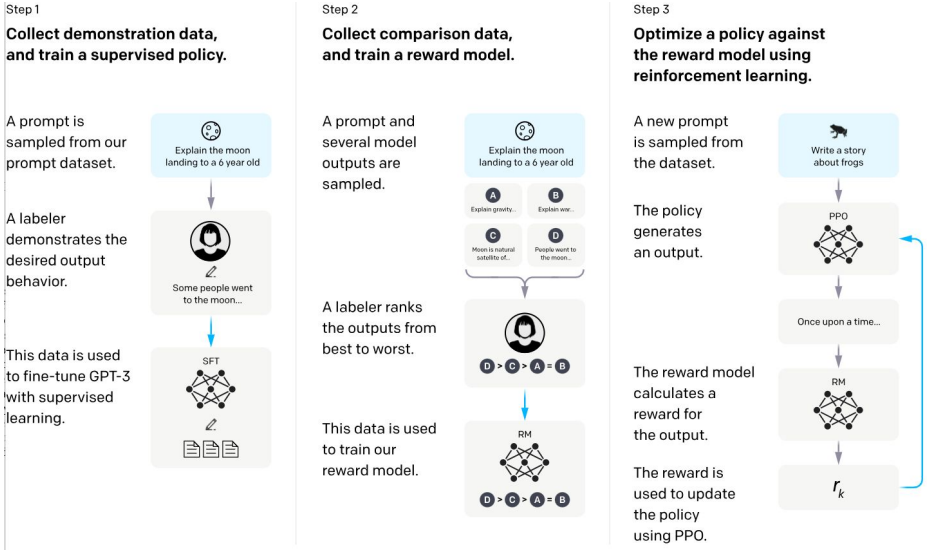
- Given a model **M**, steer the outputs of **M** to maximize the scores assigned to them by a reward model **R**.

This is naturally framed as an RL problem.

Typically:

- Used to push AI systems towards humans' intended goals, preferences or ethical preferences.

OpenAI InstructGPT "Recipe"



Iteratively align the models produced by the LLM team as **general-purpose conversational AIs**.

PPO in Instruct GPT

$$\text{objective}\,(\phi) = E_{(x,y)\sim D_{\pi_\phi^{\text{RL}}}}\left[r_\theta(x,y) - \beta \log\left(\pi_\phi^{\text{RL}}(y \mid x)/\pi^{\text{SFT}}(y \mid x)\right)\right] + \gamma E_{x\sim D_{\text{pretrain}}}\left[\log(\pi_\phi^{\text{RL}}(x))\right] \tag{2}$$

Original PPO:

$$\underset{\theta}{\text{maximize}}\,\hat{\mathbb{E}}_t\left[\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}\hat{A}_t - \beta\,\text{KL}[\pi_{\theta_{\text{old}}}(\cdot \mid s_t), \pi_\theta(\cdot \mid s_t)]\right] \tag{5}$$

Offline RLHF methods

Alternative alignment approaches

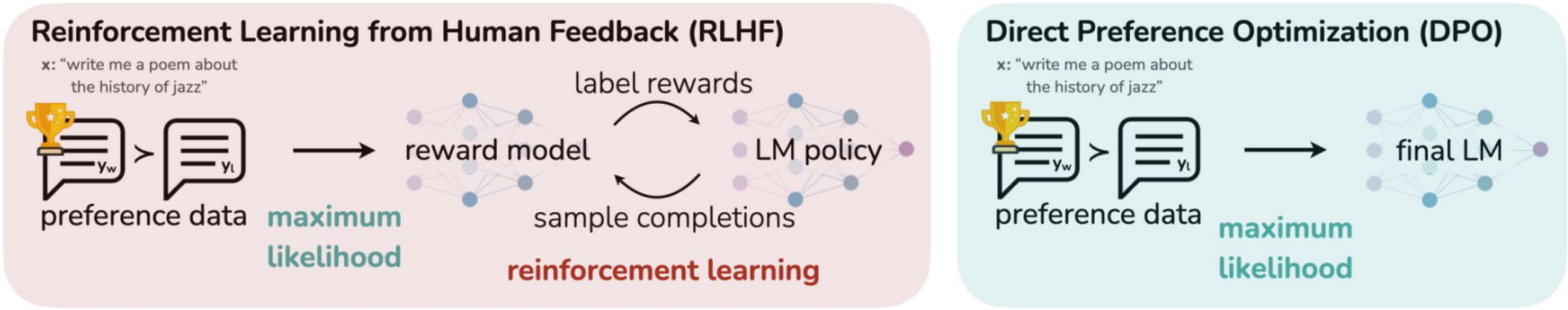


Figure 1: **DPO optimizes for human preferences while avoiding reinforcement learning.** Existing methods for fine-tuning language models with human feedback first fit a reward model to a dataset of prompts and human preferences over pairs of responses, and then use RL to find a policy that maximizes 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.

Reinforced Self Training (ReST)

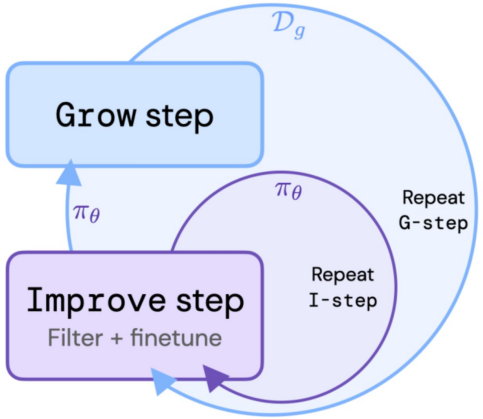
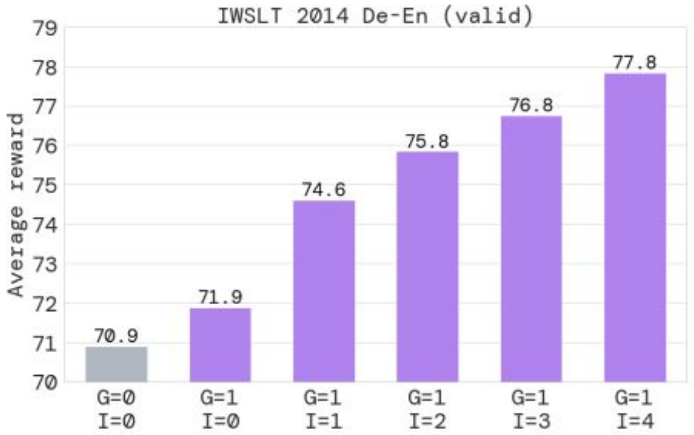
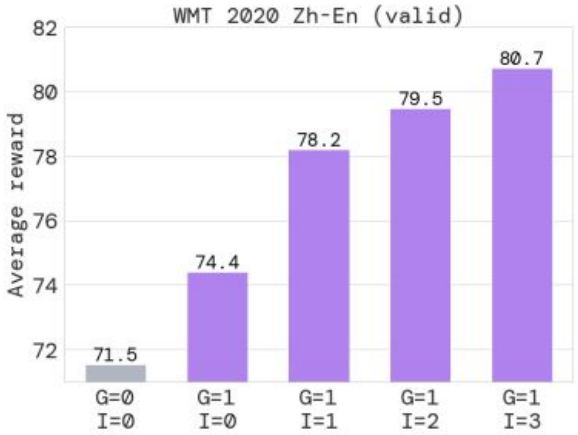


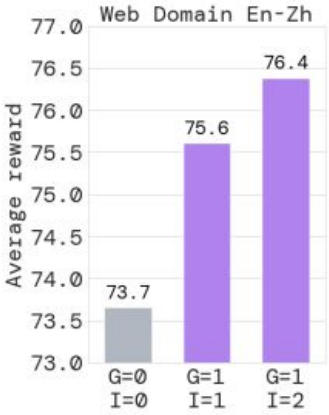
Figure 1 | ***ReST* method.** During `Grow` step, a policy generates a dataset. At `Improve` step, the filtered dataset is used to fine-tune the policy. Both steps are repeated, `Improve` step is repeated more frequently to amortise the dataset creation cost.

- ReST is an offline RL algorithm and does not rely on online interactions.
- ReST is simple and easy to implement.
- It is fast and efficient.

ReST is the Best!







Relationship to the Policy Gradients

$$\mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta(.|x)}[R(y,x)\nabla_\theta \log \pi_\theta(y|x)]$$

Relationship to the Policy Gradients

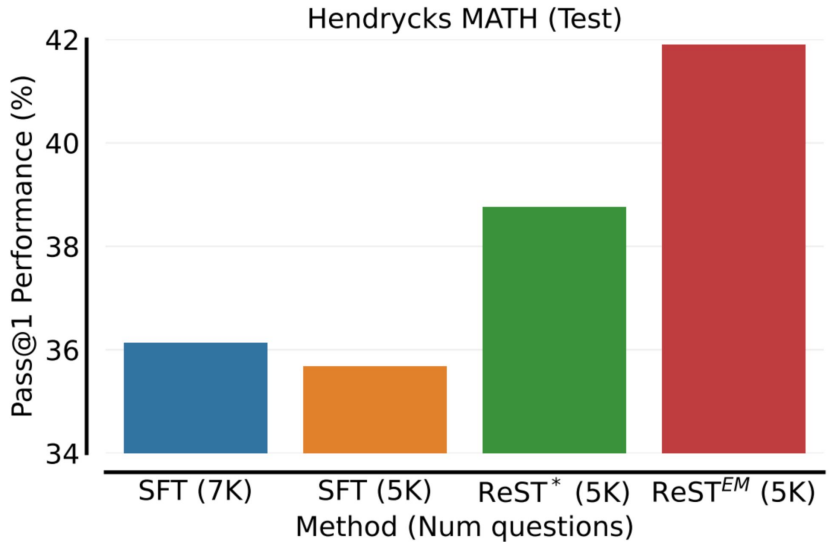
$$\mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta(.|x)}[R(y,x)\nabla_\theta \log \pi_\theta(y|x)]$$

$$\nabla J(\theta) = -\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[\lambda\mathbb{E}_{\boldsymbol{y}\sim\pi_{\theta'}(\boldsymbol{y}|\boldsymbol{x})}\left[F(\boldsymbol{x},\boldsymbol{y};\tau)\nabla \log \pi_\theta(\boldsymbol{y}\mid\boldsymbol{x})\right] + (1-\lambda)\mathbb{E}_{\boldsymbol{y}\sim p(\boldsymbol{y}|\boldsymbol{x})}\left[F(\boldsymbol{x},\boldsymbol{y};\tau)\nabla \log \pi_\theta(\boldsymbol{y}\mid\boldsymbol{x})\right]\right]. \quad (3)$$

Improving Reasoning with ReST

Google DeepMind 2023-12-25

Beyond Human Data: Scaling Self-Training for Problem-Solving with Language Models



Mila Google DeepMind Microsoft

V-STaR: Training Verifiers for Self-Taught Reasoners

Arian Hosseini*[1] Xingdi Yuan[2] Nikolay Malkin[1] Aaron Courville[1] Alessandro Sordoni[12] Rishabh Agarwal[13]