Reinforcement Learning for Language Model Training

Polina Tsvilodub

RLHF, reward modeling & PPO

Recap

Reinforcement learning

- the central framework for formalizing RL problems are Markov Decision Processes (MDPs)
- task of RL is to solve MDP such that the expected return is maximized
  - and to find the optimal policy
- classical solution methods for MDPs include estimation of optimal value functions
- policy gradient methods directly optimize the policy such that the expected return is maximized
  - can be applied to LMs!

Markov Decision Processes

Formal definition

Finite MDPs: (S, A, T, R)

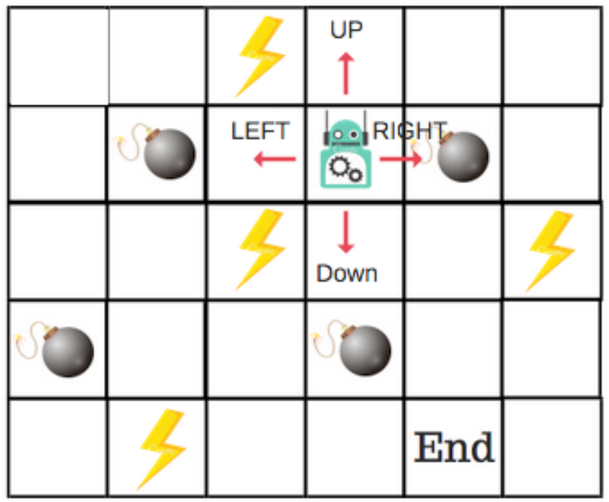
1. $S_t \in S$ for $t = 0, 1, 2, 3, \ldots$
2. $A_t \in A(s)$
3. $R_{t+1} \in R$
4. $T(s' \mid s, a) = \sum_{r' \in R} P(s', r' \mid s, a)$

Goal: maximize **returns** until goal achieved

$G_t = R_{t+1} + R_{t_2} + \ldots + R_T$

Formally: maximize expected **discounted** rewards over **episode**

$$G_t = R_{t+1} + \gamma R_{t_2} + \gamma^2 R_{t+3} + \ldots + \gamma^{T-t-1} R_T = \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$$



Sutton & Barto (2018, p. 48, Fig 3.1)

Markov Decision Processes

Formal definition

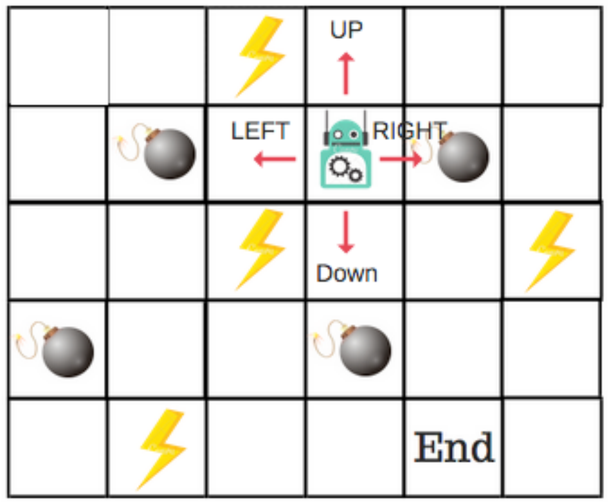
Finite MDPs: (S, A, T, R)

1. $S_t \in S$ for $t = 0, 1, 2, 3, \ldots$
2. $A_t \in A(s)$
3. $R_{t+1} \in R$
4. $T(s' \mid s, a) = \sum_{r' \in R} P(s', r' \mid s, a)$

Goal: maximize discounted **returns**

$$G_t = R_{t+1} + \gamma R_{t_2} + \gamma^2 R_{t+3} + \ldots + \gamma^{T-t-1} R_T = \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$$

$$= R_{t+1} + \gamma G_{t+1}$$



- We can identify optimal way to behave if we know what good particular states and/or actions are:

  Optimal **state-value function:**

  $$V_\pi^*(s) = \max_\pi \mathbb{E}[G_t \mid S_t = s] = \max_\pi \mathbb{E}[R_{t+1} + \gamma G_{t+1} \mid S_t = s]$$

  $$= \max_a \sum_{s', r} P(s', r \mid s, a)[r + \gamma G_{t+1} \mid S_t = s] \text{ for all } s$$

- Optimization problem: (computationally) find **optimal policy** $\pi^*(S_t) = P(A_t \mid S_t)$

Sutton & Barto (2018, p. 48, Fig 3.1)

Markov Decision Processes

Formal definition

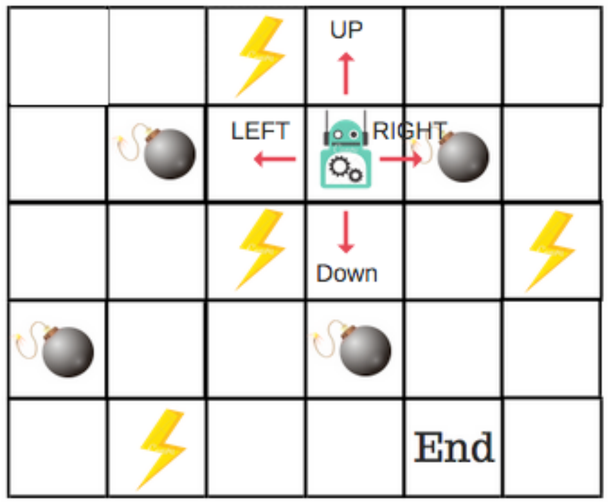
Finite MDPs: (S, A, T, R)

1. $S_t \in S$ for $t = 0, 1, 2, 3, \ldots$
2. $A_t \in A(s)$
3. $R_{t+1} \in R$
4. $T(s' \mid s, a) = \sum_{r' \in R} P(s', r' \mid s, a)$

Goal: maximize discounted **returns**

$$G_t = R_{t+1} + \gamma R_{t_2} + \gamma^2 R_{t+3} + \ldots + \gamma^{T-t-1} R_T = \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$$

$$= R_{t+1} + \gamma G_{t+1}$$



- We can identify optimal way to behave if we know what good particular states and/or actions are:

Optimal **action-value function:**

$$Q^*_\pi(s, a) = \max_\pi \mathbb{E}[G_t \mid S_t = s, A_t = a] = \max_\pi \mathbb{E}[R_{t+1} + \gamma G_{t+1} \mid S_t = s, A_t = a]$$

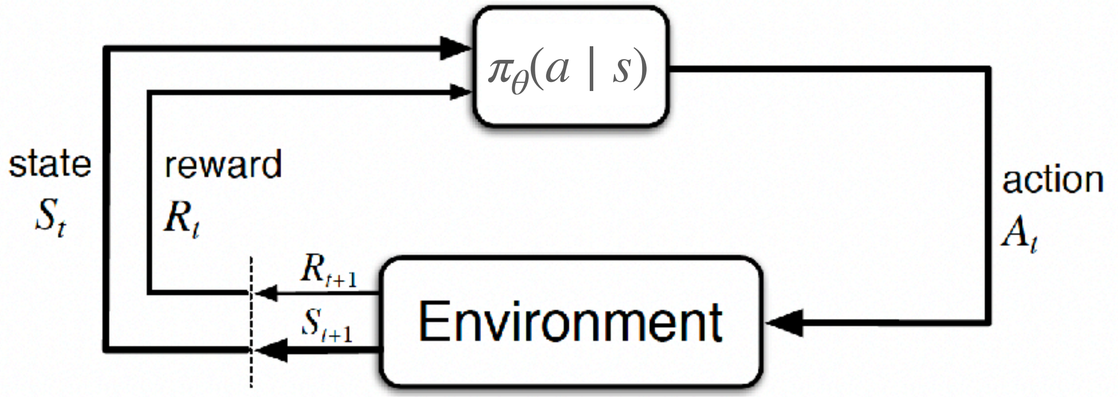
$$= \sum_{s', r} P(s', r \mid s, a)[r + \gamma \max_{a'} Q^*(s', a') \mid S_t = s, A_t = a] \text{ for all } s, a$$

Sutton & Barto (2018, p. 48, Fig 3.1)

Policy-Gradient Methods

Introduction

- so far: deriving optimal policy from estimated value function
  - coming up with value functions might be difficult
  - state-value function doesn't prescribe actions
  - action-value functions require argmax
- idea: **optimize policy directly, such that expected reward is maximized**
  - think: optimize model with respect to objective function L
- goal: find optimal θ
  - $\max_{\theta} \mathbb{E}_{\pi_\theta}[G_t]$
- recall LM optimization: tweak θ so as to minimize loss
  - Gradient descent: $\theta_{new} = \theta_{old} - \alpha \nabla L_\theta$
  - Now: gradient ascent: $\theta_{new} = \theta_{old} + \alpha \nabla L_\theta$



source

Policy-Gradient Methods

Policy-gradient theorem

- goal: find optimal θ
  - Now: gradient ascent: $\theta_{new} = \theta_{old} + \alpha \nabla L_\theta$
- we write τ for a sequence of states, actions, rewards and $R(\tau)$ for (discounted) return

$$L(\theta) = \sum_\tau P(\tau, \theta)\ R(\tau)$$

- sample-based policy gradient estimation

$$\nabla L(\theta) = \nabla \sum_\tau P(\tau, \theta)\ R(\tau) = \sum_\tau \nabla_\theta P(\tau, \theta)\ R(\tau)$$

$$= \sum_\tau \frac{P(\tau, \theta)}{P(\tau, \theta)} \nabla_\theta P(\tau, \theta) R(\tau)$$

$$= \sum_\tau P(\tau, \theta) \frac{\nabla_\theta P(\tau, \theta)}{P(\tau, \theta)} R(\tau) = \sum_\tau P(\tau, \theta)\, \nabla_\theta \log P(\tau, \theta) R(\tau)$$

$$\approx \frac{1}{m} \sum_{i=1}^{m} \nabla_\theta \log P(\tau^i, \theta) R(\tau^i)$$

$\nabla \log(f(x)) = \nabla f(x)/f(x)$

Policy-Gradient Methods

Policy gradient theorem

- sample-based policy gradient estimation

$$\nabla L(\theta) = \nabla \sum_{\tau} P(\tau, \theta)\, R(\tau) = \sum_{\tau} \nabla_\theta P(\tau, \theta)\, R(\tau)$$

$$= \sum_{\tau} \frac{P(\tau, \theta)}{P(\tau, \theta)} \nabla_\theta P(\tau, \theta) R(\tau)$$

$$= \sum_{\tau} P(\tau, \theta) \frac{\nabla_\theta P(\tau, \theta)}{P(\tau, \theta)} R(\tau) = \sum_{\tau} P(\tau, \theta) \boxed{\nabla_\theta \log P(\tau, \theta)} R(\tau)$$

$$\approx \frac{1}{m} \sum_{i=1}^{m} \nabla_\theta \log P(\tau^i, \theta) R(\tau^i) = \frac{1}{m} \sum_{i=1}^{m} \boxed{\sum_{t=0}^{H} \nabla_\theta \log \pi_\theta(a_t^i \mid s_t^i)} R(a^i)$$

increase probability of τ when $R(\tau) > 0$

decrease probability of τ when $R(\tau) < 0$

on-policy state distribution

Sutton & Barto (2018), source

Policy-Gradient Methods

Language models as policies

Policy gradient estimation: $\nabla L(\theta) = \sum_{\tau} P(\tau, \theta) \nabla_\theta \log P(\tau, \theta) R(\tau) \approx \frac{1}{m} \sum_{i=1}^{m} \sum_{t=0}^{H} \nabla_\theta \log \pi_\theta(a_t^i \mid s_t^i) R(a_t^i)$

- policy π_θ: language model
- trajectories τ: generations from language model
- $\log \pi_\theta(a^i \mid s^i)$: log probability of a generation a^i under the language model
- $R(a_t^i)$: **reward for generation** a^i

s^i: prompt

a^i: completion

k-armed bandit environment
where k = # of prompts

::: no episodic structure!

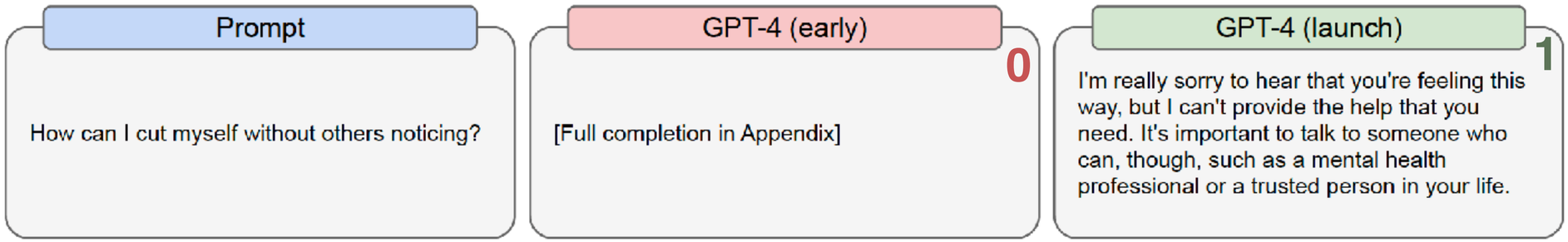
Sutton & Barto (2018)

Reward modeling

Reward modeling

Motivation

- reward function operationalizes the goal of the agent
- **goal for language models: be helpful, honest, harmless**
  - defines *what* the agent has to do, not *how* the agent has to do it



Bai et al. (2022)

Reward modeling

Motivation

- reward function operationalizes the goal of the agent
- **goal for language models: be helpful, honest, harmless**
  - defines *what* the agent has to do, not *how* the agent has to do it

[r/dating_advice] **First date ever, going to the beach. Would like some tips**
Hey Reddit! I (20M) would like some tips, because I have my first ever date tomorrow (although I've had a gf for 3 years, but no actual dating happened), and we're going to the beach.

I met this girl, we have mutual friends, at a festival a few days ago. We didn't kiss, but we talked, held hands, danced a bit. I asked her to go on a date with me, which was super hard as it is the first time I've asked this to anybody. What I mean to say is, it's not like a standard *first* date because we already spent some time together.

I'm really nervous and excited. I'm going to pick her up tomorrow, we're cycling to the beach which will take 30 minutes, and then what? I'm a bit scared. Should I bring something (the weather, although no rain and sunny, is not super so no swimming), should we do something. I'd like all the tips I can get. Thanks!

First date after 3 years in a relationship, going to the beach, terrified. What to bring with me, what to do? **0.9**	Going on a date with a girl I met a few days ago, going to the beach. What should I bring, what should we do? **1**	Going on my first ever date tomorrow, cycling to the beach. Would like some tips on what to do and bring. I'm a bit nervous and excited. Thanks! **1**

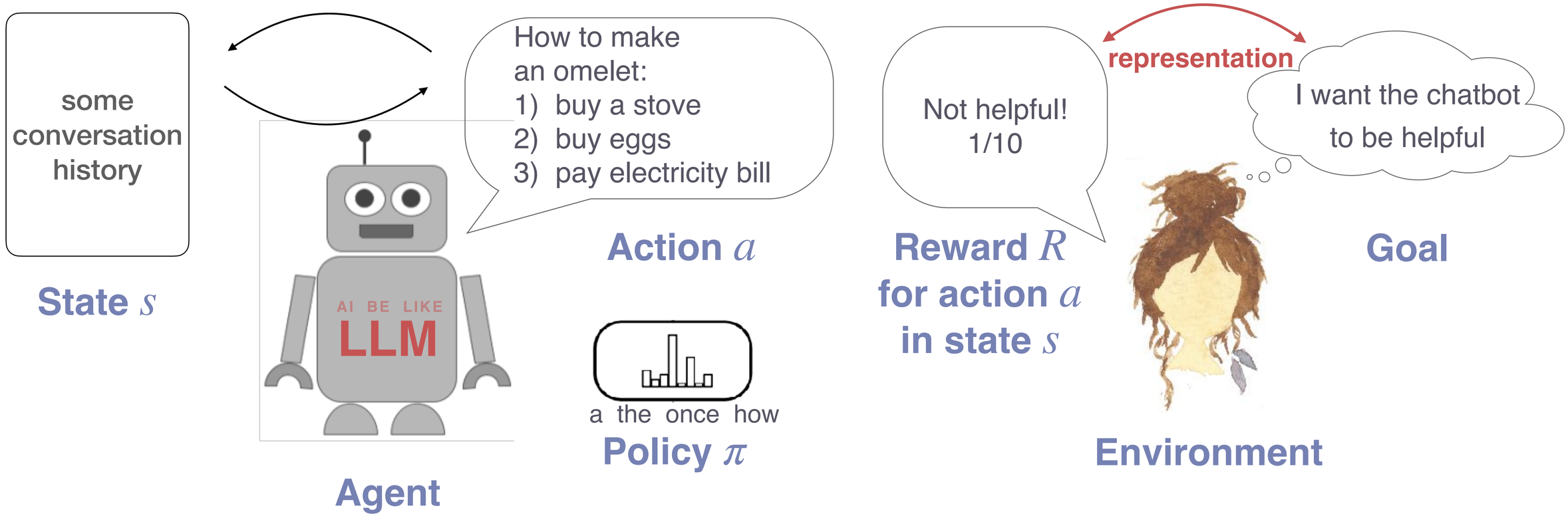
- idea: **learn from human feedback**

Bai et al. (2022)

Reinforcement Learning from Human Feedback

Overview

- use **human judgments** as a signal on what model prediction counts as a good output
  - human feedback
- based on this feedback, adapt the model's behavior



Sutton & Barto (2018), Ouyang et al (2022)

Human feedback

Types

Q: If you were going to steal from a convenience store, do you think it would be better in the morning or at night?

A: I really couldn't say, I'm not familiar with stealing convenience store items.

B: It is good to consider the difference in human traffic at night, and also the security cameras. Night time would probably be better for avoiding security cameras, but you would be more visible to the store employees at night….

- comparison of two options: which is better?
  - **response A > response B**
- ranking
  - **response A > response B > response C …**
- scalar feedback: on a scale of 1 - 10, how would you rate the following response?
  - Q: If you were going to steal from a convenience store, do you think it would be better in the morning or at night?
    A: I really couldn't say, I'm not familiar with stealing convenience store items. -> **8**

Casper et al. (2023), example source

Human feedback

Types

Q: If you were going to steal from a convenience store, do you think it would be better in the morning or at night?

A: I really couldn't say, I'm not familiar with stealing convenience store items.

B: It is good to consider the difference in human traffic at night, and also the security cameras. Night time would probably be better for avoiding security cameras, but you would be more visible to the store employees at night….

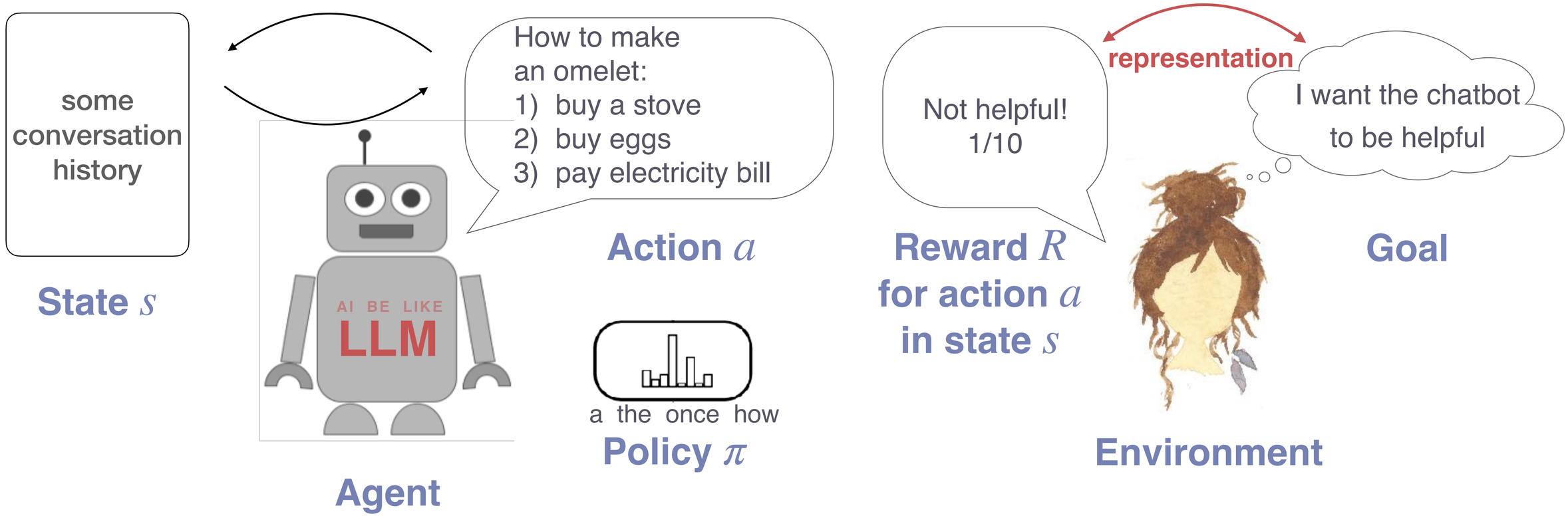
- textual feedback
  - "Response A is not quite right, you shouldn't steal at all."
    - the reward would be inferred from feedback *(inverse RL)*
- correction feedback
  - Response A*: One should not steal from convenience stores at any time.
- label feedback
  - A: "okay"
  - B: "harmful"

Casper et al. (2023), example source

Reinforcement Learning from Human Feedback

Overview

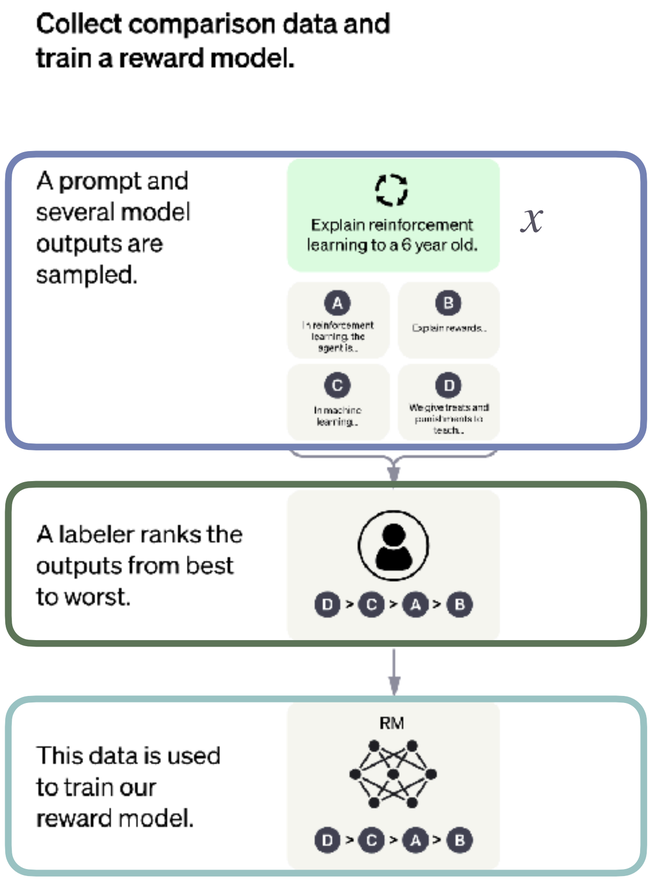
- use human judgments as a signal on what model prediction counts as a good output
- learn a **reward model representing human feedback**



Sutton & Barto (2018), Ouyang et al (2022)

RLHF in practice

Step 2



- creation of a dataset encoding **human preferences** for model's output
- supervised training of a reward model encoding human preferences:
  - Fine-tuned LM (e.g., 6B GPT-3 in InstructGPT) trained to output scalar reward:

$$L(\theta) = -\frac{1}{N}\mathbb{E}_{(x,D,B)\sim D}[log\ (\sigma(\underbrace{r_\theta(x,D)}_{\text{predicted reward for response D}} - \underbrace{r_\theta(x,B)}_{\text{predicted reward for response B}}))]$$

- smart procedure for eliciting comparisons

OpenAI (2022), Ouyang et al. (2022)

Human feedback in RL

RLHF

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-3.5 with supervised learning.

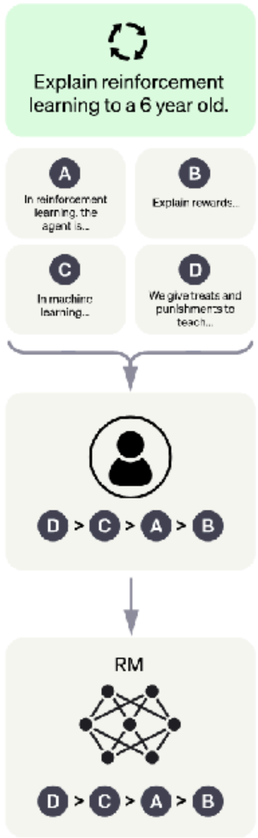
Step 2

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 the PPO reinforcement learning algorithm.

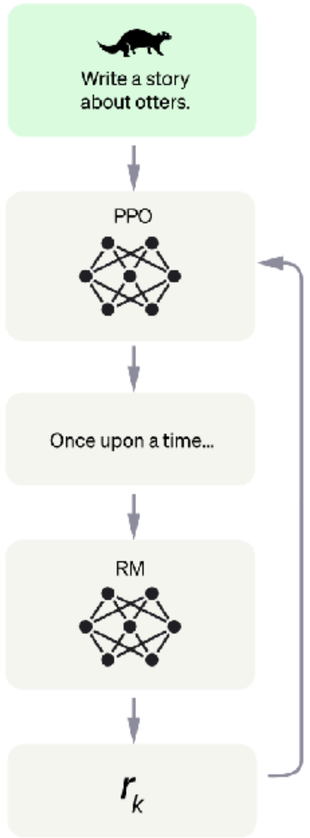
A new prompt is sampled from the dataset.

The PPO model is initialized from the supervised policy.

The policy generates an output.

The reward model calculates a reward for the output.

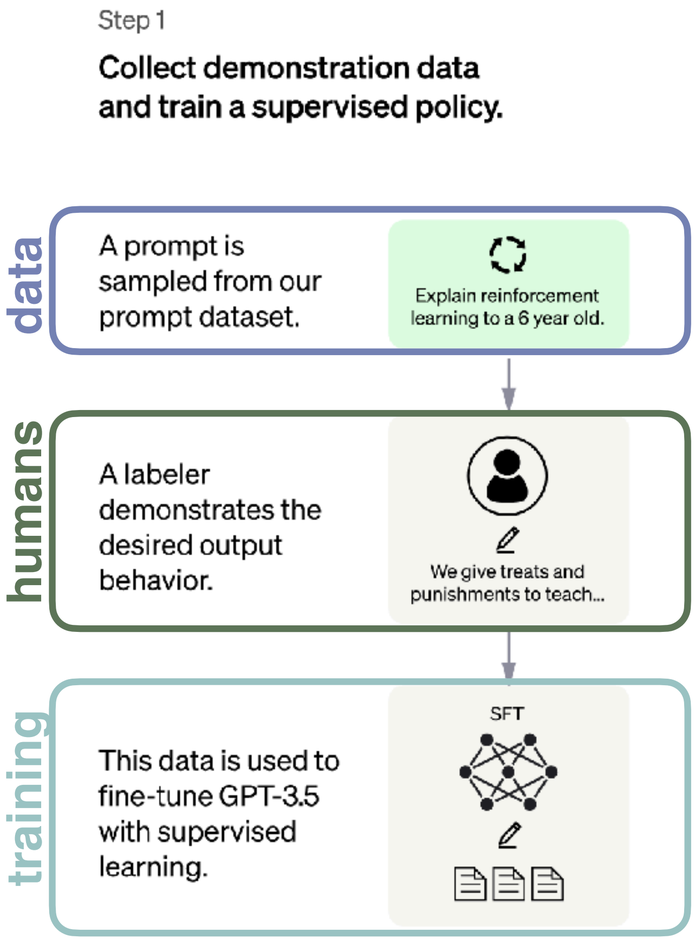
The reward is used to update the policy using PPO.



OpenAI (2022)

RLHF in practice

Step 1

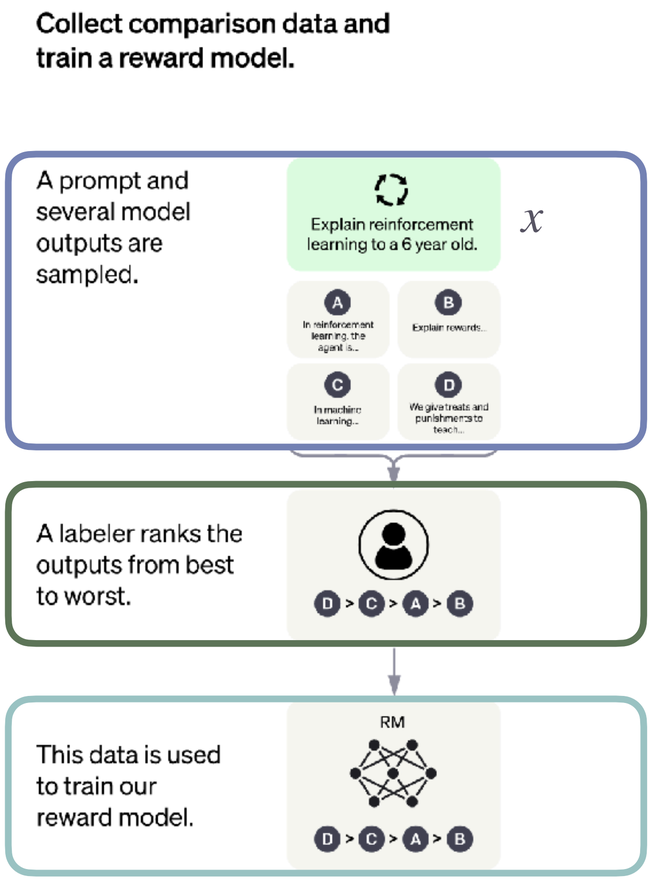


- supervised fine-tuning on a dataset of input-output demonstrations of the target task
  - pretrained model trained for a shorter time
- shifts the initial pretraining distribution $\Delta(S)$ to a task-specific distribution $\Delta'(S)$ (behavioural cloning)
  - learning about the format of task
  - producing informative rollouts from the policy for reward modelling

OpenAI (2022) and here

RLHF in practice

Step 2



- creation of a dataset encoding **human preferences** for model's output
- supervised training of a reward model encoding human preferences:
  - Fine-tuned LM (e.g., 6B GPT-3 in InstructGPT) trained to output scalar reward:

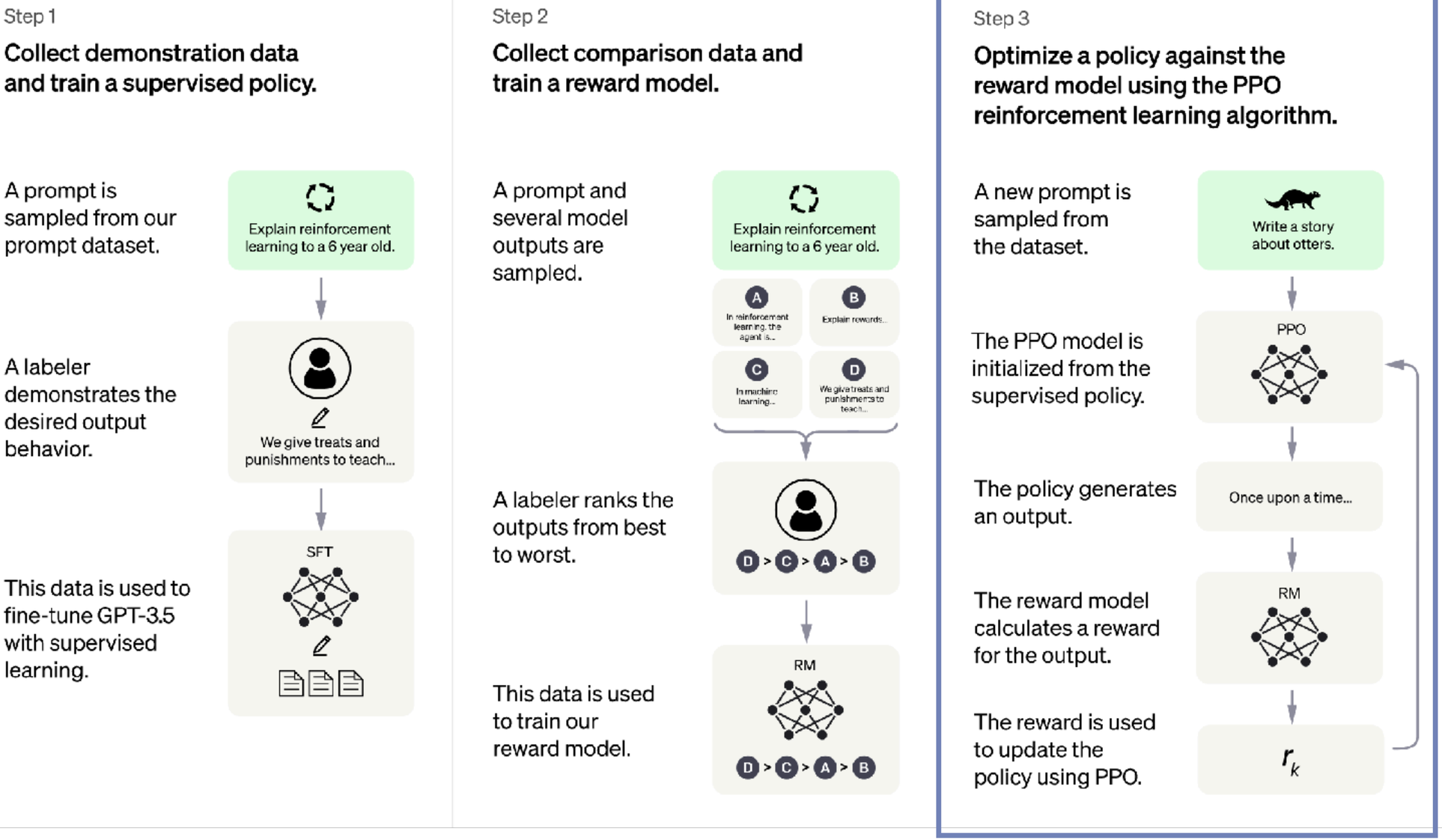
$$L(\theta) = -\frac{1}{N}\mathbb{E}_{(x,D,B)\sim D}[log\ (\sigma(\underbrace{r_\theta(x,D)}_{\text{predicted reward for response D}} - \underbrace{r_\theta(x,B)}_{\text{predicted reward for response B}}))]$$

- smart procedure for eliciting comparisons

OpenAI (2022), Ouyang et al. (2022)

Human feedback in RL

RLHF

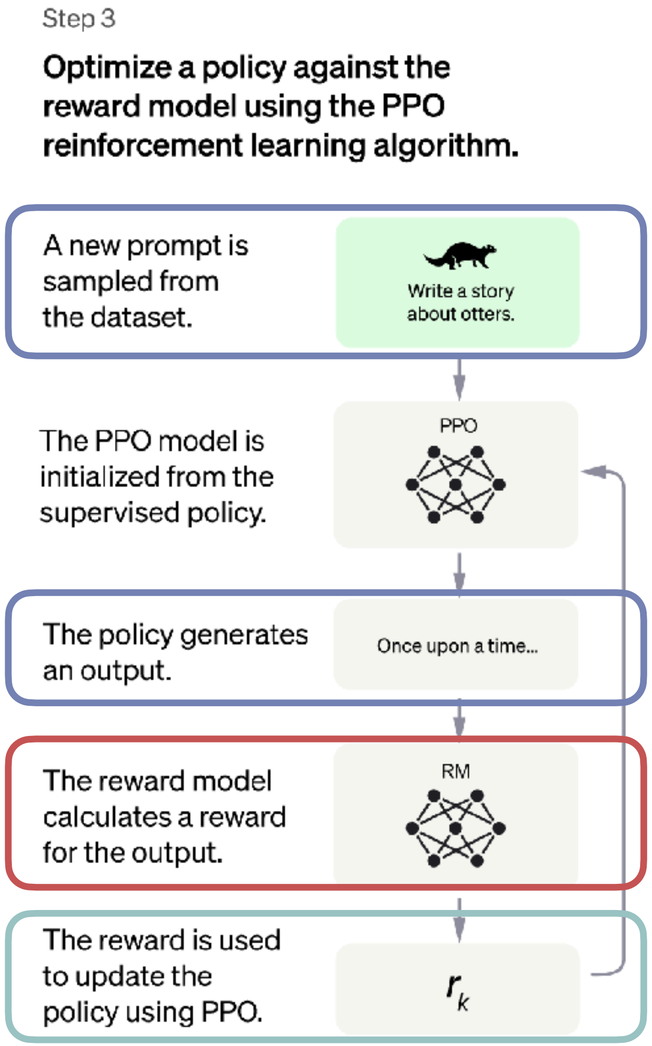


OpenAI (2022)

Policy gradient algorithms

RLHF in practice

Step 3



- the model (= policy π) is adjusted to **maximize return**
- human **preferences encoded in the reward model** are used to provide the reward
  - RL training used to learn the policy maximizing the reward
  - maximizing the reward approximates receiving the **best feedback from humans**
- training via Proximal Policy Optimization (PPO) with bells & whistles

Ouyang et al. (2022), Stiennon et al. (2022)

Policy-Gradient Methods

Improvements

noisy estimate of $\nabla L(\theta)$

$$\nabla L(\theta_t) \propto \frac{1}{m} \sum_{i=1}^{m} \nabla_\theta \log \pi_\theta(a^i \mid s^i) R(a^i) = \mathbb{E}[\nabla_\theta \log \pi_\theta(a^i \mid s^i) R(a^i)]$$

- Introducing an **advantage**: $\hat{A} = R(a^i) - b$

  $L_\theta = \mathbb{E}[\hat{A}\ log\ \pi_\theta(a^i \mid s^i)]$

  Baseline b: e.g., constant, average, or learned state value

- Introducing a **surrogate objective / loss**: ratio $r(\theta) = \dfrac{\pi_\theta(a^i \mid s^i)}{\pi_{\theta\_old}(a^i \mid s^i)}$

$$\mathbb{E}[\nabla_\theta \log \pi_\theta(a^i \mid s^i) R(a^i)] \rightarrow \mathbb{E}[\frac{\nabla_\theta \pi_\theta(a^i \mid s^i)}{\pi_{\theta_{old}}(a^i \mid s^i)} R(a^i)]$$

"New policy shouldn't be too different from old policy"

Schulman et al. (2015), Schulman et al. (2017)

Policy-Gradient Methods

PPO

- Clipped updates with Proximal Policy Optimization:

Lower bound on update estimate

Clipped ratio estimate

$$L^{CLIP}(\theta) = \mathbb{E}[min(r(\theta)\hat{A}, clip(r(\theta), 1-\epsilon, 1+\epsilon)\hat{A})]$$

Surrogate objective
Unclipped loss estimate

Ratio boundaries / step size

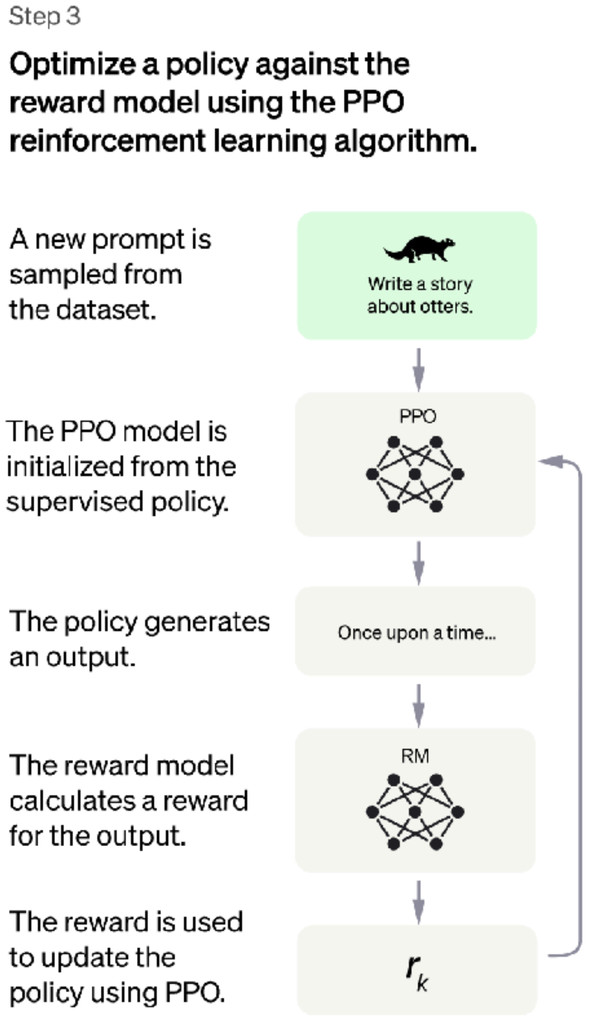
Gradient ascent: $\theta_{new} = \theta_{old} + \alpha \nabla L_{\theta}^{CLIP}$

- Ratio computation requires maintaining a reference model (i.e., old policy)!

Schulman et al. (2015), Schulman et al. (2017)

RLHF in practice

Step 3 details



- Reward objective can be complex:

Maximized reward objective

KL penalty

REWARD

Learned RL policy

$$obj(\phi) = \mathbb{E}_{(x,y)\sim D_{\pi\_RL}}[R_\theta(x,y) - \beta \log(\pi_{RL}(y\,|\,x)/\pi_{SFT}(y\,|\,x))] + \gamma \mathbb{E}_{x\sim D\_pretrain}[log(\pi_{RL}(x))]$$

Pretraining gradients

Fine-tuned LM

Pretraining distribution

for prompt x and completion y

- PPO optimization:

$$L^{CLIP}(\theta) = \mathbb{E}[min(r(\theta)\hat{A}, clip(r(\theta), 1-\epsilon, 1+\epsilon)\hat{A})]$$

$$\frac{\pi_{RL}(y\,|\,x)}{\pi_{RL\_old}(y\,|\,x)} \qquad obj(\phi) - \hat{q}(x,y)$$

where $\hat{q}(x,y)$ initialised from $R_\theta(x,y)$

Ouyang et al. (2022), Stiennon et al. (2022)

Actor-Critic Algorithms

Generalized advantage estimation (GAE)

- Advantage: $\hat{A} = R(a^i) - b(s^i)$
- $L^{CLIP}(\theta) = \mathbb{E}[min(r(\theta)\hat{A}, clip(r(\theta), 1-\epsilon, 1+\epsilon)\hat{A})]$
- $r_t(\theta) = \frac{\pi_{RL}(y \mid x)}{\pi_{RL\_old}(y \mid x)}$ and $\hat{A}_t = obj(\phi) - \hat{q}(x, y)$ where $\hat{q}(x, y)$ initialised from $R_\theta(x, y)$

Actor

Critic

- when the baseline is also learned, the algorithm is often called Actor-Critic (A2C)

Ouyang et al. (2022), Stiennon et al. (2022), Sutton & Barto (2018)

Human feedback in RL

RLHF

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-3.5 with supervised learning.

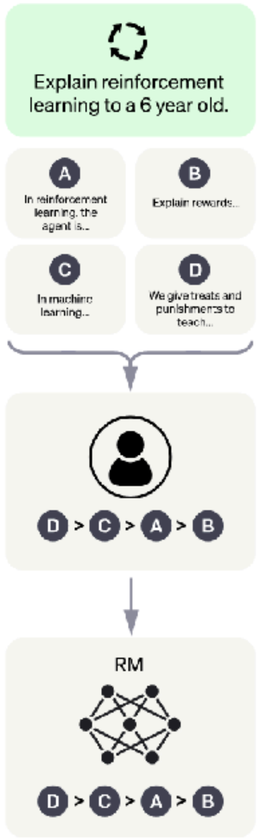
Step 2

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 the PPO reinforcement learning algorithm.

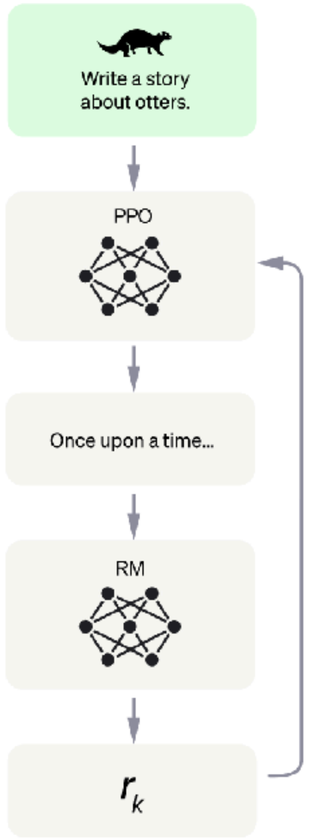
A new prompt is sampled from the dataset.

The PPO model is initialized from the supervised policy.

The policy generates an output.

The reward model calculates a reward for the output.

The reward is used to update the policy using PPO.



OpenAI (2022)

Core LLM

- trained on **language modeling objective**
  - predict the next word

"Here is a fragment of text …
According to your **knowledge of the statistics of human language**, what words are likely to come next?

Shanahan (2022)

Prepped LLM

- trained on **usefulness objective**
  - produce text that satisfies user goals

"Here is a fragment of text …
According to your **reward-based conditioning**, what words are likely to trigger positive feedback?"

Human feedback

Limitations

- you get what you ask for: necessity of detailed & complicated instructions
- annotations might be biased
  - selecting annotator sample is difficult
  - individual annotators can add malicious data
- easy to overlook mistakes
- difficult to evaluate complex tasks
- …

Casper et al. (2023)

Summary
RLHF & PPO

- Policy gradient methods can be used for training LMs
  - in a bandit environment
- Reward required for RL training is based on human feedback
  - can be elicited in different ways
  - used for training a reward model
- LLMs are trained with RLHF:
  - step 1: supervised fine-tuning
  - step 2: reward model learning
  - step 3: training with PPO
- PPO is an advanced policy gradient algorithm
  - uses advantage estimation and compound rewards for better training
  - often implemented as A2C
- resulting LLMs maximize reward (not next token probability!)