



Reinforcement Learning & RL with Human Feedback (RLHF)





DL4DS – Spring 2025

DS542 Gardos

Prince, *Understanding Deep Learning*, Creative Commons CC-BY-NC-ND license. (C) MIT Press

Other Content Cited

April Dates

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
	April 1	2	3	4 GANs	5	6
7	8	9 VAEs	10 Discussion	11 Diffusion Models	12	13
14	15	16 Graph Neural Nets (VizWiz Leaders Share)	17 Discussion	18 Office Hours	19	20
21	22	23 RL/RLHF 	24 Discussion	25 ★ Project Presentations 1 ★ 	26	27
28	29	30 ★ Project Presentations 2 ★ 	May 1 Discussion??	2 Study Period	3 Study Period	4
5	6 Final Exams	7 Final report & Repo ** 	8	9	10	11

** Might be earlier. Depends on when grades are due.

Project Presentations

Final project info updated on Gradescope and website.

Format:

≤ 3 minutes screencast/video

≤ 2 minutes additional presentation

~2 minutes Q&A

April 25 – 75 minutes

April 30 – 75 minutes

Outline

- RL basic concepts
- Deep reinforcement learning from human preferences (2017)
- Training language models to follow instructions with human feedback (2022)

Outline

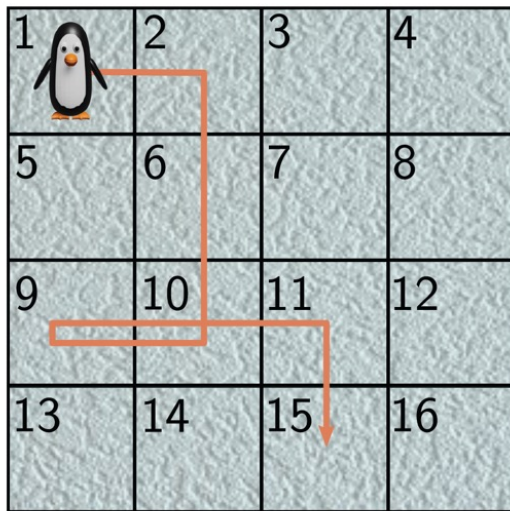
- RL basic concepts
- Deep reinforcement learning from human preferences (2017)
- Training language models to follow instructions with human feedback (2022)

In a nutshell...

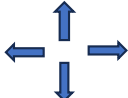
RL is the study of agents and how they learn by trial and error.

It formalizes the idea that rewarding or penalizing an agent for its behavior makes it more or less likely, respectively, to repeat that behavior in the future.

Markov Process



$s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8$
 $\tau = [1, 2, 6, 10, 9, 10, 11, 15]$

Can only go 

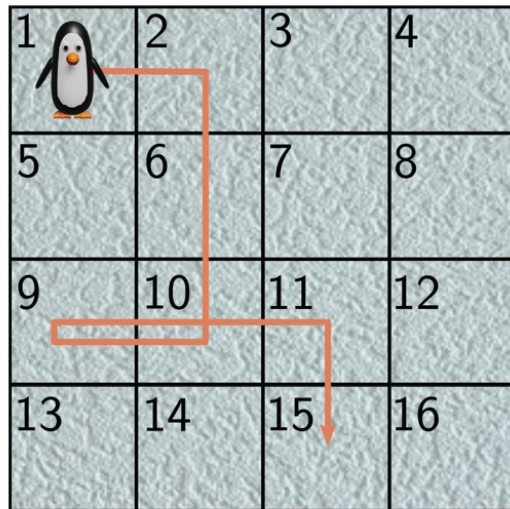
World is described by a set of states s

Changes between states are represented by *transition probabilities* $\Pr(s_{t+1}|s_t)$

Markov process produces a sequence of states s_1, s_2, s_3, \dots

A *trajectory* is the sequence of states
 $\tau = [s_1, s_2, s_3, \dots]$

Markov Process – Transition Probabilities



$s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8$
 $\tau = [1, 2, 6, 10, 9, 10, 11, 15]$

$s_{t+1} \backslash s_t$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	0	0.33	0	0	0.33	0	0	0	0	0	0	0	0	0	0	0
2	0.5	0	0.33	0	0	0.25	0	0	0	0	0	0	0	0	0	0
3	0	0.33	0	0.5	0	0	0.25	0	0	0	0	0	0	0	0	0
4	0	0	0.33	0	0	0	0	0.33	0	0	0	0	0	0	0	0
5	0.5	0	0	0	0	0.25	0	0	0.33	0	0	0	0	0	0	0
6	0	0.33	0	0	0.33	0	0.25	0	0	0.25	0	0	0	0	0	0
7	0	0	0.33	0	0	0.25	0	0.33	0	0	0.25	0	0	0	0	0
8	0	0	0	0.5	0	0	0.25	0	0	0	0	0.33	0	0	0	0
9	0	0	0	0	0.33	0	0	0	0	0.25	0	0	0.5	0	0	0
10	0	0	0	0	0	0.25	0	0	0.33	0	0.25	0	0	0.33	0	0
11	0	0	0	0	0	0	0.25	0	0.25	0	0.33	0	0	0	0.33	0
12	0	0	0	0	0	0	0	0.33	0	0	0.25	0	0	0	0	0.5
13	0	0	0	0	0	0	0	0	0.33	0	0	0	0	0.33	0	0
14	0	0	0	0	0	0	0	0	0	0.25	0	0	0.5	0	0.33	0
15	0	0	0	0	0	0	0	0	0	0	0.25	0	0	0.33	0	0.5
16	0	0	0	0	0	0	0	0	0	0	0	0.33	0	0	0.33	0

$Pr(s_{t+1}|s_t)$

Transition probability from square 1 to square n

Equally likely to go in any allowable direction.

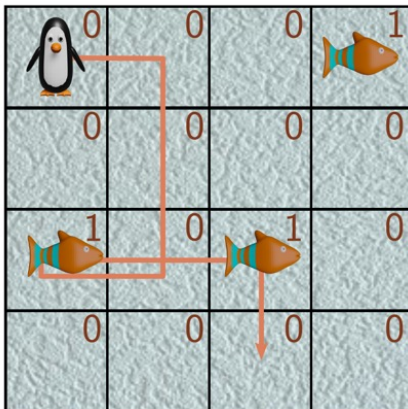
Markov reward process

Distribution of rewards at next time step given current state: $\Pr(r_{t+1}|s_t)$

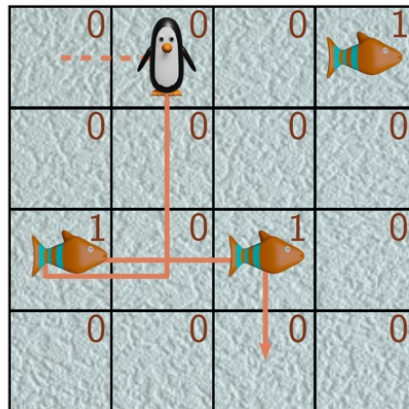
The *return* G_t is the sum of discounted future rewards

$$G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \quad \text{where} \quad \gamma \in (0,1]$$

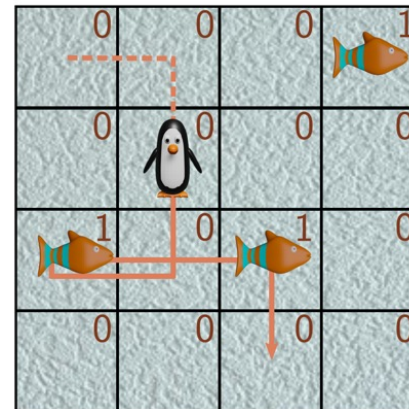
a) $G_1 = 0 + \gamma \cdot 0 + \gamma^2 \cdot 0 + \gamma^3 \cdot 0 + \gamma^4 \cdot 1 + \gamma^5 \cdot 0 + \gamma^6 \cdot 1 + \gamma^7 \cdot 0 = 1.19$



b) $G_2 = 0 + \gamma \cdot 0 + \gamma^2 \cdot 0 + \gamma^3 \cdot 1 + \gamma^4 \cdot 0 + \gamma^5 \cdot 1 + \gamma^6 \cdot 0 = 1.31$



c) $G_3 = 0 + \gamma \cdot 0 + \gamma^2 \cdot 1 + \gamma^3 \cdot 0 + \gamma^4 \cdot 1 + \gamma^5 \cdot 0 = 1.47$



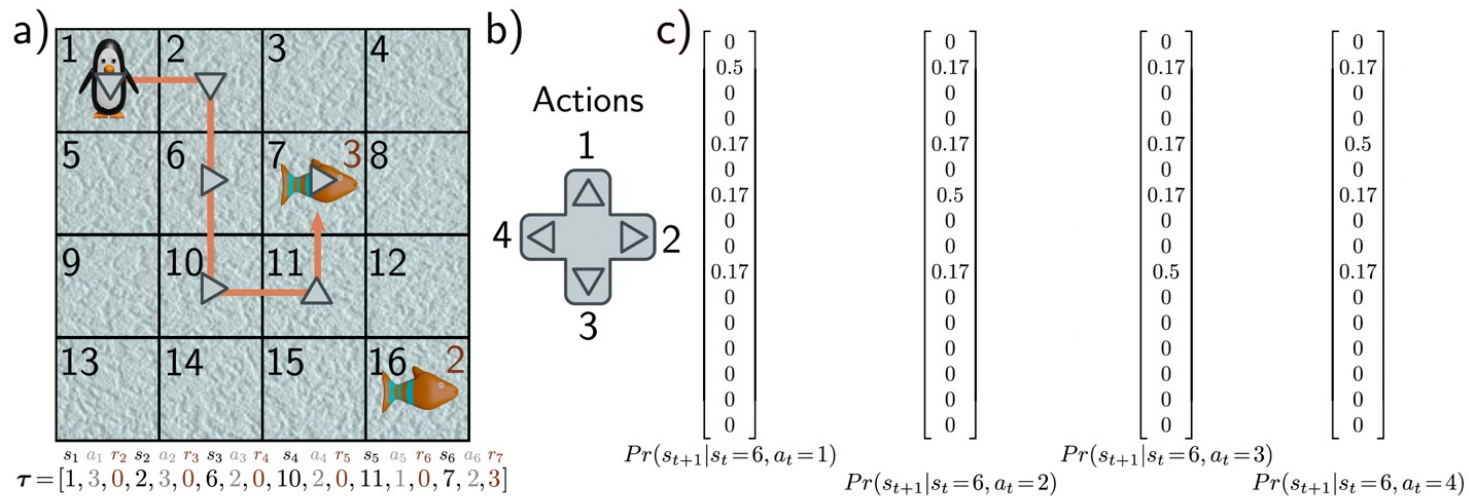
$s_1 \ r_1 \ s_2 \ r_2 \ s_3 \ r_3 \ s_4 \ r_4 \ s_5 \ r_5 \ s_6 \ r_6 \ s_7 \ r_7 \ s_8 \ r_8 \ s_9$
 $\tau = [1, 0, 2, 0, 6, 0, 10, 0, 9, 1, 10, 0, 11, 1, 15, 0]$

Trajectory now comprised of state and the next reward

Markov decision process (MDP)

Adds a set of possible actions at each time step that changes transition and reward probabilities

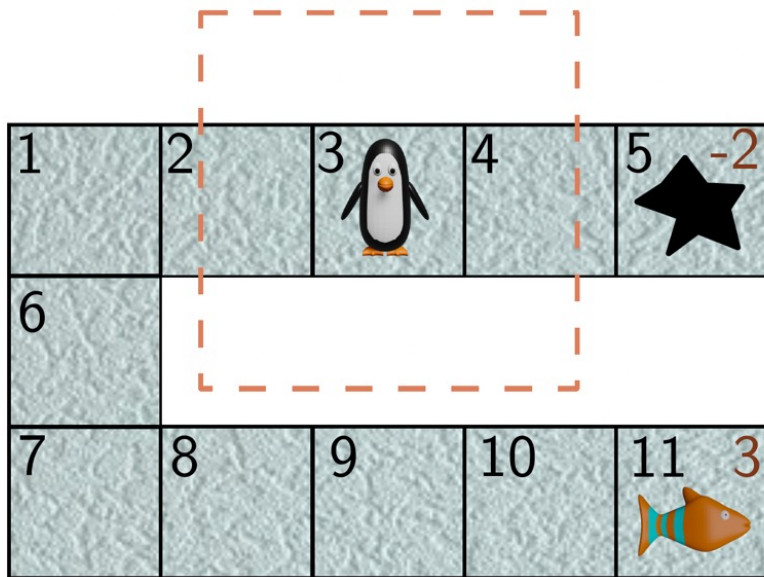
$$\Pr(s_{t+1}|s_t, a_t) \quad \text{and} \quad \Pr(r_{t+1}|s_t, a_t)$$



Trajectory now consists of states, actions and rewards

Action is not deterministic

Partially observable Markov decision process (POMDP)



The state is not directly visible, but instead receives an observation o_t drawn from $\Pr(o_t|s_t)$

Penguin can only see what is in the dashed box.

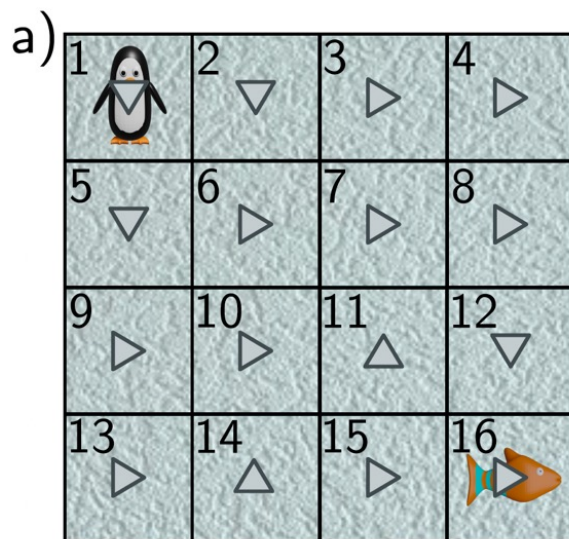
Indistinguishable from what it would see from box 9.

Policy

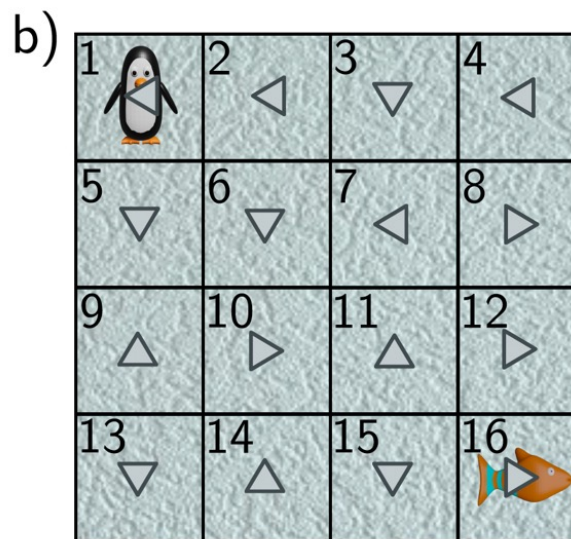
The rules that determine the agent's (e.g. penguin's) action for each state: $\pi[a|s]$

Can be *deterministic* or *stochastic*

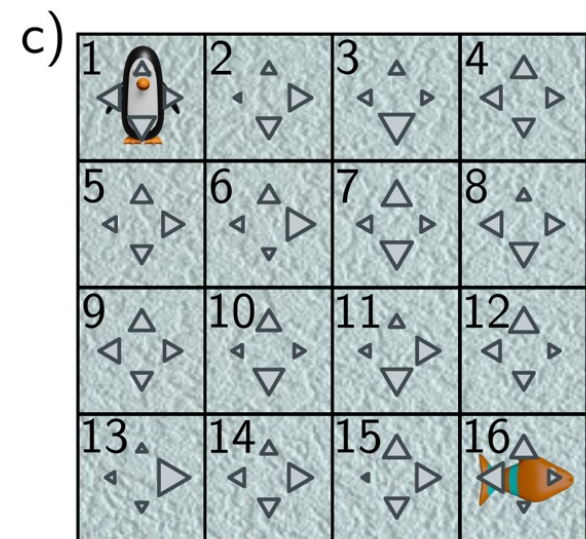
Can be *stationary* or *non-stationary (time dependent)*



Better deterministic policy



Poorer deterministic policy



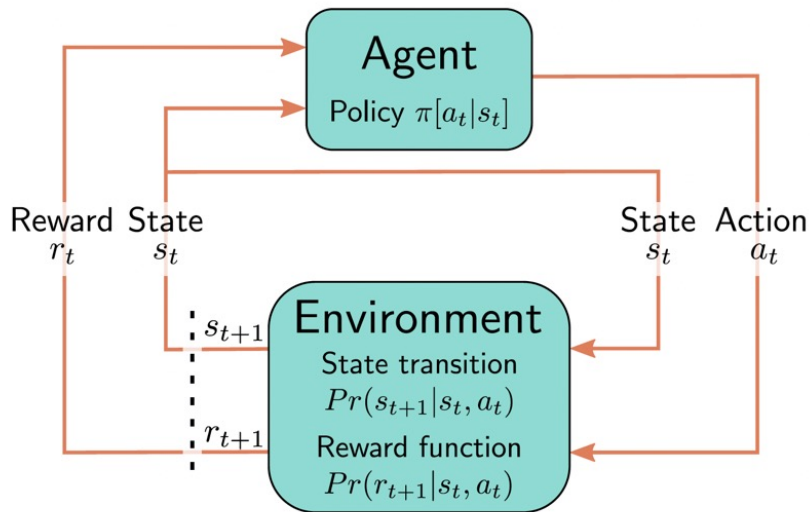
Stochastic policy

Full reinforcement learning loop

Agent receives the state (or observation) and reward.

Then (optionally modifies the policy and) choose next action.

Environment then assigns next state and reward according to transition probabilities.



RLHF as RL

“We can think of the main model as an agent that takes sequential actions (choose tokens) and receives a delayed reward from the reward model when the last token is chosen”

Outline

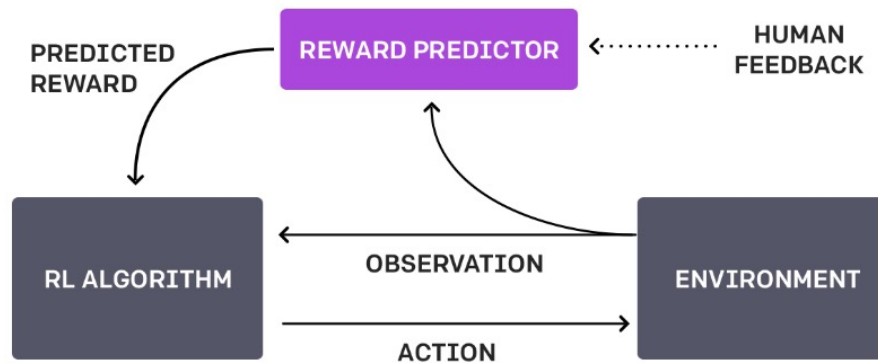
- RL basic concepts
- Deep reinforcement learning from human preferences (2017)
- Training language models to follow instructions with human feedback (2022)

Aligning to Human Preferences

“One step towards building safe AI systems is to remove the need for humans to write goal functions, since using a simple proxy for a complex goal, or getting the complex goal a bit wrong, can lead to undesirable and even dangerous behavior.

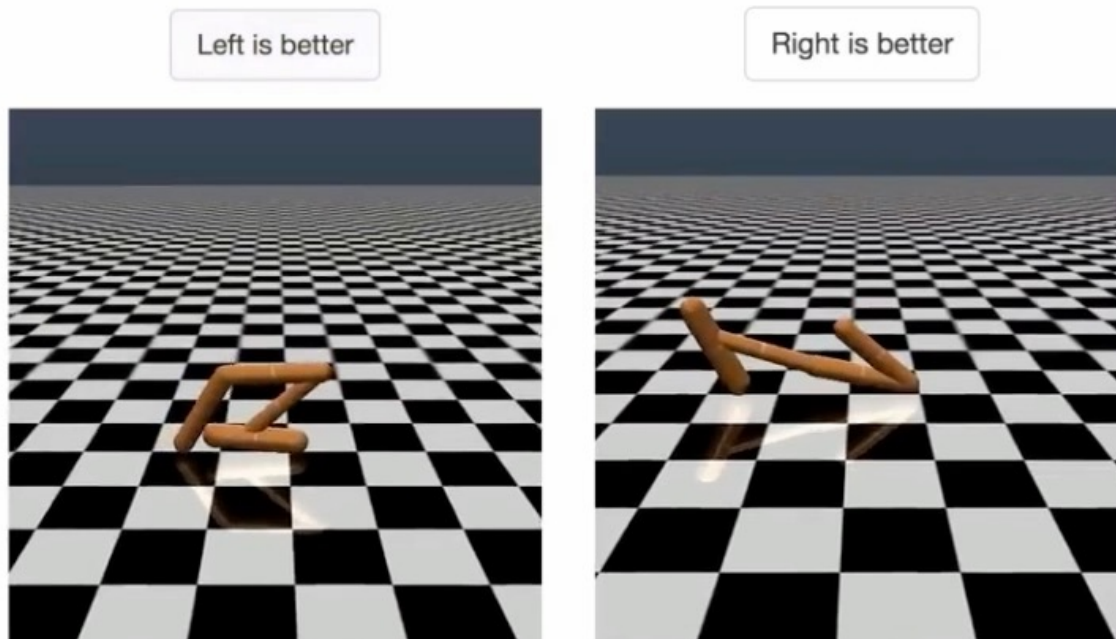
..., we’ve developed an algorithm which can infer what humans want by being told which of two proposed behaviors is better.”

Originally developed to improve RL systems



- Starts by acting randomly
- Gives 2 examples to a human who votes on which is closer to achieving goal
- AI builds reward model to match human votes
- Continues to seek feedback on trajectory pairs that are most uncertain

Originally developed to improve RL systems

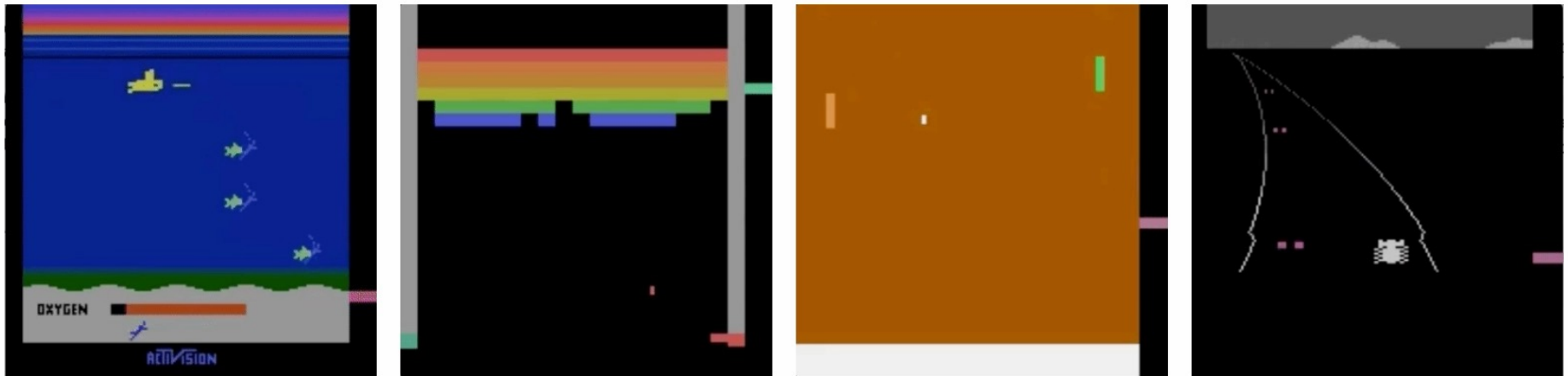


- Trained with ~1 hour of evaluator time
- Background policy accumulated ~70 hours of experience

<https://openai.com/research/learning-from-human-preferences>, 2017

Christiano et al (OpenAI), "Deep Reinforcement Learning from Human Preferences," 2017

Originally developed to improve RL systems

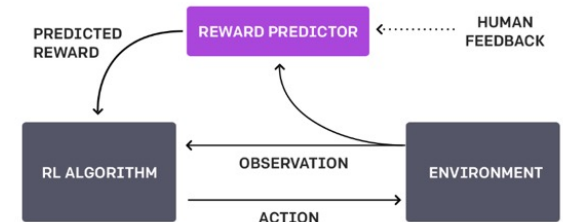


- Learned Atari with only human reward model (right vertical bar)
- Without access to game score as reward
- Human feedback sometimes does better than normal reward function

<https://openai.com/research/learning-from-human-preferences>, 2017

Christiano et al (OpenAI), "Deep Reinforcement Learning from Human Preferences," 2017

Deep RL from Human Preferences



Assume human overseer who can express preferences between trajectory segments

$$\sigma = ((o_0, a_0), (o_1, a_1), \dots, (o_{k-1}, a_{k-1})) \in (\mathcal{O} \times \mathcal{A})^k$$

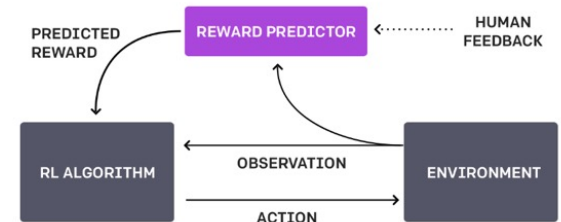
We write

$$\sigma^1 \succ \sigma^2$$

to indicate that the human preferred trajectory σ^1 to σ^2 .

Goal of Agent: Produce trajectories preferred by human, while making as few queries as possible to the human.

Deep RL from Human Preferences



Preferences are *generated* by a reward function $r : \mathcal{O} \times \mathcal{A} \rightarrow \mathbb{R}$ if

$$\left((o_0^1, a_0^1), \dots, (o_{k-1}^1, a_{k-1}^1) \right) \succ \left((o_0^2, a_0^2), \dots, (o_{k-1}^2, a_{k-1}^2) \right)$$

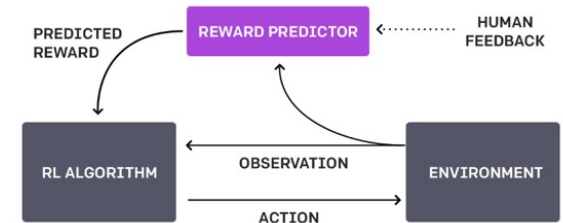
whenever

$$r(o_0^1, a_0^1) + \dots + r(o_k^1, a_k^1) > r(o_0^2, a_0^2) + \dots + r(o_{k-1}^2, a_{k-1}^2)$$

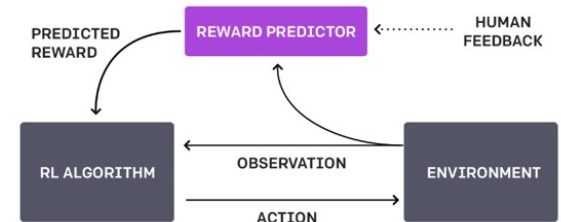
Deep RL from Human Preferences -- Method

At each point in time,

- maintain a policy $\pi : \mathcal{O} \rightarrow \mathcal{A}$
 - and a reward function estimate $\hat{r} : \mathcal{O} \times \mathcal{A} \rightarrow \mathbb{R}$
- each parameterized by deep neural networks.



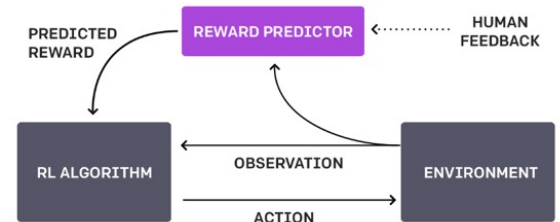
Deep RL from Human Preferences -- Method



The policy and reward estimate networks are updated by three processes

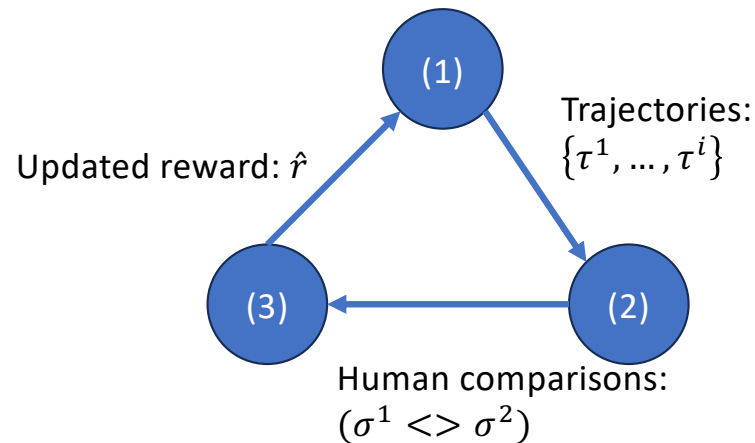
1. Policy π interacts with the environment to produce a set of trajectories $\{\tau^1, \dots, \tau^i\}$, where π is updated with traditional RL to maximize sum of predicted rewards
2. Select pairs of segments (σ^1, σ^2) from trajectories $\{\tau^1, \dots, \tau^i\}$ and query human for comparison.
3. Parameters of reward estimate \hat{r} are optimized via supervised learning to fit human comparisons

Deep RL from Human Preferences -- Method



The policy and reward estimate networks are updated by three processes

1. Learn policy π and produce trajectories $\{\tau^1, \dots, \tau^i\}$
2. Select pairs of segments (σ^1, σ^2) and query human
3. Update reward estimate \hat{r} to match human comparisons



Processes run asynchronously

Outline

- RL basic concepts
- Deep reinforcement learning from human preferences (2017)
- Training language models to follow instructions with human feedback (2022)

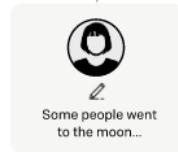
3 Steps

Step 1
**Collect demonstration data,
and train a supervised policy.**

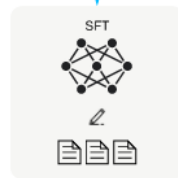
A prompt is
sampled from our
prompt dataset.



A labeler
demonstrates the
desired output
behavior.



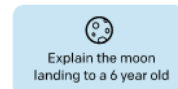
This data is used
to fine-tune GPT-3
with supervised
learning.



Supervised Fine-Tuning

Step 2
**Collect comparison data,
and train a reward model.**

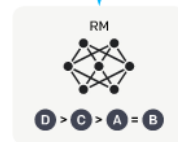
A prompt and
several model
outputs are
sampled.



A labeler
ranks the
outputs from
best to worst.



This data is used
to train our
reward model.



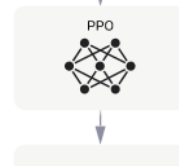
Reward Model (RM)
Training

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.

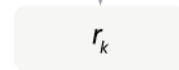


Once upon a time...

The reward model
calculates a
reward for
the output.



The reward is
used to update
the policy
using PPO.



Reinforcement Learning via
Proximal Policy Optimization
on this Reward Model

Supervised (Instruction) Fine Tuning

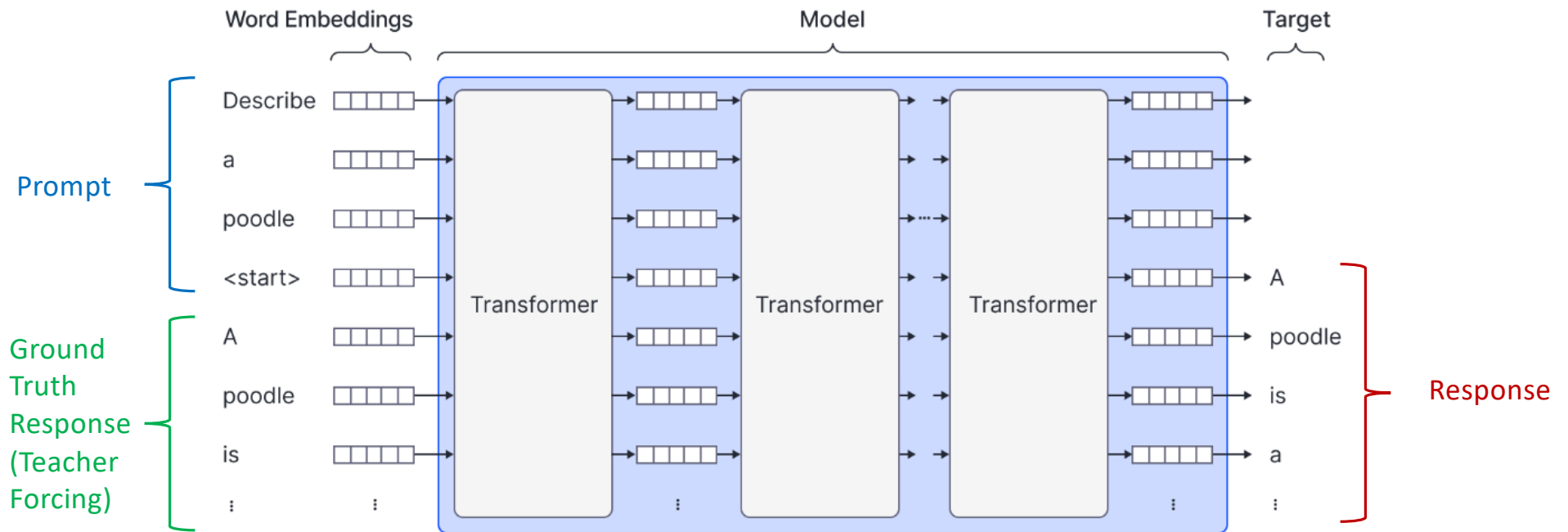
Create a dataset of ~10,000 prompts and fine-tuned responses.

Denoting the tokens for the i^{th} prompt by $\mathbf{x}_i = [x_{i,1}, x_{i,2}, \dots]$ and the tokens in the corresponding response by $\mathbf{y}_i = [y_{i,1}, y_{i,2}, \dots, y_{i,T_i}]$, the loss function can now be written as:

$$L[\phi] = - \sum_{i=1}^I \sum_{t=1}^{T_i} \log \left[Pr(y_{i,t+1} | \mathbf{x}_i, y_{i,1..t}, \phi) \right].$$

where once more ϕ represents the model parameters.

Supervised (Instruction) Fine Tuning



Reward Modeling

Train a neural network, e.g. start with the SFT model with the last layers replaced.

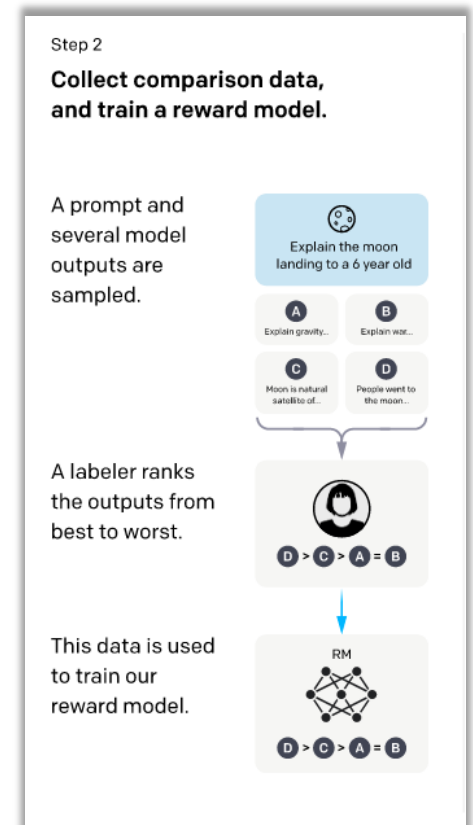
Present labelers between $K = 4$ and $K = 9$ responses to rank.

This produces $\binom{K}{2}$ comparisons for each prompt.

The loss function for the reward model is:

$$\text{loss}(\theta) = -\frac{1}{\binom{K}{2}} E_{(x, y_w, y_l) \sim D} [\log(\sigma(r_\theta(x, y_w) - r_\theta(x, y_l)))]$$

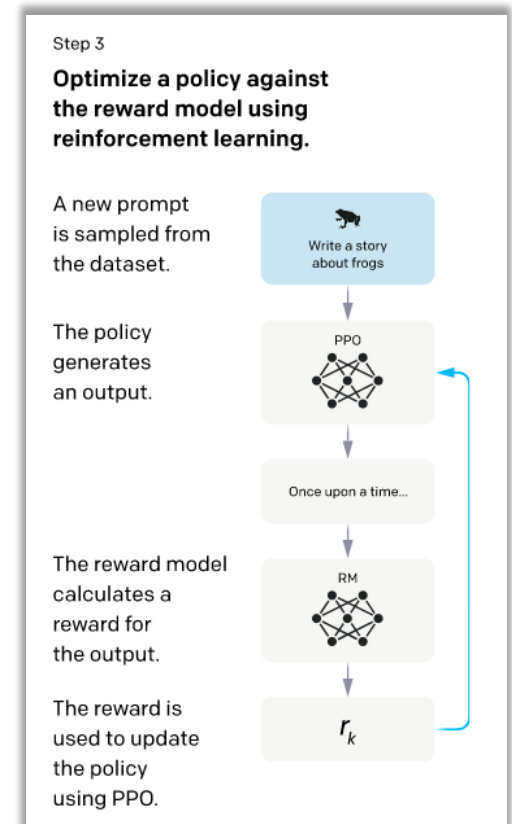
where $r_\theta(x, y)$ is the scalar output of the reward model for prompt x and completion y with parameters θ , y_w is the preferred completion (winner) versus y_l , and D is the data set of human comparisons.



Reinforcement Learning (RL)

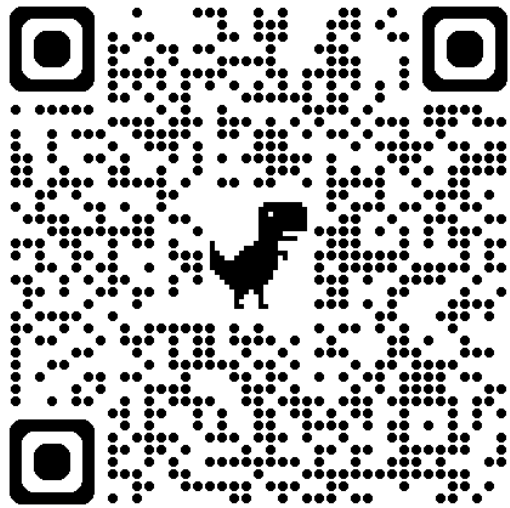
Continue training the SFT model to maximize the following objective:

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

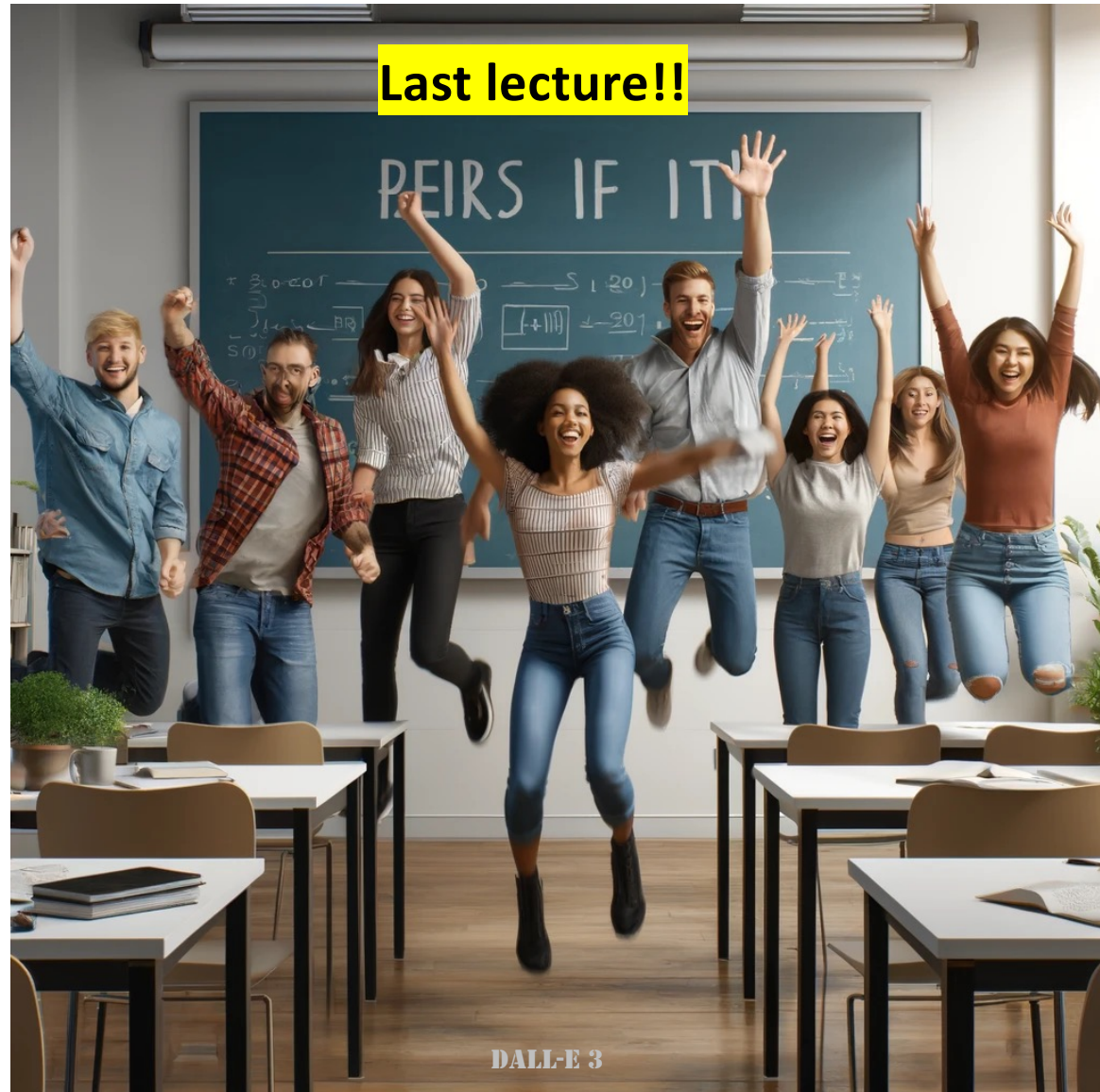


Next

- Project Presentations
- Fill out course evaluations



[Link](#)



DALL-E 3