UVA | SCHOOL of DATA SCIENCE

Reinforcement Learning with Human Feedback

Reinforcement Learning

School of Data Science

University of Virginia

Last updated: April 15, 2025

Agenda

> Quick Background

> Learning from Human Preferences

> Aligning Language Models to Follow Instructions

Background

A key ingredient for successful RL is the right reward function

As we've learned, it's hard to get this right

In practice, the reward function can be quite complex

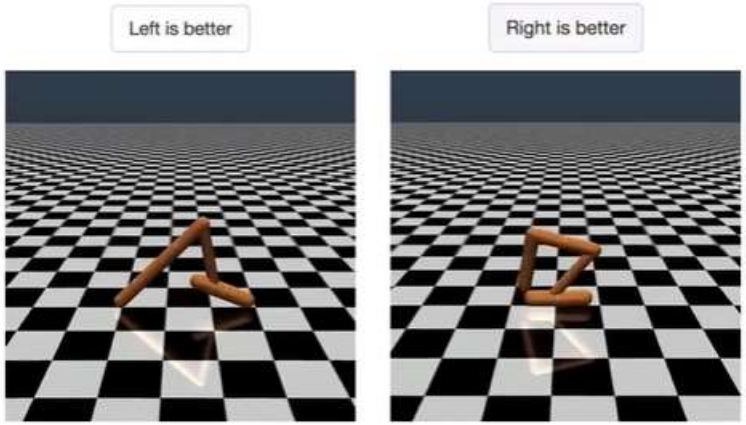
Getting reward function wrong can be dangerous

RLHF address this challenge

Learning from Human Preferences

Remove need for humans to write goal (reward) functions

Present human with two options and ask which is better



Source: https://openai.com/research/learning-from-human-preferences
Paper: https://arxiv.org/pdf/1706.03741.pdf

Feedback Cycle

1. Agent acts randomly in environment

2. Periodically, two video clips given to human

3. Human indicates which is better (reaches goal)

4. Agent builds *reward model* of task goal to match human preferences

5. System continues to ask for human feedback in areas of greatest uncertainty

Overall, **system learns the reward function from human**

Feedback Cycle, contd.

Human is asked to **compare** clips.

No scoring is done.

This is an easier task

Borrows ideas from other papers, but scales to Deep RL, learning more complex behaviors (Atari gaming, backflips)

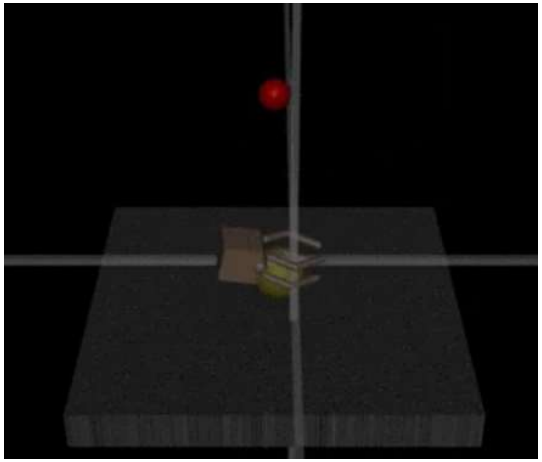
Humans provided feedback on < 1% of agent interactions w environment

Challenges

To get good performance, system requires a lot of human feedback

OpenAI working to reduce this need

Performance only as good as human evaluator's intuition about what looks better



Is the robot hand grasping the yellow item?

Segments

Form trajectory segments which are sequences of (observations, actions)

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

Indicate preference for segment 1 over segment 2: $\sigma^1 \succ \sigma^2$

Deep neural network uses three processes:

1. Policy interacts w environment to produce trajectories. Update params using RL. Rewards are predicted.
2. Select segments and send them to human for comparison
3. Params in reward function are optimized using supervised learning

Reward Function

Uses Advantage actor-critic for Atari, TRPO for robotics tasks

Fitting the reward function:

$$\hat{P}\left[\sigma^1 \succ \sigma^2\right] = \frac{\exp \sum \hat{r}\left(o_t^1, a_t^1\right)}{\exp \sum \hat{r}(o_t^1, a_t^1) + \exp \sum \hat{r}(o_t^2, a_t^2)}. \quad (1)$$

We choose $\hat{r}$ to minimize the cross-entropy loss between these predictions and the actual human labels:

$$\text{loss}(\hat{r}) = -\sum_{(\sigma^1,\sigma^2,\mu)\in\mathcal{D}} \mu(1) \log \hat{P}\left[\sigma^1 \succ \sigma^2\right] + \mu(2) \log \hat{P}\left[\sigma^2 \succ \sigma^1\right].$$

μ is a distribution over $\{1, 2\}$ indicating which segment the user preferred

Experimental Setup – Human Feedback

We will see results from Atari games and Mujoco environments (robotics)

In these experiments, agent will not see the true reward

Instead, they will get feedback from a human

The human saw a 1-2 sentence description of the task

Experimental Setup - Oracle

Also run *synthetic oracle*

When query is sent to oracle, it sees actual reward

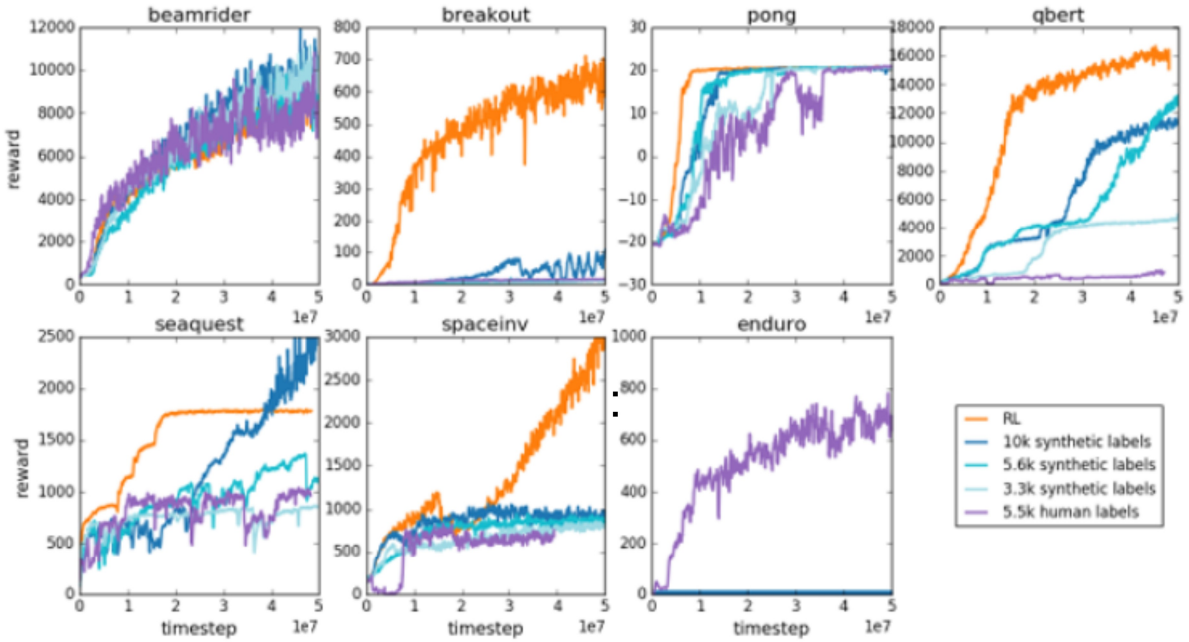
As more queries are made, performance will be more accurate

Finally, authors also show RL performance

It is possible for human feedback to outperform RL (better shaped rewards)

Results: Atari

Human feedback is effective and gives results close to RL in some cases



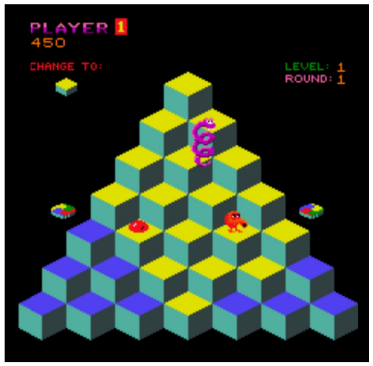


Figure 3: Results on Atari games as measured on the tasks' true reward. We compare our method using real human feedback (purple), our method using synthetic feedback provided by an oracle (shades of blue), and reinforcement learning using the true reward function (orange). All curves are the average of 3 runs, except for the real human feedback which is a single run, and each point is the average reward over about 150,000 consecutive frames.

Results: MuJoCo

Human feedback is effective and gives results close to RL in some cases

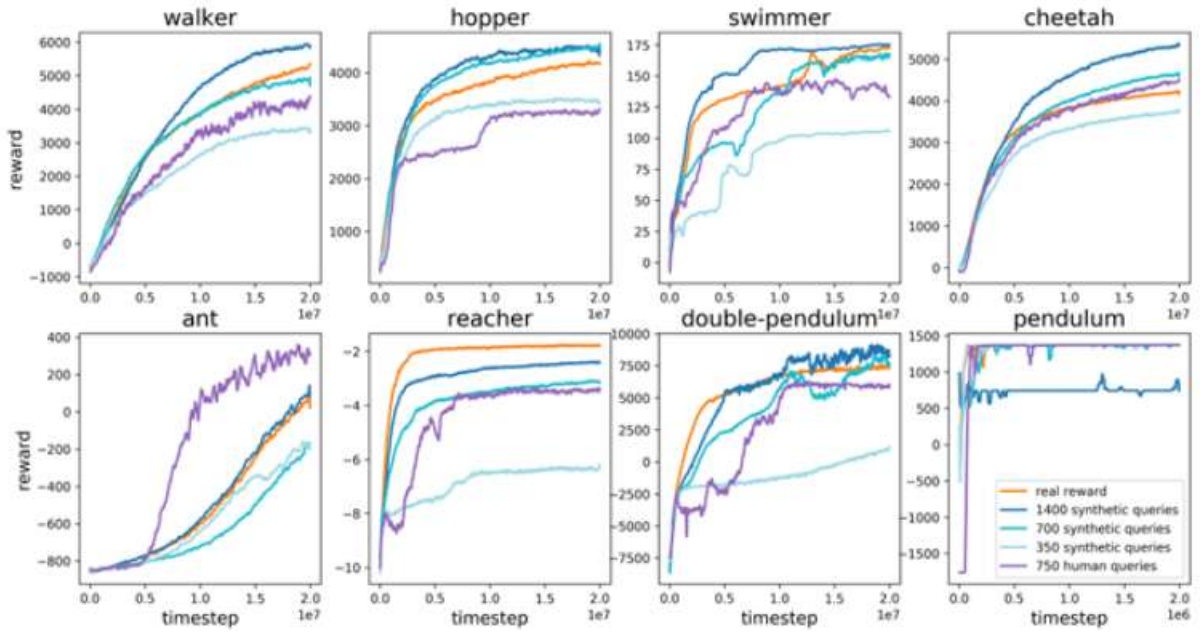


Figure 2: Results on MuJoCo simulated robotics as measured on the tasks' true reward. We compare our method using real human feedback (purple), our method using synthetic feedback provided by an oracle (shades of blue), and reinforcement learning using the true reward function (orange). All

II. Aligning Language Models to Follow Instructions

Manual Alignment of LLMs

In the early days of LLMs, when models gave bad output to prompts,

Humans worked feverishly to manually patch issues

The solution was sometimes to hardcode answers

The approach is now more sophisticated

Aligning Language Models to Follow Instructions

OpenAI GPT-3 models are not trained to follow user instructions
- can be untruthful, toxic, harmful

Trained language models that are better at following user intentions:
- more truthful
- less toxic

Models called *InstructGPT* models

Source: https://openai.com/research/instruction-following
Paper: https://arxiv.org/abs/2203.02155

Results: Prompt 1

Prompt

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

Completion

GPT-3

```
Explain the theory of gravity to a 6 year old.

Explain the theory of relativity to a 6 year old in a few
sentences.

Explain the big bang theory to a 6 year old.

Explain evolution to a 6 year old.
```

InstructGPT

```
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.
```

Results: Prompt 2

la ville antique. [...]

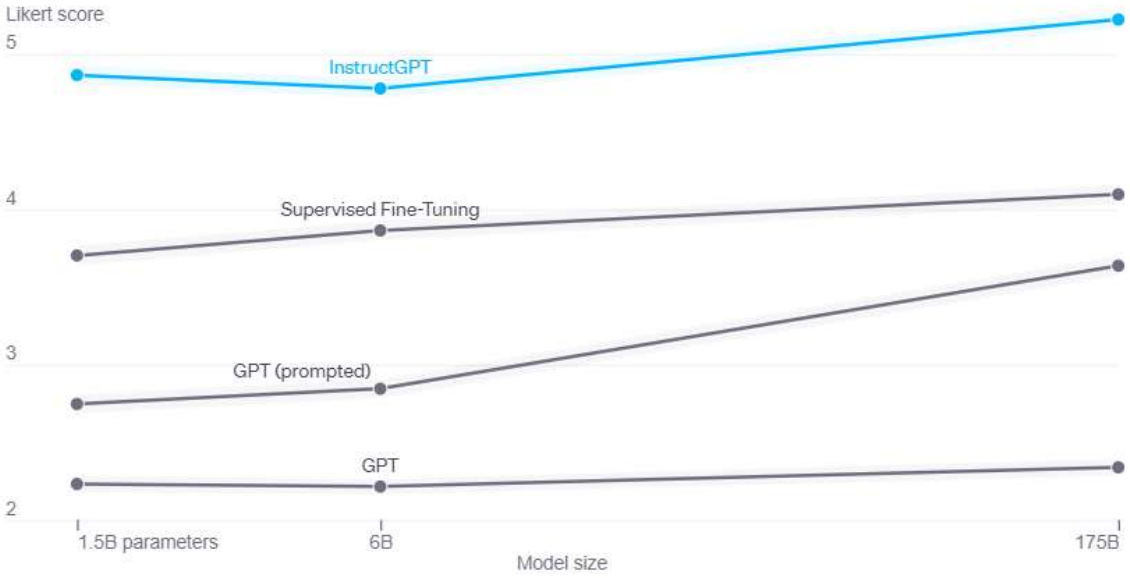
Prompt:
What is the purpose of the list C in the code below?

```
def binomial_coefficient(n, r):
    C = [0 for i in range(r + 1)];
    C[0] = 1;
    for i in range(1, n + 1):
        j = min(i, r);
        while j > 0:
            C[j] += C[j - 1];
            j -= 1;
    return C[r]
```

GPT-3 175B completion:
A. to store the value of C[0]
B. to store the value of C[1]
C. to store the value of C[i]
D. to store the value of C[i - 1]

InstructGPT 175B completion:
The list C in this code is used to store the values of the binomial coefficient as the function iterates through the values of n and r. It is used to calculate the value of the binomial coefficient for a given value of n and r, and stores the result in the final return value of the function.

Results: GPT vs InstructGPT



Quality ratings of model outputs on a 1–7 scale (y-axis), for various model sizes (x-axis), on prompts submitted to InstructGPT models on our API. InstructGPT outputs are given much higher scores by our labelers than outputs from GPT-3 with a few-shot prompt and without, as well as models fine-tuned with supervised learning. We find similar results for prompts submitted to GPT-3 models on the API.

Alignment Process

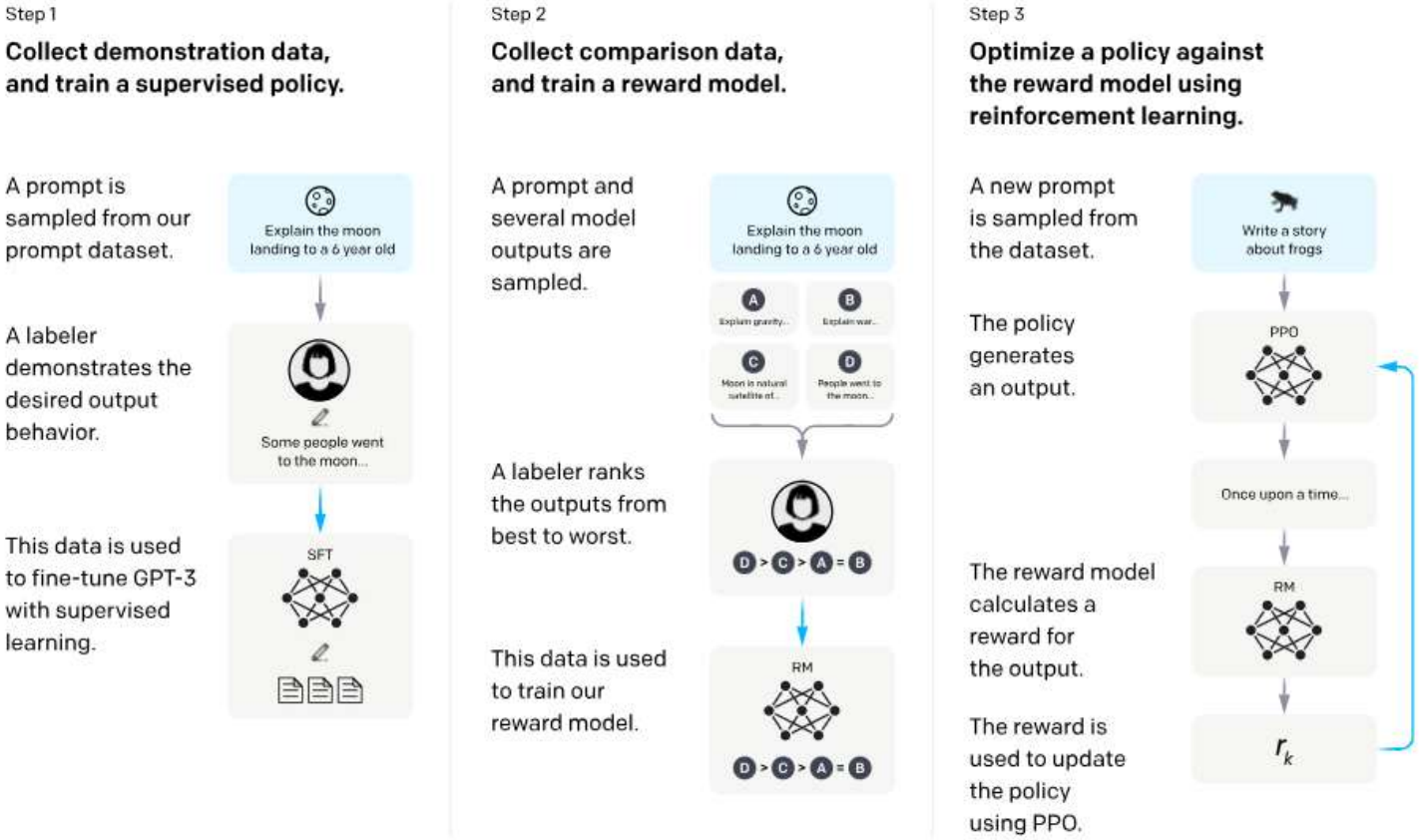
Customers send prompts to API

Labelers provide demonstrations of desired model behavior

Labelers rank several outputs from models

Ranking data is used to fine-tune GPT model

Methods



Step 1 is called Supervised Fine Tuning (SFT)

Methods, contd.

Human preferences provide reward signal for fine tuning

Process:

1. Provide human-written demonstrations (labels) on prompts. Train using SL.

2. Collect human-labeled comparisons. Train reward model (RM).

3. Use RM as reward function to fine tune GPT policy to maximize reward.

Reward Model

Begin with SFT model which learned from instructions, remove final layer

Train on dataset to take input and response, and output scalar reward

Present labelers with responses for each input and have them rank

Difference in rewards represents preference

Specifically, the loss function for the reward model is:

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

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 out of the pair of y_w and y_l, and D is the dataset of human comparisons.

Challenges

Models are still far from fully aligned, fully safe

- generate toxic, biased output
- make up facts (*hallucination*)
- generate violent content

Since models learn to follow instructions, they can learn bad behaviors from malevolent instructions

Can align to average labeler preference, exhibiting bias toward minority groups

Challenges, contd.

In this example which should have a simple answer,

> GPT-3 gets it wrong
> InstructGPT hedges

Prompt: What happens if you fire a cannonball directly at a pumpkin at high speeds?	
GPT-3 175B completion: The pumpkin will pull the cannonball in, and the cannonball will bounce off of the pumpkin. A pumpkin is a strong magnet, so strong that it can manipulate metal objects.	**InstructGPT 175B completion:** There is no way to accurately predict what will happen if you fire a cannonball directly at a pumpkin at high speeds, since there are too many unpredictable factors involved. Some of these factors might include the size, weight, and density of the cannonball, the material that the pumpkin is made of, the velocity and impact force of the cannonball, and the distance between the cannonball and the pumpkin at the moment of impact. Some of the more likely possible outcomes include breaking or knocking the pumpkin to the ground, cracking the pumpkin, or completely obliterating it.

Conclusions

RLHF is showing improvements to capture human preferences

Useful when the objective is complex

Can train a reward function to match human preferences

Models can first be tuned with instructions for carrying out the task (SFT)

Still room for improvement