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

NeurIPS 2022

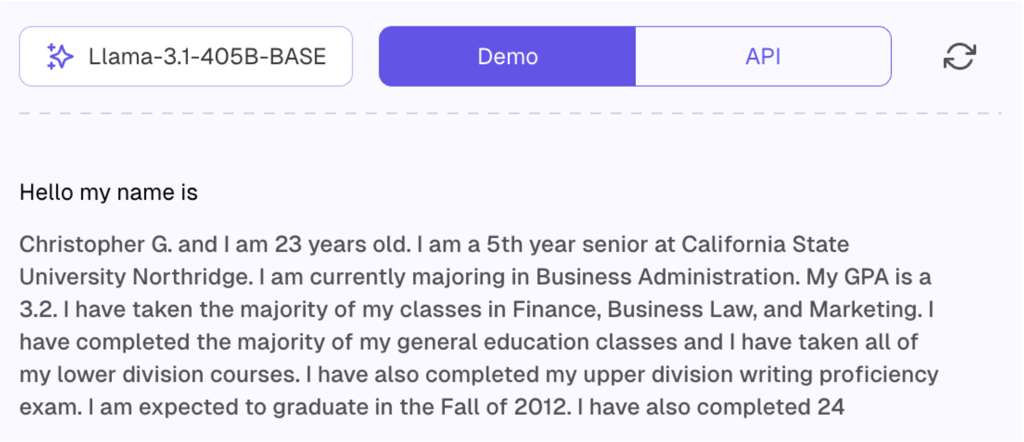
HUMANE Lab 최종현

2025.03.14 랩 세미나

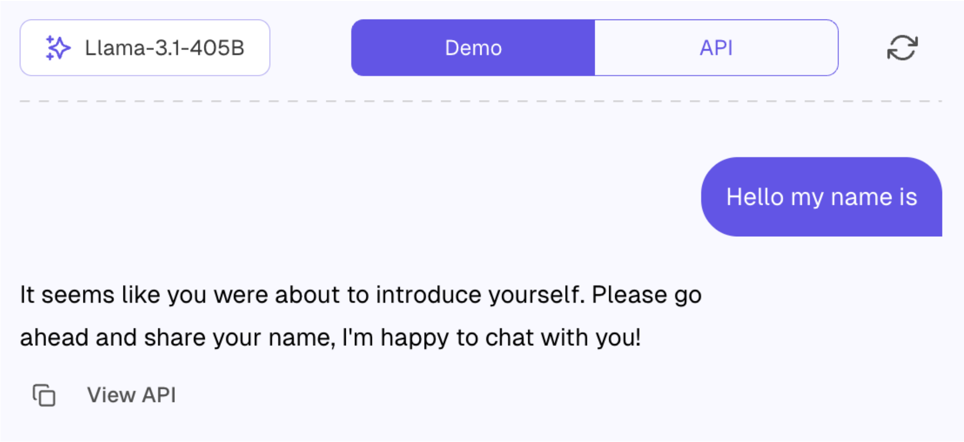
Background

- Large models like GPT-3 weren't following user instructions
- Could generate bias, toxic, or misleading content
- Wants models to be helpful, honest, harmless
- Introduces InstructGPT
    - a version of GPT-3 fine-tuned with human-feedback
    - improves instruction following capabilities
    - uses methods such as SFT, RM training, and RLHF

Why it’s needed

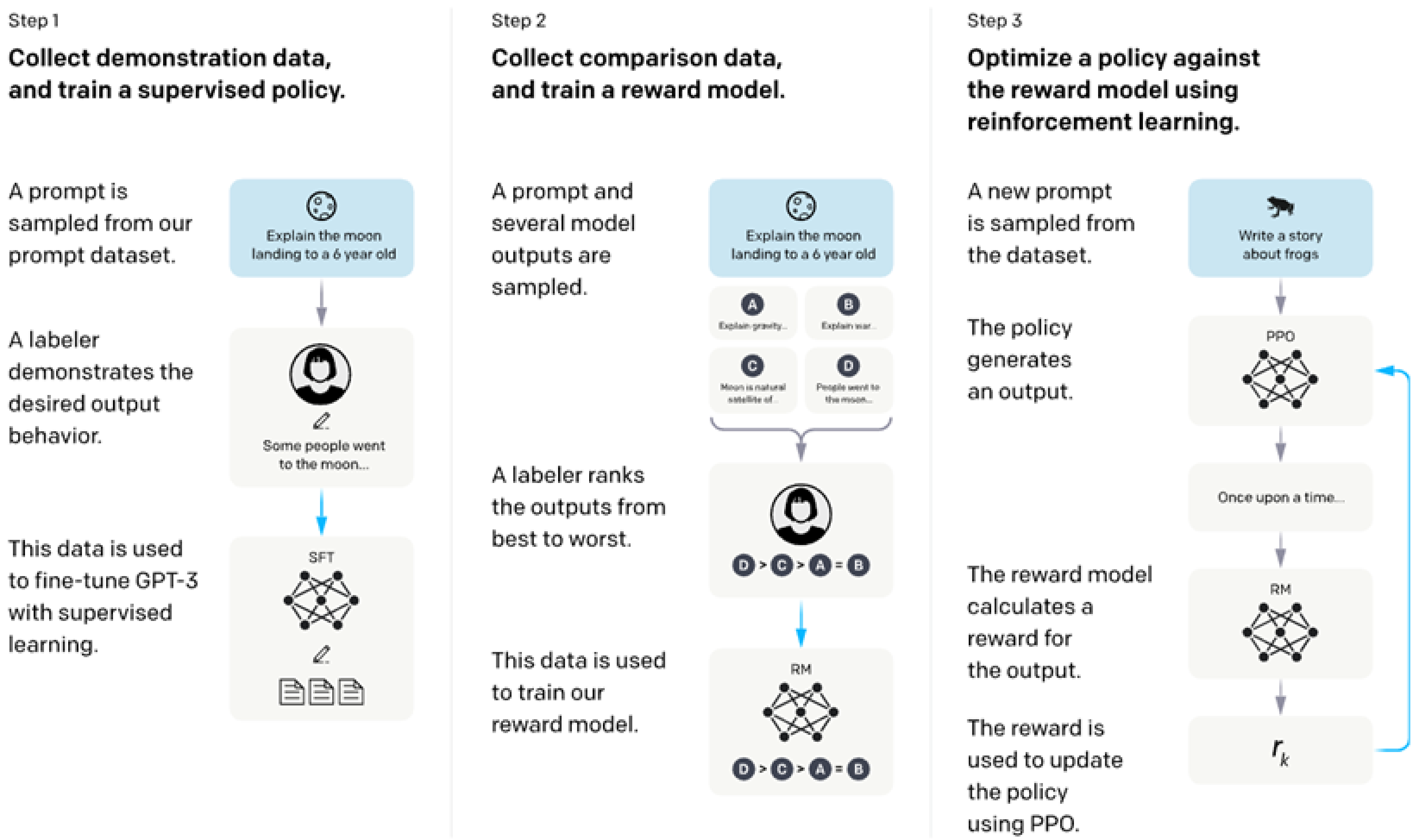


Llama 3.1 405B Base



Llama 3.1 405B

Method



Step 1
Collect demonstration data, and train a supervised policy.
A prompt is sampled from our prompt dataset.
Explain the moon landing to a 6 year old
A labeler demonstrates the desired output behavior.
Some people went to the moon...
This data is used to fine-tune GPT-3 with supervised learning.
SFT
Step 2
Collect comparison data, and train a reward model.
A prompt and several model outputs are sampled.
Explain the moon landing to a 6 year old
A
Explain gravity...
B
Explain war...
C
Moon is natural satellite of...
D
People went to the moon...
A labeler ranks the outputs from best to worst.
D > C > A = B
This data is used to train our reward model.
RM
D > C > A = B
Step 3
Optimize a policy against the reward model using reinforcement learning.
A new prompt is sampled from the dataset.
Write a story about frogs
The policy generates an output.
PPO
Once upon a time...
The reward model calculates a reward for the output.
RM
The reward is used to update the policy using PPO.
r_k

Dataset

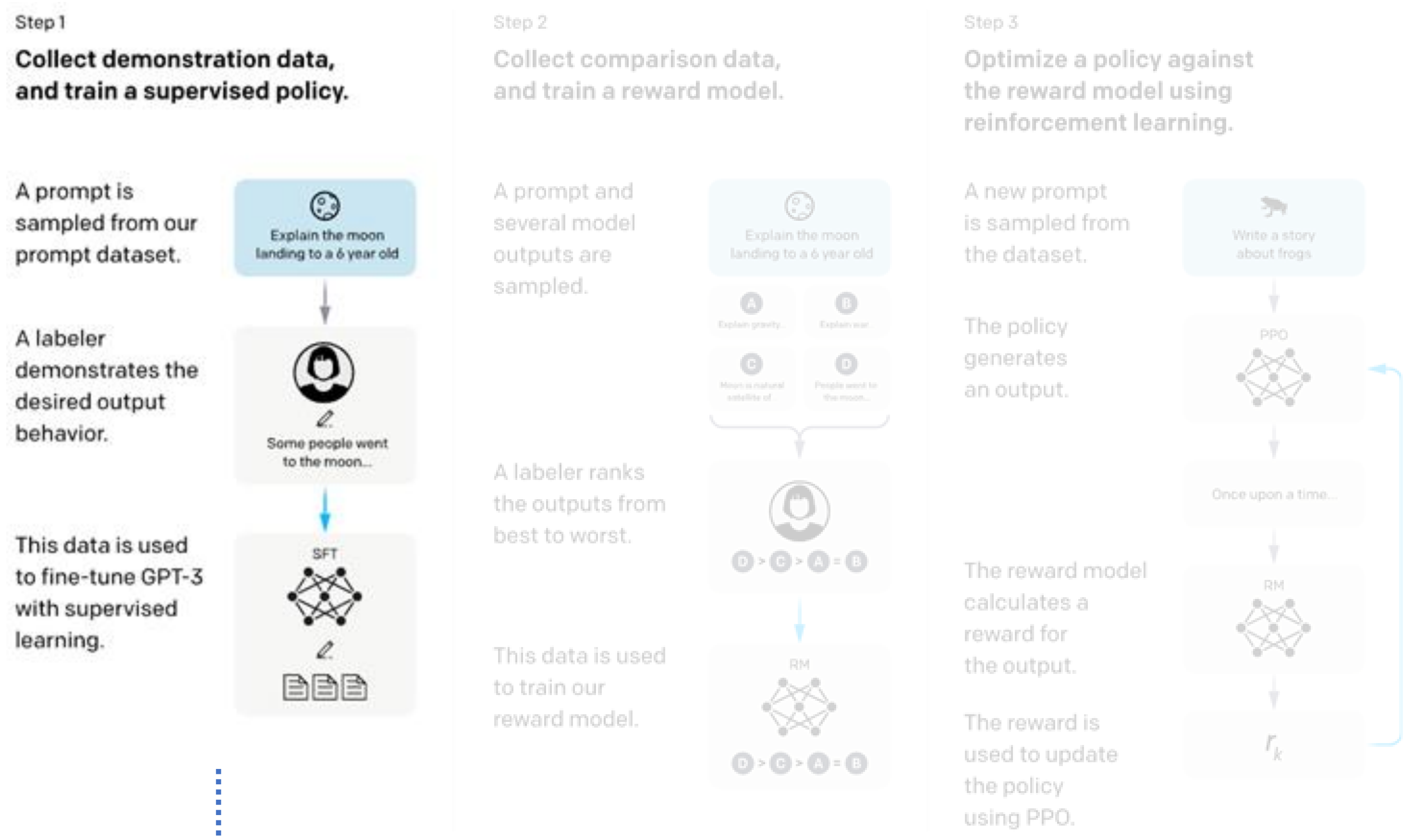
- Collect prompts from OpenAI API Playground and human labelers
- Human labelers prompts include:
  - Plain prompts
  - Few-shot prompts
  - User-based prompts
- Prompt filtering (e.g., personal information, deduplication)
- Split dataset based on user ID – train / validation / test
- Split again for SFT, RM, PPO dataset

Dataset

- SFT – 13k training data (API + labeler)
- RM – 33k training data (API + labeler)
- PPO – 31k training data (API)

SFT Data			RM Data			PPO Data		
split	source	size	split	source	size	split	source	size
train	labeler	11,295	train	labeler	6,623	train	customer	31,144
train	customer	1,430	train	customer	26,584	valid	customer	16,185
valid	labeler	1,550	valid	labeler	3,488			
valid	customer	103	valid	customer	14,399			

Step 1

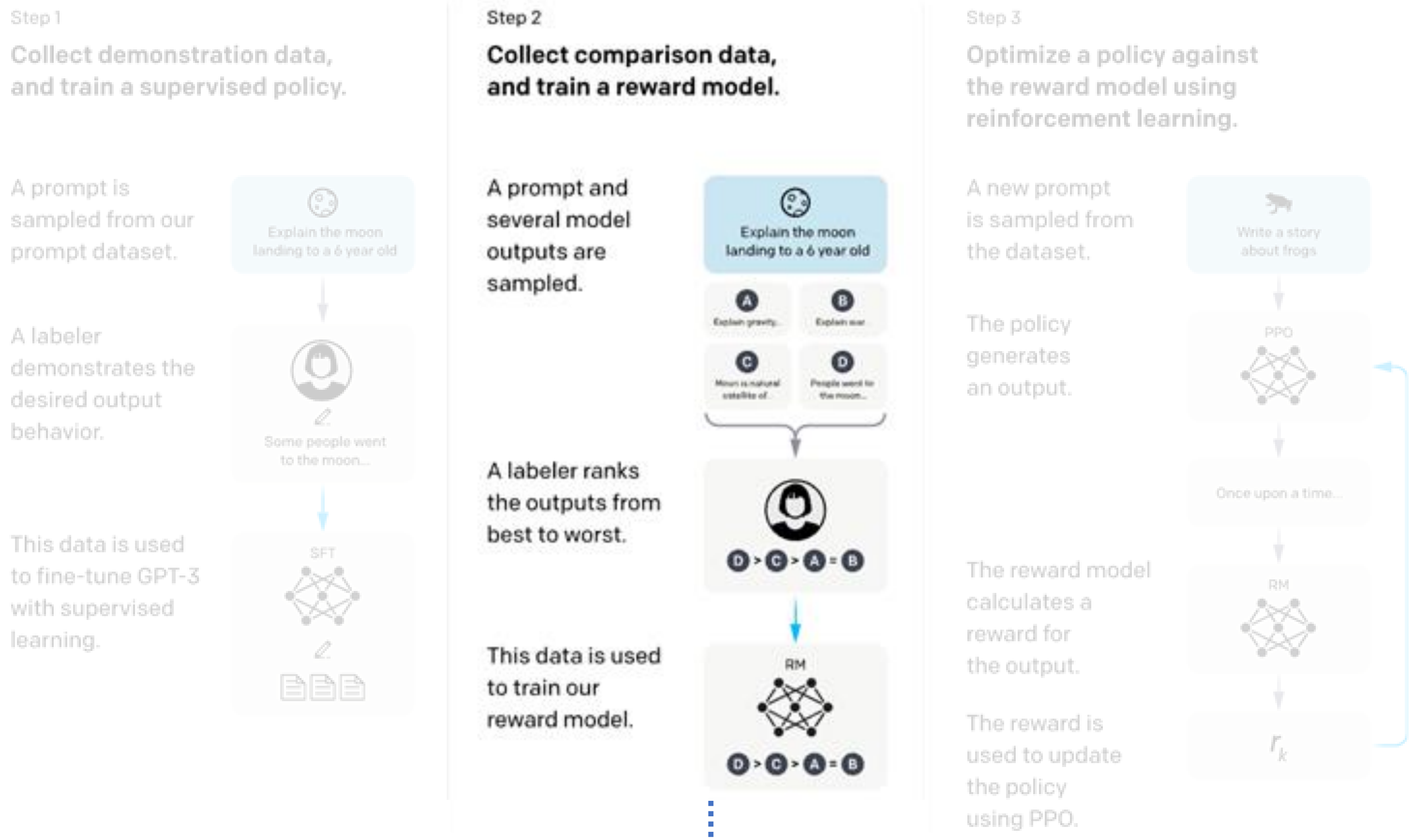


Supervised Fine-Tuning (SFT)

Step 1

- Fine-tune GPT-3 using supervised learning with SFT dataset
- Training configurations:
  - 16 epochs – overfits after 1 epoch but training for more epochs helps both RM score and human preference ratings
- Starting point for next step – reward model training

Step 2



Reward Model (RM) Training

Basics of RL - Terms

- Agent: model that makes decisions
- Environment: system that agent interacts with (user prompts)
- State: current situation
- Action: choices made by agent (generated text)
- Reward: feedback on how good or bad an action was
- Policy: strategy the agent uses to choose actions based on states
  - $\pi(a|s) = P(a_t = a|s_t = s)$
  - probability that agent selects action a at time step t, given its state s

Step 2

- Generate multiple candidate outputs per prompt (K=4 to K=9 outputs)
- Human annotators rank the outputs from best to worst
- Convert labeler rankings to pairwise preference data $\binom{K}{2}$
- Train the reward model to predict which response is preferred

Step 2

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

- Loss function for the reward model
- x: prompt
- y_w: human-preferred completion
- y_l: less preferred completion
- $r_\theta(x, y)$: human-preferred completion
- σ: sigmoid that converts scores into probs between 0 and 1

Step 2

Prompt: “Explain why exercise is good for health”

A: “Exercise improves health, and helps with weight management”

B: “It’s recommended by doctors”

C: “You should exercise daily for your health”

D: ″Exercising boots health by improving sleep”

Step 2

Prompt: “Explain why exercise is good for health”

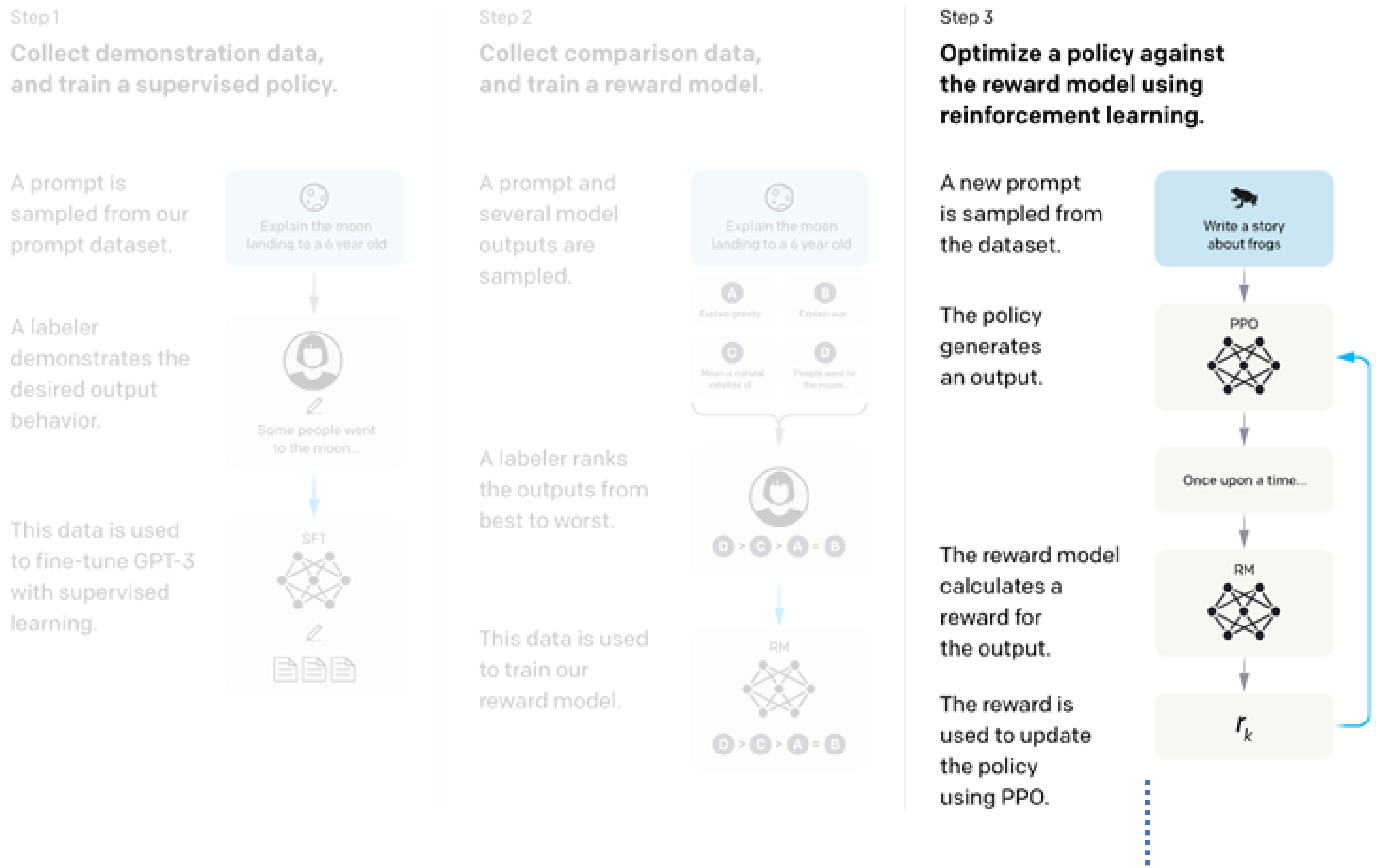
A: “Exercise improves health, and helps with weight management”

B: “It’s recommended by doctors”

C: “You should exercise daily for your health”

D: ”Exercising boots health by improving sleep”

Step 3



PPO Fine-Tuning

Step 3

- Two components
  - SFT model: current language model
  - RM model: outputs score indicating how well the generated text aligns with humans
- How it's done
  - Environment present prompt
  - Model generates a completion
  - Reward model evaluates the completion
  - Reward is given back to the policy → fine-tunes this with PPO

Step 3

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

- Objective function to optimize policy in RLHF
- Reward score from reward model (RM)
- KL divergence prevents the RL policy from deviating too much from the SFT model
- Ensures RL-trained policy to retain general knowledge and fluency from the original pretraining distribution (prevents forgetting)

Step 3

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

PPO

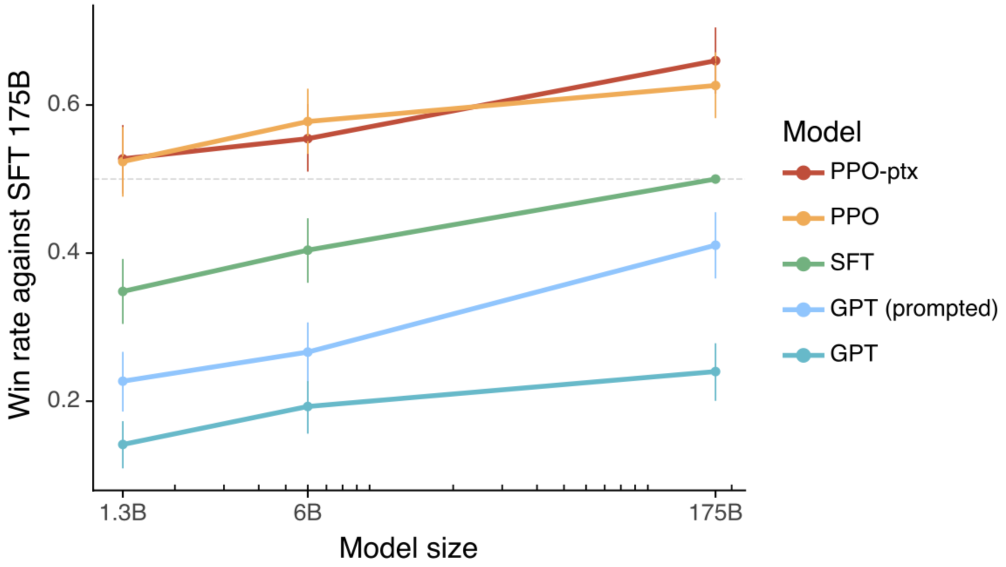
$$\text{objective}(\phi) = E_{(x,y)\sim D_{\pi_\phi^{\mathrm{RL}}}}\left[r_\theta(x,y) - \beta \log\left(\pi_\phi^{\mathrm{RL}}(y \mid x)/\pi^{\mathrm{SFT}}(y \mid x)\right)\right] +$$
$$\boxed{\gamma E_{x\sim D_{\text{pretrain}}}\left[\log(\pi_\phi^{\mathrm{RL}}(x))\right]}$$

PPO-ptx

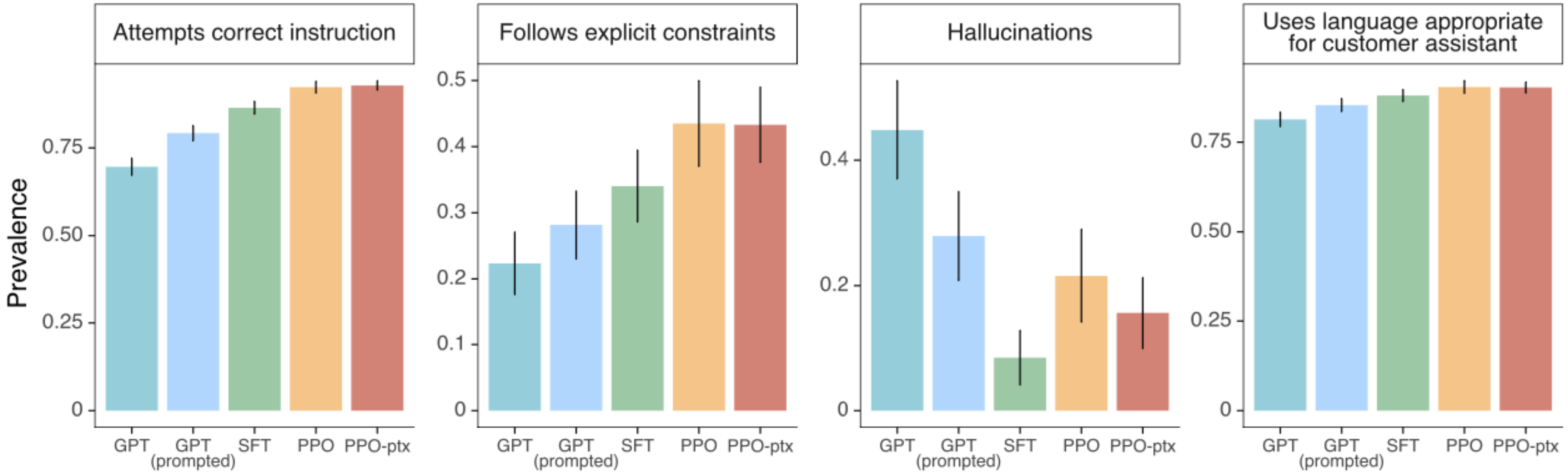
pretraining mix (ptx)

Result

- InstructGPT (1.3B) > GPT-3 (175B)
- PPO-ptx models are preferred (vs GPT-3)
- Scaling model doesn't fix misalignment
- Alignment matters more than just scaling



Result



- InstructGPT is preferred across instruction categories
- More reliable and easier to control than GPT-3
- Less likely to hallucinate

Conclusion

- InstructGPT model using SFT, RM, PPO
- Aligns with human preferences
- Alignment is more important than just scaling
- RLHF can make models more helpful, truthful, and safe
- Foundation of modern instruction-following AI models

Q&A