CS 333: Natural Language Processing

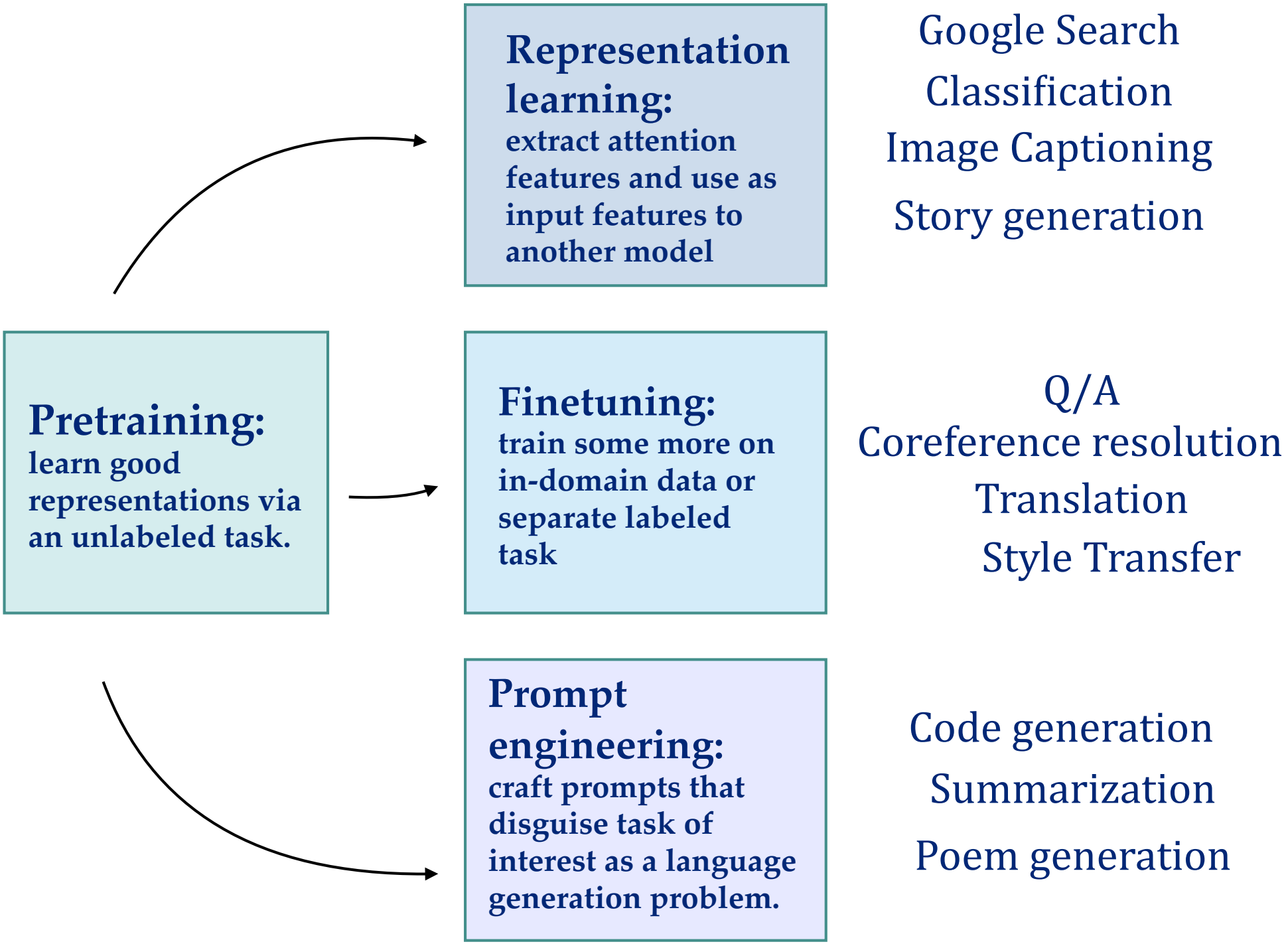
Fall 2025

Prof. Carolyn Anderson
Wellesley College

Reminders

- HW 8 is due on Monday, 11/24 - extended to Wed.
- My next help hours: Monday 4-5:30

The Deep Learning Pipeline


Pretraining:
learn good representations via an unlabeled task.
Representation learning:
extract attention features and use as input features to another model
Finetuning:
train some more on in-domain data or separate labeled task
Prompt engineering:
craft prompts that disguise task of interest as a language generation problem.
Google Search
Classification
Image Captioning
Story generation
Q/A
Coreference resolution
Translation
Style Transfer
Code generation
Summarization
Poem generation

Fine-tuning = Further training

Fine-tuning:
train some more on in-domain data or separate labeled task

Simple Fine-tuning

Just train on more (labeled) data

Proxy Tuning

Fine-tune a smaller model and use it steer a larger model

Reinforcement Learning

Tune the model with a reward function

Alignment through Reinforcement Learning

Aligning Models with Human Preferences

Users tend to have preferences about generated text that go beyond its statistical frequency.

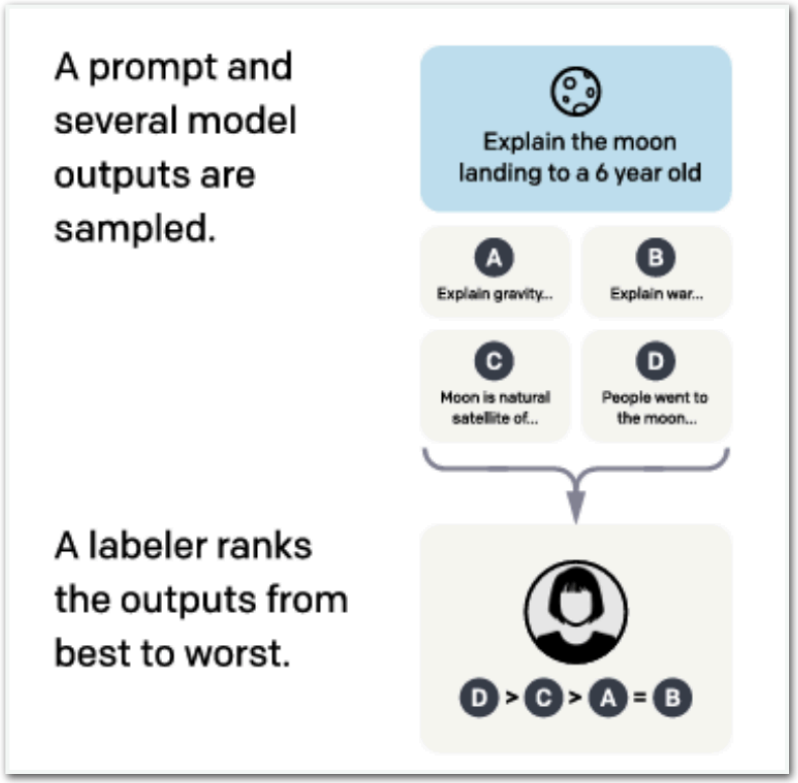
Companies also have preferences for how their models respond.

How can we align model behavior with these preferences?

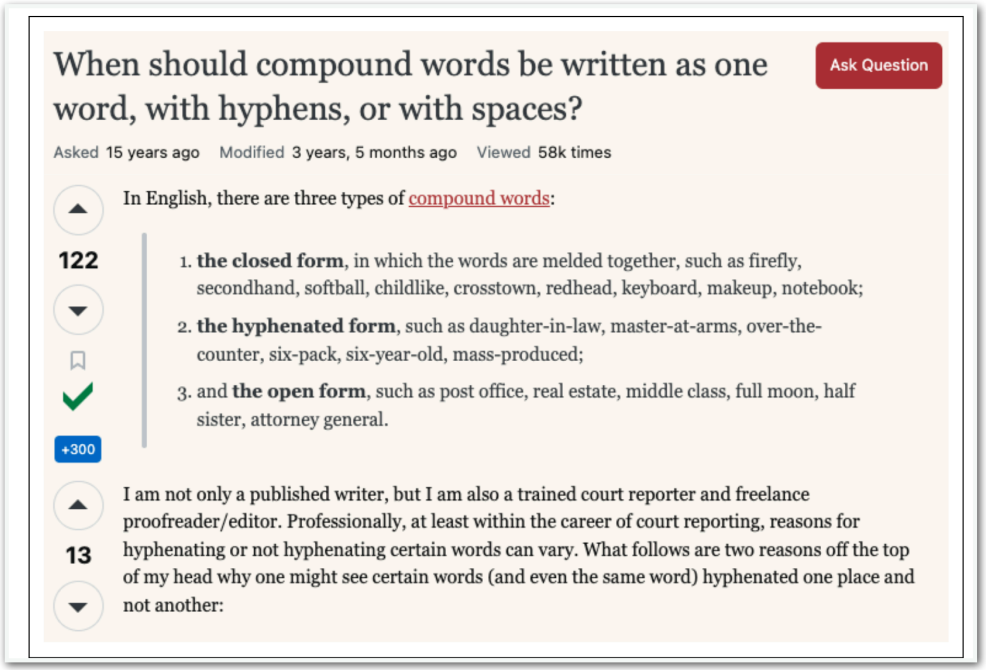
Mark Dredze
@mdredze

Reinforcement Learning from Human Feedback

Step 1: Obtain data on human preferences

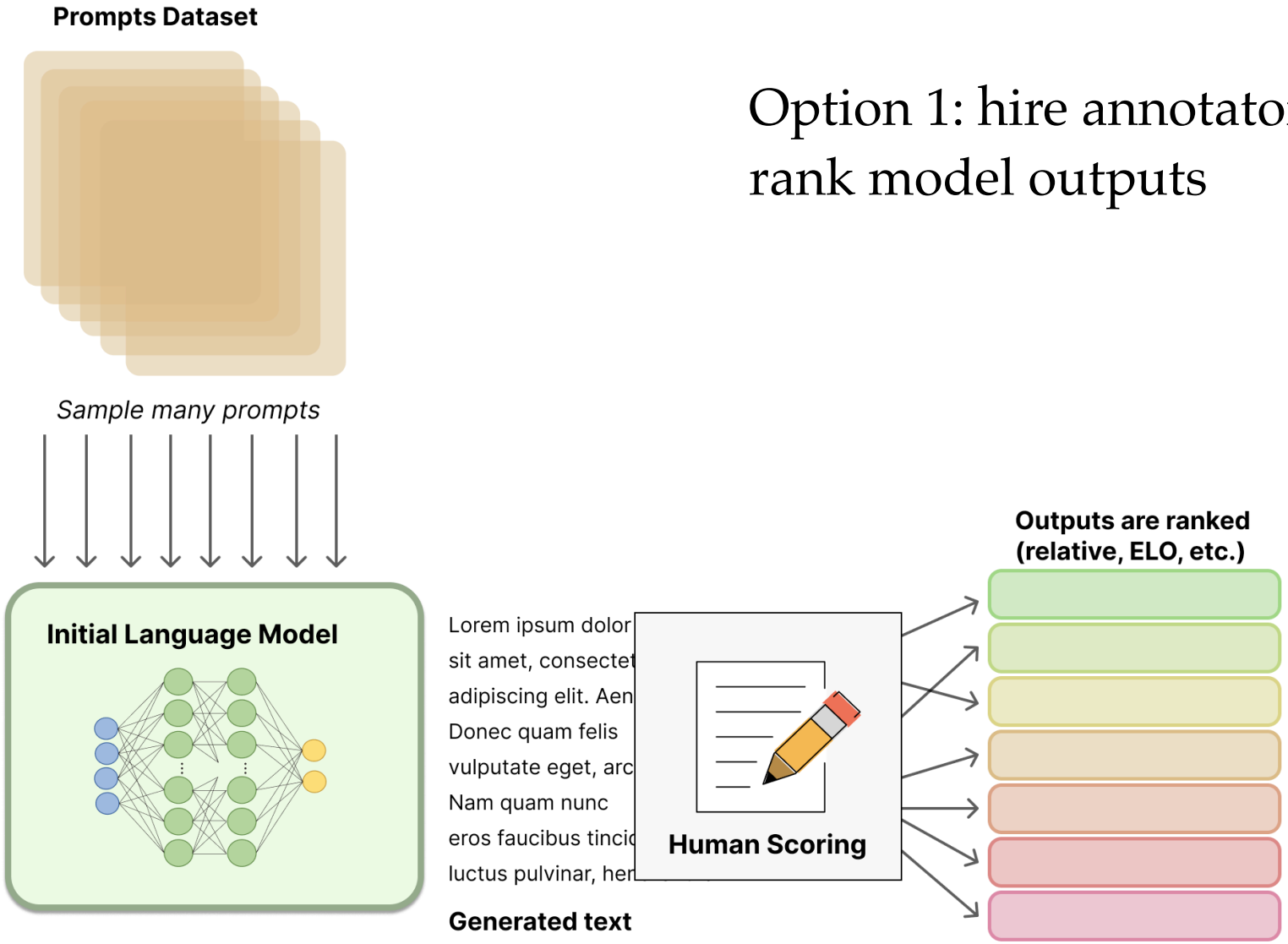


Option 1: hire annotators to rank model outputs



Option 2: find natural datasets of preference data

Step 1: Obtain data on human preferences



Option 1: hire annotators to rank model outputs

https://huggingface.co/blog/rlhf

Potential Impact on Annotators

Artificial intelligence (AI)

This article is more than **2 years old**

'It's destroyed me completely': Kenyan moderators decry toll of training of AI models

Employees describe the psychological trauma of reading and viewing graphic content, low pay and abrupt dismissals

Niamh Rowe

Wed 2 Aug 2023 11.00 EDT

Share

The images pop up in Mophat Okinyi's mind when he's alone, or when he's about to sleep.

Okinyi, a former content moderator for Open AI's ChatGPT in Nairobi, Kenya, is one of four people in that role who have filed a petition to the Kenyan government calling for an investigation into what they describe as exploitative conditions for contractors reviewing the content that powers artificial intelligence programs.

The 51 moderators in Nairobi working on Sama's OpenAI account were tasked with reviewing texts, and some images, many depicting graphic scenes of violence, self-harm, murder, rape, necrophilia, child abuse, bestiality and incest, the petitioners say.

The moderators say they weren't adequately warned about the brutality of some of the text and images they would be tasked with reviewing, and were offered no or inadequate psychological support. Workers were paid between $1.46 and $3.74 an hour, according to a Sama spokesperson.

Step 2: Model Preferences

Bradley-Terry Model:

Given two outputs o_i and o_j with associated scores z_i and z_j:

$$P(o_i > o_j \mid x) = \sigma(z_i - z_j)$$

logistic sigmoid of the difference in the scores

Step 3: Learn a Reward Function

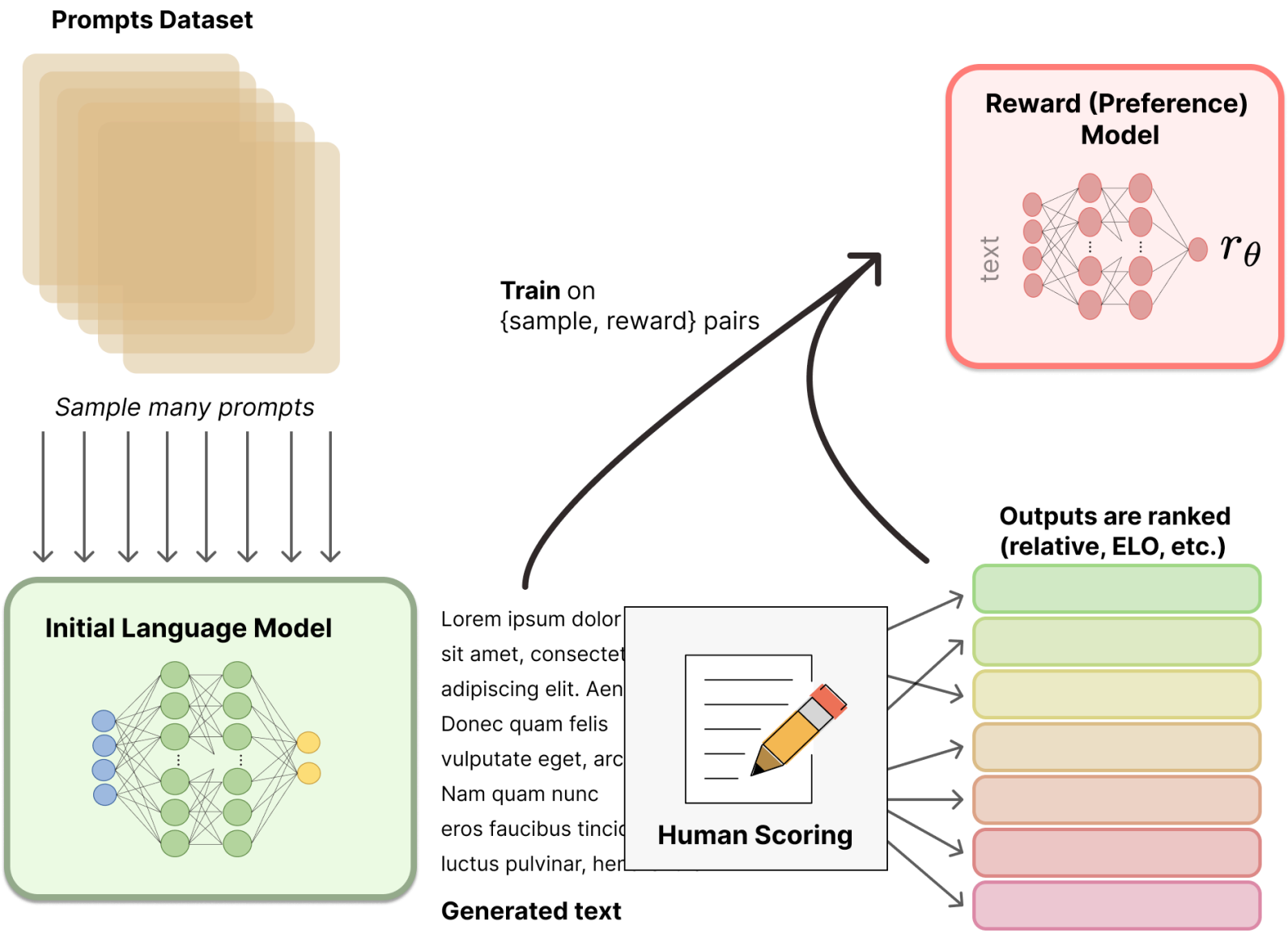
Goal: Learn a model that predicts the rewards of different inputs.

$$P(o_i \succ o_j | x) = \sigma(z_i - z_j)$$
$$= \sigma(r(o_i, x), r(o_j, x))$$

where r is the reward assigned to each input/output pair

In principle, the reward model could be any kind of classification model. In practice, we often fine-tune a pretrained LLM.

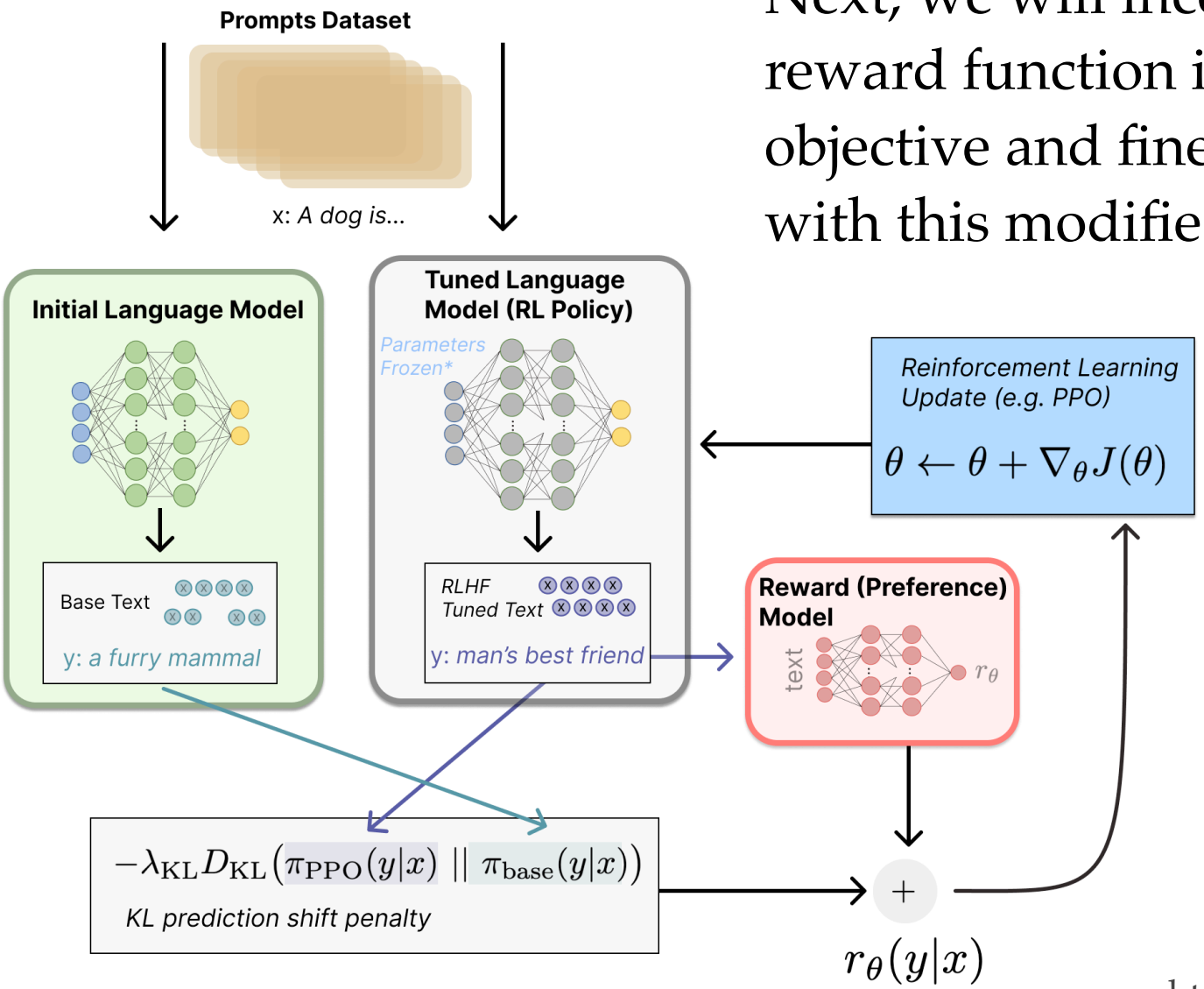
Step 3: Learn a Reward Function



https://huggingface.co/blog/rlhf

Step 4: Use Reward Function to Fine-Tune

Next, we will incorporate the reward function into the training objective and fine-tune our model with this modified objective.



https://huggingface.co/blog/rlhf

Reinforcement Learning

Policy Gradient Learning

- There are various RL methods
- Maybe the most common nowadays are ***policy gradient methods***
- Maximize some performance measure via gradient **ascent**
- The most common performance measure is the value of the start state:

$$J(\theta) = v_{\pi_\theta}(s_0)$$

- So during learning we want to find θ such that

$$\theta = \arg\max_\theta J(\theta) = \arg\max_\theta v_{\pi_\theta}(s_0)$$

- One of the simplest algorithms to do this is REINFORCE [Williams 1992]

Reinforcement Learning

Proximal Policy Optimization (PPO)

- PPO [Schulman et al. 2017] is a contemporary RL algorithm
- The most common choice for LLMs
- Empirically provides several advantages over REINFORCE
  - Increased stability and reliability, reduction in gradient estimates variance, and faster learning
  - But, has more hyper-parameters and requires to estimate the value function $v_\pi(s)$
  - Recent work shows REINFORCE remains competitive [Ahmadian et al. 2024], when used well

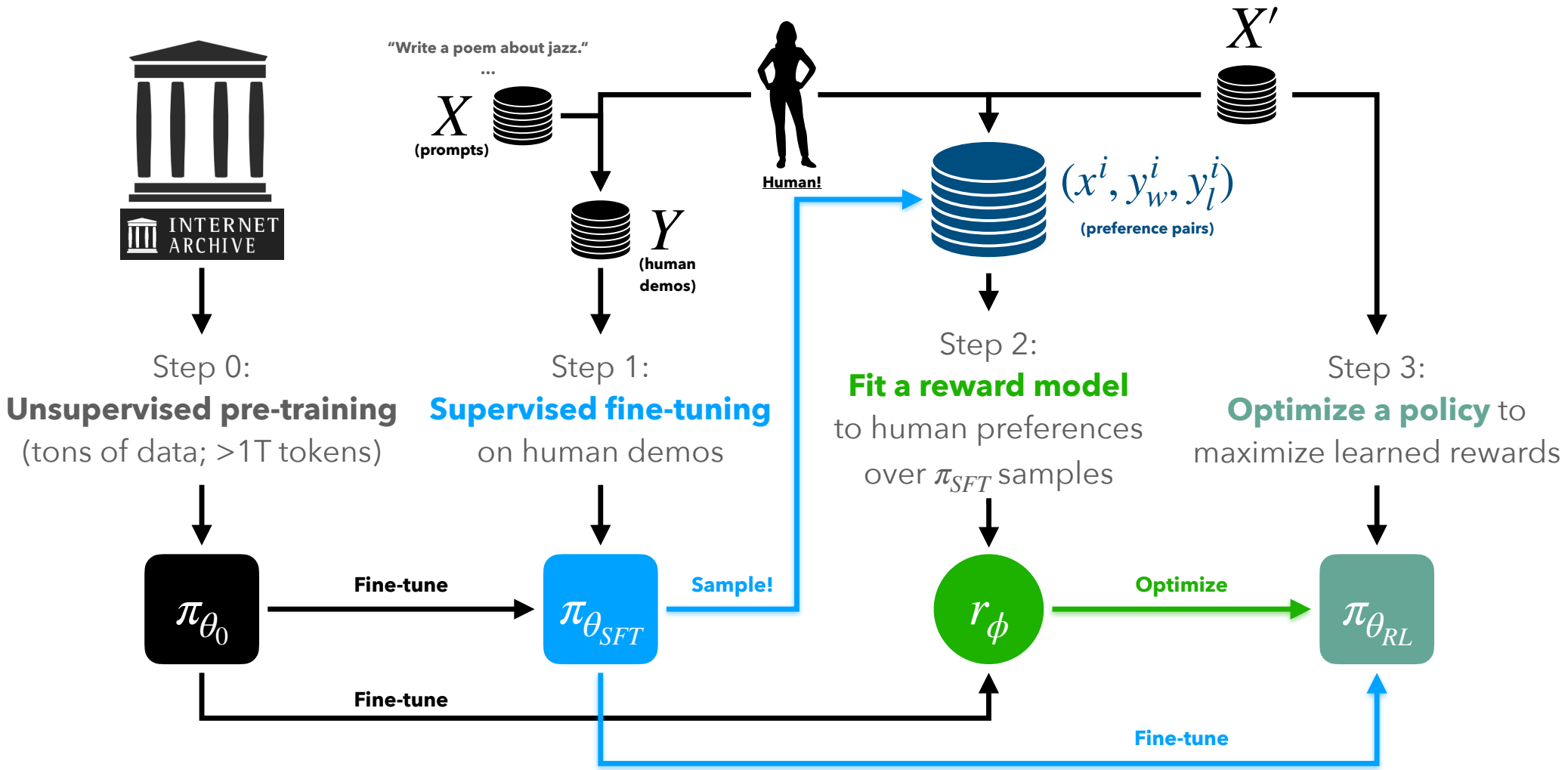
Step 4: Use Reward Function to Fine-Tune

Problem: LLM can drift too far and forget what it learned in pretraining.

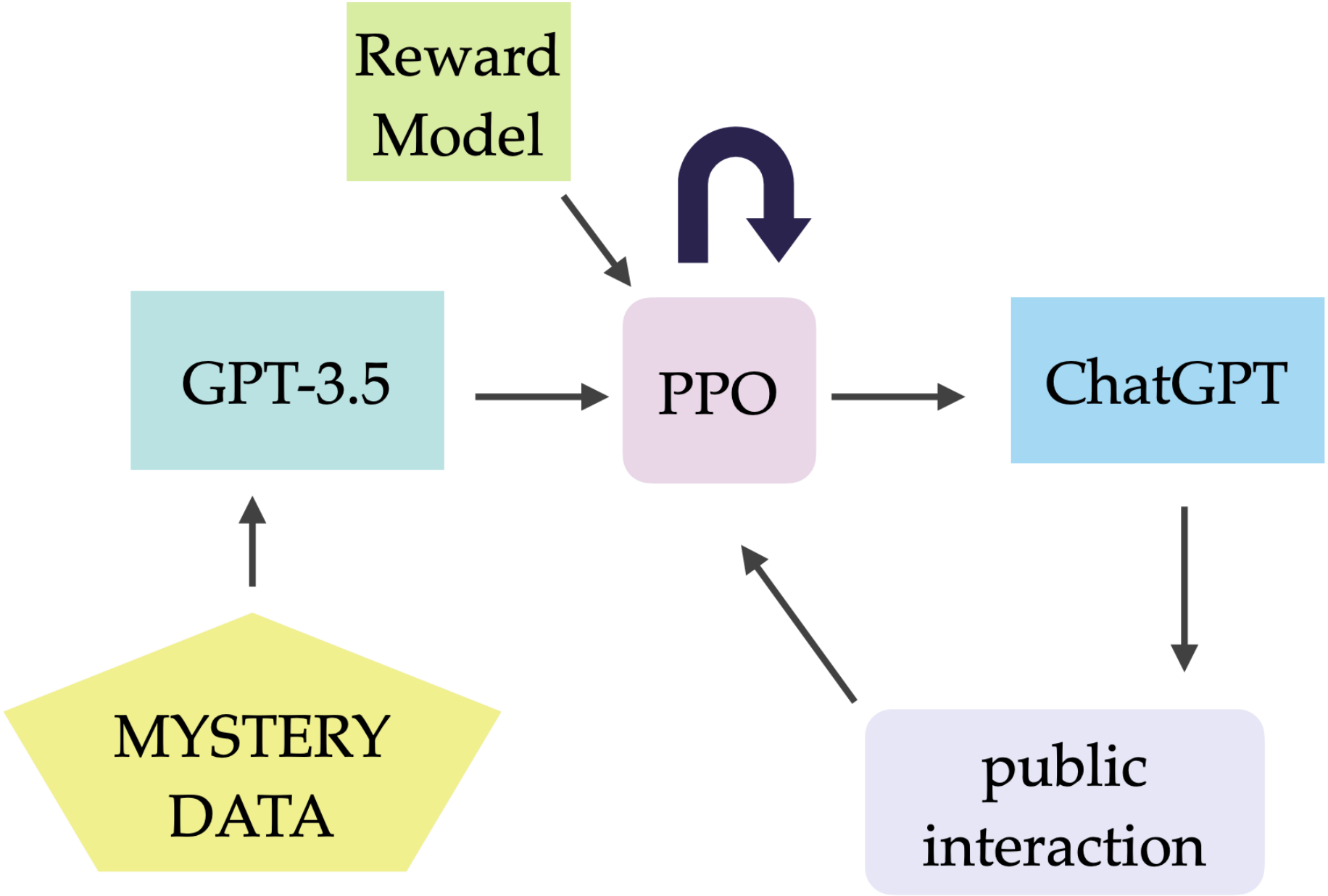
Solution: Incorporate the **Kullback-Leibler divergence** between the pretrained model and the reward-tuned model into the reward function (i.e. reward models that stay fairly similar to the original model).

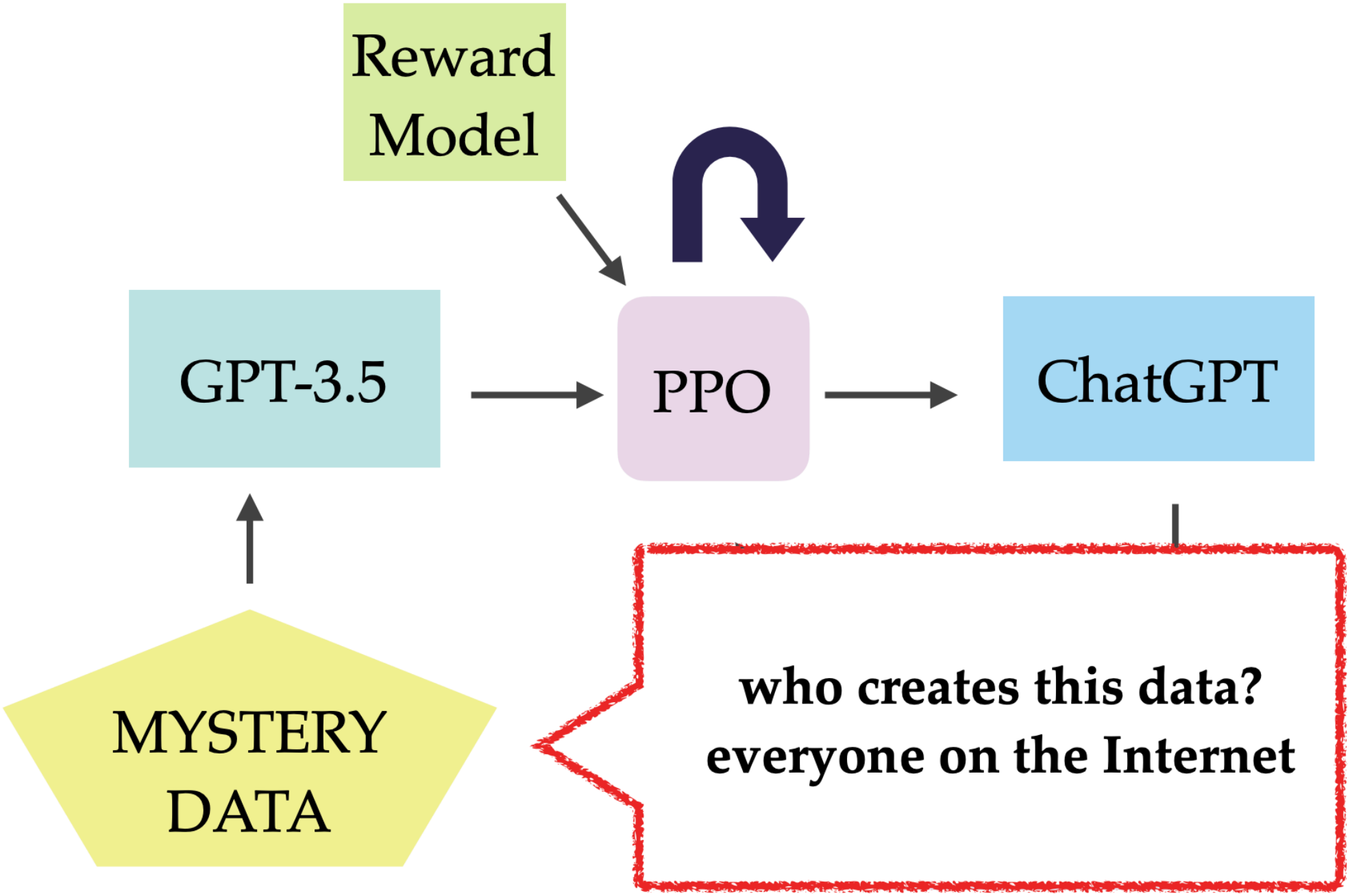
KL divergence: measures the distance between two probability distributions.

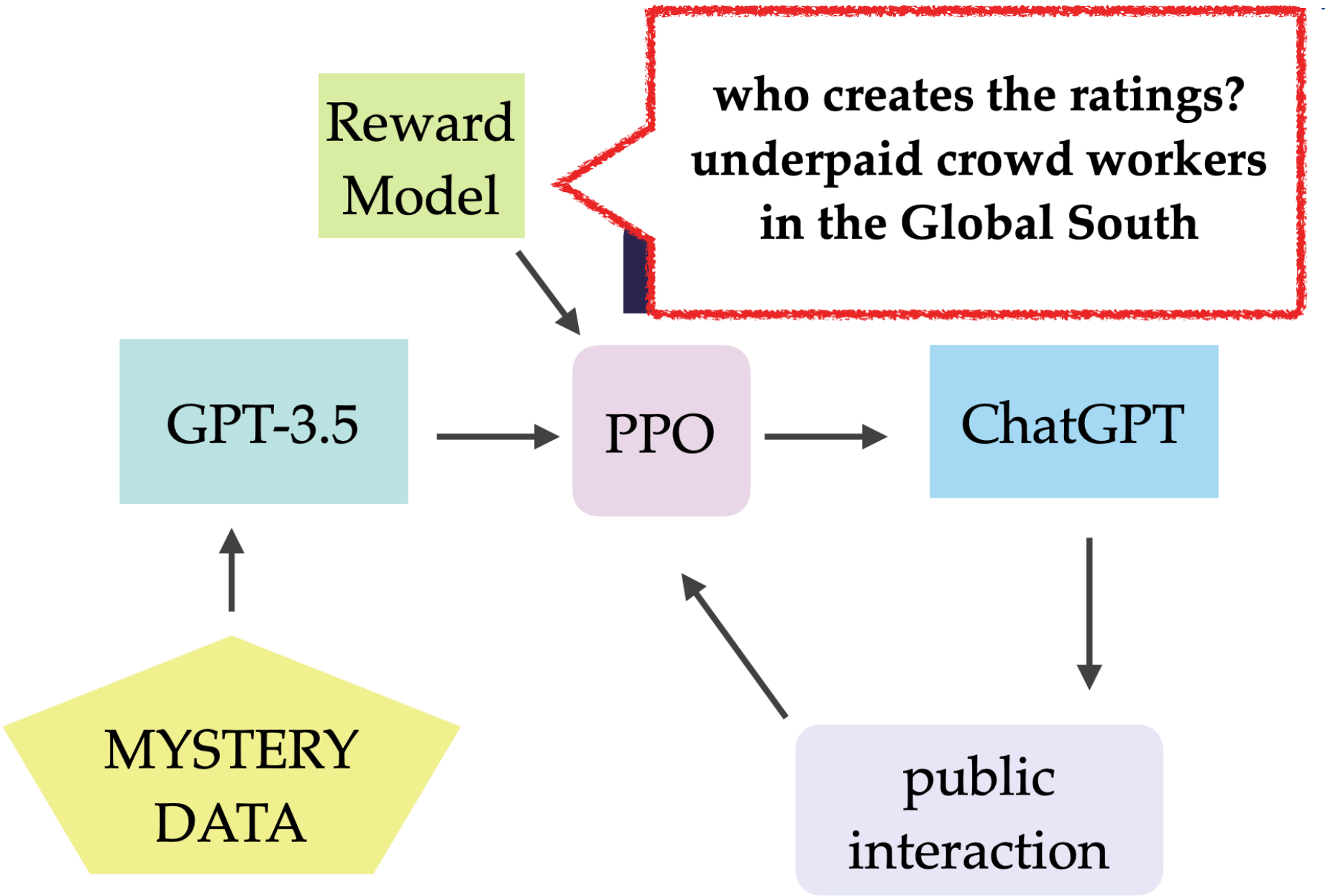
Reinforcement Learning from Human Feedback Process

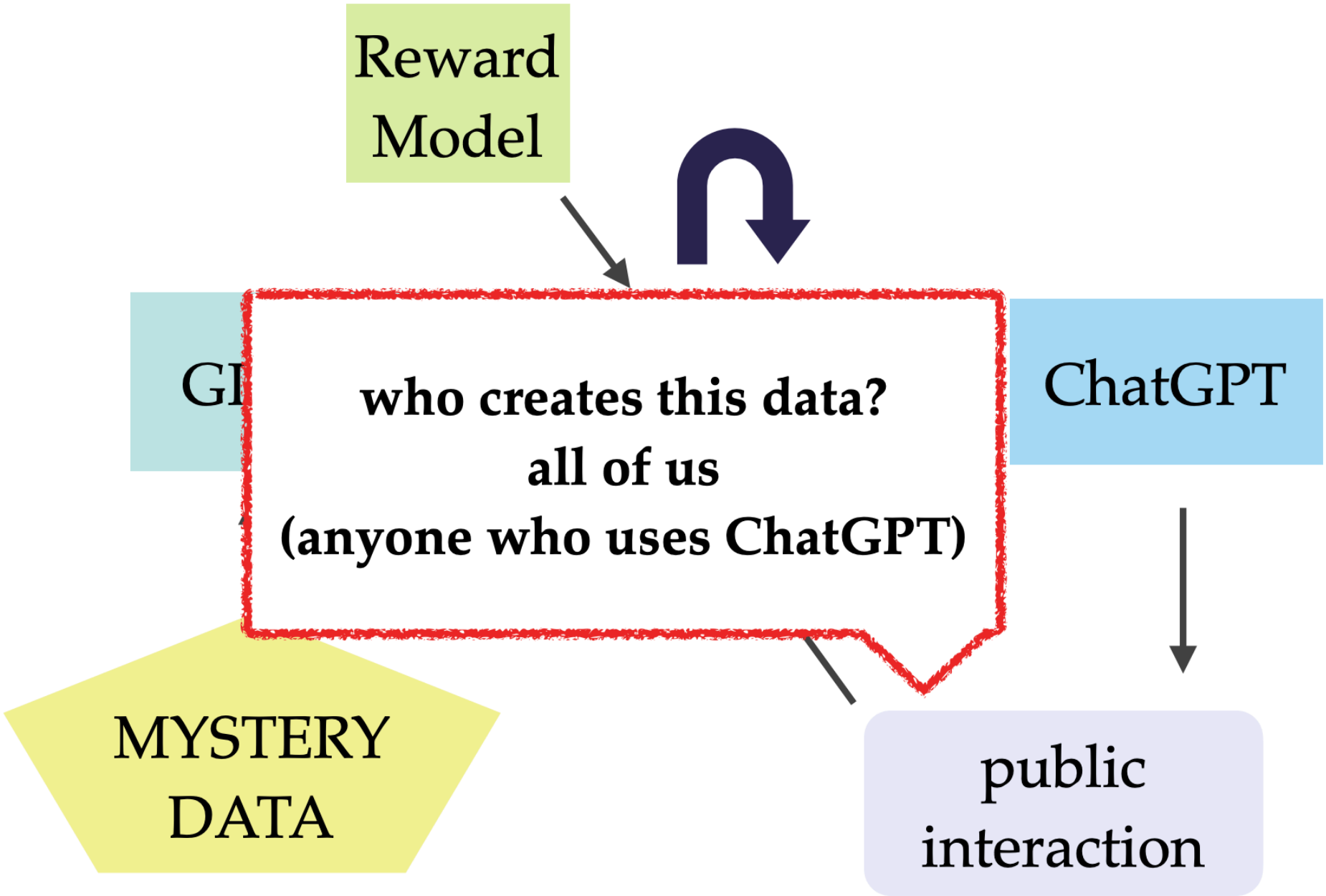


[Figure from Eric Mitchell]


Reward Model
GPT-3.5
PPO
ChatGPT
MYSTERY DATA
public interaction


Reward Model
GPT-3.5
PPO
ChatGPT
MYSTERY DATA
who creates this data?
everyone on the Internet


Reward Model
who creates the ratings? underpaid crowd workers in the Global South
GPT-3.5
PPO
ChatGPT
MYSTERY DATA
public interaction


Reward Model
GI
who creates this data?
all of us
(anyone who uses ChatGPT)
ChatGPT
MYSTERY DATA
public interaction

Reinforcement Learning from Human Feedback

Takeaways

- A pretty complex process with many tricky implementation details
- Hard to get it to work — both reward modeling and RL
- Very costly — both compute and data annotation
- But, works really well
- There are alternatives that involve building preference signals into fine-tuning directly, rather than training a separate reward model (Direct Policy Optimization).
- Basically all SOTA models at this point go through some kind of reinforcement learning for preferences (DPO or PPO).

Prompt Engineering

Few-Shot Prompting

Often called "in context learning" now.

Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.

```
1  Translate English to French:      ← task description
2  cheese =>                         ← prompt
```

One-shot

In addition to the task description, the model sees a single example of the task. No gradient updates are performed.

```
1  Translate English to French:      ← task description
2  sea otter => loutre de mer        ← example
3  cheese =>                         ← prompt
```

Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

```
1  Translate English to French:      ← task description
2  sea otter => loutre de mer        ← examples
3  peppermint => menthe poivrée      ←
4  plush girafe => girafe peluche    ←
5  cheese =>                         ← prompt
```

Fine-tuning

The model is trained via repeated gradient updates using a large corpus of example tasks.

Few-Shot Prompting

Few-shot prompting almost always helps performance.

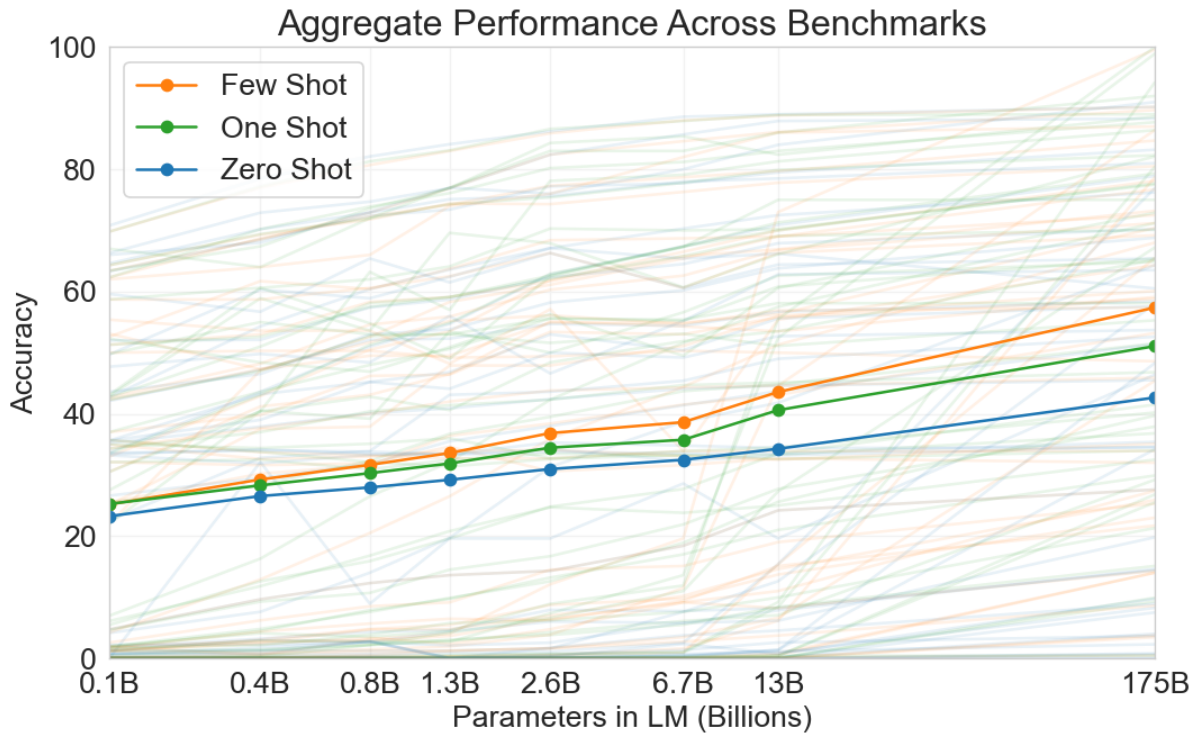


Figure 1.3: Aggregate performance for all 42 accuracy-denominated benchmarks While zero-shot performance improves steadily with model size, few-shot performance increases more rapidly, demonstrating that larger models are more proficient at in-context learning. See Figure 3.8 for a more detailed analysis on SuperGLUE, a standard NLP benchmark suite.

[Brown et al. 2020]

Few-Shot Prompting

However, it can be highly sensitive to the choice of examples, their ordering, and the format of the prompt.

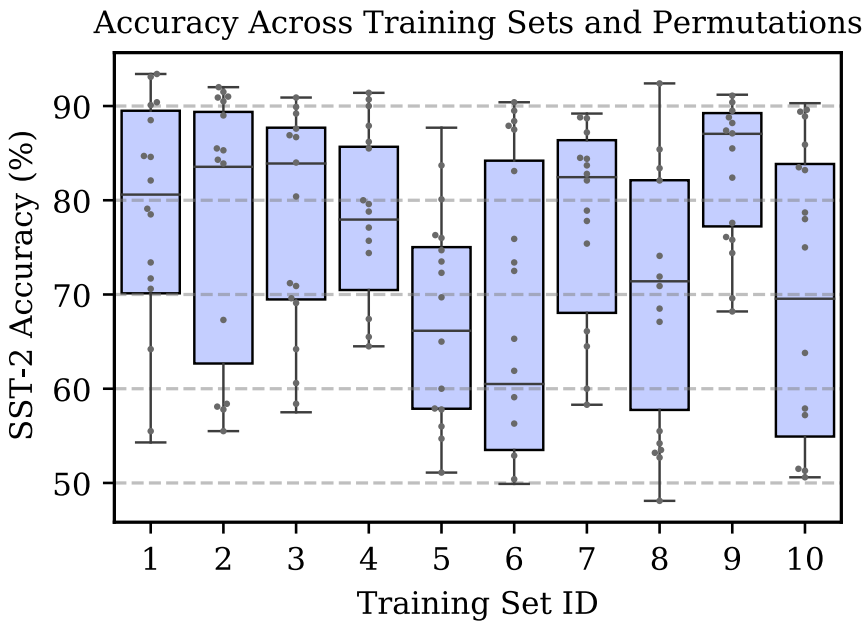


Figure 2. There is high variance in GPT-3's accuracy as we change the prompt's **training examples**, as well as the **permutation** of the examples. Here, we select ten different sets of four SST-2 training examples. For each set of examples, we vary their permutation and plot GPT-3 2.7B's accuracy for each permutation (and its quartiles).

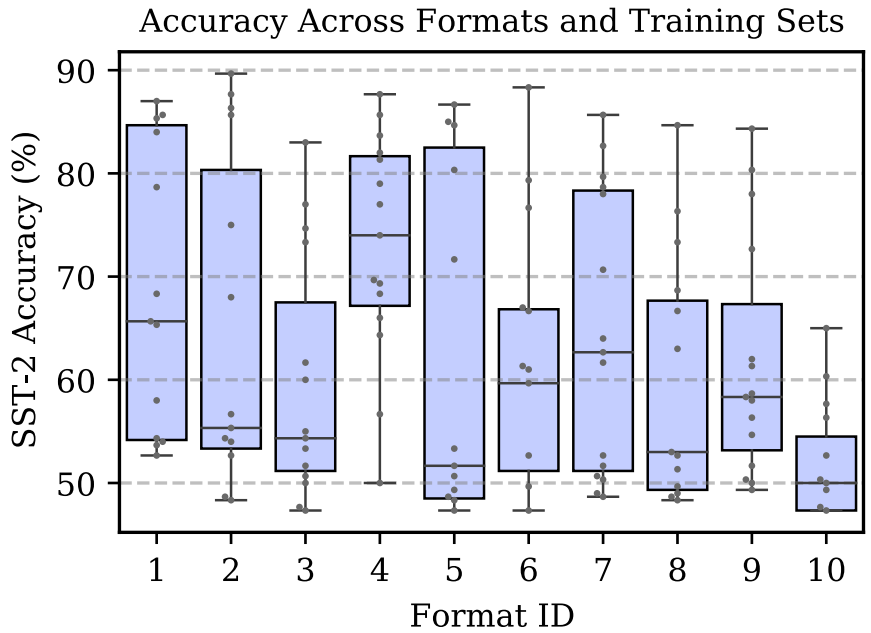


Figure 3. There is high variance in GPT-3's accuracy as we change the **prompt format**. In this figure, we use ten different prompt formats for SST-2. For each format, we plot GPT-3 2.7B's accuracy for different sets of four training examples, along with the quartiles.

[Zhao et al. 2021]

Few-Shot Prompting

Also, it seems not to work well for vision-language models.

GlyphPattern: An Abstract Pattern Recognition for Vision-Language Models

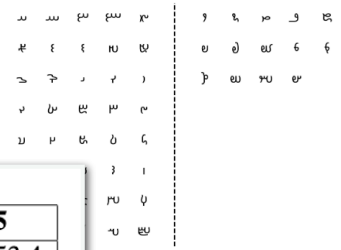
Zixuan Wu
Northeastern University
zi.wu@northeastern.edu

Yoolim Kim
Wellesley College
ykim6@wellesley.edu

Carolyn Jane Anderson
Wellesley College
carolyn.anderson@wellesley.edu

Abstract

Vision-Language Models (VLMs) have made rapid progress in reasoning across visual and textual data. While VLMs perform well on vision tasks that they are trained on, our results highlight key challenges in abstract pat-



rs on the right side in the image are
s that contain a loop or circular

Example GlyphPattern item in the le
. The writing system is Avestan.

Model	0-shot	1	3	5
Gemini-1.5	47.1	52.0	**53.7**	53.4
Gemini-1.5 CoT	46.6	-	39.3	-
GPT-4o	**55.6**	53.4	55.4	52.9
GPT-4o CoT	50.0	-	49.1	-
Idefics2	**31.5**	30.1	31.2	30.0
Idefics3	**34.3**	33.5	32.7	33.3
InstructBLIP	24.1	-	-	-
LLaVA-NeXT	**32.9**	27.0	28.0	26.8
Molmo-O	42.6	-	-	-
Molmo-D	**46.0**	-	-	-

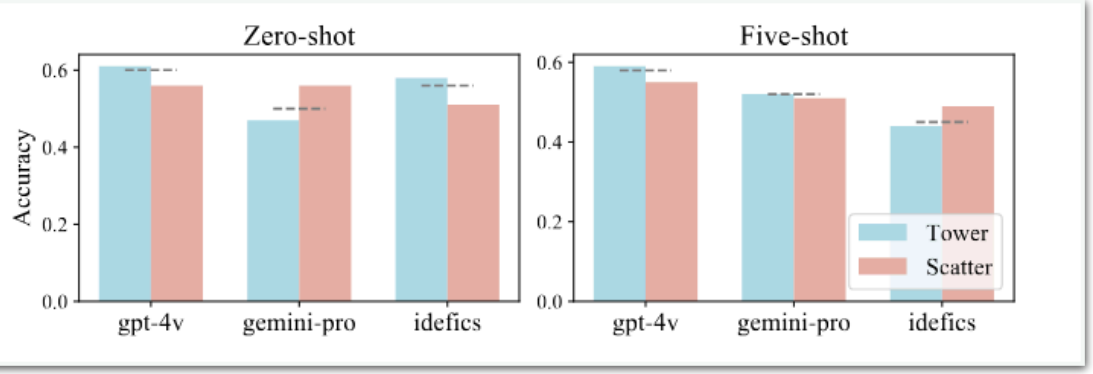
Table 1: Zero-shot and few-shot accuracy by model.

A Surprising Failure?
Multimodal LLMs and the NLVR Challenge

Anne Wu, Kianté Brantley, and Yoav Artzi
Department of Computer Science and Cornell Tech, Cornell University
{annewu, yoav}@cs.cornell.edu, kdb82@cornell.edu

Abstract

This study evaluates three state-of-the-art MLLMs — GPT-4V, Gemini Pro, and the open-source model IDEFICS — on the compositional natural language vision reasoning task NLVR. Given a human-written sentence paired with a synthetic image, this task requires the model to determine the truth value of the sentence with respect to the image. Despite the strong performance demonstrated by these models, we observe they perform poorly on NLVR, which was constructed to require compositional and spatial reasoning, and to be robust for semantic and systematic biases.



Chain-of-Thought Reasoning

One idea is to make the model generate reasoning before an answer. This guarantees that the answer is conditioned on the reasoning.

Some people think this could improve the quality of the answer. However, other work has shown that the answer is not always consistent with the given reasoning.

Standard Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The answer is 27. ✗

Chain-of-Thought Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11. The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The cafeteria had 23 apples originally. They used 20 to make lunch. So they had 23 - 20 = 3. They bought 6 more apples, so they have 3 + 6 = 9. The answer is 9. ✓

[Wei et al. 2022]

(a) Few-shot

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?
A: The answer is 11.

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A:

(Output) The answer is 8. ✗

(b) Few-shot-CoT

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?
A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11. The answer is 11.

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A:

(Output) The juggler can juggle 16 balls. Half of the balls are golf balls. So there are 16 / 2 = 8 golf balls. Half of the golf balls are blue. So there are 8 / 2 = 4 blue golf balls. The answer is 4. ✓

(c) Zero-shot

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A: The answer (arabic numerals) is

(Output) 8 ✗

(d) Zero-shot-CoT (Ours)

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A: ***Let's think step by step.***

(Output) There are 16 balls in total. Half of the balls are golf balls. That means that there are 8 golf balls. Half of the golf balls are blue. That means that there are 4 blue golf balls. ✓

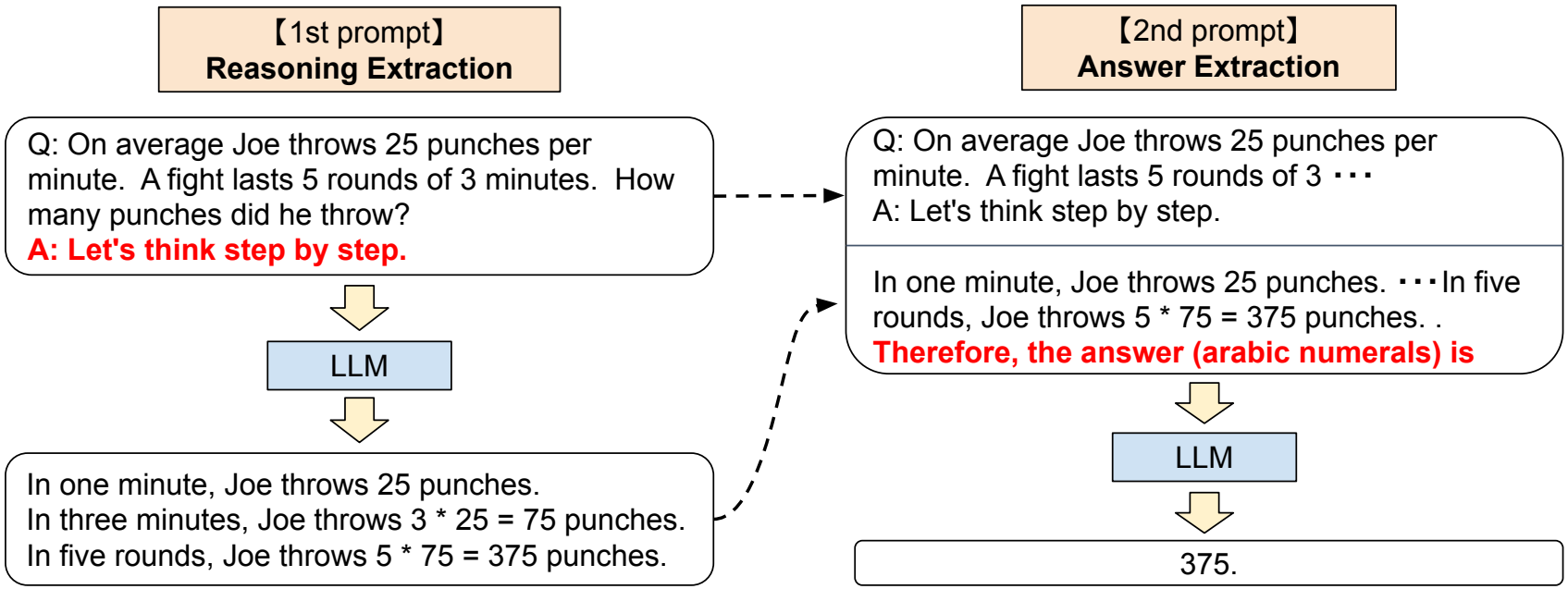
[Kojima et al. 2022]

Chain-of-Thought Prompting

- CoT requires examples explicitly enumerating the reasoning steps
- Turn out reasoning steps can often be elicited from models; just “tell” the model to reason in steps

Chain-of-Thought Prompting

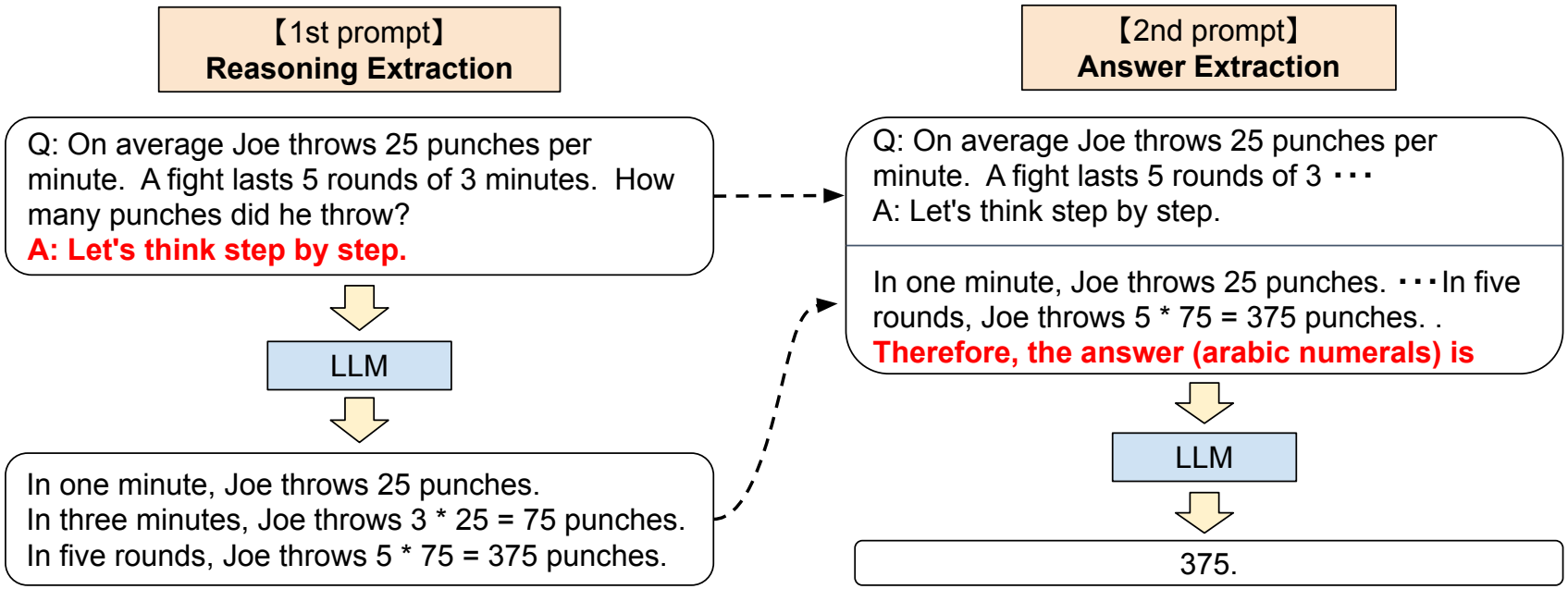
- Main idea: just "tell" the model to reason in steps
- Challenge: the answer is often entangled in the reasoning text, so how do we extract it?



Chain-of-Thought Prompting

- Main idea: just "tell" the model to reason in steps
- Challenge: the answer is often entangled in the reasoning text, so how do we extract it?

Just use an LLM!



Does CoT Help?

Solving and Generating NPR Sunday Puzzles with Large Language Models

Jingmiao Zhao and Carolyn Jane Anderson
Computer Science Department
Wellesley College
Wellesley, MA 02482 USA
carolyn.anderson@wellesley.edu

Abstract

We explore the ability of large language models to solve and generate puzzles from the NPR Sunday Puzzle game show using PUZZLEQA, a dataset comprising 15 years of on-air puzzles. We evaluate four large language models using PUZZLEQA, in both multiple choice and free response formats, and explore two prompt engineering techniques to improve free response performance: chain-of-thought reasoning and prompt summarization. We find that state-of-the-art large language models can solve many PUZZLEQA puzzles: the best model, GPT-3.5, achieves 50.2% loose accuracy. However, in our few-shot puzzle generation experiment, we find no evidence that models can generate puzzles: GPT-3.5 generates puzzles with answers that do not conform to the generated rules. Puzzle generation remains a challenging task for future work.

Puzzle Description: Today's puzzle involves "consonyms," which are words that have the same consonants in the same order but with different vowels. Every answer is the name of a country.
Question: MINGLE
Answer: MONGOLIA

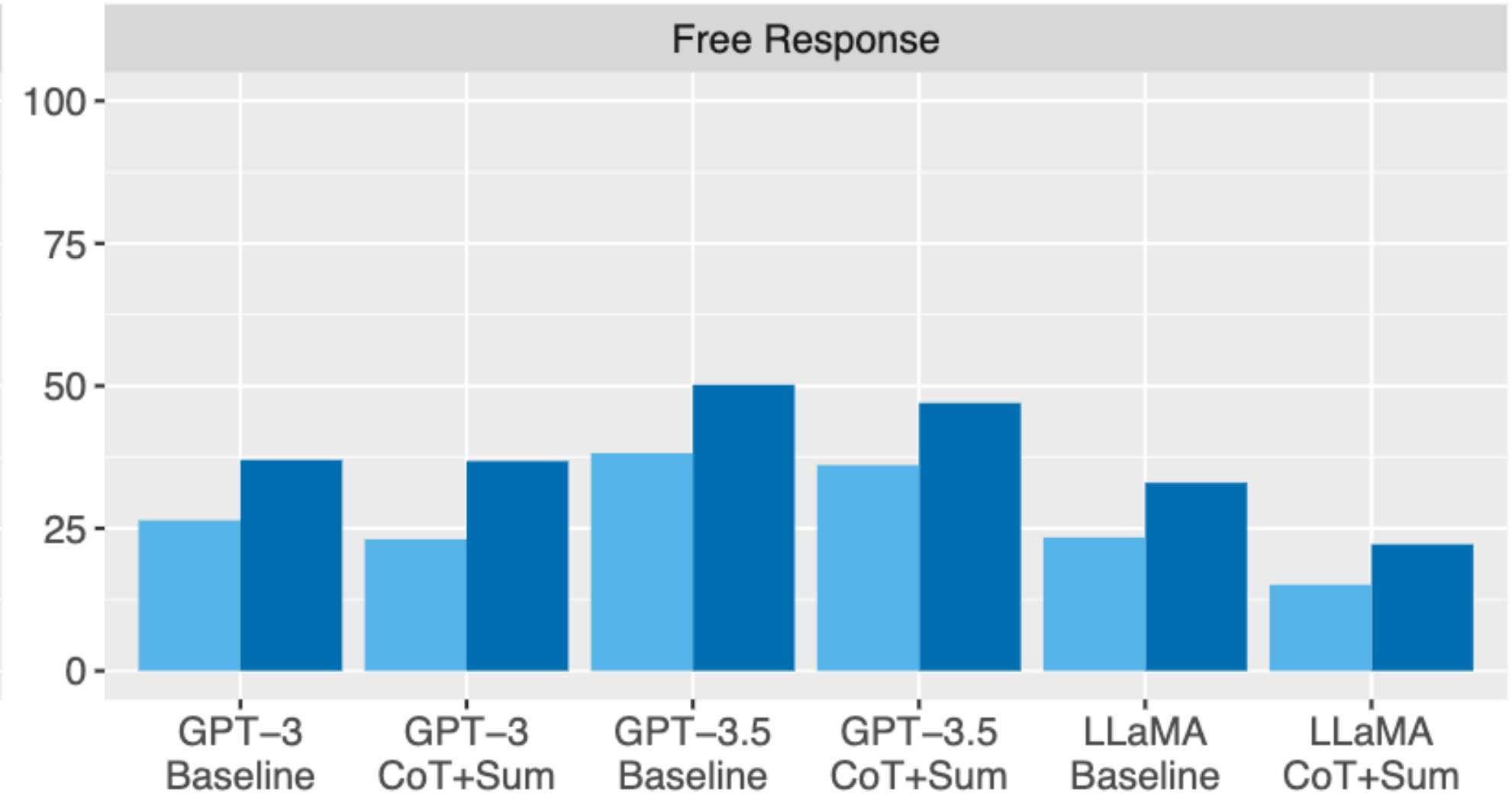
Figure 1: NPR Sunday Puzzle from March 12, 2023

Benchmarking AI through Games

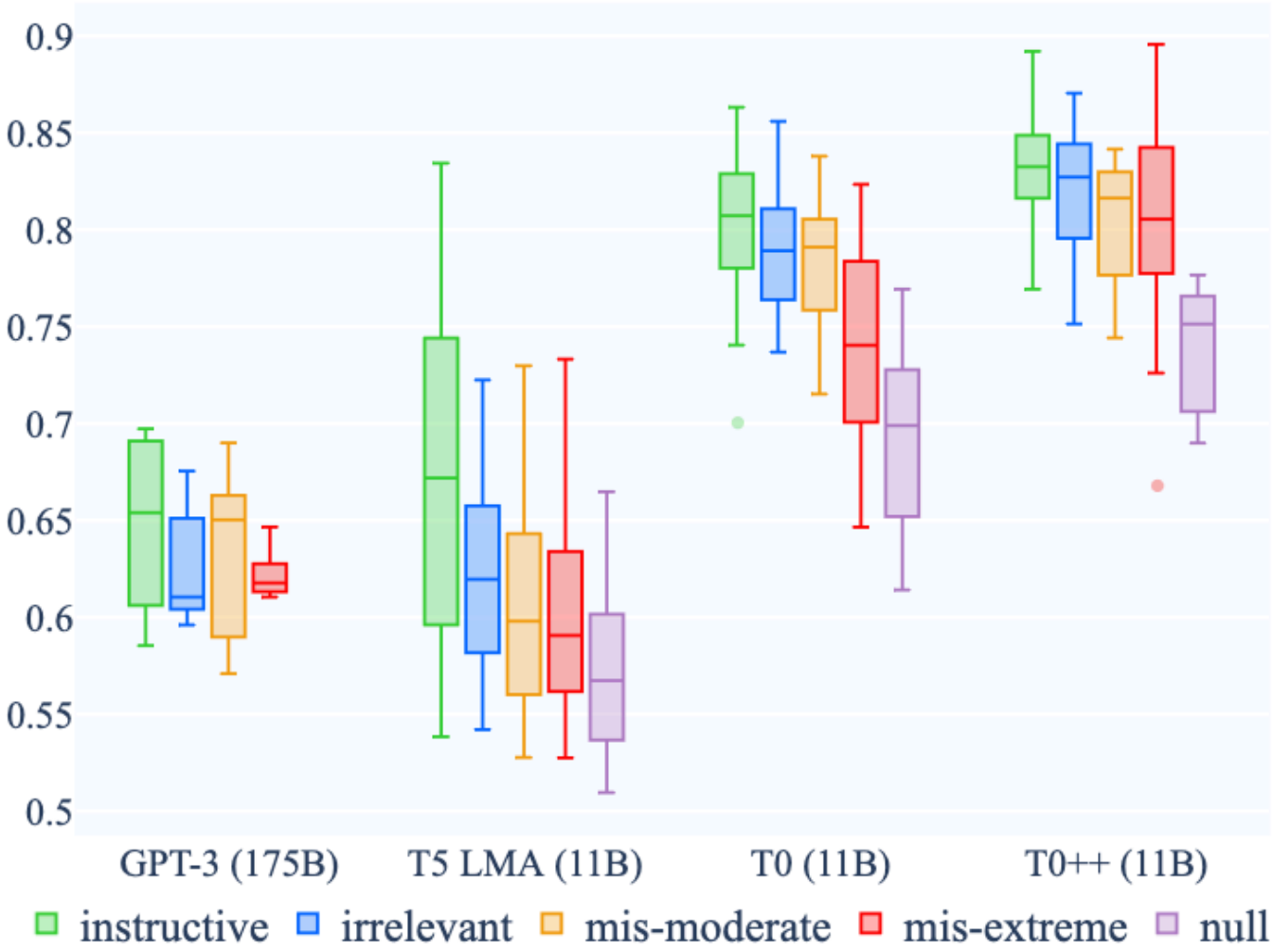
Our work continues the tradition of evaluating AI progress through puzzles and games (Ferrucci 2012; Rodriguez et al. 2021; Rozner, Potts, and Mahowald 2021; Sobieszek and Price 2022). Contemporary LLMs have demonstrated strong performance on a wide variety of language tasks, including

Does CoT Help?

Maybe not?



What Are Prompts Really Doing?

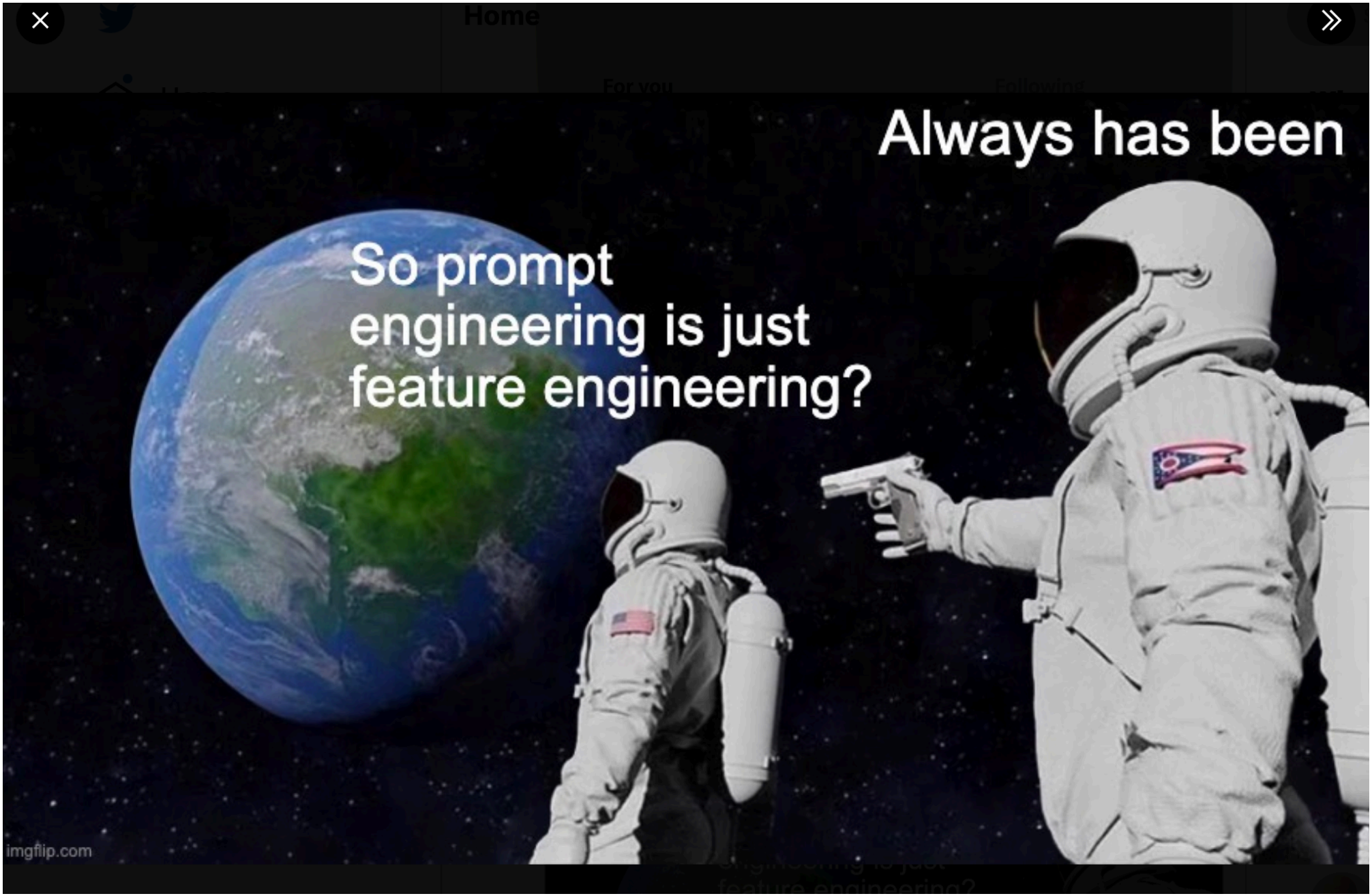


Results from Webson & Pavlick (2022)

Continuous Prompting

Humans write discrete prompts, which are then turned into text embeddings.

What if we tried to directly **learn** good text embeddings?



Mark Dredze
@mdredze