CS 1671/2071
Human Language Technologies

Session 19: Prompting and post-training LLMs

Michael Miller Yoder

March 26, 2025

University of Pittsburgh | School of Computing and Information

Course logistics

- Project progress report is **due tomorrow, Thu Mar 27**. See the project website for instructions
  - **Part 1:** Data statistics and exploratory data analysis (EDA)
  - **Part 2:** A result from baseline/initial approach
  - **Part 3:** Proposal on how to use LLMs for your task
  - **Part 4:** Open questions and challenges
- Project resources
  - Class OpenAI API account to use ($150 total) is coming soon. In the meantime use Gemini free tier or other free LLMs
  - 5 TB class storage is available on CRCD at /ix/cs1671_2025s

Course logistics

- In-person exam will be **next Wed Apr 2**
  - One page of double-sided notes will be permitted
  - Covers Modules 2-4
  - Review session is next Mon Mar 31 during class
- Homework 3 has been released and is **due Apr 10**
  - LLM prompting
  - $5 free credits for OpenAI are no longer a thing 😭
  - Wait to start until the class OpenAI account is active
  - I will extend the due date to reflect the delay

Overview: Prompting and post-training LLMs

- Prompting
  - In-context learning, zero-shot and few-shot learning
  - Chain-of-thought prompting
- Post-training and model alignment for LLMs
  - Instruction tuning
  - RLHF
- Coding activity: programmatic prompting using Gemini API

In-context learning, zero-shot and few-shot learning

Two ways of adapting an LLM to your use case

1. Finetune the parameters of the model with additional data
    - See parameter-efficient finetuning for finetuning very large models

2. Add more information in your prompt to the LLM, such as demonstrations (in-context learning)

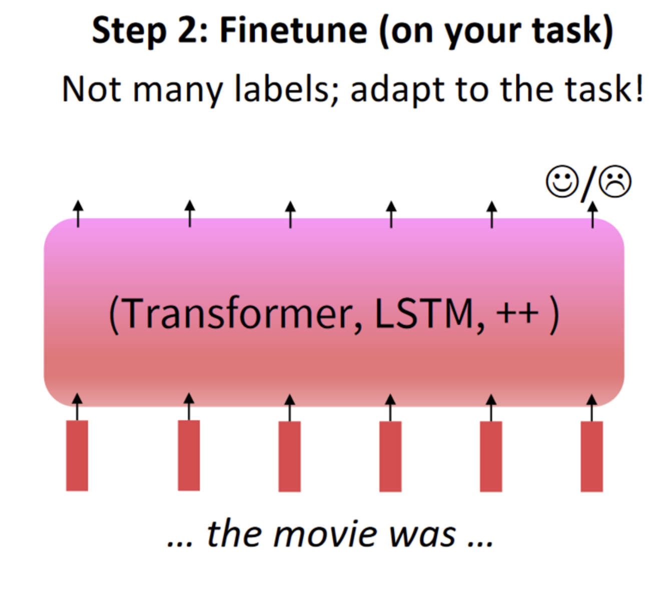


Figure from John Hewitt

Emergent zero-shot learning from GPT2

- GPT-2 [Radford et al. 2019] is a 1.5B parameter language model trained on 40G text data (webtext)
- One key emergent ability in GPT-2 is zero-shot learning: the ability to do many tasks with **no examples** and **no gradient updates**, by simply:
  - Specifying the right sequence prediction problem in the prompt
    - Question answering

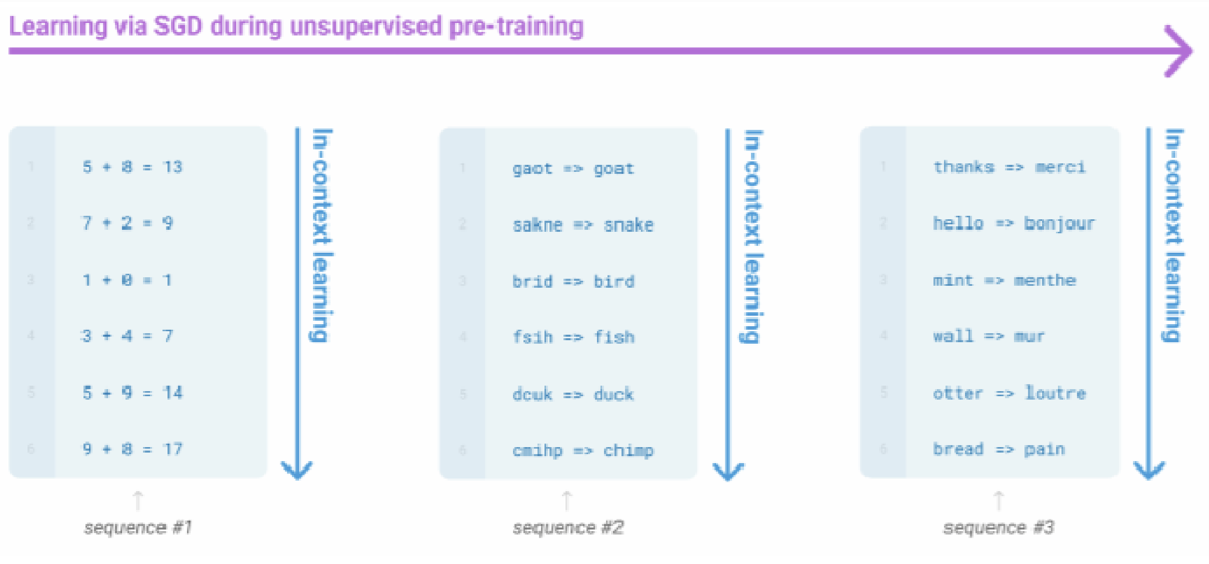
      ```
      Passage: Tom Brady... Q: Where was Tom Brady born? A: ...
      ```

    - Summarization: <article> tl;dr <summary>

Slide adapted from Stanford Natural Language Processing with Deep Learning course via Lorraine Li

In-context learning (few-shot prompting)

Very large language models seem to perform some kind of learning **without gradient steps** simply from examples you provide within their contexts (prompts). The in-context steps seem to specify the task to be performed.



Slide adapted from John Hewitt

Chain-of-thought prompting

Limits of prompting for harder tasks?

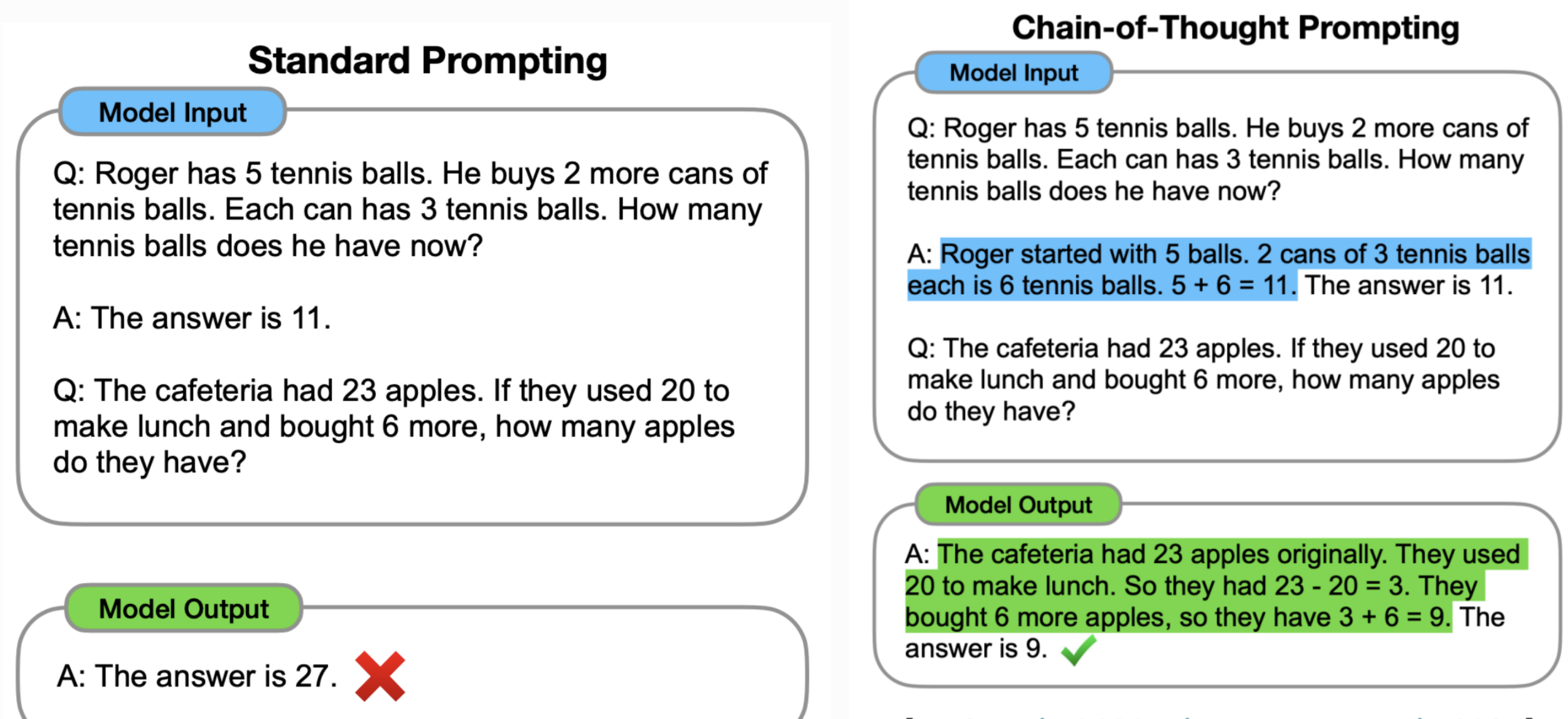
- Some tasks seem too hard for LLM to learn through prompting alone.
- Especially tasks involving richer, multi-step reasoning

```
19583 + 29534 = 49117
98394 + 49384 = 147778
29382 + 12347 = 41729
93847 + 39299 = ?
```

Solution: change the prompt!

Slide adapted from Stanford Natural Language Processing with Deep Learning course

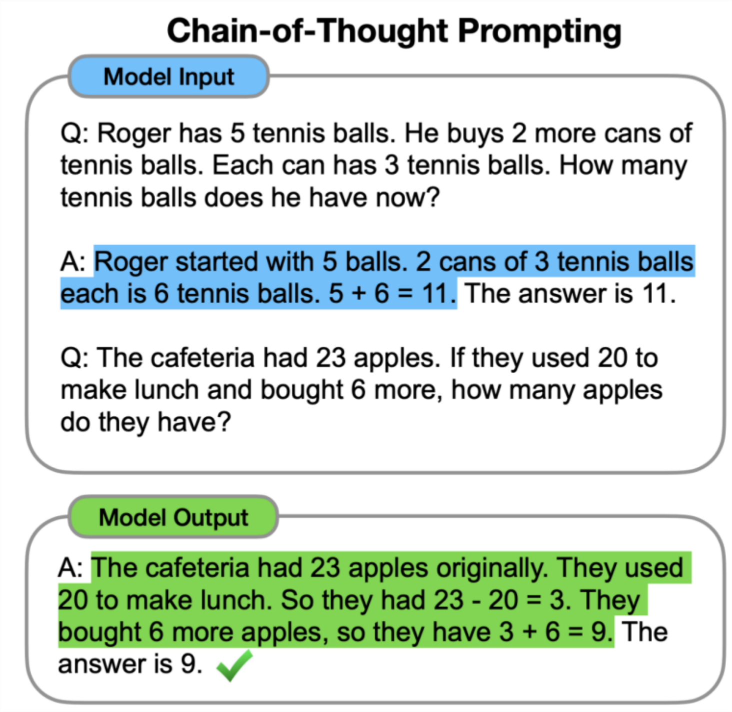
Chain-of-thought prompting



[Wei et al., 2022; also see Nye et al., 2021]

Slide adapted from Stanford Natural Language Processing with Deep Learning course via Lorraine Li

Chain-of-thought prompting



Do we even need examples of reasoning? Can we just ask the model to reason through things?

Slide adapted from Stanford Natural Language Processing with Deep Learning course via Lorraine Li

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. ✓

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.*** *There are 16 balls in total. Half of the balls are golf balls. That means there are 8 golf balls. Half of the golf balls are blue. That means there are 4 blue golf balls.* ✓

[Kojima et al., 2022]

Slide adapted from Stanford Natural Language Processing with Deep Learning course via Lorraine Li

Downside of prompt-based learning

1. **Inefficiency:** The prompt needs to be processed *every time* the model makes a prediction.

2. **Poor performance**: Prompting generally performs worse than fine-tuning [Brown et al., 2020].

3. **Sensitivity** to the wording of the prompt [Webson & Pavlick, 2022], order of examples [Zhao et al., 2021; Lu et al., 2022], etc.

4. **Lack of clarity** regarding what the model learns from the prompt. Even random labels work [Zhang et al., 2022; Min et al., 2022]!

Slide adapted from Stanford Natural Language Processing with Deep Learning course via Lorraine Li

Post-training and model alignment for LLMs: language modeling is not enough

Language modeling ≠ assisting users

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.
```

- Language models are not aligned with user intent [Ouyang et al. 2022]
- Post-training LLMs to the rescue!

Slide adapted from Jesse Mu

Language model as multitask assistants?

- How do we get from
    - University of Pittsburgh is located in ___, Pennsylvania. [World knowledge]
    - I put __ fork down on the table. [syntax]
- To...

Slide adapted from Stanford Natural Language Processing with Deep Learning course

Language model as world models?

Language models may do rudimentary modeling of agents, beliefs, and actions.

Pat watches a demonstration of a bowling ball and a leaf being dropped at the same time in a vacuum chamber. Pat, who is a physicist, predicts that *the bowling ball and the leaf will fall at the same rate.*

Changing the last sentence of the prompt, we get:

...Pat, who has never seen this demonstration before, predicts that *the bowling ball will fall to the ground first. This is incorrect. In a vacuum chamber, there is no air*

Slide adapted from Stanford Natural Language Processing with Deep Learning course

"Language Models as Agent Models" Andreas 2022

Language model as world models?

Language models may do mathematical proofs if pretrained on math content.

Input:
Let $f(r) = \sum_{j=2}^{2008} \frac{1}{j^r} = \frac{1}{2^r} + \frac{1}{3^r} + \cdots + \frac{1}{2008^r}$. Find $\sum_{k=2}^{\infty} f(k)$.

LLEMMA 34B solution:
We have

$$\begin{aligned}
\sum_{k=2}^{\infty} f(k) &= \sum_{k=2}^{\infty} \sum_{j=2}^{2008} \frac{1}{j^k} = \sum_{j=2}^{2008} \sum_{k=2}^{\infty} \frac{1}{j^k} = \sum_{j=2}^{2008} \frac{1}{j^2} \sum_{k=0}^{\infty} \frac{1}{j^k} = \sum_{j=2}^{2008} \frac{1}{j^2} \frac{1}{1-\frac{1}{j}} \\
&= \sum_{j=2}^{2008} \frac{1}{j(j-1)} = \sum_{j=2}^{2008} \left(\frac{1}{j-1} - \frac{1}{j} \right) \\
&= \left(\frac{1}{1} - \frac{1}{2} \right) + \left(\frac{1}{2} - \frac{1}{3} \right) + \cdots + \left(\frac{1}{2007} - \frac{1}{2008} \right) \\
&= 1 - \frac{1}{2008} \\
&= \boxed{\frac{2007}{2008}}.
\end{aligned}$$

Final Answer: The final answer is $\frac{2007}{2008}$.

Slide adapted from Stanford Natural Language Processing with Deep Learning course

"Llemma: An Open Language Model For Mathematics" Azebayev et al 2023

Language model as world models?

Language models can generate working code.

monalisa

Write unit tests for this function

GitHub Copilot

```python
import unittest
import datetime

class TestParseExpenses(unittest.TestCase):
    def test_parse_expenses_with_valid_input(self):
        expenses_string = '2023-01-02 -34.01 USD\n2023-01-
        expected_output = [(datetime.datetime(2023, 1, 2)
        self.assertEqual(parse_expenses(expenses_string),

    def test_parse_expenses_with_empty_input(self):
        expenses_string = ''
        expected_output = []
        self.assertEqual(parse_expenses(expenses_string),

if __name__ == '__main__':
    unittest.main()
```

Slide adapted from Stanford Natural Language Processing with Deep Learning course

Copilot

Language model as multitask assistants?

I need to throw a dinner party for 6 people who are vegetarian. Can you suggest a 3-course menu with a chocolate dessert?

[Microsoft Bing]

(Also see OpenAI's ChatGPT, Google's Bard, Anthropic's Claude)

Sure, I can suggest a 3-course menu with a chocolate dessert for your vegetarian dinner party. Here are some possible options:

- Starter: **Wild Mushroom Tartlets with Onion Sauce** [1] or **Vegan Popcorn Tofu Nuggets** [2]
- Main: **Vegan Butternut Squash Mac** [2] or **Vegetarian Three-Bean Chili** [3]
- Dessert: **Chocolate Lava Cake** [4] or **Chocolate Pasta with Chocolate Hazelnut Cream Sauce, White Chocolate Shavings, and Fresh Berries** [4]

I hope this helps you plan your party. Enjoy! 😋

Learn more: 1. booths.co.uk +10 more

Slide adapted from Stanford Natural Language Processing with Deep Learning course

Post-training (model alignment)

Two techniques to align LLMs with human preferences (what we want them to do):

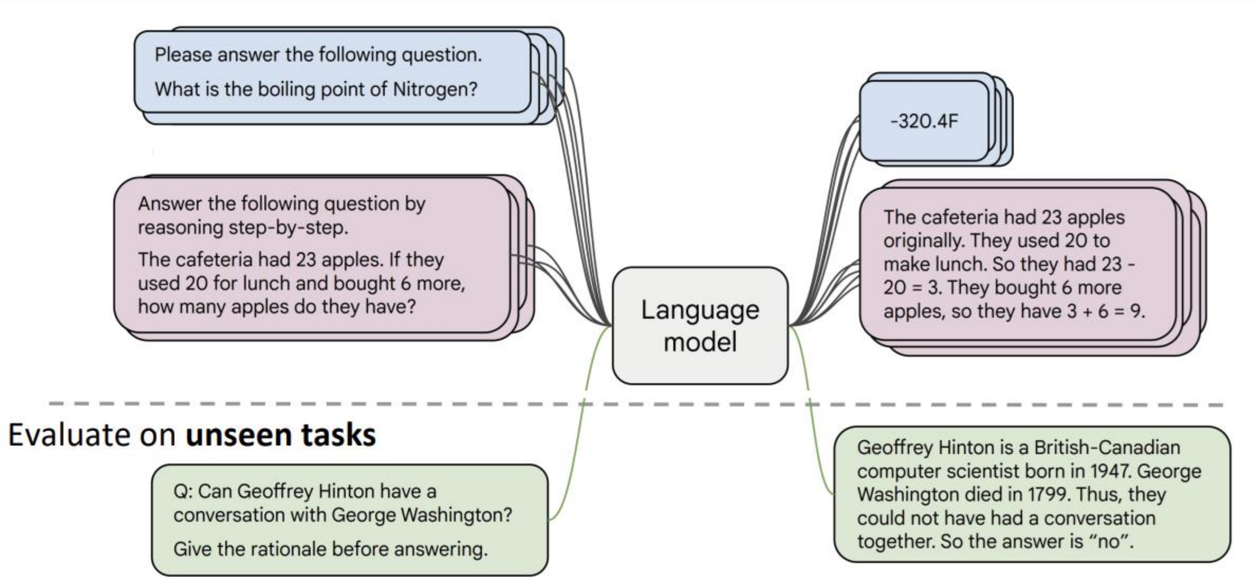
1. Instruction tuning
   - Models are finetuned on a corpus of instructions/questions and desired responses
2. Preference alignment (RLHF)
   - Separate model is trained to decide how much a candidate response aligns with human preferences
   - This reward model is used to finetune the base model

Slide based on Jurafsky & Martin

Instruction tuning

Instruction tuning (supervised finetuning, SFT)

- Collect examples of (instruction, output) pairs across many tasks and finetune an LM
- Still just LM objective (predict the next word)



Slide adapted from Jesse Mu, Chung et al. 2022

Limitations of instruction finetuning

- Expensive to collect ground-truth data for tasks
  - Though you can include existing datasets of tasks like question answering
  - And LLMs are now commonly used to generate instruction tuning datasets
- Tasks like open-ended creative generation have no right answer.
  - Write me a story about a dog and her pet grasshopper.
- Language modeling penalizes all token-level mistakes equally, but some errors are worse than others
- Even with instruction finetuning, there is a mismatch between the LM objective and the objective of “satisfy human preferences”!
- Can we **explicitly attempt to satisfy human preferences?**

Slide adapted from Jesse Mu, Jurafsky & Martin

Reinforcement learning from human feedback (RLHF)

Optimizing for human preferences

- Let’s say we were training a language model on some task (e.g. summarization).
- For each LM sample s, imagine we had a way to obtain a human reward of that summary: $R(s) \in \mathbb{R}$, higher is better.

```
SAN FRANCISCO,
California (CNN) --
A magnitude 4.2
earthquake shook the
San Francisco
...
overturn unstable
objects.
```

```
An earthquake hit
San Francisco.
There was minor
property damage,
but no injuries.
```

s_1

$$R(s_1) = 8.0$$

```
The Bay Area has
good weather but is
prone to
earthquakes and
wildfires.
```

s_2

$$R(s_2) = 1.2$$

- Now we want to maximize the expected reward of samples from our LM

Slide adapted from Jesse Mu

How do we model human preferences?

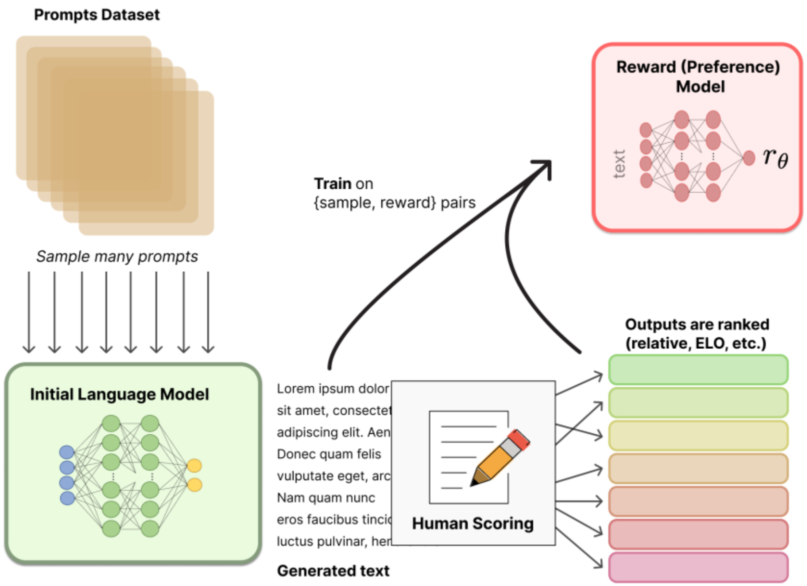
Ask annotators to rank different responses by their preference

An earthquake hit San Francisco. There was minor property damage, but no injuries.	>	A 4.2 magnitude earthquake hit San Francisco, resulting in massive damage.	>	The Bay Area has good weather but is prone to earthquakes and wildfires.

Slide adapted from Jesse Mu

Reward model

- Takes in a sequence of text and produces a scalar representing human preference for that text
Training data:
    - Prompts (can come from real users of OpenAI's LLMs, e.g.)
    - LLM-generated responses to those prompts, ranked by human annotators



Slide based on https://huggingface.co/blog/rlhf

LLMs are finetuned to optimize the reward model using reinforcement learning

- Why reinforcement learning? It's good at handling arbitrary reward functions like "human preference"
- Finetuning language modeling (predicting the next word) with the goal of optimizing preference for the entire output
    - Is like optimizing steps to take in a game to optimize winning, the classic example of reinforcement learning
- Often uses the Proximal Policy Optimization (PPO) reinforcement learning algorithm
- Tries to not stray too far from the original language model

Slide based on https://huggingface.co/blog/rlhf

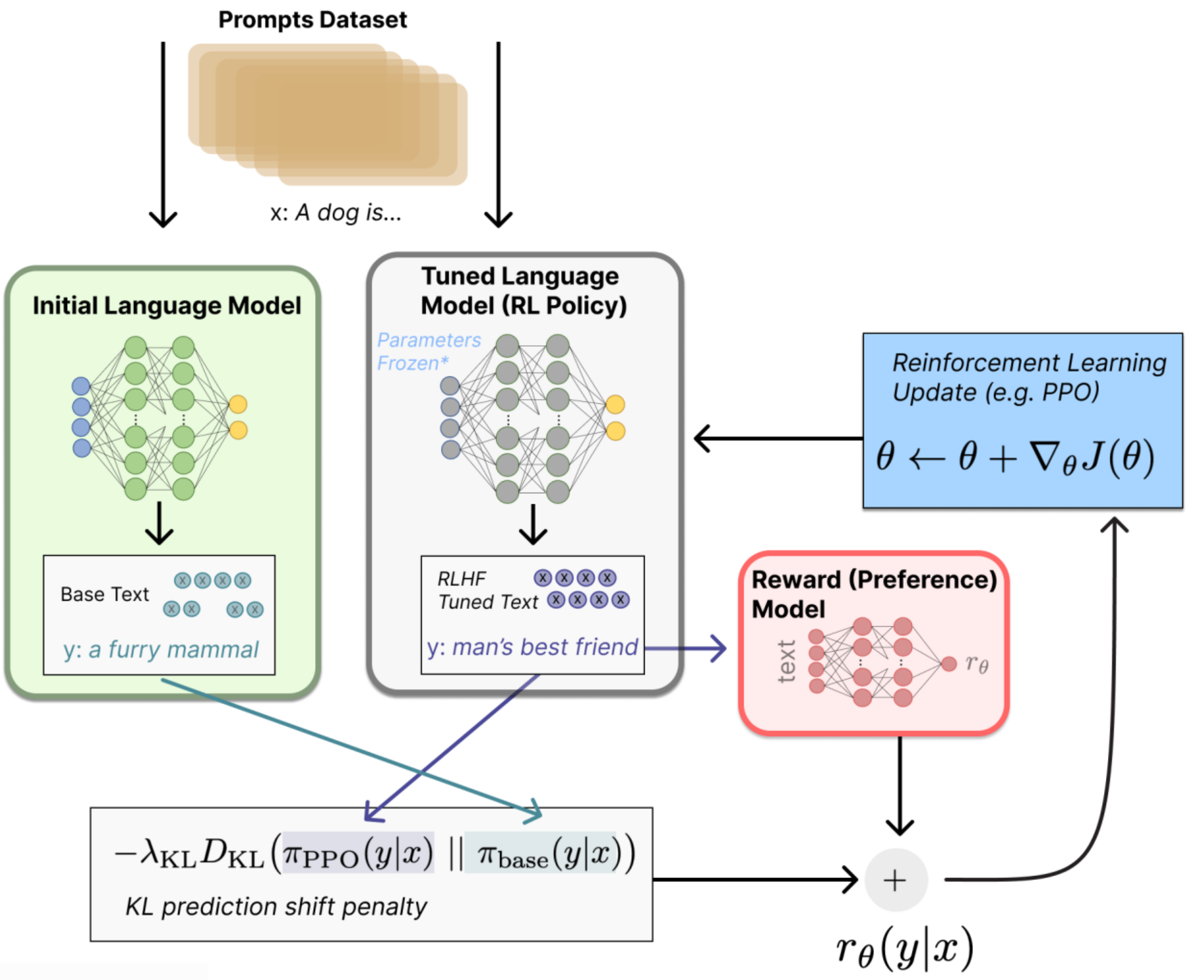
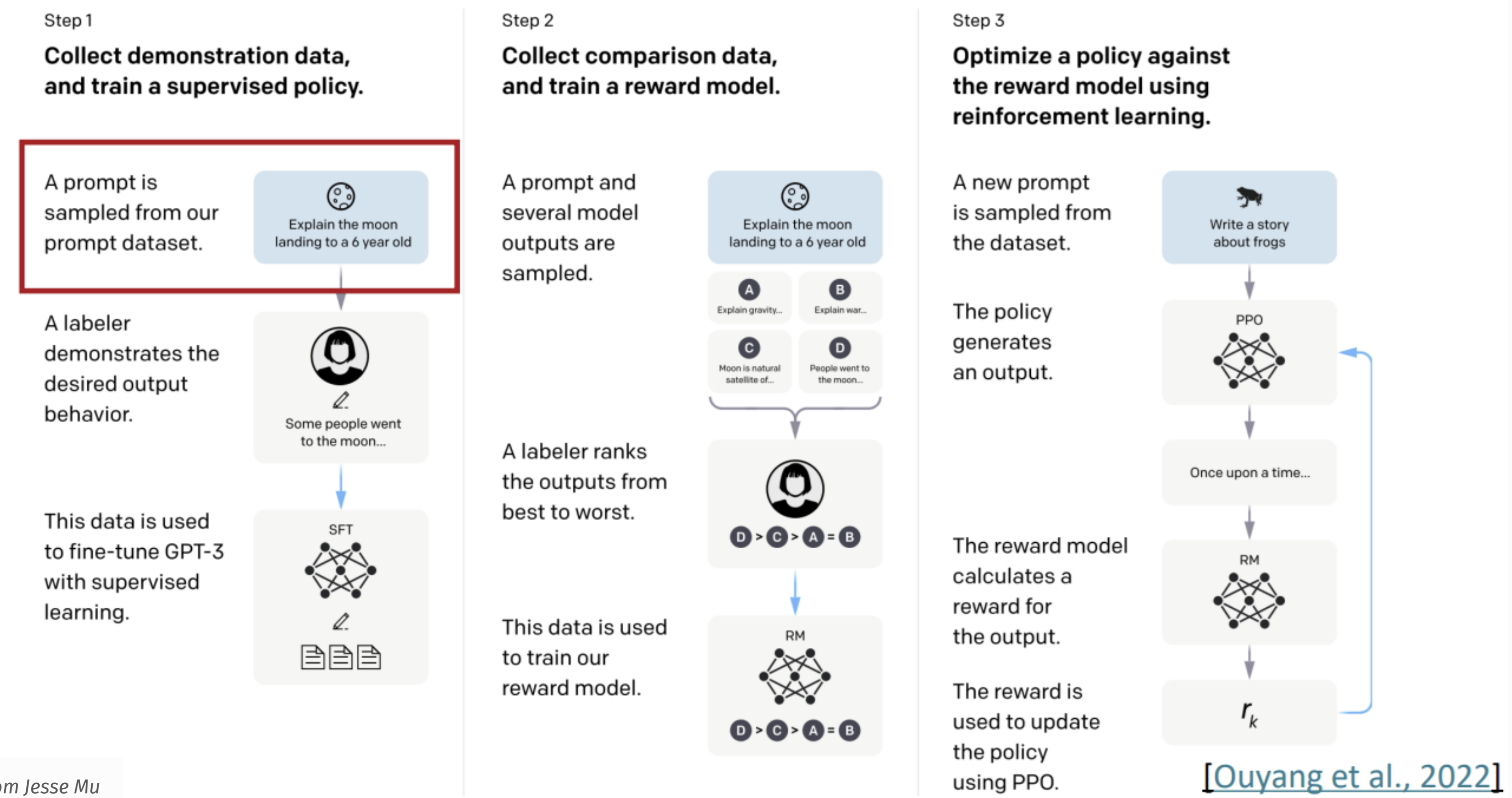


Image from https://huggingface.co/blog/rlhf

InstructGPT: scaling up RLHF to tens of thousands of tasks

30k tasks!



[Ouyang et al., 2022]

Slide adapted from Jesse Mu

InstructGPT gets us closer to dialogue intent

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.
```

Slide adapted from Jesse Mu

InstructGPT can accomplish many tasks

PROMPT *Write a short poem about a wise frog.*

COMPLETION

GPT-3

```
Write a short story in which a character has two different names.

Write a short story in which you try to get something back that you
have lost.

Write a short story in which a character has a bad dream.
```

InstructGPT

```
The frog is a symbol of wisdom
He knows all the secrets of the world
He is a master of disguise
And a great teacher of life
He is a symbol of transformation
And the bringer of change
He is the frog who has seen it all
And knows the meaning of it all
```

Slide adapted from Jesse Mu

ChatGPT: Instruction Finetuning + RLHF for dialog agents

ChatGPT: Optimizing Language Models for Dialogue

Note: OpenAI (and similar companies) are keeping more details secret about ChatGPT training (including data, training parameters, model size)—perhaps to keep a competitive edge...

Methods

We trained this model using Reinforcement Learning from Human Feedback (RLHF), using the same methods as InstructGPT, but with slight differences in the data collection setup. We trained an initial model using supervised fine-tuning: human AI trainers provided conversations in which they played both sides—the user and an AI assistant. We gave the trainers access to model-written suggestions to help them compose their responses. We mixed this new dialogue dataset with the InstructGPT dataset, which we transformed into a dialogue format.

(Instruction finetuning!)

https://openai.com/blog/chatgpt/

Slide adapted from Jesse Mu

ChatGPT: Instruction Finetuning + RLHF for dialog agents

ChatGPT: Optimizing Language Models for Dialogue

Note: OpenAI (and similar companies) are keeping more details secret about ChatGPT training (including data, training parameters, model size)—perhaps to keep a competitive edge…

Methods

To create a reward model for reinforcement learning, we needed to collect comparison data, which consisted of two or more model responses ranked by quality. To collect this data, we took conversations that AI trainers had with the chatbot. We randomly selected a model-written message, sampled several alternative completions, and had AI trainers rank them. Using these reward models, we can fine-tune the model using Proximal Policy Optimization. We performed several iterations of this process.

(RLHF!)

https://openai.com/blog/chatgpt/

Limitations of RL + Reward Modeling

- Human preferences are unreliable!
- “Reward hacking” is a common problem in RL
- Chatbots are rewarded to produce responses that seem authoritative and helpful, regardless of truth
- This can result in making up facts + hallucinations

Slide adapted from Jesse Mu

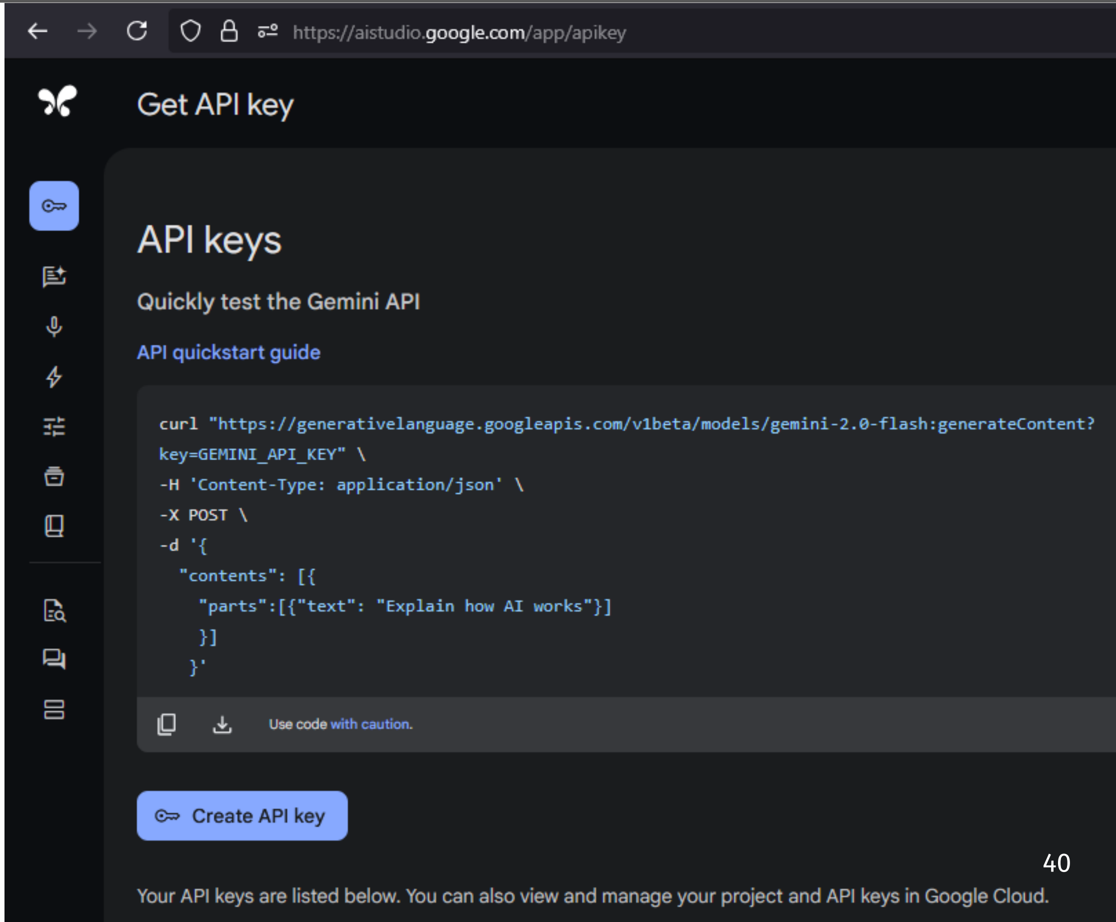
Conclusion

- Zero-shot prompting is simply asking an LLM to do something without providing any examples
- Few-shot prompting (in-context learning) is where a few examples are provided, which can improve LLM output
- Chain-of-though prompting, providing reasoning in examples, can also improve LLM output
- Instruction tuning, finetuning of LLMs with prompt-response pairs, can help align language models with human preferences
- Large language models can be trained to provide more useful responses using reinforcement learning from human feedback (RLHF)

Coding activity: prompt Gemini

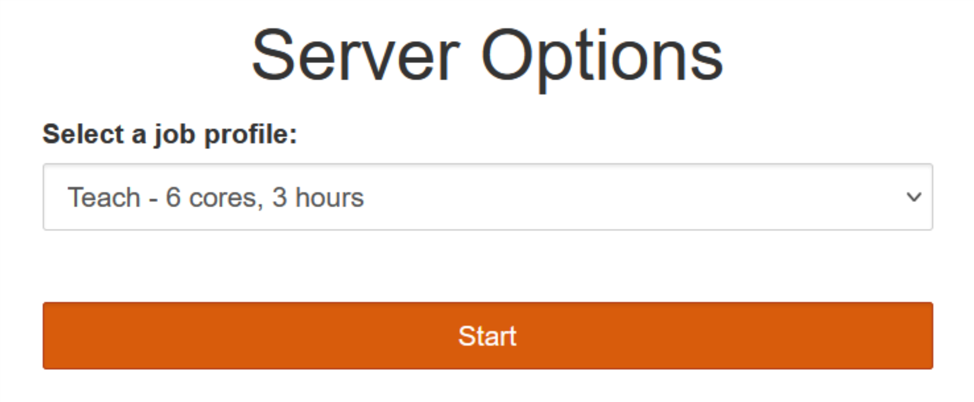
First, create an API key for Google Gemini

- Use this link to create an API key
- You will need to sign in to a Google account
  - If you don't have or want one, look on with a neighbor



Notebook for this class: prompt Gemini through an API

- Click on this nbgitpuller link or find the link on the course website
- Start a regular CPU ‘Teach – 6 cores, 3 hours’ server. There is no need for a GPU



- Open **session19_prompting.ipynb**