SFU NatLangLab

CMPT 413/713: Natural Language Processing

Instruction tuning

Spring 2024

2024-03-11

Slides adapted from Anoop Sarkar

From LLMs to Helpful Assistants

How to build chatGPT from an LLM base model

https://www.youtube.com/watch?v=bZQun8Y4L2A

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.
```

https://openai.com/research/instruction-following

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

https://arxiv.org/abs/2203.02155

GPT models (after GPT-3)

OpenAI

- Supervised fine-tuning on human conversations
- Data where human will pretend to be user or AI assistant

InstructGPT and GPT-3.5 [2022]

- Align responses to human feedback
- Instruction fine-tuning
- Reinforcement learning from human feedback
- Used in initial ChatGPT

- Human rank generated output
- Use reinforcement learning to improve generation

GPT-4 [March 2023]

- Multimodal with images and text (GPT-4V)
- Larger, better model

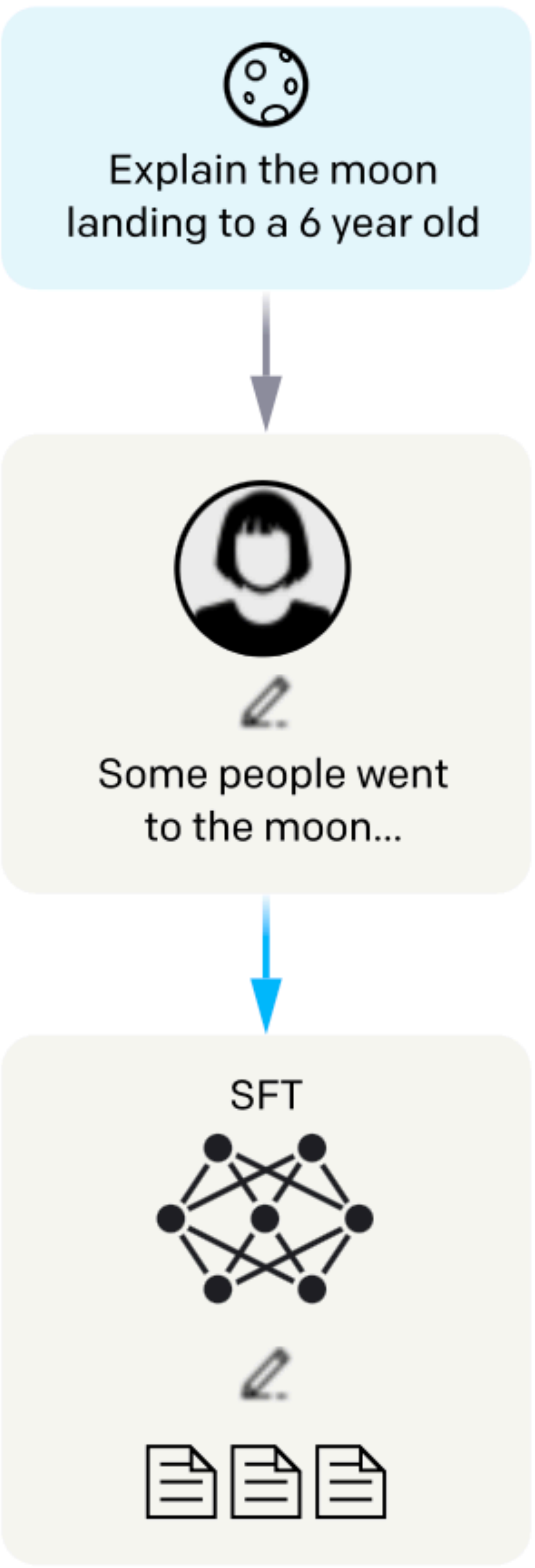
Step 1

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.



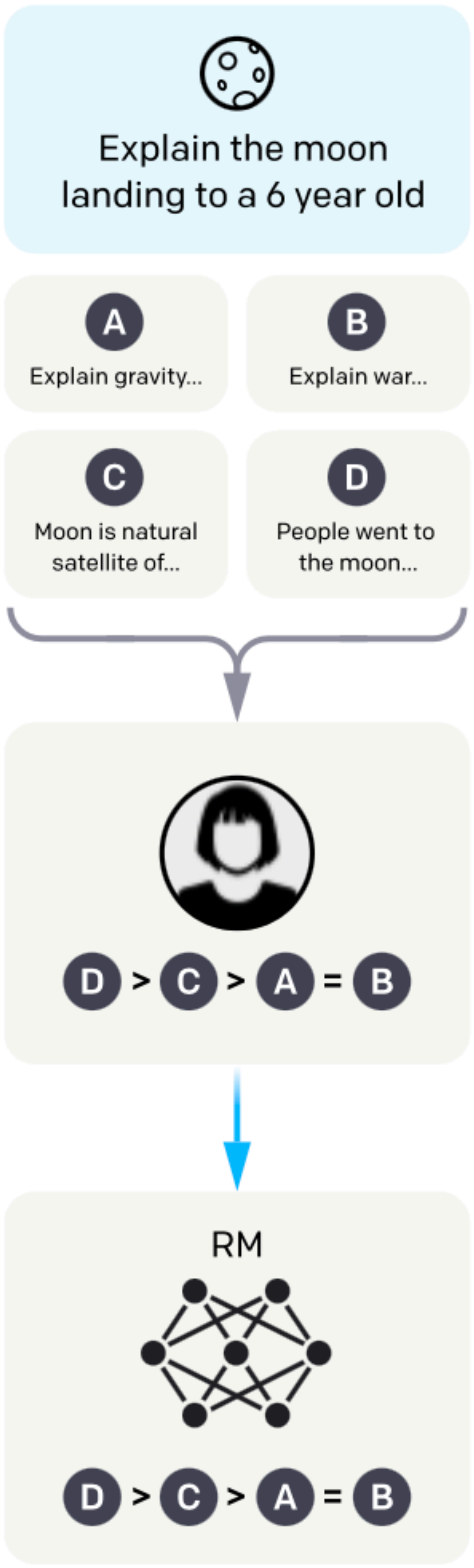
Step 2

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.



Step 3

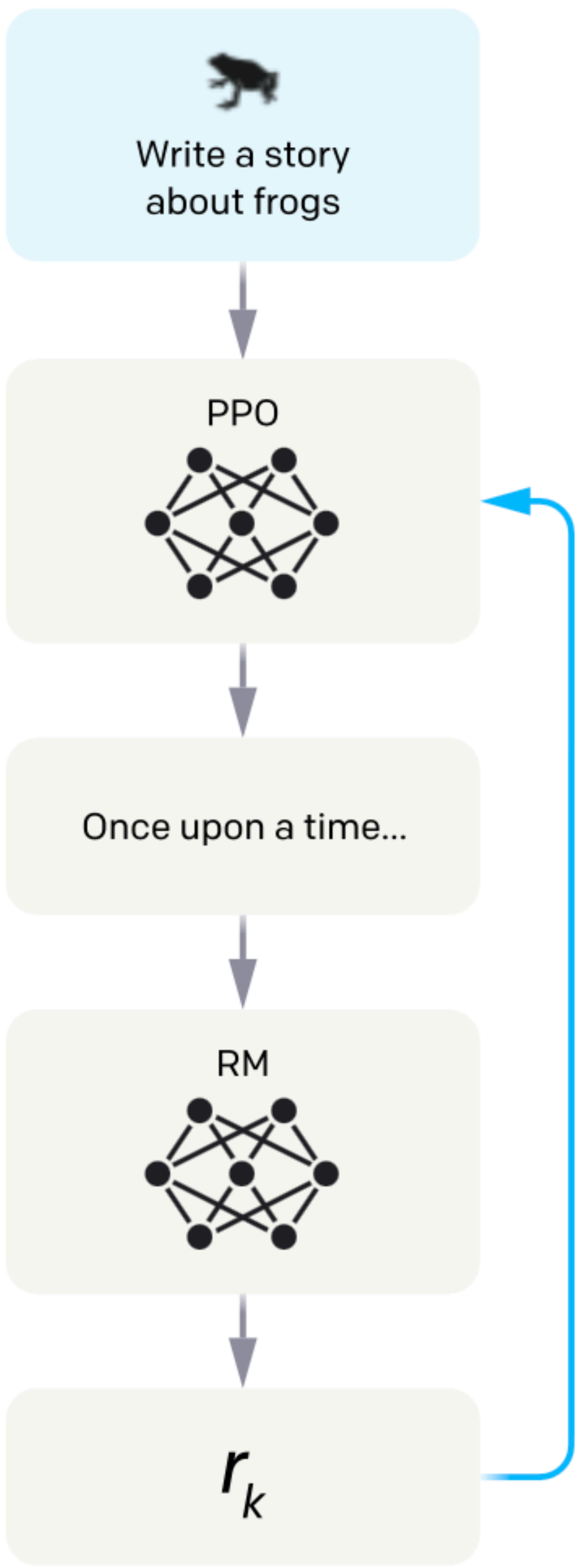
Optimize a policy against the reward model using reinforcement learning.

A new prompt is sampled from the dataset.

The policy generates an output.

The reward model calculates a reward for the output.

The reward is used to update the policy using PPO.



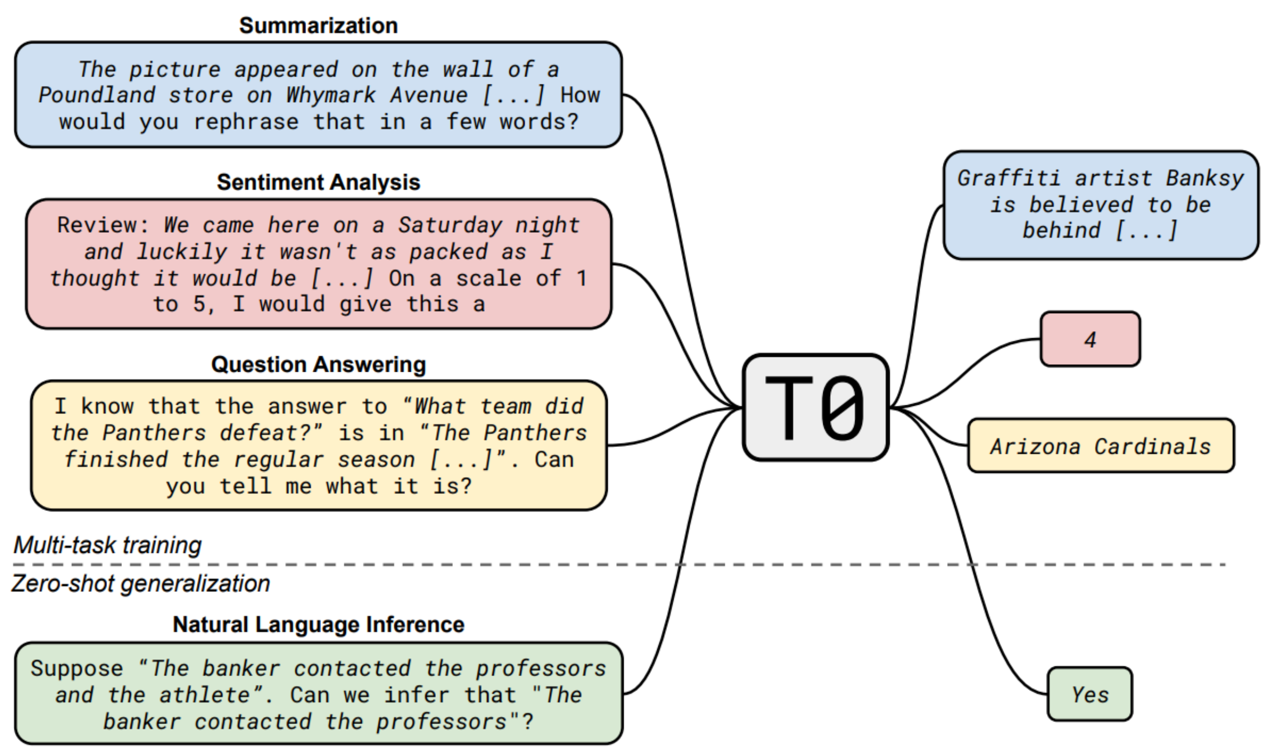
https://openai.com/research/instruction-following

GPT Assistant training pipeline

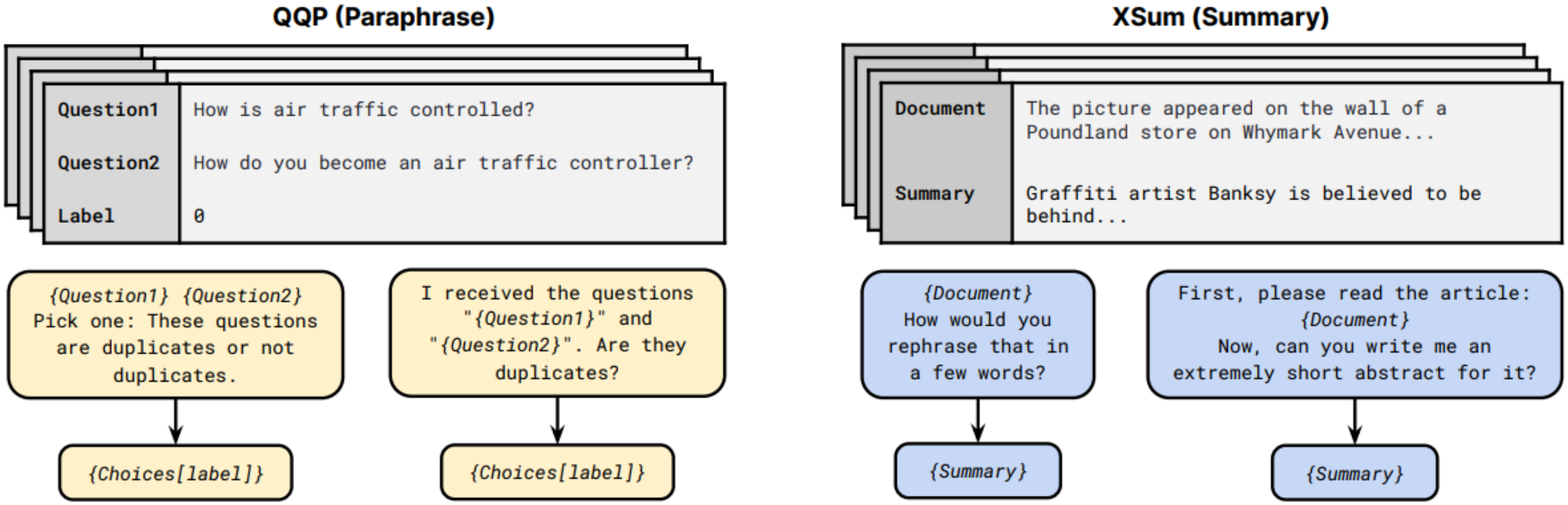
Stage	Pretraining	Supervised Finetuning	Reward Modeling	Reinforcement Learning
Dataset	**Raw internet** text trillions of words low-quality, large quantity	**Demonstrations** Ideal Assistant responses, ~10-100K (prompt, response) written by contractors low quantity, high quality	**Comparisons** 100K –1M comparisons written by contractors low quantity, high quality	**Prompts** ~10K-100K prompts written by contractors low quantity, high quality
Algorithm	**Language modeling** predict the next token	**Language modeling** predict the next token	**Binary classification** predict rewards consistent w preferences	**Reinforcement Learning** generate tokens that maximize the reward
		init from	init from	init from SFT use RM
Model	**Base model**	**SFT model**	**RM model**	**RL model**
Notes	1000s of GPUs months of training ex: GPT, LLaMA, PaLM **can deploy this model**	1-100 GPUs days of training ex: Vicuna-13B **can deploy this model**	1-100 GPUs days of training	1-100 GPUs days of training ex: ChatGPT, Claude **can deploy this model**

Supervised Fine-Tuning
(instruction tuning without human data)

Instruction tuning

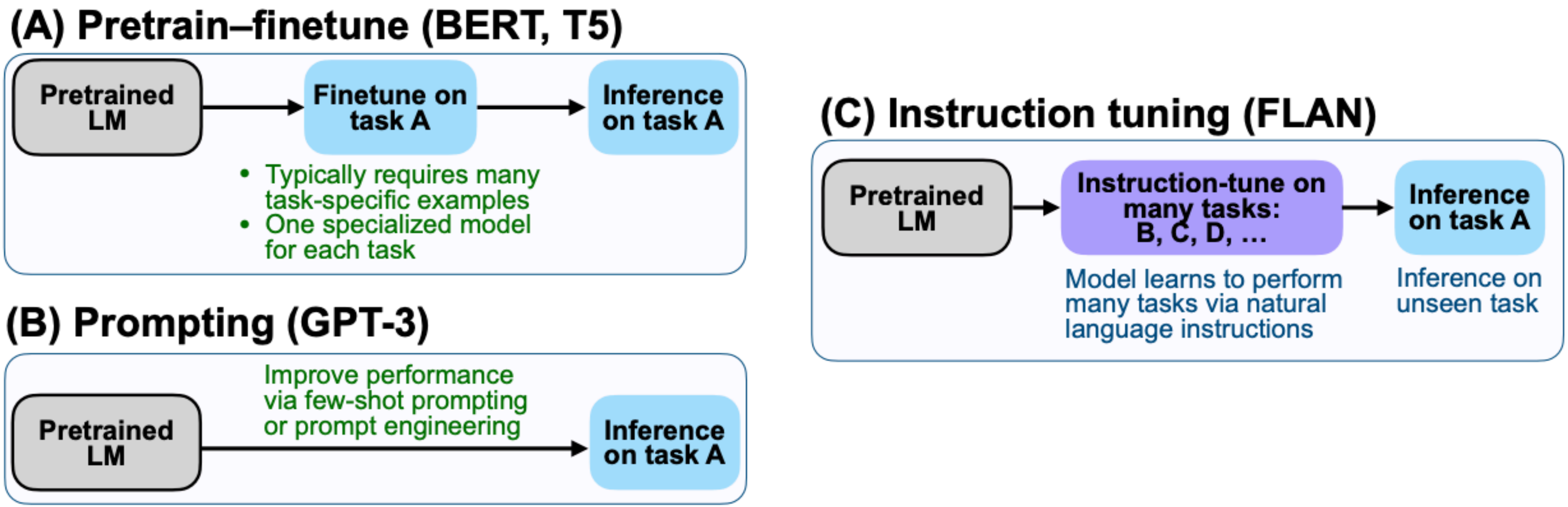


- Use templates to make them into instruction based dataset
- Text based format makes it natural for humans



Multitask prompted training enables zero-shot task generalization, Sahn et al. Google, ICLR 2022

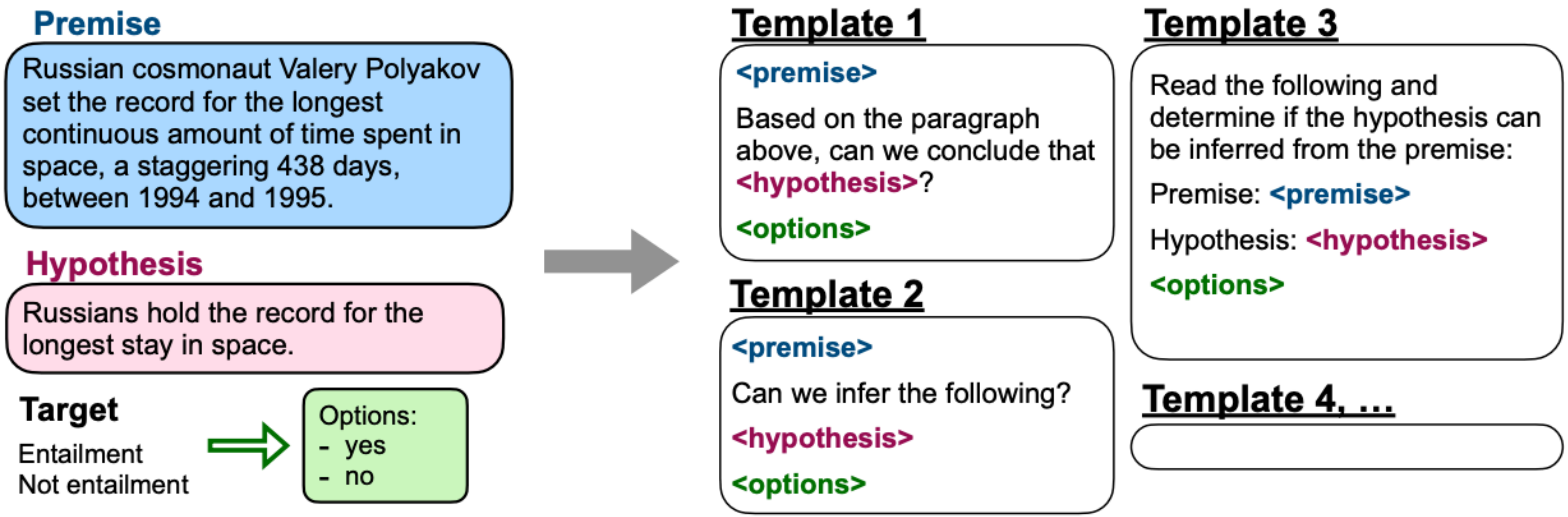
Instruction tuning



Finetuned language models are zero-shot learners, Wei et al. Google, ICLR 2022

Instruction tuning

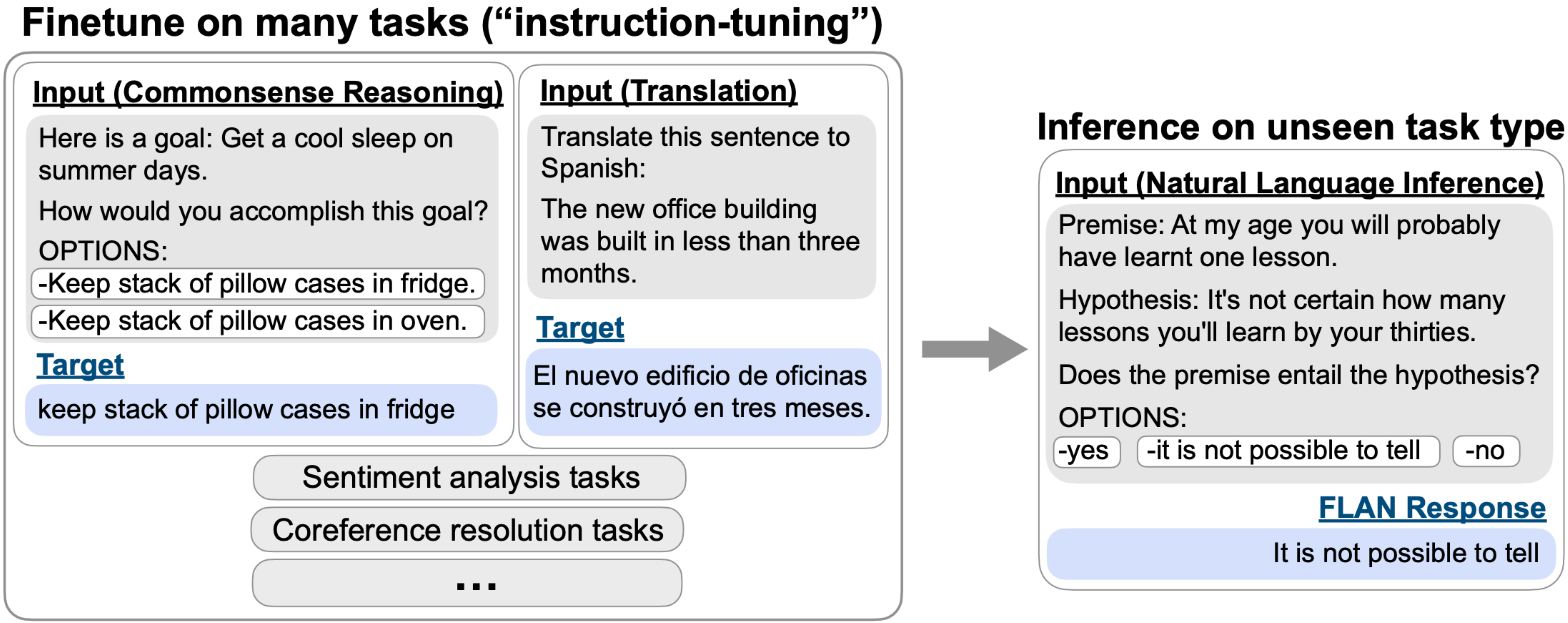
- Use templates to make them into instruction based dataset
- Text based format makes it natural for humans



Finetuned language models are zero-shot learners, Wei et al. Google, ICLR 2022

Instruction tuning

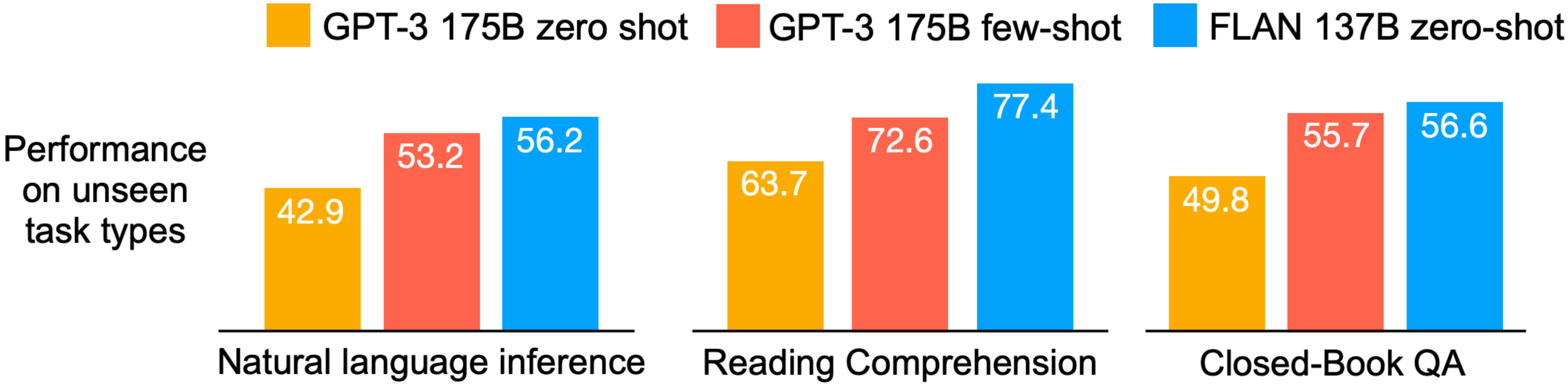
- Can be used on an unseen task type



Finetuned language models are zero-shot learners, Wei et al. Google, ICLR 2022

Instruction tuning

- Can be used on an unseen task type



Finetuned language models are zero-shot learners, Wei et al. Google, ICLR 2022

Supervised Fine-Tuning
(instruction tuning with human data)

Prompt:

"Can you write a short introduction about the relevance of the term "monopsony" in economics? Please use examples related to potential monopsonies in the labour market and cite relevant research."

Response:

""Monopsony" refers to a market structure where there is only one buyer for a particular good or service. In economics, this term is particularly relevant in the labor market, where a monopsony employer has significant power over the wages and working conditions of their employees. The presence of a monopsony can result in lower wages and reduced employment opportunities for workers, as the employer has little incentive to increase wages or provide better working conditions. Recent research has identified potential monopsonies in industries such as retail and fast food, where a few large companies control a significant portion of the market (Bivens & Mishel, 2013). In these industries, workers often face low wages, limited benefits, and reduced bargaining power, leading to a situation where they are dependent on the employer for their livelihood. This dependence can result in further suppression of wages and a decline in working conditions. Overall, the concept of monopsony is essential to understanding the dynamics of labor markets and the impact of market power on workers. Further research is needed to understand the extent and impact of monopsonies on the economy and to develop policies to address this issue. References: Bivens, J., & Mishel, L. (2013). The Pay of Corporate Executives and Financial Professionals as Evidence of Rents in Top 1 Percent Incomes. Journal of Economic Perspectives, 27(3), 57-78."

[example from OpenAssistant Conversations Dataset (OASST1)]

Collect prompts from those submitted to OpenAI API (Playground Interface)

To bootstrap process, initial prompts from human labelers to train initial InstructGPT

Table 1: Distribution of use case categories from our API prompt dataset.

Use-case	(%)
Generation	45.6%
Open QA	12.4%
Brainstorming	11.2%
Chat	8.4%
Rewrite	6.6%
Summarization	4.2%
Classification	3.5%
Other	3.5%
Closed QA	2.6%
Extract	1.9%

Table 2: Illustrative prompts from our API prompt dataset. These are fictional examples inspired by real usage—see more examples in Appendix A.2.1.

Use-case	Prompt
Brainstorming	List five ideas for how to regain enthusiasm for my career
Generation	Write a short story where a bear goes to the beach, makes friends with a seal, and then returns home.
Rewrite	This is the summary of a Broadway play: """ {summary} """ This is the outline of the commercial for that play: """

Instructions to evaluate output text outputs for prompts

Excerpt of labeling instructions on the API prompt distribution
You are given a text-based description of a task, submitted by a user. This task description may be in the form of an explicit instruction (e.g. "Write a story about a wise frog."). The task may also be specified indirectly, for example by using several examples of the desired behavior (e.g. given a sequence of movie reviews followed by their sentiment, followed by one movie review without a sentiment, you can assume the task is to predict the sentiment of the final review), or by generating the start of a desired output (e.g. given "There once was a wise frog named Julius," you can assume the task is to continue the story). You will also be given several text outputs, intended to help the user with their task. Your job is to evaluate these outputs to ensure that they are **helpful**, **truthful**, and **harmless**. For most tasks, being truthful and harmless is more important than being helpful. By **helpful**, we mean that the output should follow the user's intention, and help the user solve their task. Some examples of helpful behavior: • Writing in clear language. • Answering the question they meant to ask, even if they mis-asked it. • Being sensitive to internationality (e.g. "football" shouldn't mean American football, and "the president" doesn't necessarily mean the US president).

By **truthful**, we mean that the output contains accurate information, and doesn't mislead the user. Some examples of truthful behavior:

- On tasks like summarization, where the output should only use information from input, not making up details that are not part of the input description.
- Not producing clearly false information about the world (e.g. making up facts or promoting conspiracies). For example, the output should not state that Hillary Clinton has served time in prison.
- Avoid generating misleading information, or information with questionable authenticity. For example, if the input asks "Why did Hillary Clinton go to jail?", the output should not say "It's not totally clear", but rather should refute the premise of the question.

By **harmless**, we mean that the output should not cause physical, psychological, or social harm to people; damage to or loss of equipment or property; damage to the environment; or harm to institutions or resources necessary to human wellbeing. Some examples of harmless behavior:

- Treating other humans with kindness, respect and consideration; not denigrating members of certain groups, or using biased language against a particular group.
- Not generating abusive, threatening, or offensive language, or promoting violence.
- Not writing sexual or violent content if it's not asked for.
- Not giving bad real-world advice, or promoting illegal activity.

Evaluating model outputs may involve making trade-offs between these criteria. These trade-offs will depend on the task. Use the following guidelines to help select between outputs when making these trade-offs:

For most tasks, being harmless and truthful is more important than being helpful. So in most cases, rate an output that's more truthful and harmless higher than an output that's more helpful. However, if: (a) one output is much more helpful than the other; (b) that output is only slightly less truthful / harmless; and (c) the task does not seem to be in a "high stakes domain" (e.g. loan applications, therapy, medical or legal advice, etc.); then rate the more helpful output higher. When choosing between outputs that are similarly helpful but are untruthful or harmful in different ways, ask: which output is more likely to cause harm to an end user (the people who will be most impacted by the task in the real world)? This output should be ranked lower. If this isn't clear from the task, then mark these outputs as tied.

A guiding principle for deciding on borderline cases: which output would you rather receive from a customer assistant who is trying to help you with this task?

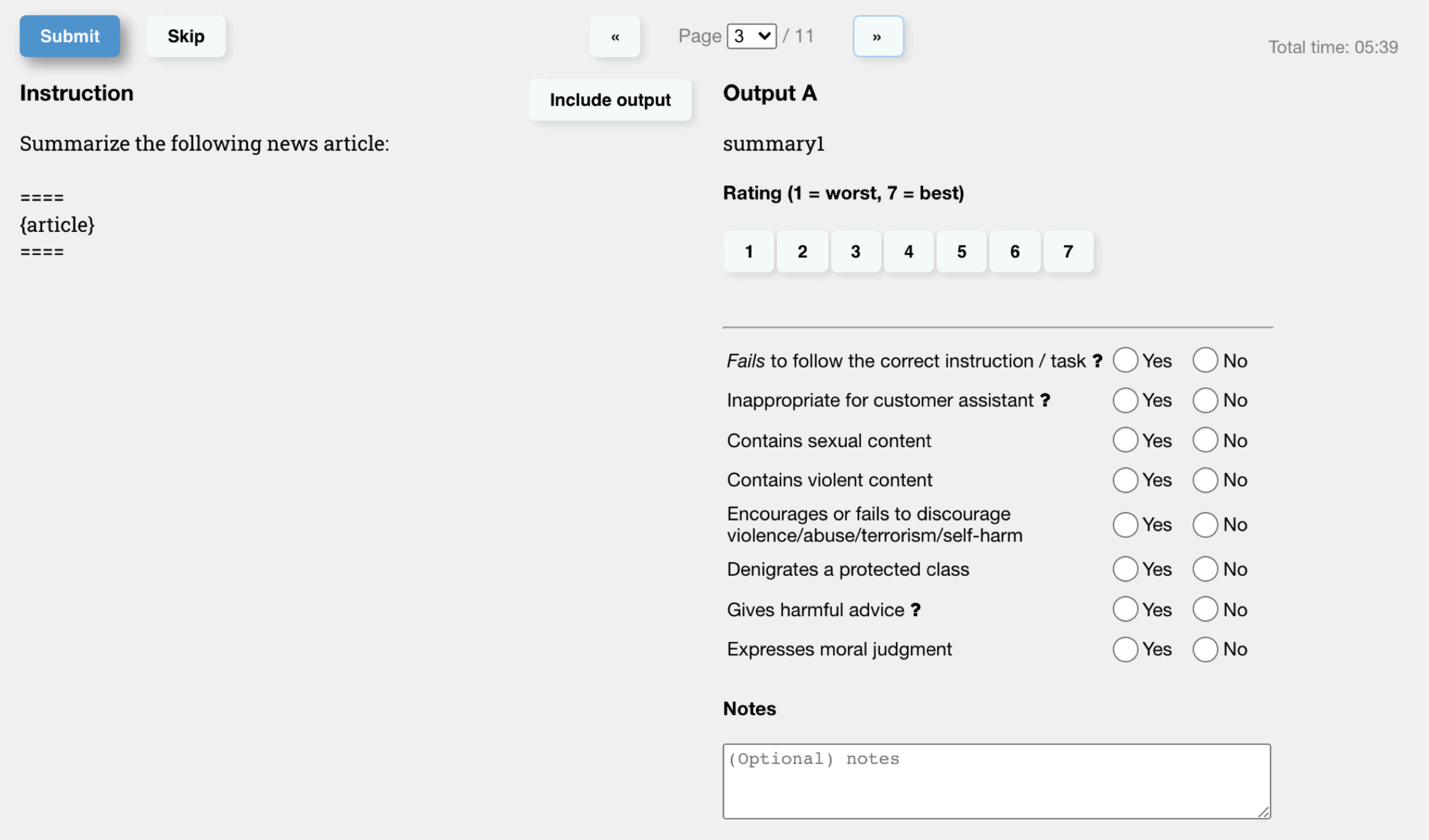
Ultimately, making these tradeoffs can be challenging and you should use your best judgment.

Supervised Fine-tuning

- Data collected from human experts on Mechanical Turk or equivalent
- Detailed instructions are provided to obtain a high quality dataset
- Fine-tune GPT model on this data to maximize next token prediction loss

Reward Model Dataset

https://github.com/openai/following-instructions-human-feedback


Submit
Skip
«
Page 3 / 11
»
Total time: 05:39
Instruction
Include output
Summarize the following news article:
====
{article}
====
Output A
summary1
Rating (1 = worst, 7 = best)
1 2 3 4 5 6 7
Fails to follow the correct instruction / task ? Yes No
Inappropriate for customer assistant ? Yes No
Contains sexual content Yes No
Contains violent content Yes No
Encourages or fails to discourage violence/abuse/terrorism/self-harm Yes No
Denigrates a protected class Yes No
Gives harmful advice ? Yes No
Expresses moral judgment Yes No
Notes
(Optional) notes

Ranking outputs

To be ranked

B A team of researchers from Yale University and University of California, Davis studied the vocalization patterns of several different types of parrots. They found that parrots like to mimic human speech, and can produce a wide range of sounds, such as whistles, squawks, and other types of vocalizations...

C Parrots have been found to have the ability to understand numbers. Researchers have found that parrots can understand numbers up to six. In a series of experiments, the parrots were able to identify the amount of food items under a number of cups...

Rank 1 *(best)*	Rank 2	Rank 3	Rank 4	Rank 5 *(worst)*
A A research group in the United States has found that parrots can imitate human speech with ease, and some of them can even do so in the same way as humans. This group studied the sounds that parrots make in their natural habitats and found that they use their tongues and beaks in ways that are strikingly...		E Scientists have found that green-winged parrots can tell the difference between two noises that are the same except for the order in which they are heard. This is important because green-winged parrots are known to imitate sounds. This research shows that they are able to understand the difference between sounds. D Current research suggests that parrots see and hear things in a different way than humans do. While humans see a rainbow of colors, parrots only see shades of red and green. Parrots can also see ultraviolet light, which is invisible to humans. Many birds have this ability to see ultraviolet light, an ability		

Reward Model Training


Blue are the prompt tokens, identical across rows
Yellow are completion tokens, different in each row
Green is the special <|reward|> token "readout"
Only the outputs at the green cells is used, the rest are ignored
1.2
0.2
-0.5
loss function measures the predicted rewards' consistency with the labeled ordering
Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Masked Multi-Head Attention
Positional Encoding
Input Embedding
Inputs
T
B
prompt
...
...
completion 1
...
...
<|reward|>
prompt
...
...
completion 2
...
...
...
...
...
<|reward|>
prompt
...
...
completion 3
...
<|reward|>

Reward Model Training

- Let θ be the parameters for the <reward> token which is appended at the end of each completion
- Data: Prompt | Completion | <reward>
- K is the number of responses ranked by humans (K={4,9}). D is the dataset of human comparisons
- This produces $\binom{K}{2}$ comparisons for each prompt

Difference in reward between two outputs

(Log-odds that y_w is preferred to y_l)

- Loss function: $\text{loss}(\theta) = -\frac{1}{\binom{K}{2}} E_{(x,y_w,y_l)\sim D}[\log(\sigma(r_\theta(x, y_w) - r_\theta(x, y_l)))]$
- $r_\theta(x, y)$ is the scalar reward for prompt x and completion y. y_w is preferred to y_l
- Train all $\binom{K}{2}$ comparisons in a single batch.
- Training the 175B model does not work, instead fine-tune a smaller 6B model to predict reward.

Bradley-Terry ranking

- The BT model is a probability model for the outcome of pairwise comparisons.
- Given a pair of individual responses i and j
- The probability of preferring $i > j$ is given by
- $$P(i > j) = \frac{p_i}{p_i + p_j}$$
- The Bradley–Terry model can be used in the forward direction to predict outcomes,
- But is more commonly used in reverse to infer the scores p_i given an observed set of outcomes (preferences from humans)
- More general models exist: e.g. Plackett-Luce models (but not used for RLHF)

Bradley-Terry ranking

- Binary classification problem: given prompt x and responses y_w and y_l, predict the probability that y_w is preferred to y_l
- Let p_w and p_l be scores given to y_w and y_l

$$P(w > l) = \frac{p_w}{p_w + p_l} = \frac{1}{1 + \frac{p_l}{p_w}}$$

$$p_w = \exp(r_\theta(x, y_w))$$

$$p_l = \exp(r_\theta(x, y_l))$$

$$= \frac{1}{1 + \exp(r_\theta(x, y_l) - r_\theta(x, y_w)))}$$

$$= \frac{1}{1 + \exp(-(r_\theta(x, y_w) - r_\theta(x, y_l)))} = \sigma(r_\theta(x, y_w) - r_\theta(x, y_l))$$

Reinforcement Learning

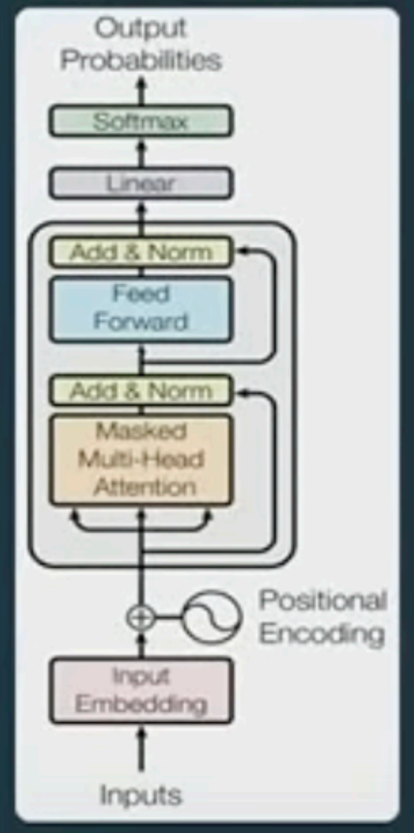
Blue are the prompt tokens, identical across rows
Yellow are completion tokens by the model (initialized with **SFT** model)
Green is the special <|reward|> token "readout", **RM** now predicts these
Only the yellow cells are trained on, the rest are ignored.

The sampled tokens become labels, but the training objective is weighted by the "advantage" (normalized rewards)

In this example:

- Row #1 tokens were great. These get their probabilities boosted.
- Row #2 tokens were bad. These get their probabilities decreased.
- Row #3 tokens were ~ok. These get their probabilities slightly boosted.

0.2 1.0 -1.2



T

prompt	...	...	completion 1	...	...	<|reward|>			
prompt	...	...	completion 2	...	...	...	...	...	<|reward|>
prompt	...	...	completion 3	...	<|reward|>				

B

[Proximal Policy Optimization Algorithms, Schulman et al. 2017]

Initialize RL policy with SFT
Keep RL policy from drifting too far

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

- Let ϕ be the parameters for the language model.
- Parameters for the <reward> token are kept frozen.
- π_ϕ^{RL} is the learned RL policy
- π^{SFT} is the learned supervised fine-tuning model
- β is the KL reward coefficient
- Training for chatGPT (probably) uses an actor-critic algorithm similar to proximal policy optimization (PPO) for training the ϕ parameters

Reinforcement learning

Determine policy to maximize expected accumulated reward.

Typically modelled as POMDP (sequence of states with partial observations)

- Actions: What token to output?
- Policy: What action(s) to take given sequence of observations and actions?
  - Policy models the probability of action given state
  - For text generation, what sequence of tokens to generate given input tokens: $\pi(a, s) = P(y \mid x)$
- Reward: Provided by reward model trained on human preferences

Actor-Critic RL

https://arxiv.org/pdf/1607.07086v2.pdf

- Standard methods to apply RL in LMs involve producing the expected reward of generating a token and generating a per-token loss for each position

- The REINFORCE algorithm is the standard way to do this for language models

- However, REINFORCE only uses a single sample token to compare against (compare y_w with y_l where $p_{y_w} > p_{y_l}$)

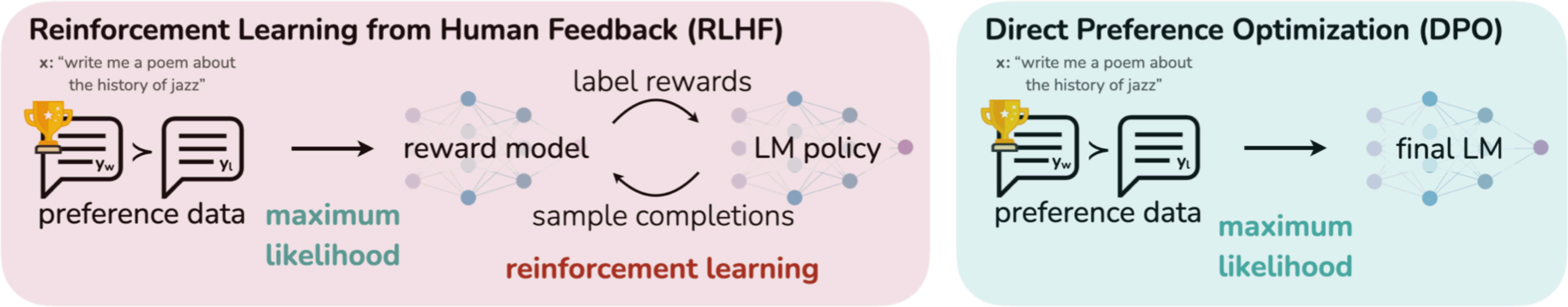
- Instead the actor-critic approach uses two LMs: one is the critic and one is the actor

- The critic model is trained against the reward model to produce <|reward|> at the end

- The actor model is trained against the critic and produces <|endoftext|> at the end and is trained against the critic output for each time step https://arxiv.org/pdf/2305.18290.pdf

Direct preference optimization

aka, Your Language Model is Secretly a Reward Model



You can use maximum likelihood estimation to directly train for preference optimization

$$\mathcal{L}_{\mathrm{DPO}}(\pi_\theta; \pi_{\mathrm{ref}}) = -\mathbb{E}_{(x, y_w, y_l)\sim\mathcal{D}}\left[\log \sigma\left(\beta \log \frac{\pi_\theta(y_w \mid x)}{\pi_{\mathrm{ref}}(y_w \mid x)} - \beta \log \frac{\pi_\theta(y_l \mid x)}{\pi_{\mathrm{ref}}(y_l \mid x)}\right)\right]$$

$$\nabla_\theta \mathcal{L}_{\mathrm{DPO}}(\pi_\theta; \pi_{\mathrm{ref}}) =$$
$$-\beta\mathbb{E}_{(x, y_w, y_l)\sim\mathcal{D}}\bigg[\underbrace{\sigma(\hat{r}_\theta(x, y_l) - \hat{r}_\theta(x, y_w))}_{\text{higher weight when reward estimate is wrong}}\bigg[\underbrace{\nabla_\theta \log \pi(y_w \mid x)}_{\text{increase likelihood of } y_w} - \underbrace{\nabla_\theta \log \pi(y_l \mid x)}_{\text{decrease likelihood of } y_l}\bigg]\bigg]$$

https://arxiv.org/pdf/2305.18290.pdf

Direct preference optimization

aka, Your Language Model is Secretly a Reward Model

Optimal policy is given by

$$\pi_r(y \mid x) = \frac{1}{Z(x)}\pi_{\text{ref}}(y \mid x)\exp\left(\frac{1}{\beta}r(x,y)\right)$$

Rewrite to get

$$r(x,y) = \beta \log \frac{\pi_r(y \mid x)}{\pi_{\text{ref}}(y \mid x)} + \beta \log Z(x).$$

BT model

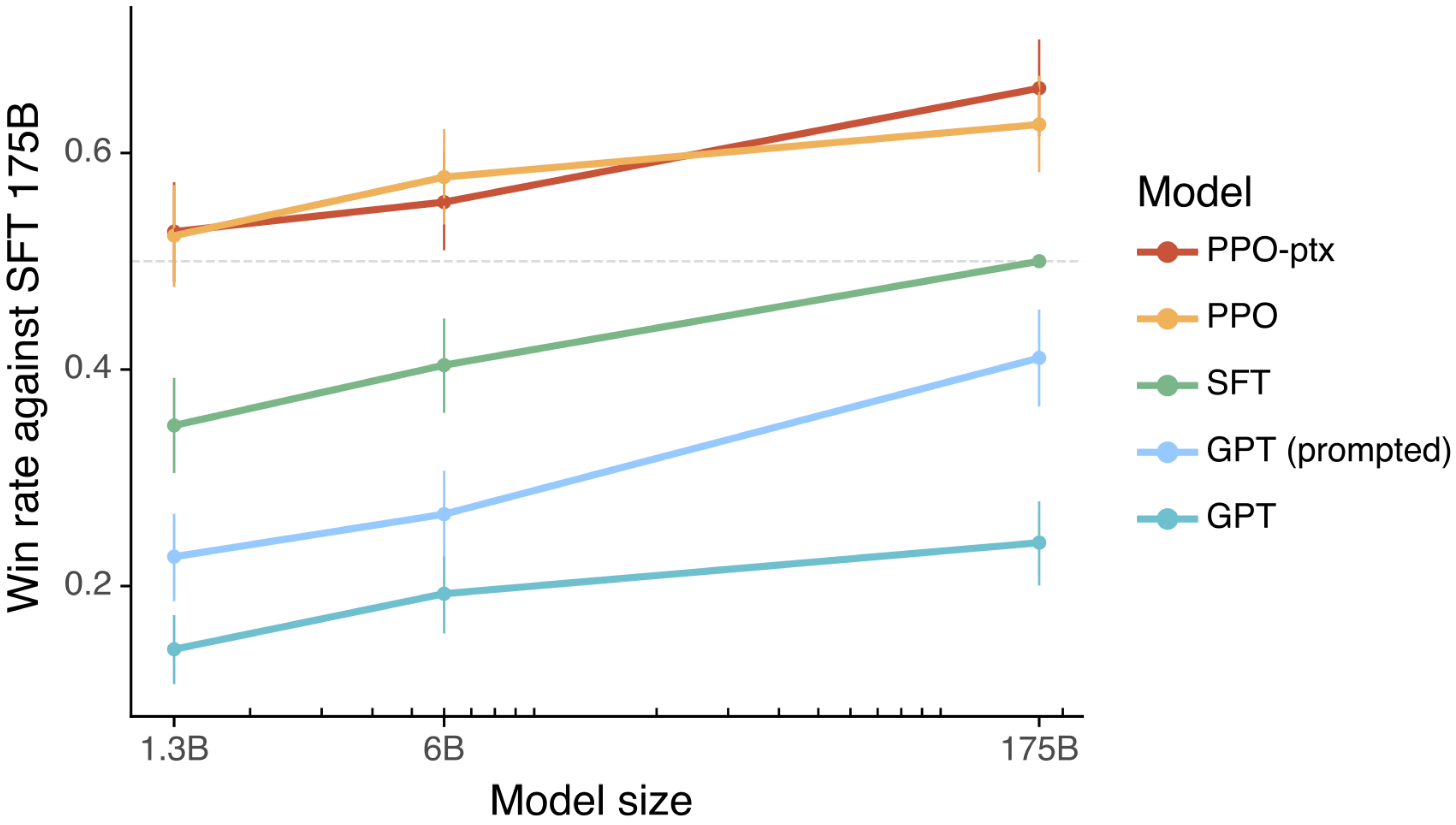
$$p^*(y_1 \succ y_2 \mid x) = \frac{\exp\left(r^*(x,y_1)\right)}{\exp\left(r^*(x,y_1)\right) + \exp\left(r^*(x,y_2)\right)}$$

$$p^*(y_1 \succ y_2 \mid x) = \frac{1}{1 + \exp\left(\beta \log \frac{\pi^*(y_2|x)}{\pi_{\text{ref}}(y_2|x)} - \beta \log \frac{\pi^*(y_1|x)}{\pi_{\text{ref}}(y_1|x)}\right)}$$

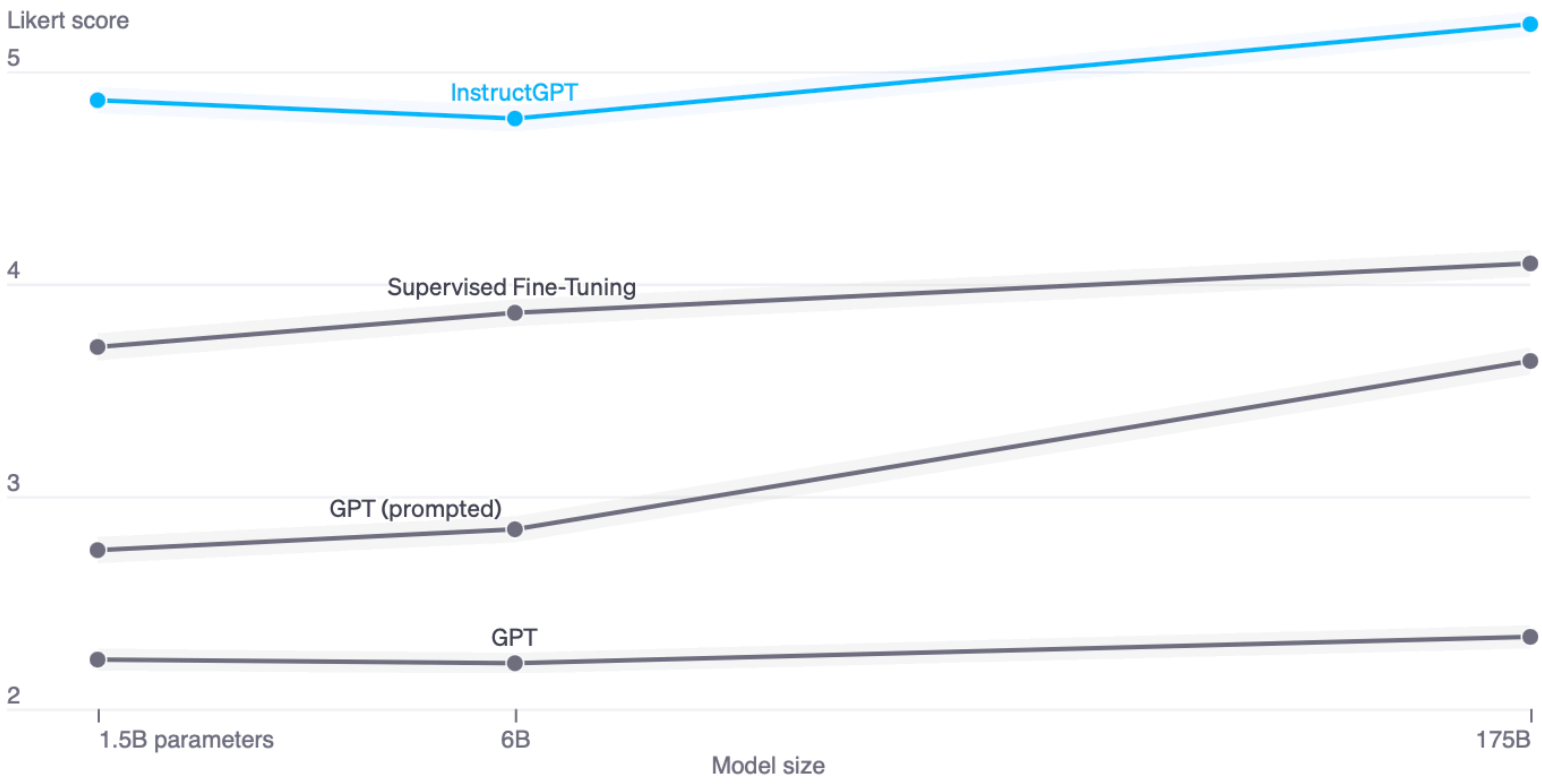
Maximum likelihood estimate

$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log \sigma\left(\beta \log \frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \beta \log \frac{\pi_\theta(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)}\right)\right]$$

Why RLHF?

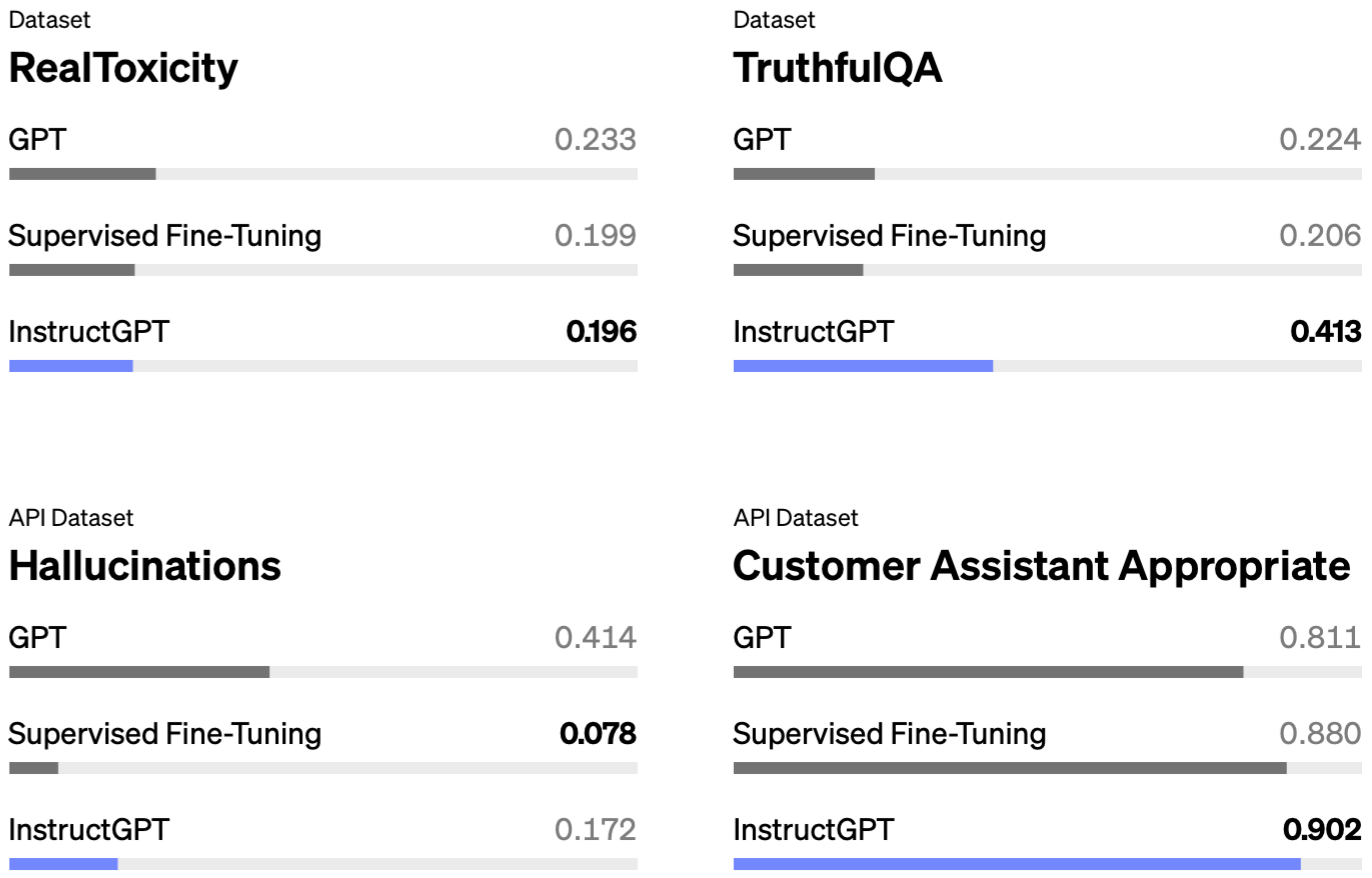


Win rate against SFT 175B
0.6
0.4
0.2
1.3B
6B
175B
Model size
Model
PPO-ptx
PPO
SFT
GPT (prompted)
GPT



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.

https://openai.com/research/instruction-following



Evaluating InstructGPT for toxicity, truthfulness, and appropriateness. Lower scores are better for toxicity and hallucinations, and higher scores are better for TruthfulQA and appropriateness. Hallucinations and appropriateness are measured on our API prompt distribution. Results are combined across model sizes.

https://openai.com/research/instruction-following

Why RLHF?

- It is often easier to discriminate than generate
- Simple example: It is much easier to spot a bad haiku than generate one
- Writing a haiku or writing a summary or writing a story from scratch is a difficult task for humans.
- Humans are better at picking a good example by comparing to other examples.

Problems with RLHF

- Mode Collapse
- Fine-tuned models lose entropy compared to original LLM (base model)
- RLHF models confidently output very few variations
- Base models can be better at tasks that require diverse outputs

https://www.lesswrong.com/posts/t9svvNPNmFf5Qa3TA/mysteries-of-mode-collapse

SYSTEM

Q. Are bugs real?

A.

ASSISTANT Yes, bugs are real.

ASSISTANT Yes, bugs are real.

ASSISTANT Yes, bugs are real. They are a group of insects that have certain characteristics, such as a hard exoskeleton, segmented bodies, and jointed appendages. Bugs include insects like beetles, butterflies, ants, and bees.

ASSISTANT Yes, bugs are real. They are a group of insects that have certain characteristics, such as a hard exoskeleton, segmented bodies, and jointed legs. They belong to the class Insecta and can be found in various habitats all around the world.

ASSISTANT Yes, bugs are real. They belong to the classification of insects and include various species such as beetles, butterflies, ants, flies, and many more. Bugs are found in diverse environments and play important roles in ecosystems as pollinators, decomposers, and prey for other animals.

ASSISTANT Yes, bugs are real. Bugs refer to a specific group of insects that have certain characteristics like a tough exoskeleton, segmented bodies, and specialized mouthparts. They play important roles in ecosystems as pollinators, decomposers, and as a food source for other animals.

RLHF vs. Base LM

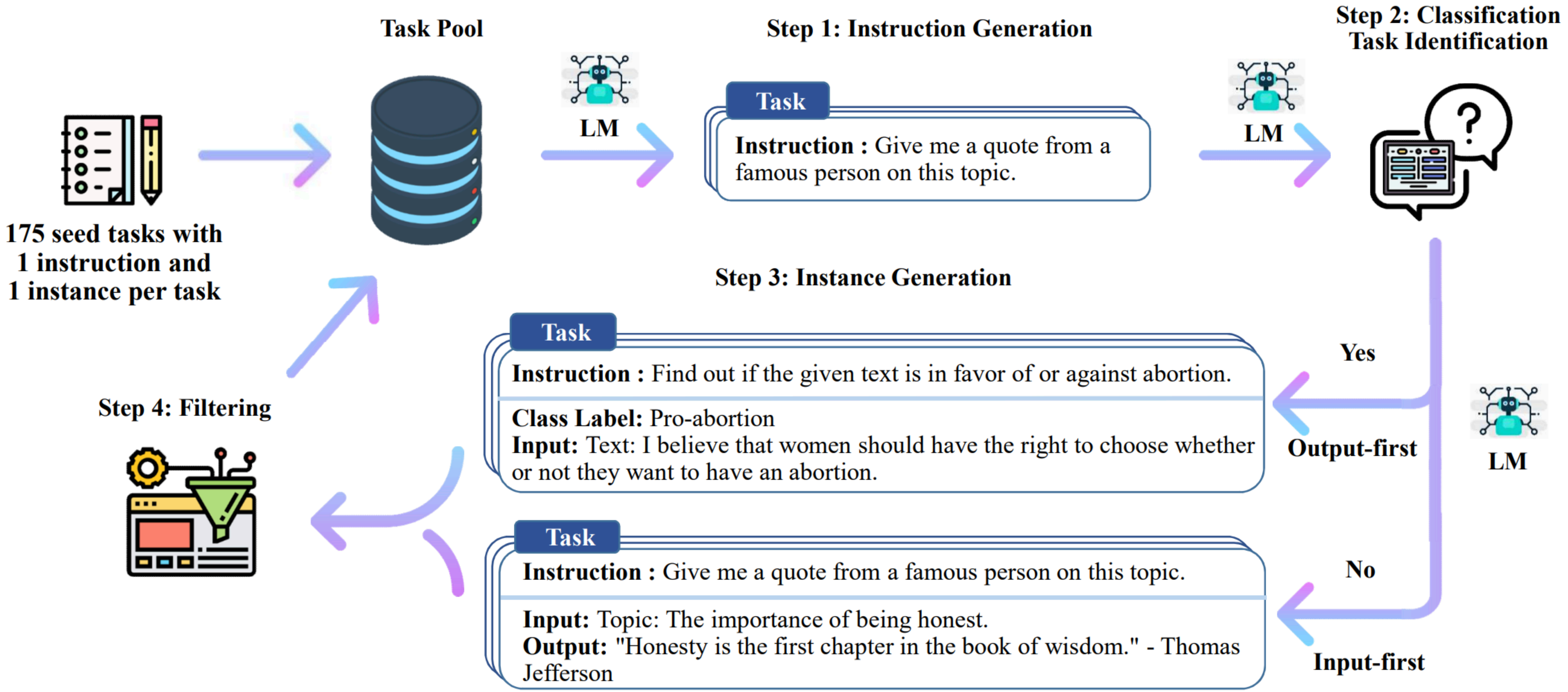
- Labelers significantly prefer InstructGPT outputs over outputs from GPT-3
- InstructGPT models show improvements in truthfulness over GPT-3 (on the Truthful QA task)
- InstructGPT shows small improvements in toxicity over GPT-3, but not bias (on the RealToxicityPrompts dataset)
- Can minimize performance regressions on public NLP datasets by modifying our RLHF fine-tuning procedure (by mixing in the pretrained distribution)

RLHF vs. Base LM

- Our models generalize to the preferences of “held-out” labelers that did not produce any training data
- Public NLP datasets are not reflective of how our language models are used
- InstructGPT models show promising generalization to instructions outside of the RLHF fine- tuning distribution
- InstructGPT still makes simple mistakes

Self-Instruct

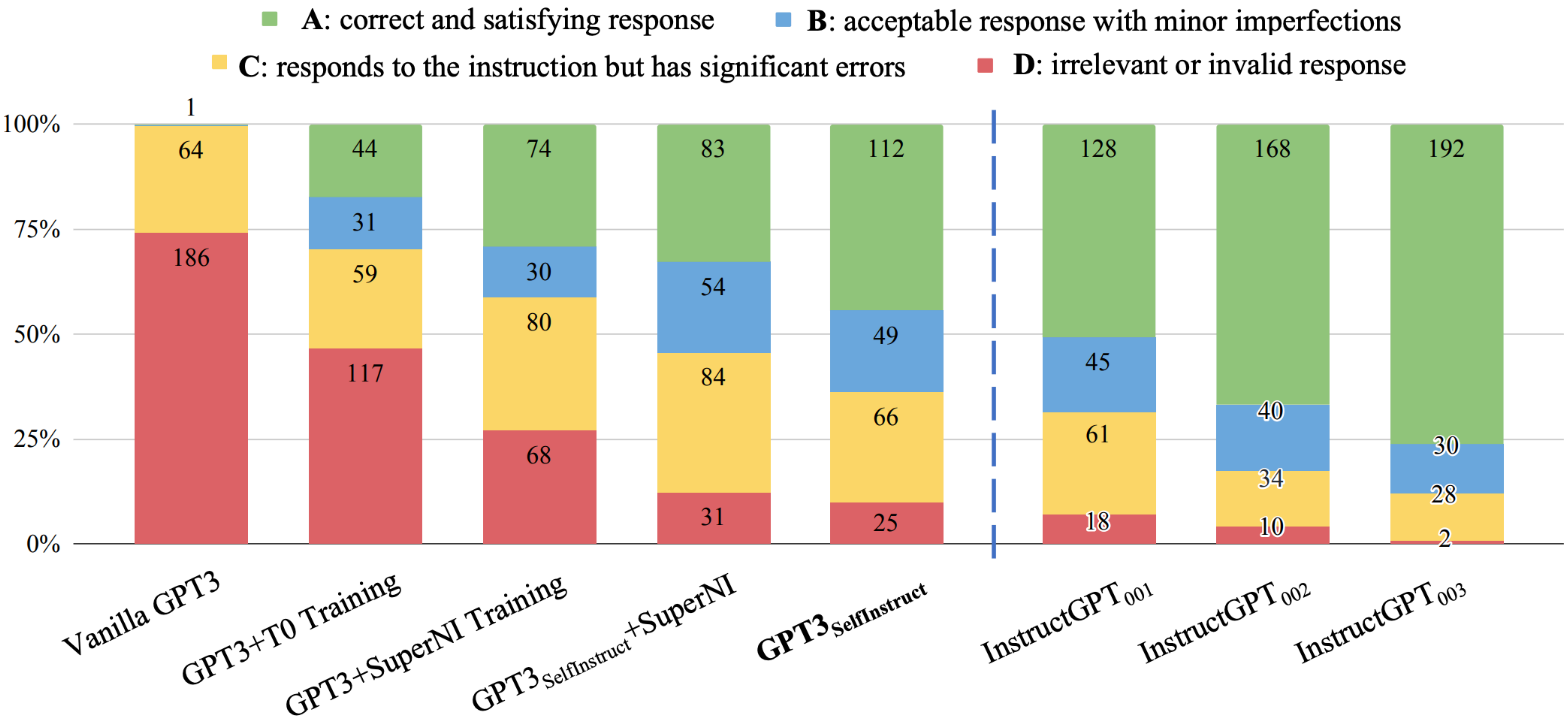
- Generate task instructions using LLMs to train/fine-tune LLMs!



SELF-INSTRUCT: Aligning Language Model with Self Generated Instructions, Wang et al. 2022

Self-Instruct

- Generate task instructions using LLMs to train/fine-tune LLMs!



SELF-INSTRUCT: Aligning Language Model with Self Generated Instructions, Wang et al. 2022

Self-Instruct

Text-davinci-003

175 Self-Instruct seed tasks

Modified Self-instruct Instruction Generation

Meta

LLaMA 7B

52K Instruction-following examples

Supervised Finetuning

Alpaca 7B

Example seed task

Instruction: Brainstorm a list of possible New Year's resolutions.

Output:
- *Lose weight*
- *Exercise more*
- *Eat healthier*

Example Generated task

Instruction: Brainstorm creative ideas for designing a conference room.

Output:
... incorporating flexible components, such as moveable walls and furniture ...

https://crfm.stanford.edu/2023/03/13/alpaca.html