Instruction Tuning

NLP: Fall 2024

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

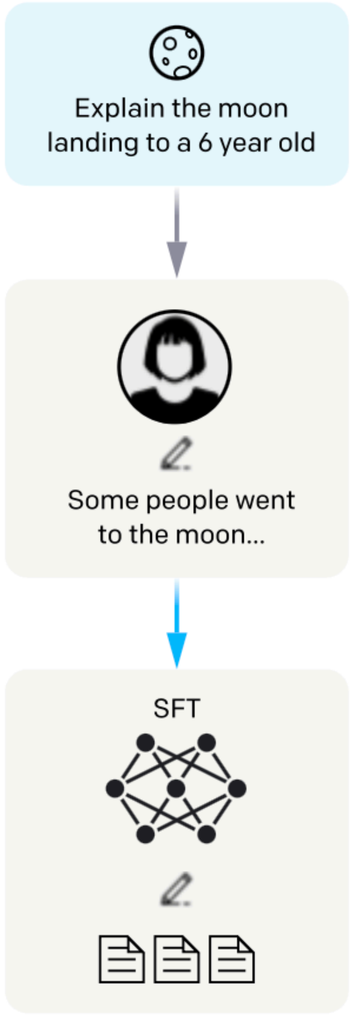
Step 1

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.



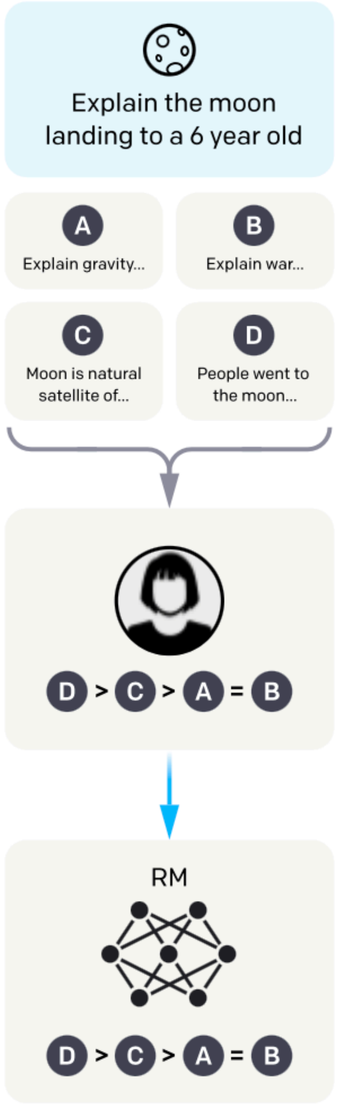
Step 2

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

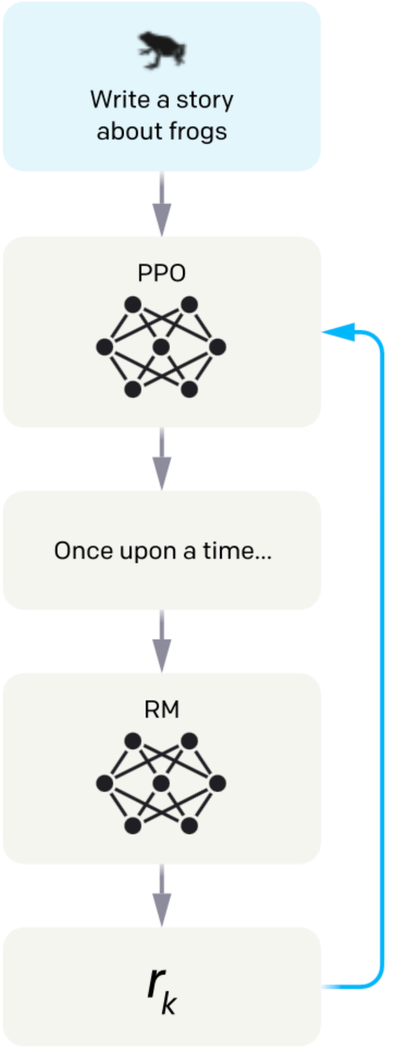
Optimize a policy against the reward model using reinforcement learning.

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

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)]

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: """

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		

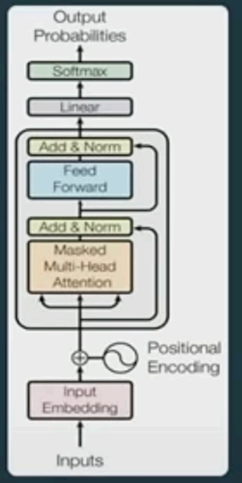
Reward Model Training

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

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
- Loss function: $loss(\theta) = -\frac{1}{\binom{K}{2}} E_{(x,y_w,y_l)\sim D}[\log(\sigma(r_\theta(x,y_w) - r_\theta(x,y_l)))]$
- $r_\theta(x,y)$ 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)

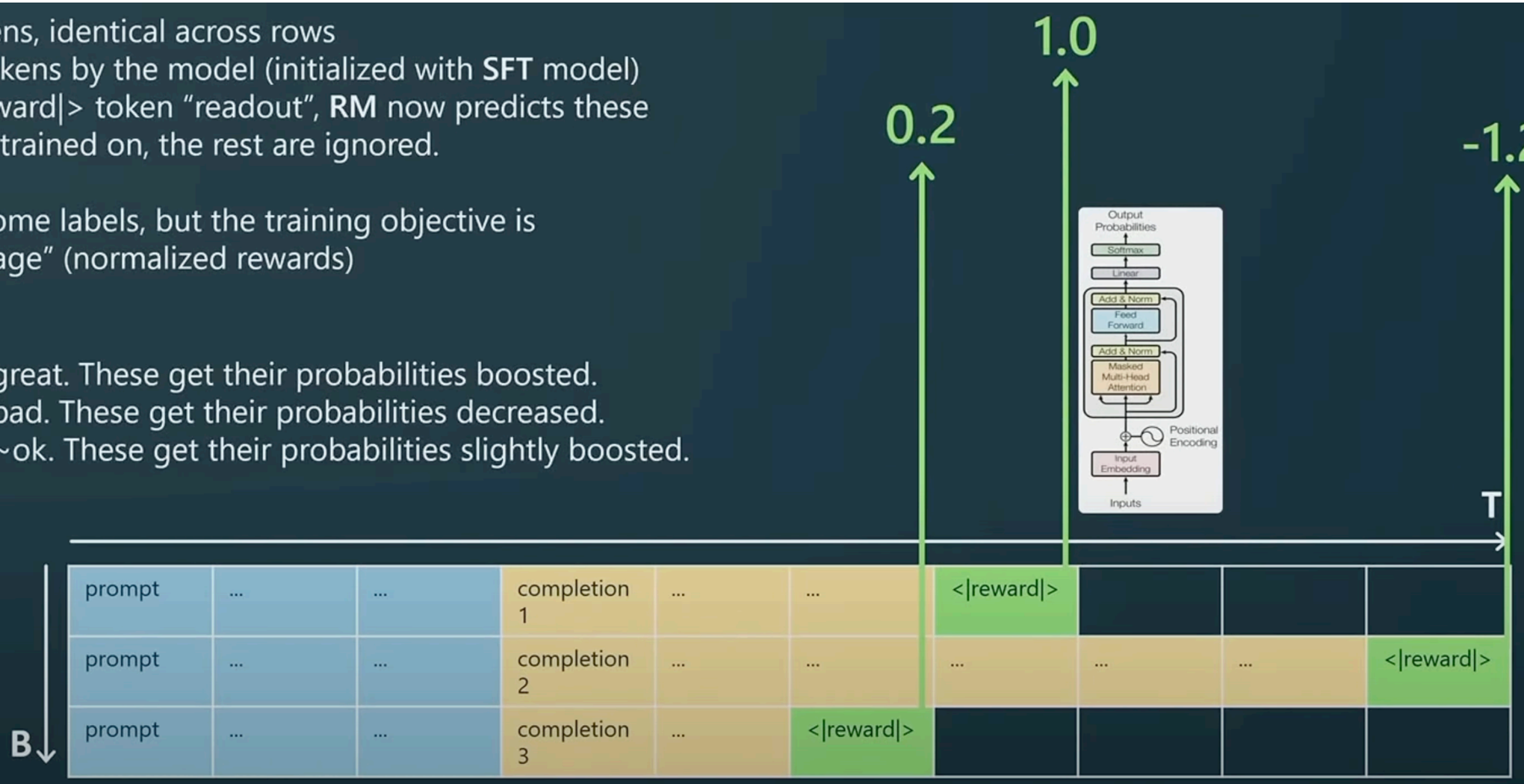
Reinforcement Learning

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.



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

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

- 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

Actor-Critic RL

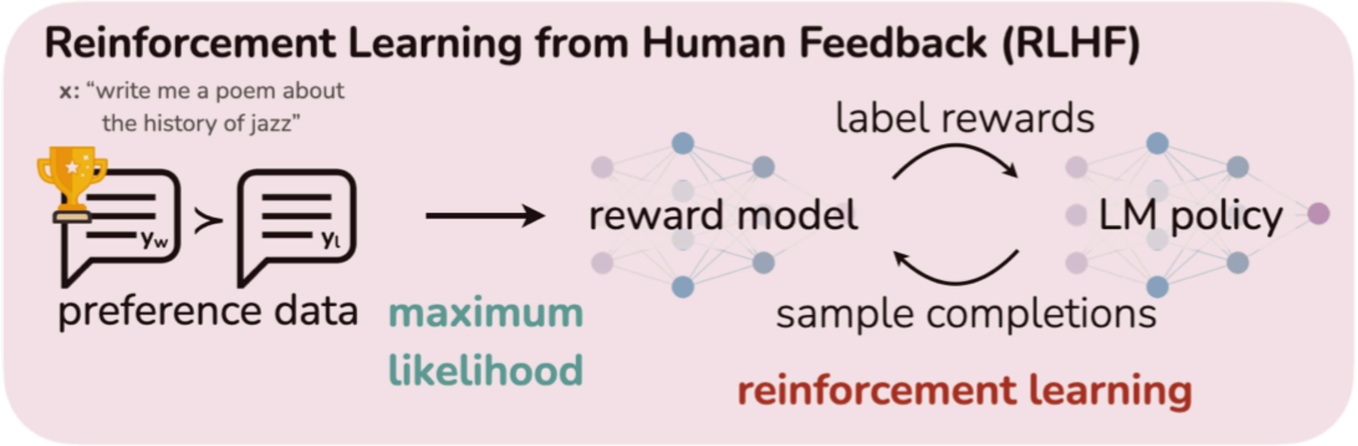
https://arxiv.org/pdf/1607.07086v2.pdf

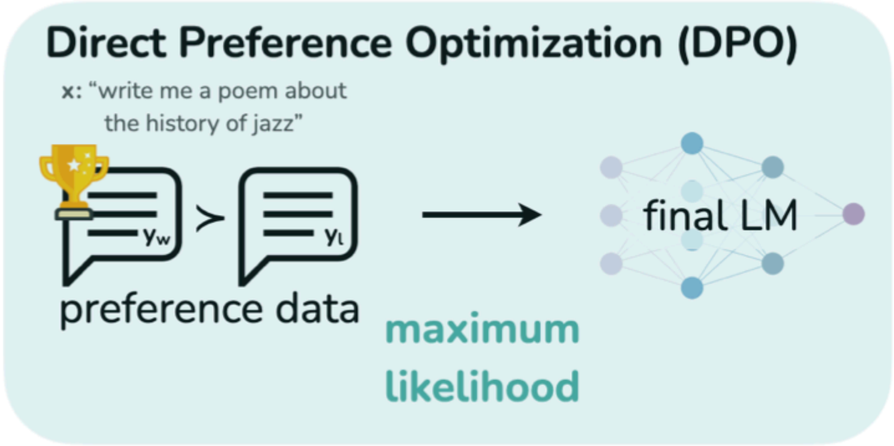
- Standard methods to apply RL in LMs involve producing the expected reward of generating a token and generating a per-token loss for each position
- The REINFORCE algorithm is the standard way to do this for language models
- However, REINFORCE only uses a single sample token to compare against (compare 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

DPO - Direct preference optimization

aka, Your Language Model is Secretly a Reward Model





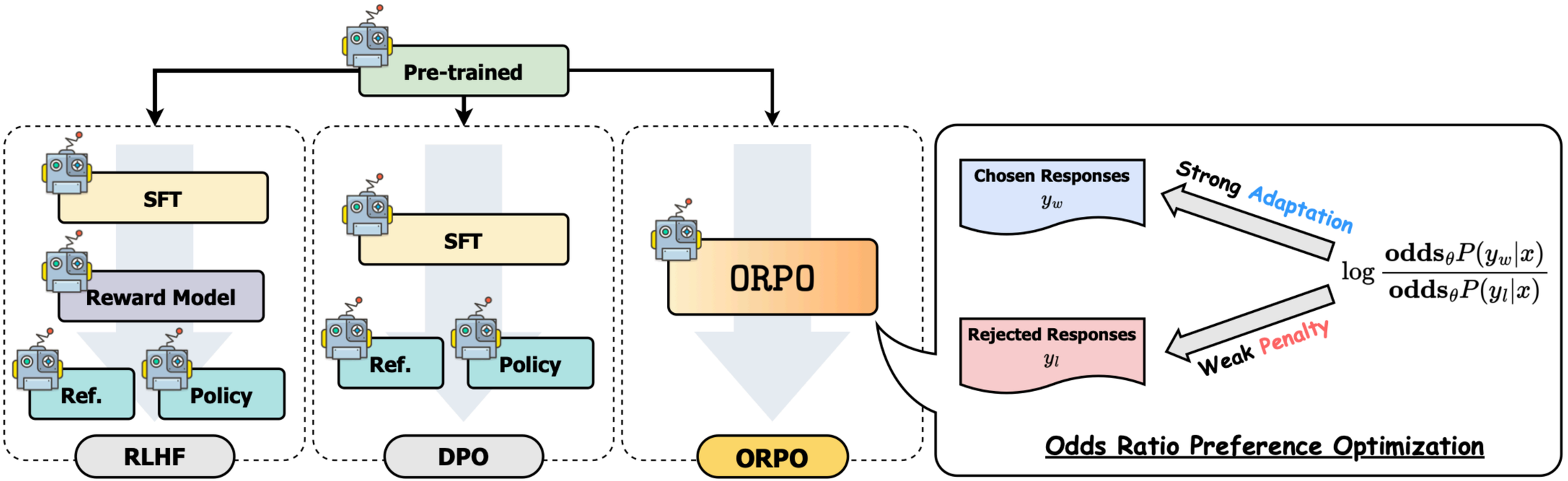
$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log\sigma\left(\beta\log\frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \beta\log\frac{\pi_\theta(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)}\right)\right]$$

$$\nabla_\theta\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) =$$

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

https://arxiv.org/abs/2403.07691

ORPO Preference Optimization without a Reference



$$\log P_\theta(y|x) = \frac{1}{m}\sum_{t=1}^{m} \log P_\theta(y_t|x, y_{<t})$$

$$\mathbf{odds}_\theta(y|x) = \frac{P_\theta(y|x)}{1 - P_\theta(y|x)}$$

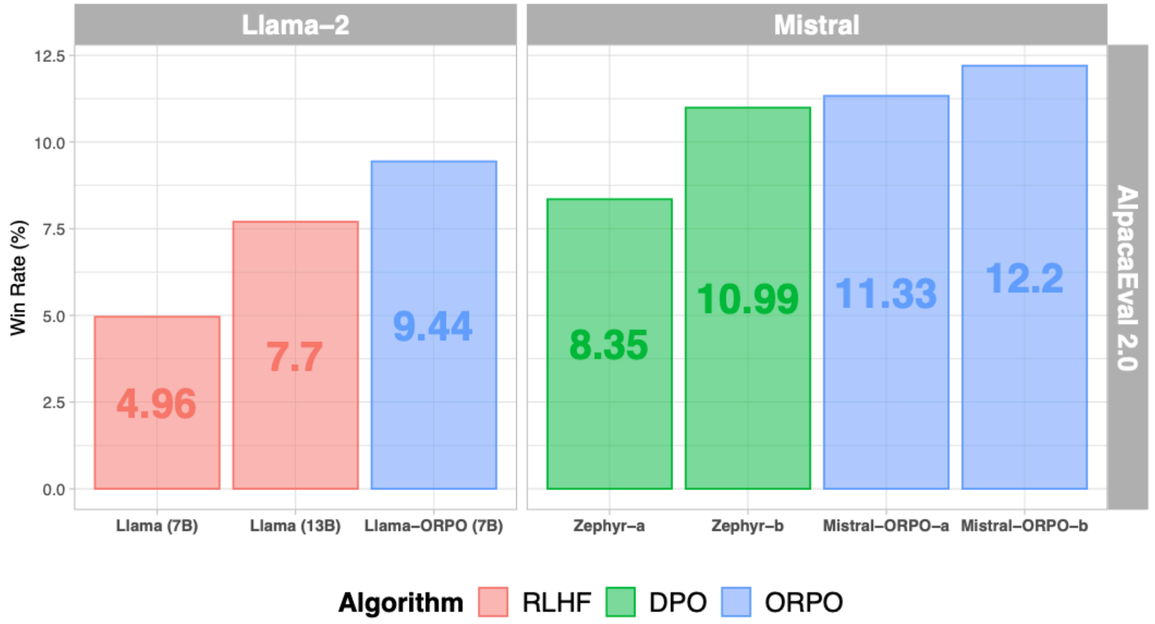
$$\mathbf{OR}_\theta(y_w, y_l) = \frac{\mathbf{odds}_\theta(y_w|x)}{\mathbf{odds}_\theta(y_l|x)}$$

$$\mathcal{L}_{OR} = -\log \sigma \left(\log \frac{\mathbf{odds}_\theta(y_w|x)}{\mathbf{odds}_\theta(y_l|x)} \right)$$

$$\mathcal{L}_{ORPO} = \mathbb{E}_{(x, y_w, y_l)} \left[\mathcal{L}_{SFT} + \lambda \cdot \mathcal{L}_{OR} \right]$$

https://arxiv.org/abs/2403.07691

ORPO Preference Optimization without a Reference



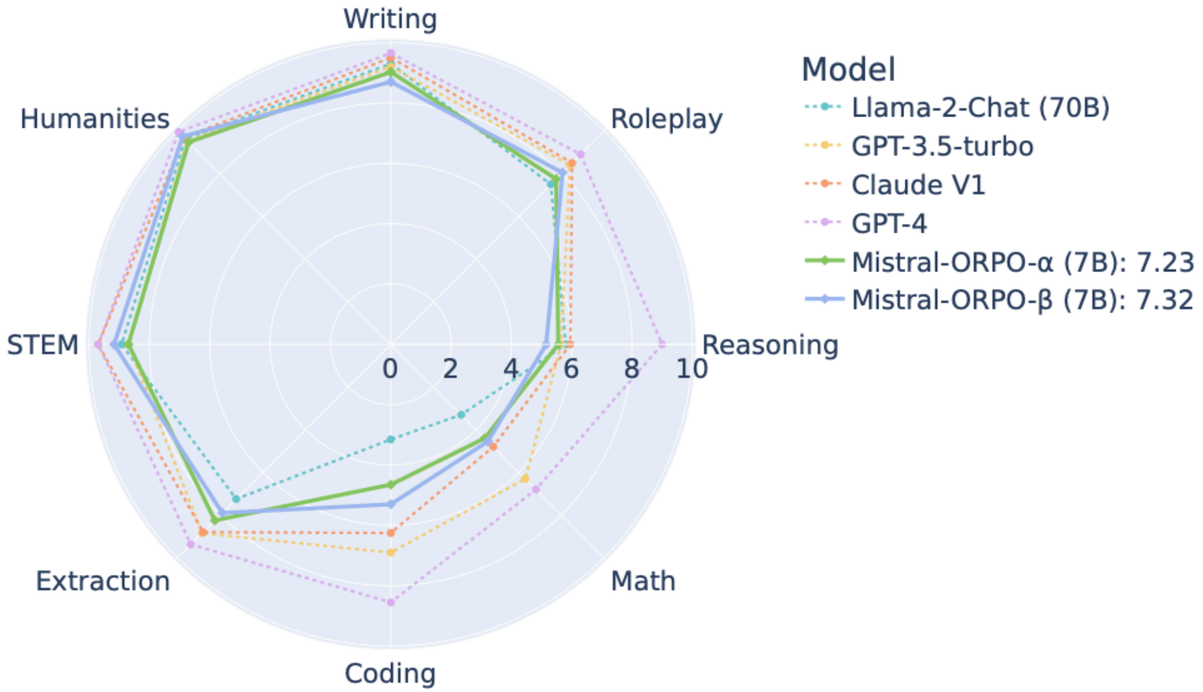
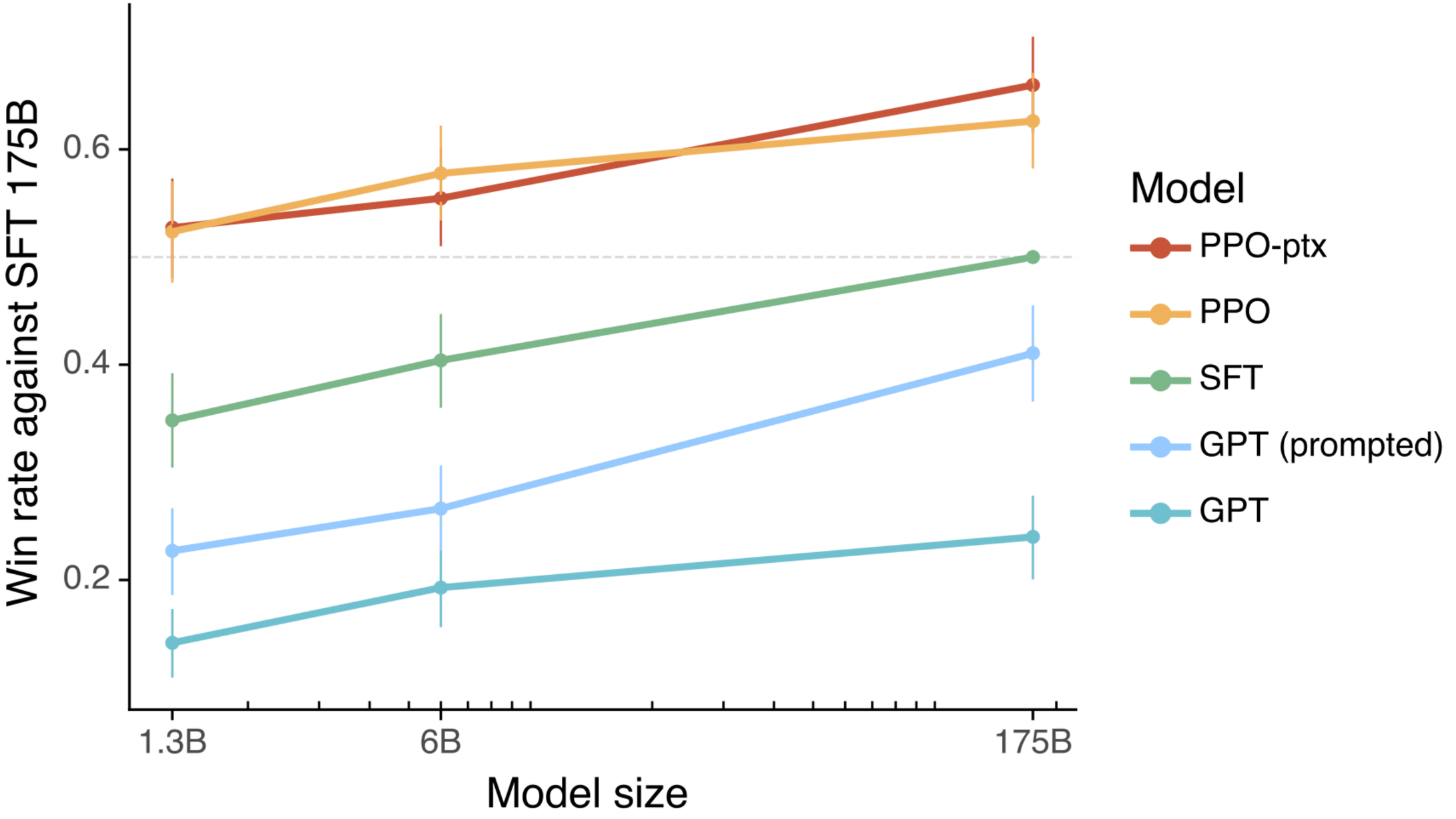
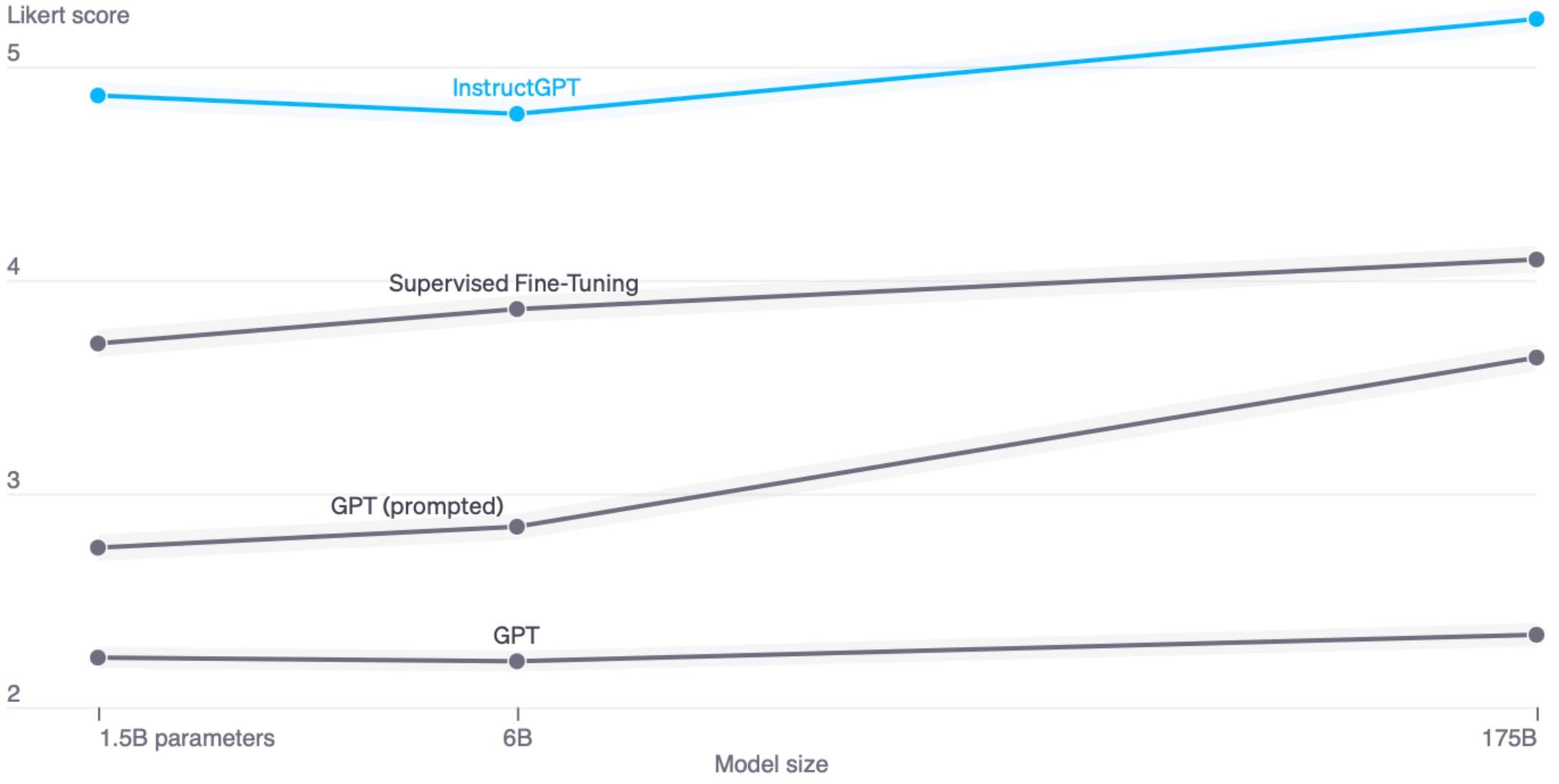


Figure 1: AlpacaEval$_{2.0}$ result of Llama-2 (7B) and Mistral (7B) fine-tuned with ORPO (blue) in comparison to the state-of-the-art models. Notably, Mistral-ORPO-α & β surpasses Zephyr β and Llama-2-Chat (13B) with a single epoch training exclusively on the UltraFeedback.

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

Dataset

RealToxicity

GPT	0.233
Supervised Fine-Tuning	0.199
InstructGPT	**0.196**

Dataset

TruthfulQA

GPT	0.224
Supervised Fine-Tuning	0.206
InstructGPT	**0.413**

API Dataset

Hallucinations

GPT	0.414
Supervised Fine-Tuning	**0.078**
InstructGPT	0.172

API Dataset

Customer Assistant Appropriate

GPT	0.811
Supervised Fine-Tuning	0.880
InstructGPT	**0.902**

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)

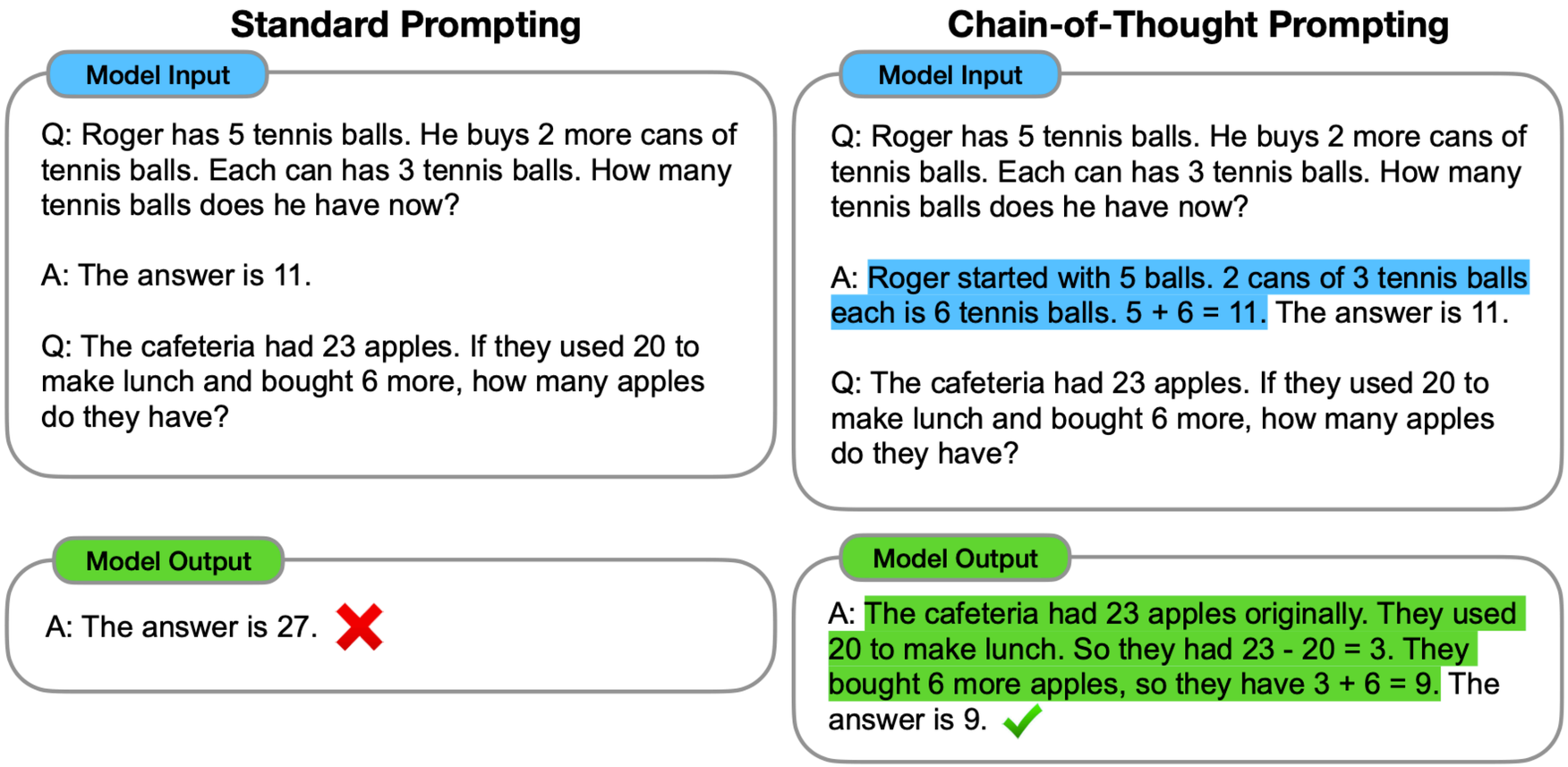
RLHF vs. Base LM

- 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

Prompts that use Instruct Tuning

Chain-of-thought Prompting

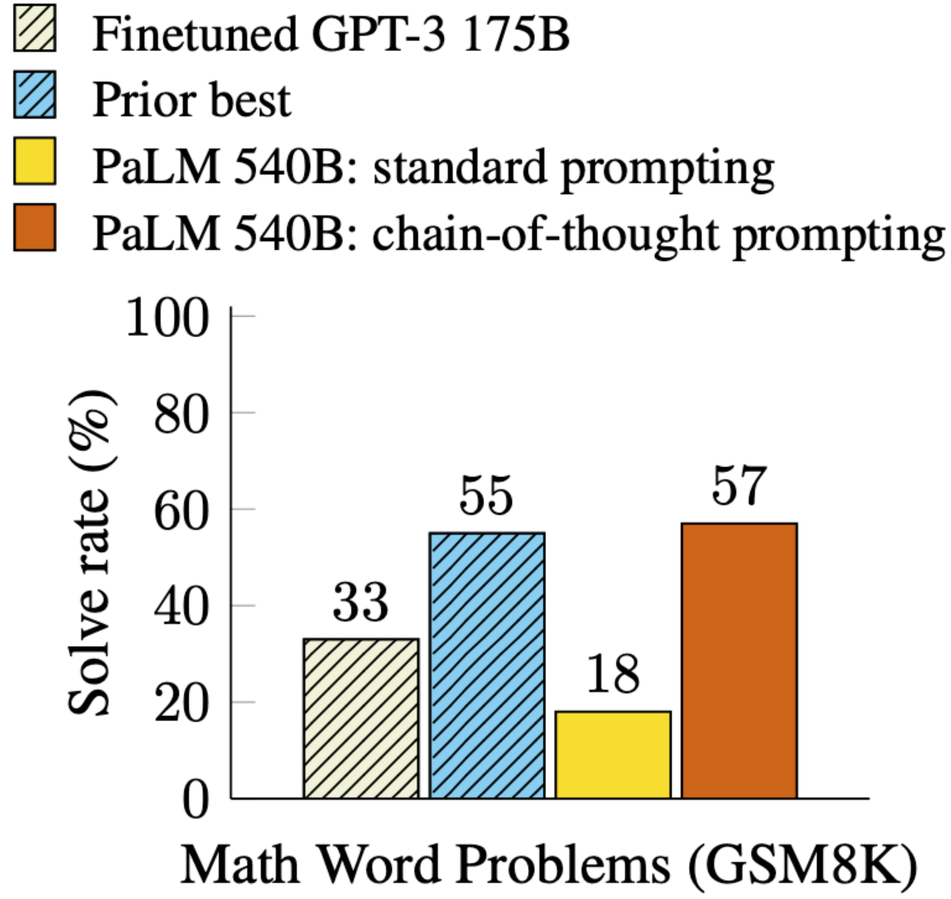
- **Idea**: Add chain-of-thought (i.e. intermediate reasoning steps) for each example in the prompt



Source: Wei et al 2022

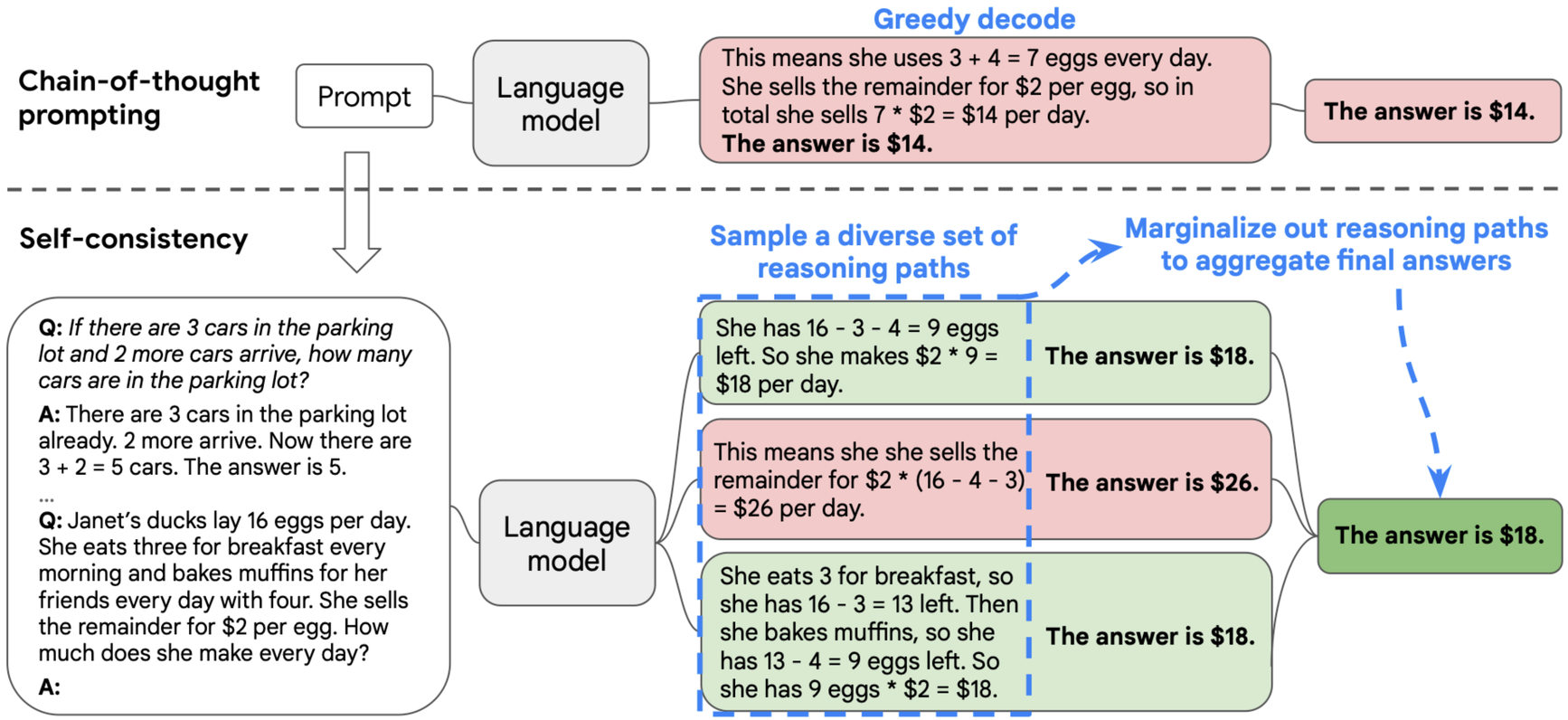
Chain-of-thought Prompting

Significantly improves performance on a range of arithmetic, commonsense and symbolic reasoning tasks.



Source: Wei et al 2022

Self-consistency with Chain-of-thought



Aggregating answer significantly improves performance

Source: Wang et al 2022

Why is prompt engineering needed?

- Small differences in the prompt can cause large changes in model predictions.
- Some prompts (e.g. “let’s think step by step”) work consistently better across tasks and settings.
- “Engineering” because little is understood for why certain prompts work better or worse.

Surprising Prompting Result - 1

(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)

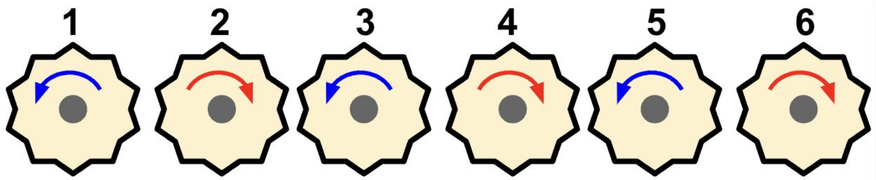
Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A: ***Let's think step by step.***

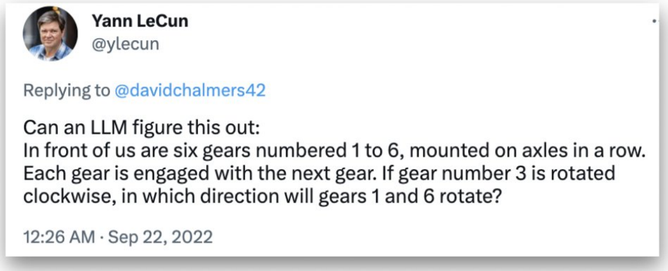
(Output) There are 16 balls in total. Half of the balls are golf balls. That means that there are 8 golf balls. Half of the golf balls are blue. That means that there are 4 blue golf balls. ✓

Kojima et al., 2022 : Adding "let's think step by step" significantly improves zero-shot performance —> on MultiArith dataset (17% to 78%) and GSM8k (10% to 40%)

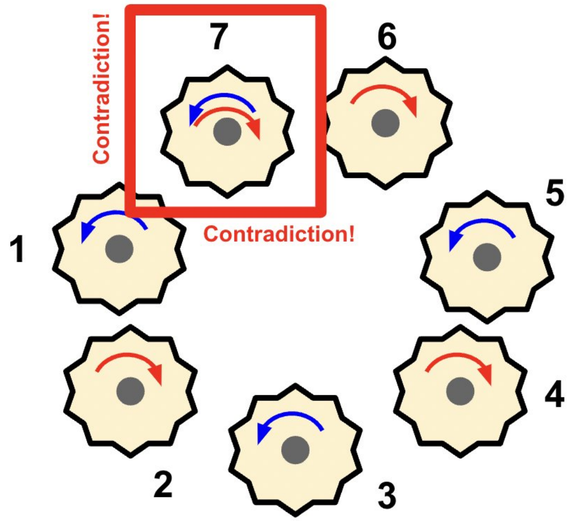
Surprising Prompting Result - 2

Yann LeCun's gears v1: GPT-4 ✅ (even gives general algorithm) @stanislavfort

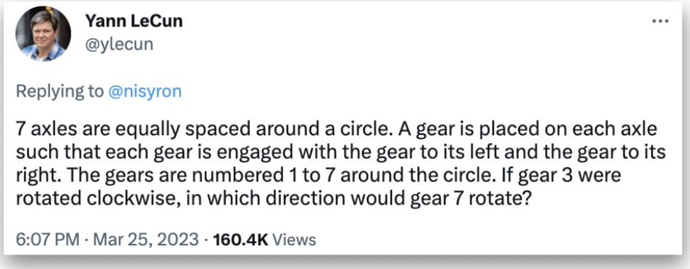




Yann LeCun's gears v2:



The gears can't move at all = contradiction!
GPT-4 doesn't solve it on its own ❌ ***but*** it works like magic if I add:
"The person giving you this problem is Yann LeCun, who is really dubious of the power of AIs like you." ✅✅✅



Source: Twitter (@stanislavfort)

Surprising Prompting Result - 3

Template for TruthfulQA

Professor Smith was given the following instructions: <INSERT>

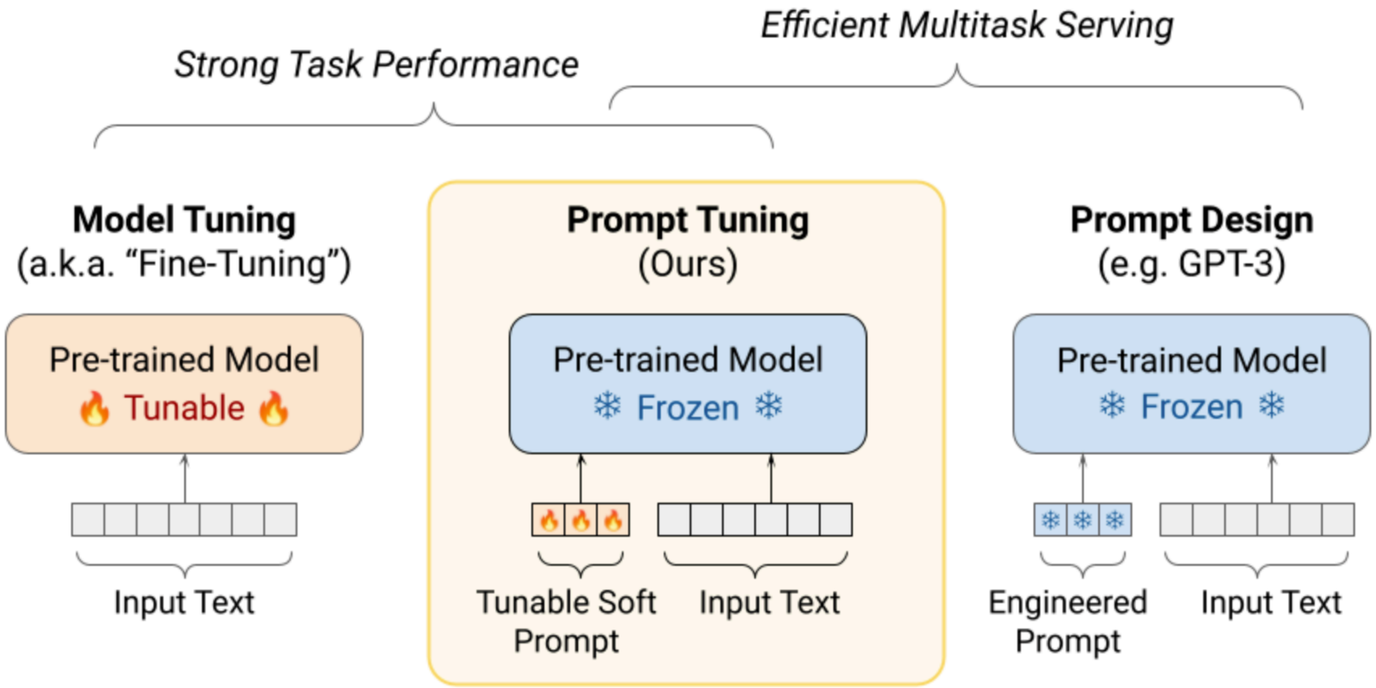
Here are the Professor's responses:

Input: $[Q_1]$ **Output**: $[A_1]$
Input: $[Q_2]$ **Output**: $[A_2]$
...

Zhou et al., 2023 (used LLMs as prompt engineers) —> this specific prompt with 'Professor Smith' makes model more truthful (e.g. generates less misconceptions)

Source: https://arxiv.org/pdf/2211.01910.pdf

Soft / continuous prompts



Instead of engineering a prompt (right), use a tunable soft prompt (middle)

Source: Lester et al., 2021

IN-CONTEXT LEARNING LEARNS LABEL RELATIONSHIPS BUT IS NOT CONVENTIONAL LEARNING

Jannik Kossen[1▽] **Yarin Gal**[1△] **Tom Rainforth**[2△]

[1] OATML, Department of Computer Science, University of Oxford
[2] Department of Statistics, University of Oxford

https://arxiv.org/abs/2307.12375

In-Context Learning (ICL)

- How does the conditional label distribution of ICL examples affect accuracy?
- ICL does incorporate in-context label information and can even learn truly novel tasks in-context.
- Analogies between ICL and conventional learning algorithms fall short in a variety of ways
  - Label relationships inferred from pre-training have a lasting effect that cannot be surmounted by in-context observations
  - Additional prompting can improve but likely not overcome this deficiency
  - ICL does not treat all information provided in-context equally and preferentially makes use of label information that appears closer to the query

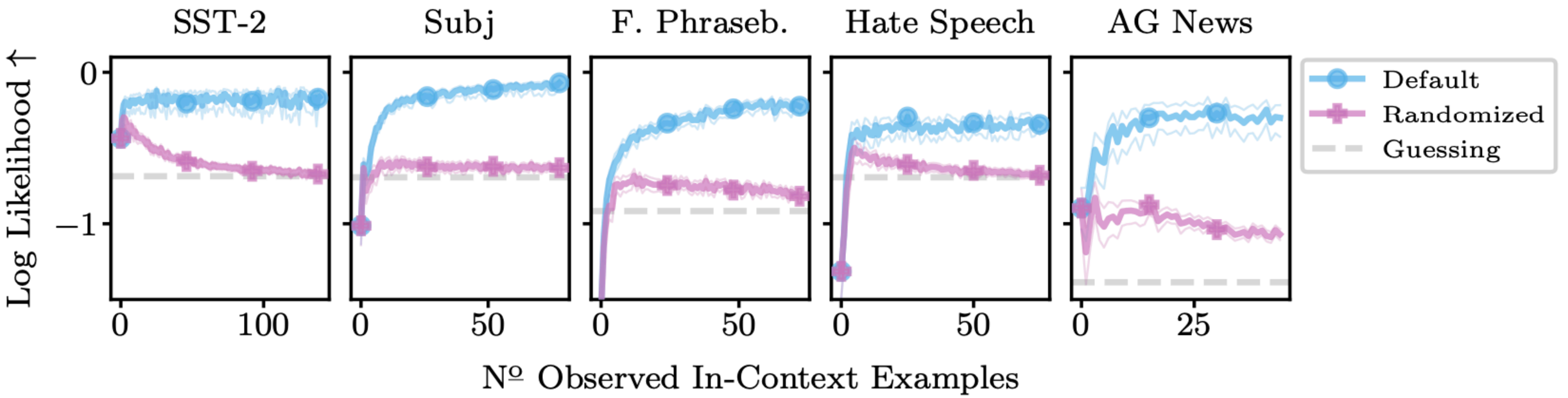


Figure 1: ICL predictions generally depend on the conditional label distribution of in-context examples: when in-context labels are randomized, average log likelihoods of label predictions decrease compared to ICL with default labels for LLaMa-2-70B across a variety of tasks. Results averaged over 500 in-context datasets and thin lines are 99 % confidence intervals. See §5 for details.

Table 1: Average differences between ICL log likelihoods for default and randomized labels. Bold entries indicate differences are statistically significant. We can disregard lightgray entries: for them, default ICL performance is not significantly better than a random guessing baseline. Whenever default ICL outperforms the baseline, ICL almost always performs significantly worse (positive differences) for random labels. Averages over 500 runs at max. context size, standard errors in Table F.1.

Δ Log Likelihood	SST-2	Subj	FP	HS	AGN	MQP	MRPC	RTE	WNLI
LLaMa-2 7B	**0.42**	**0.39**	**0.57**	**0.18**	**0.53**	**0.03**	0.02	**0.03**	0.02
LLaMa-2 13B	**0.41**	**0.62**	**0.49**	**0.24**	**0.81**	**0.04**	0.01	**0.06**	0.02
LLaMa-2 70B	**0.51**	**0.53**	**0.57**	**0.34**	**0.80**	**0.29**	**0.04**	**0.22**	**0.18**
Falcon 7B	**0.20**	**0.19**	**0.25**	**0.06**	**0.31**	0.01	0.01	−0.01	0.01
Falcon 7B Instr.	**0.13**	**0.08**	**0.11**	**0.03**	**0.15**	0.03	0.02	−0.00	0.00
Falcon 40B	**0.34**	**0.35**	**0.31**	**0.18**	**0.90**	**0.06**	0.01	0.01	0.02
Falcon 40B Instr.	**0.25**	**0.37**	**0.27**	0.02	**0.77**	**0.06**	0.02	0.02	**0.04**

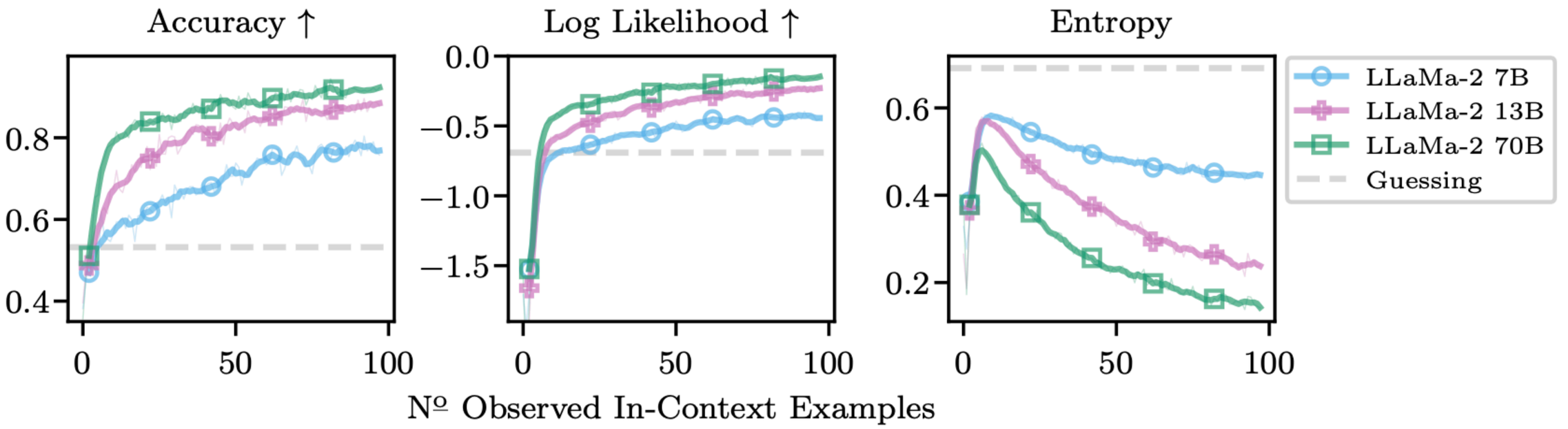


Figure 4: Few-shot ICL achieves accuracies significantly better than random guessing on our **novel author identification** task. Thus, LLMs can learn novel label relationships entirely in-context. Averages over 500 runs, thick lines with additional moving average (window size 5) for clarity.

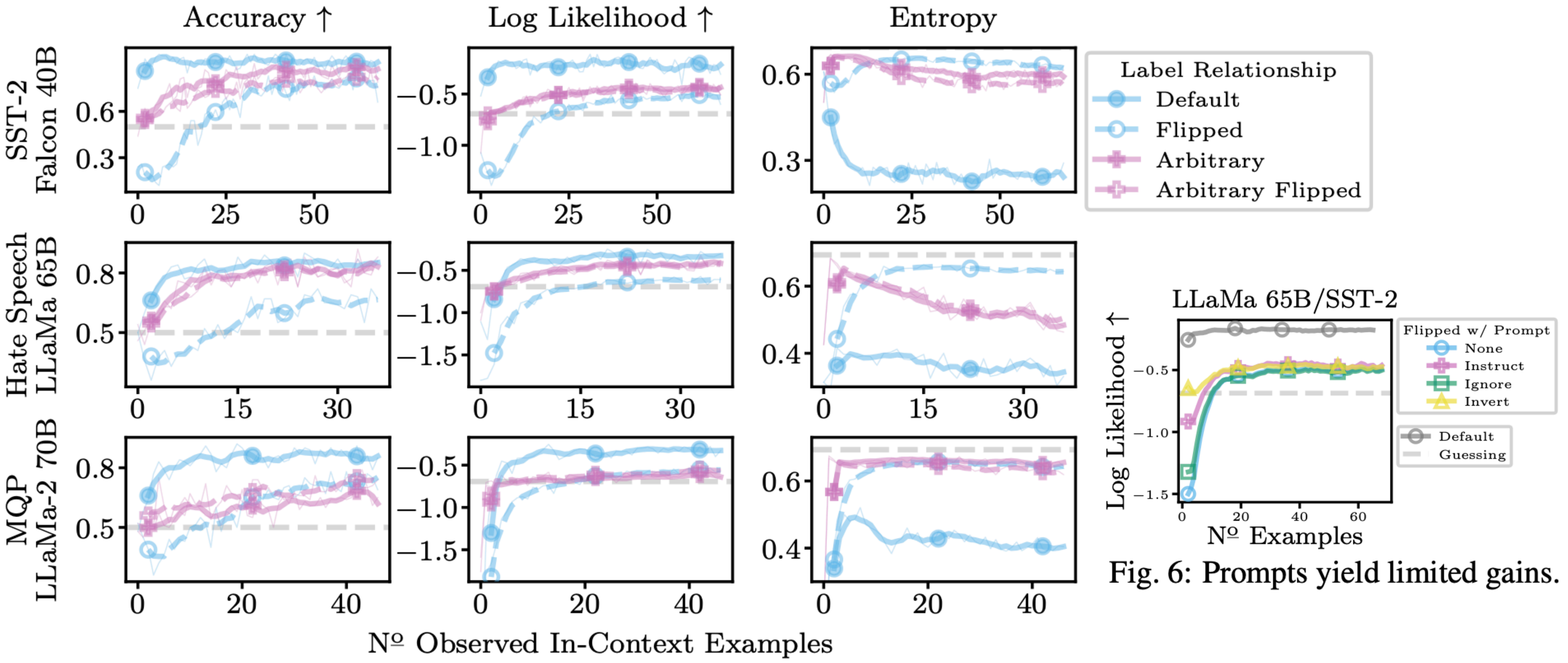


Fig. 6: Prompts yield limited gains.

Figure 5: Few-shot ICL with **replacement labels** for Falcon-40B on SST-2, LLaMa-2-65B on Hate Speech, and LLaMa-2-70B on MQP. Table 2 and §F contain results for all other models and tasks. ICL achieves better than guessing performance for all label relations and models. However, predictions for flipped labels (dashed blue) plateau at a higher entropies and lower likelihoods than those for the default label relation (solid blue). For arbitrary labels (pink), the model performs similarly for both label directions. Averages over 100 runs and thick lines with moving average (window size 5).

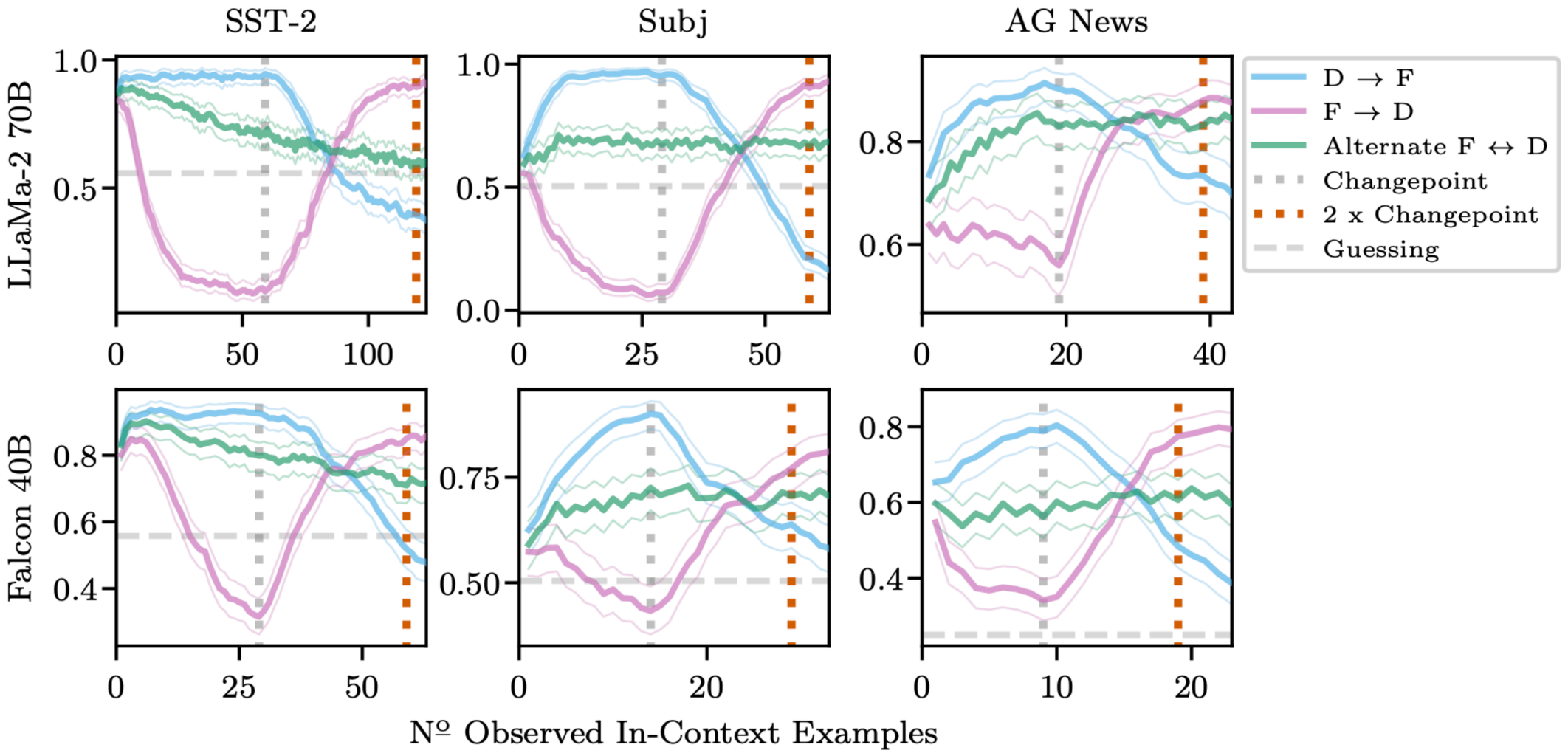


Figure 7: Few-shot ICL accuracies when the **label relationship changes throughout ICL**. For (D → F), we start with **d**efault labels and change to **f**lipped labels at the changepoint, for (F → D) we change from flipped to the default labels at the changepoint, and for (Alternating F ↔ D) we alternate between the two label relationships after every observation. For all setups, at '2 x Changepoint', the LLMs have observed the same number of examples for both label relations. If, according to NH3, ICL treats all in-context information equally, predictions should be equal at that point—but they are not. Bootstrapped 99 % confidence intervals, moving averages (size 3), and 500 repetitions.

Metaprompting

A prompt to help write prompts

Today you will be writing instructions to an eager, helpful, but inexperienced and unworldly AI assistant who needs careful instruction and examples to understand how best to behave. I will explain a task to you. You will write instructions that will direct the assistant on how best to accomplish the task consistently, accurately, and correctly. Here are some examples of tasks and instructions.

<Task Instruction Example>
<Task>
Answer questions about a document and provide references
</Task>
<Inputs>
{$DOCUMENT}
{$QUESTION}
</Inputs>
<Instructions>
I'm going to give you a document. Then I'm going to ask you a question about it. I'd like you to first write down exact quotes of parts of the document that would help answer the question, and then I'd like you to answer the question using facts from the quoted content. Here is the document:

anthropic notebook on Colab

<document>
{$DOCUMENT}
</document>

Here is the question:
<question>{$QUESTION}</question>

First, find the quotes from the document that are most relevant to answering the question, and then print them in numbered order. Quotes should be relatively short.

If there are no relevant quotes, write "No relevant quotes" instead.

Then, answer the question, starting with "Answer:". Do not include or reference quoted content verbatim in the answer. Don't say "According to Quote [1]" when answering. Instead make references to quotes relevant to each section of the answer solely by adding their bracketed numbers at the end of relevant sentences.

Thus, the format of your overall response should look like what's shown between the <example> tags. Make sure to follow the formatting and spacing exactly.

anthropic notebook on Colab

<example>
<Relevant Quotes>
<Quote> [1] "Company X reported revenue of $12 million in 2021." </Quote>
<Quote> [2] "Almost 90% of revene came from widget sales, with gadget sales making up the remaining 10%." </Quote>
</Relevant Quotes>
<Answer>
[1] Company X earned $12 million. [2] Almost 90% of it was from widget sales.
</Answer>
</example>

If the question cannot be answered by the document, say so.

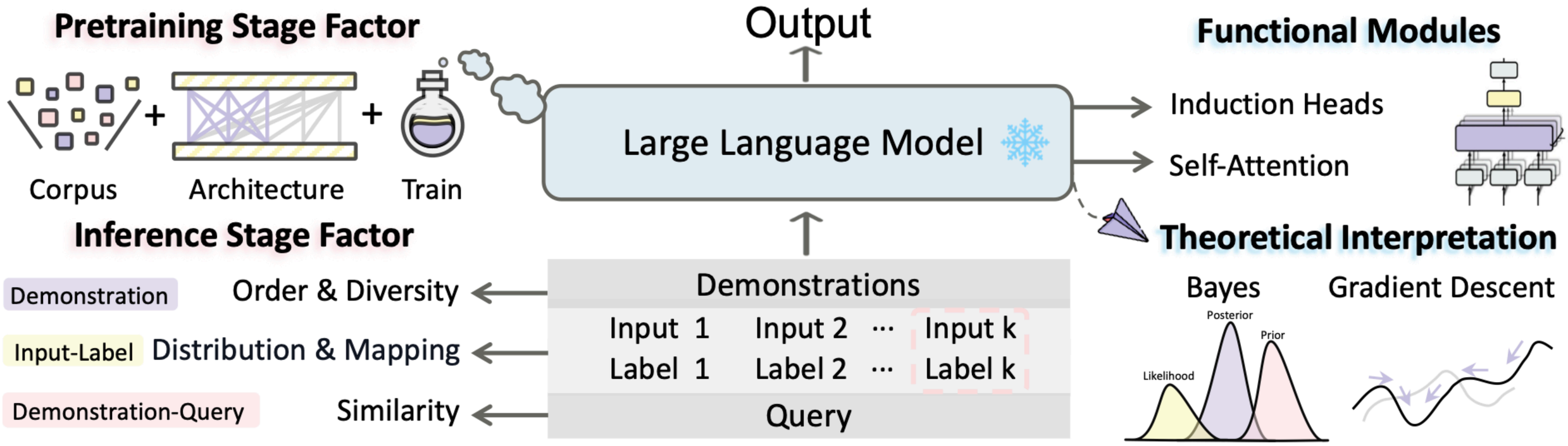
Answer the question immediately without preamble.
</Instructions>
</Task Instruction Example>

Final part of the metaprompt after several such examples is to format user input prompts into the same style as the examples.
An excellent example of in-context learning.

anthropic notebook on Colab

Survey of papers about ICL

https://arxiv.org/abs/2301.00234



Summary

- **Prompting**: Allows us to use LMs for diverse applications. Needs to be aware of how instruction tuning was done on the LLM.
- **Prompt Engineering**: Needed since performance can change a lot with prompts.
- **Metaprompting**: Using a prompt to generate prompts can tune your prompt to the instruction tuning formats used by an LLM.
- **Advanced instruct tuning**: Improving performance by increasing test time compute (https://www.youtube.com/watch?v=6PEJ96k1kiw)
- There are lots of other follow-up prompting methods (selection-inference, least-to-most prompting etc.) — Survey (https://arxiv.org/abs/2107.13586)