

Chain of thought prompting, Instruction tuning

493 / 599 May 30 2023
Ludwig Schmidt

From a base model to ChatGPT

Base model = pre-trained language model without fine-tuning (GPT-3, LLaMA, etc.)

What does it take to get a ChatGPT-style application from a base model?

Timeline from GPT-3 to ChatGPT:

May 2020: OpenAI releases GPT-3

January 2022: OpenAI releases InstructGPT, an instruction-tuned version of GPT-3
Also proposes RLHF (reinforcement learning from human feedback)

Along the way: Updated base model (trained on code etc.) → “GPT-3.5”

November 2022: OpenAI releases ChatGPT

text-davinci-002

An InstructGPT model

Released as API in
early 2022

No RLHF (yet)

Basically same
functionality as
ChatGPT, but not
an explicit chat
product.

User interface is key!

The screenshot shows the InstructGPT playground interface. At the top, there are buttons for 'Load a preset...', 'Save', 'View code', 'Share', and a three-dot menu. On the right, there are sections for 'Mode' (set to 'Complete'), 'Model' (set to 'text-davinci-003'), and various configuration sliders: Temperature (1), Maximum length (256), Stop sequences (empty), Top P (1), Frequency penalty (0), Presence penalty (0), Best of (1), Inject start text (checked), Inject restart text (checked), and Show probabilities (Off). The main area shows a conversation:

- Write a tag line for an ice cream shop
- "Let your taste buds be thrilled!"
- Can you incorporate the theme "ice"?
- "Experience a chill that's simply ice-citing!"
- Great, can you also make a German version of it?
- "Erleben Sie eine Kälte, die einfach eisig aufregend ist!"

At the bottom, there is a note: "Looking for ChatGPT? Try it now" with a link icon. There are also 'Submit' and feedback icons (refresh, thumbs up/down) at the bottom.

Converting base models into chatbots is active research

A lot of research since the release of ChatGPT

Many contradictory findings, some aspects of ChatGPT-style models are still unclear

Evaluations / benchmarks for chatbots currently poor

Current consensus:

- (Almost) all capabilities come from the base model
- Base models (without instruction tuning / RLHF) can be hard to use / extract capabilities from
- Instruction tuning / RLHF makes models more user-friendly / chatbot-like

GPT-4 paper:

The model's capabilities on exams appear to stem primarily from the pre-training process and are not significantly affected by RLHF. On multiple choice questions, both the base GPT-4 model and the RLHF model perform equally well on average across the exams we tested (see Appendix B).

Our evaluations suggest RLHF does not significantly affect the base GPT-4 model's capability - see Appendix B for more discussion.

To test the impact of RLHF on the capability of our base model, we ran the multiple-choice question portions of our exam benchmark on the GPT-4 base model and the post RLHF GPT-4 model. The results are shown in Table 8. Averaged across all exams, the base model achieves a score of 73.7% while the RLHF model achieves a score of 74.0%, suggesting that post-training does not substantially alter base model capability.

For free-response questions, it is difficult to compare the base and RLHF models on an even footing, as our methodology for sampling free-response answers likely benefits from the model’s ability to do instruction following.

Exam	Base model	RLHF model		
LSAT (MCQ)	67.0 %	72.0 %		
SAT EBRW – Reading Portion	92.3 %	90.4 %		
SAT EBRW – Writing Portion	90.9 %	84.1 %		
SAT Math (MCQ)	91.4 %	86.2 %		
Graduate Record Examination (GRE) Quantitative	57.5 %	67.5 %		
Graduate Record Examination (GRE) Verbal	87.5 %	90.0 %		
USNCO Local Section Exam 2022	51.7 %	63.3 %		
AP Art History (MCQ)	72.5 %	66.2 %		
AP Biology (MCQ)	98.3 %	96.7 %		
AP Calculus BC (MCQ)	66.7 %	57.8 %		
AP Chemistry (MCQ)	58.3 %	71.7 %		
AP English Language and Composition (MCQ)	55.6 %	51.1 %		
AP English Literature and Composition (MCQ)	63.6 %	69.1 %		
			AP Environmental Science (MCQ)	72.5 %
			AP Macroeconomics (MCQ)	83.3 %
			AP Microeconomics (MCQ)	90.0 %
			AP Physics 2 (MCQ)	62.2 %
			AP Psychology (MCQ)	98.0 %
			AP Statistics (MCQ)	60.0 %
			AP US Government (MCQ)	85.5 %
			AP US History (MCQ)	89.1 %
			AP World History (MCQ)	94.5 %
			MKSAP Questions (MCQ)	77.9 %
			AMC 10	28.0 %
			AMC 12	20.0 %
			Introductory Sommelier (theory knowledge)	90.5 %
			Certified Sommelier (theory knowledge)	83.2 %
			Advanced Sommelier (theory knowledge)	74.8 %
			Average	73.7 %
				74.0 %

Chain-of-Thought Prompting Elicits Reasoning in Large Language Models

Jason Wei

Xuezhi Wang

Dale Schuurmans

Maarten Bosma

Brian Ichter

Fei Xia

Ed H. Chi

Quoc V. Le

Denny Zhou

Google Research, Brain Team
`{jasonwei,dennyzhou}@google.com`

Abstract

We explore how generating a *chain of thought*—a series of intermediate reasoning steps—significantly improves the ability of large language models to perform complex reasoning. In particular, we show how such reasoning abilities emerge naturally in sufficiently large language models via a simple method called *chain-of-thought prompting*, where a few chain of thought demonstrations are provided as exemplars in prompting.

Experiments on three large language models show that chain-of-thought prompting improves performance on a range of arithmetic, commonsense, and symbolic reasoning tasks. The empirical gains can be striking. For instance, prompting a PaLM 540B with just eight chain-of-thought exemplars achieves state-of-the-art accuracy on the GSM8K benchmark of math word problems, surpassing even finetuned GPT-3 with a verifier.

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. 

Figure 1: Chain-of-thought prompting enables large language models to tackle complex arithmetic, commonsense, and symbolic reasoning tasks. Chain-of-thought reasoning processes are highlighted.

- Finetuned GPT-3 175B
- Prior best
- PaLM 540B: standard prompting
- PaLM 540B: chain-of-thought prompting

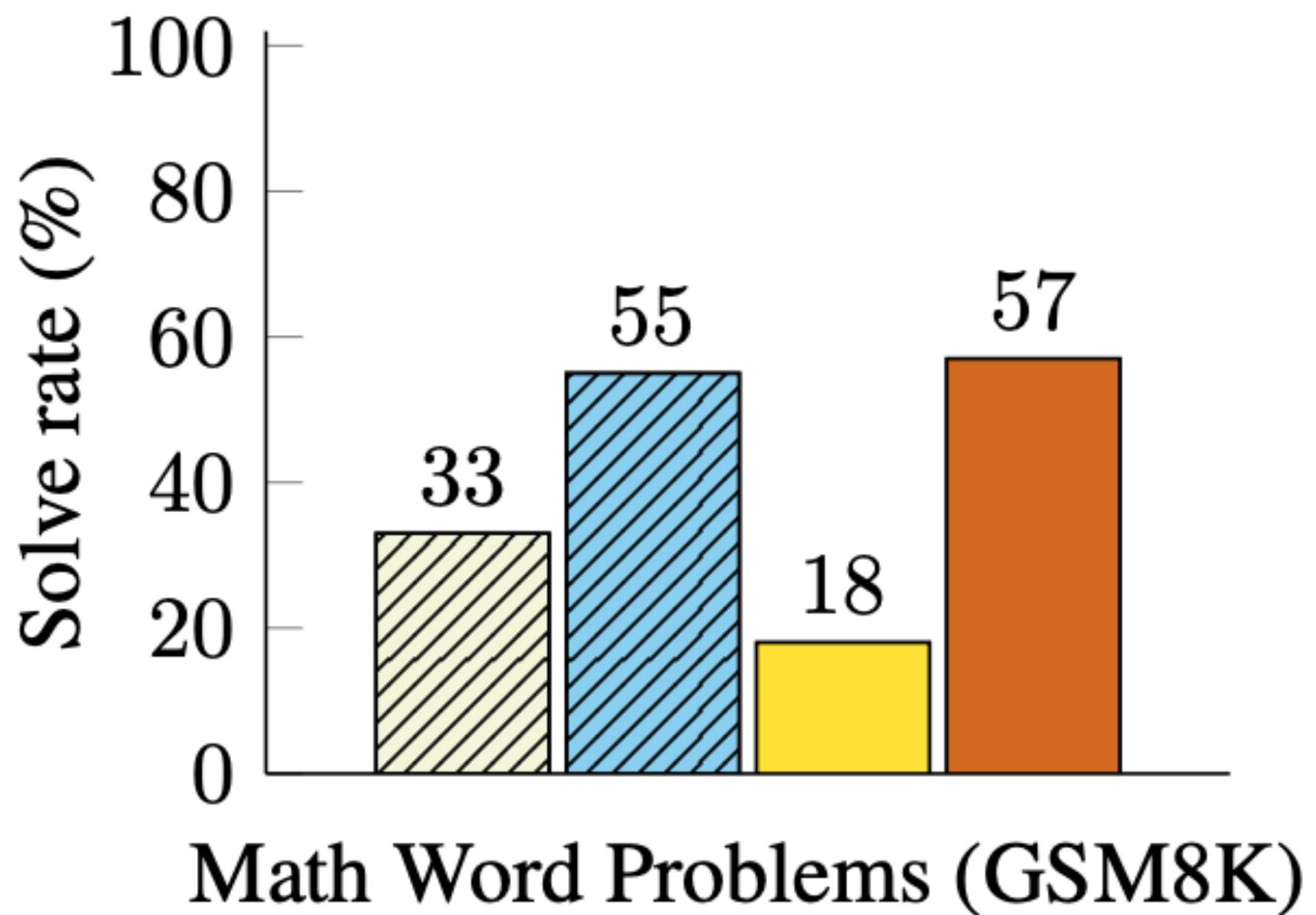


Figure 2: PaLM 540B uses chain-of-thought prompting to achieve new state-of-the-art performance on the GSM8K benchmark of math word problems. Finetuned GPT-3 and prior best are from Cobbe et al. (2021).

Math Word Problems (free response)

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.

Math Word Problems (multiple choice)

Q: How many keystrokes are needed to type the numbers from 1 to 500?
Answer Choices: (a) 1156 (b) 1392 (c) 1480
(d) 1562 (e) 1788

A: There are 9 one-digit numbers from 1 to 9. There are 90 two-digit numbers from 10 to 99. There are 401 three-digit numbers from 100 to 500. $9 + 90(2) + 401(3) = 1392$. The answer is (b).

CSQA (commonsense)

Q: Sammy wanted to go to where the people were. Where might he go?
Options: (a) race track (b) populated areas
(c) desert (d) apartment (e) roadblock

A: The answer must be a place with a lot of people. Race tracks, desert, apartments, and roadblocks don't have a lot of people, but populated areas do. So the answer is (b).

StrategyQA

Q: Yes or no: Would a pear sink in water?

A: The density of a pear is about 0.6 g/cm³, which is less than water. Thus, a pear would float. So the answer is no.

Date Understanding

Q: The concert was scheduled to be on 06/01/1943, but was delayed by one day to today. What is the date 10 days ago in MM/DD/YYYY?

A: One day after 06/01/1943 is 06/02/1943, so today is 06/02/1943. 10 days before today is 05/23/1943. So the answer is 05/23/1943.

Sports Understanding

Q: Is the following sentence plausible? "Joao Moutinho caught the screen pass in the NFC championship."

A: Joao Moutinho is a soccer player. The NFC championship is part of American football, not soccer. So the answer is no.

SayCan (Instructing a robot)

Human: How would you bring me something that isn't a fruit?

Explanation: the user wants something to eat that isn't a fruit. An energy bar is not a fruit, so I will bring the user an energy bar.

Plan: 1. find(energy bar) 2. pick(energy bar) 3. find(user) 4. put(energy bar) 5. done().

Last Letter Concatenation

Q: Take the last letters of the words in "Lady Gaga" and concatenate them.

A: The last letter of "Lady" is "y". The last letter of "Gaga" is "a". Concatenating them is "ya". So the answer is ya.

Coin Flip (state tracking)

Q: A coin is heads up. Maybelle flips the coin. Shalonda does not flip the coin. Is the coin still heads up?

A: The coin was flipped by Maybelle. So the coin was flipped 1 time, which is an odd number. The coin started heads up, so after an odd number of flips, it will be tails up. So the answer is no.

Figure 3: Examples of $\langle \text{input}, \text{chain of thought}, \text{output} \rangle$ triples for arithmetic, commonsense, and symbolic reasoning benchmarks. Chains of thought are highlighted. Full prompts in Appendix G.

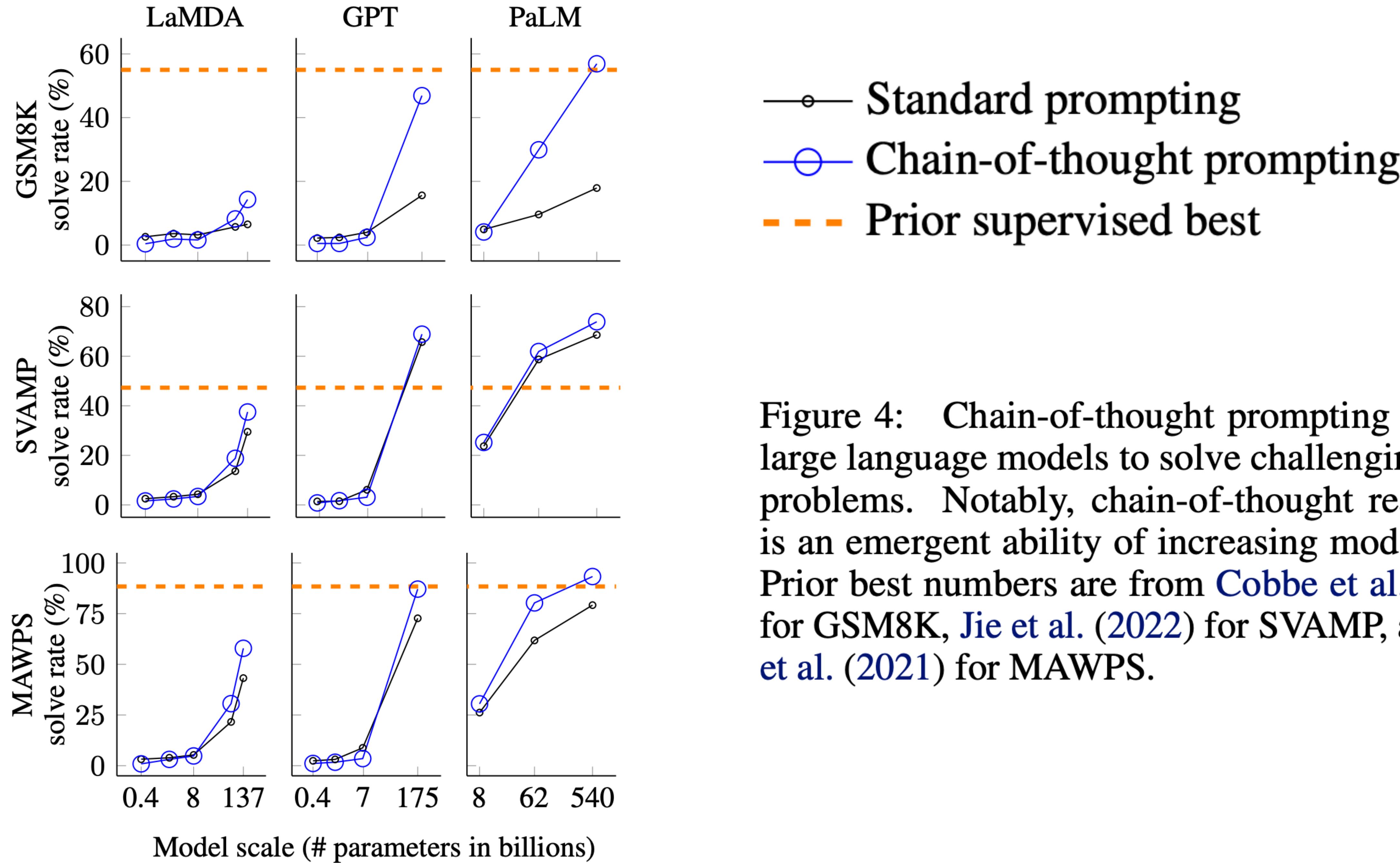


Figure 4: Chain-of-thought prompting enables large language models to solve challenging math problems. Notably, chain-of-thought reasoning is an emergent ability of increasing model scale. Prior best numbers are from Cobbe et al. (2021) for GSM8K, Jie et al. (2022) for SVAMP, and Lan et al. (2021) for MAWPS.

Table 3: Standard prompting versus chain of thought prompting on the four subsets of the MAWPS benchmark. The point of stratifying the MAWPS benchmark is to show that performance gains are minimal on easy one-step or two-step problems where large language models already achieve high performance (e.g., SingleOp, SingleEq, and AddSub).

Model		SingleOp		SingleEq		AddSub		MultiArith	
		standard	CoT	standard	CoT	standard	CoT	standard	CoT
UL2	20B	24.9	27.2	18.0	20.2	18.5	18.2	5.0	10.7
LaMDA	420M	2.8	1.0	2.4	0.4	1.9	0.7	5.8	1.5
	2B	4.6	4.1	2.4	3.3	2.7	3.2	5.8	1.8
	8B	8.0	7.0	4.5	4.4	3.4	5.2	5.2	2.4
	68B	36.5	40.8	23.9	26.0	17.3	23.2	8.7	32.4
	137B	73.2	76.2	48.8	58.7	43.0	51.9	7.6	44.9
GPT	350M	3.2	1.8	2.0	0.2	2.0	1.5	2.3	0.8
	1.3B	5.3	3.0	2.4	1.6	2.3	1.5	2.2	0.5
	6.7B	13.5	3.9	8.7	4.9	8.6	2.5	4.5	2.8
	175B	90.9	88.8	82.7	86.6	83.3	81.3	33.8	91.7
Codex	-	93.1	91.8	86.8	93.1	90.9	89.1	44.0	96.2
PaLM	8B	41.8	46.6	29.5	28.2	29.4	31.4	4.2	15.8
	62B	87.9	85.6	77.2	83.5	74.7	78.2	7.3	73.7
	540B	94.1	94.1	86.5	92.3	93.9	91.9	42.2	94.7

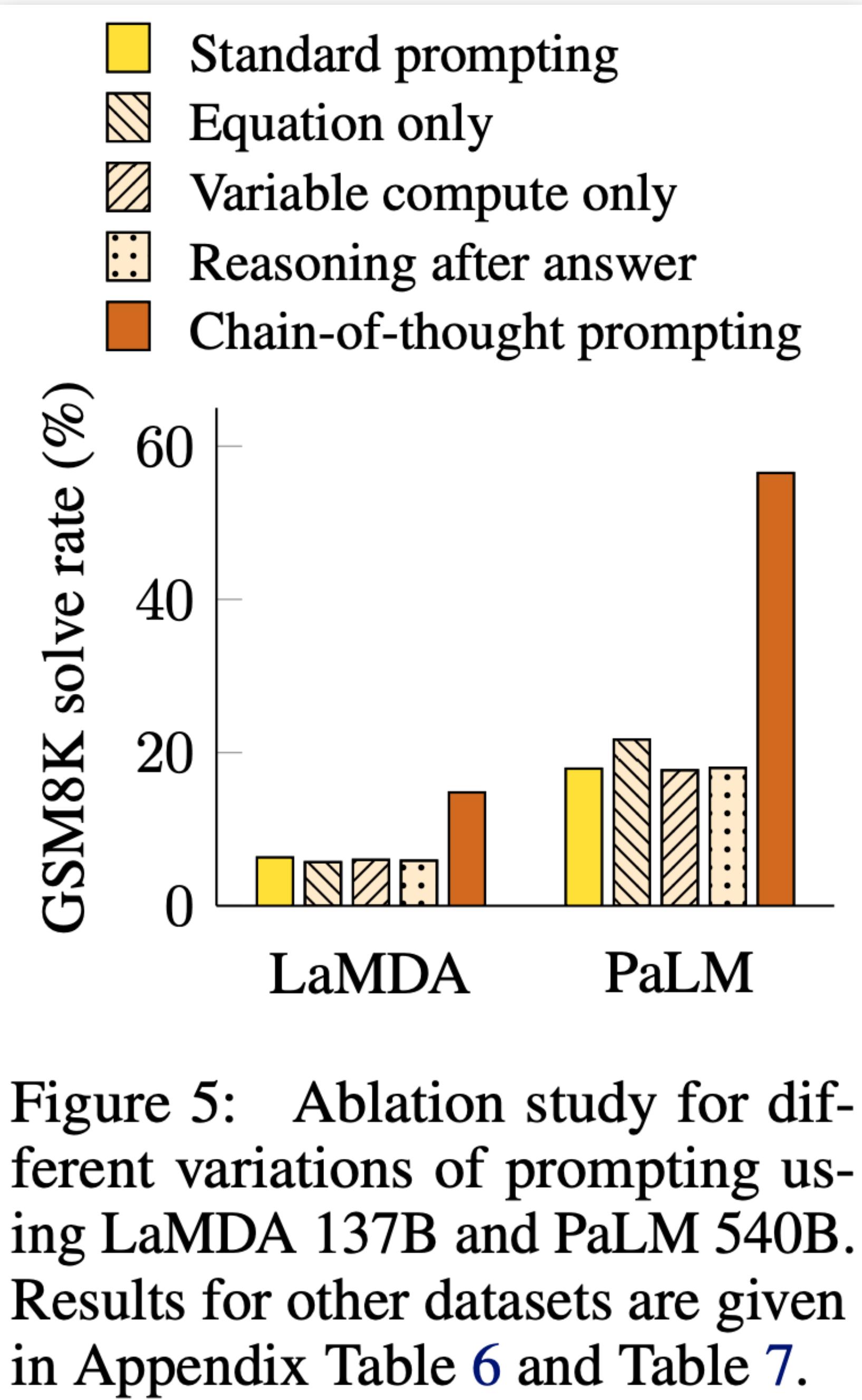


Figure 5: Ablation study for different variations of prompting using LaMDA 137B and PaLM 540B. Results for other datasets are given in Appendix Table 6 and Table 7.

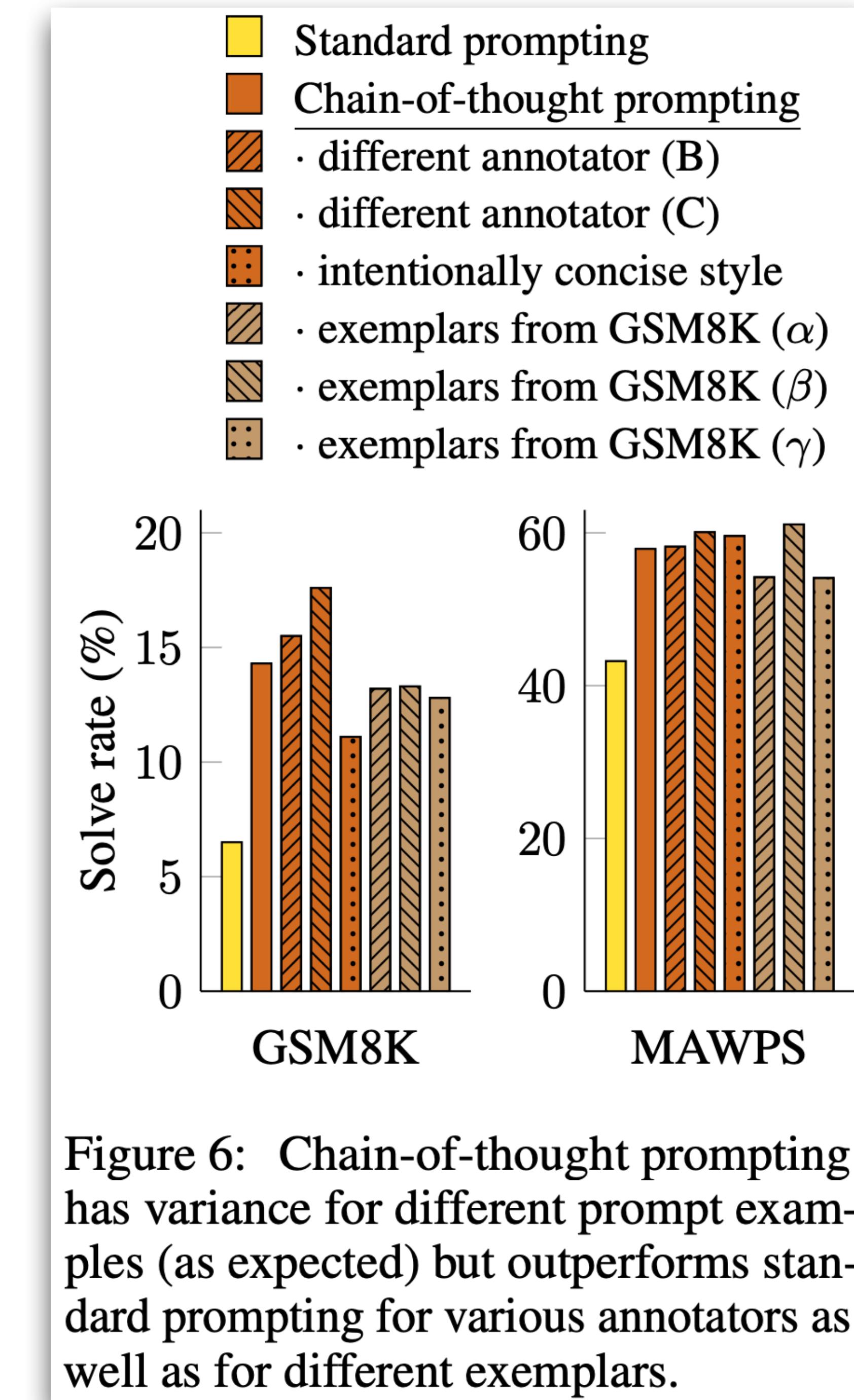


Figure 6: Chain-of-thought prompting has variance for different prompt examples (as expected) but outperforms standard prompting for various annotators as well as for different exemplars.

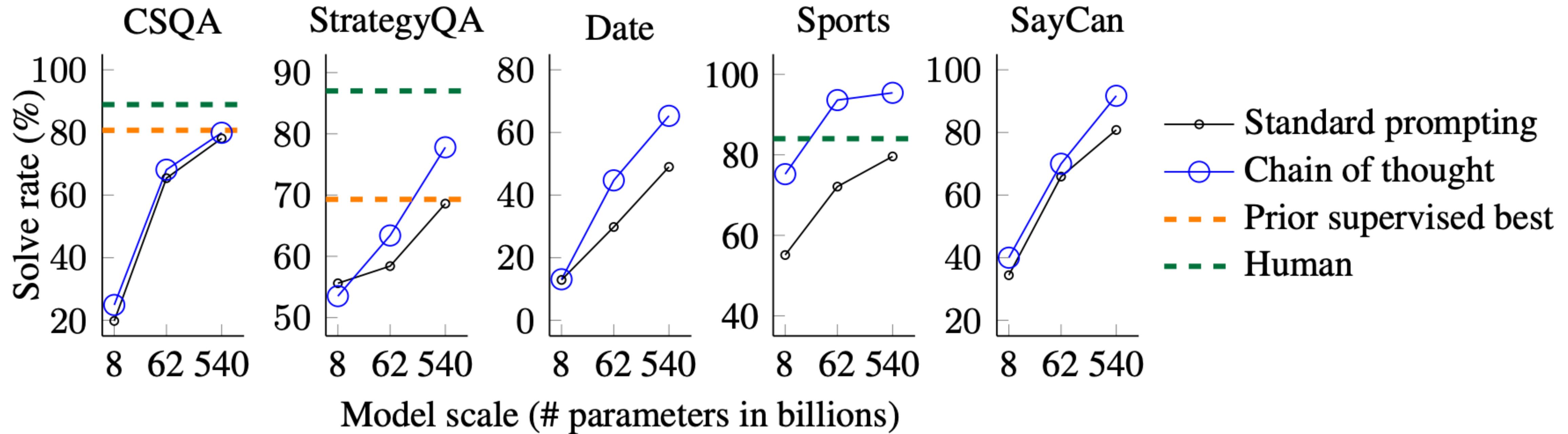


Figure 7: Chain-of-thought prompting also improves the commonsense reasoning abilities of language models. The language model shown here is PaLM. Prior best numbers are from the leaderboards of CSQA (Talmor et al., 2019) and StrategyQA (Geva et al., 2021) (single-model only, as of May 5, 2022). Additional results using various sizes of LaMDA, GPT-3, and PaLM are shown in Table 4.

Tasks. We use the following two toy tasks.

- **Last letter concatenation.** This task asks the model to concatenate the last letters of words in a name (e.g., “*Amy Brown*” → “*yn*”). It is a more challenging version of first letter concatenation, which language models can already perform without chain of thought.³ We generate full names by randomly concatenating names from the top one-thousand first and last names from name census data (<https://namecensus.com/>).
- **Coin flip.** This task asks the model to answer whether a coin is still heads up after people either flip or don’t flip the coin (e.g., “*A coin is heads up. Phoebe flips the coin. Osvaldo does not flip the coin. Is the coin still heads up?*” → “*no*”).

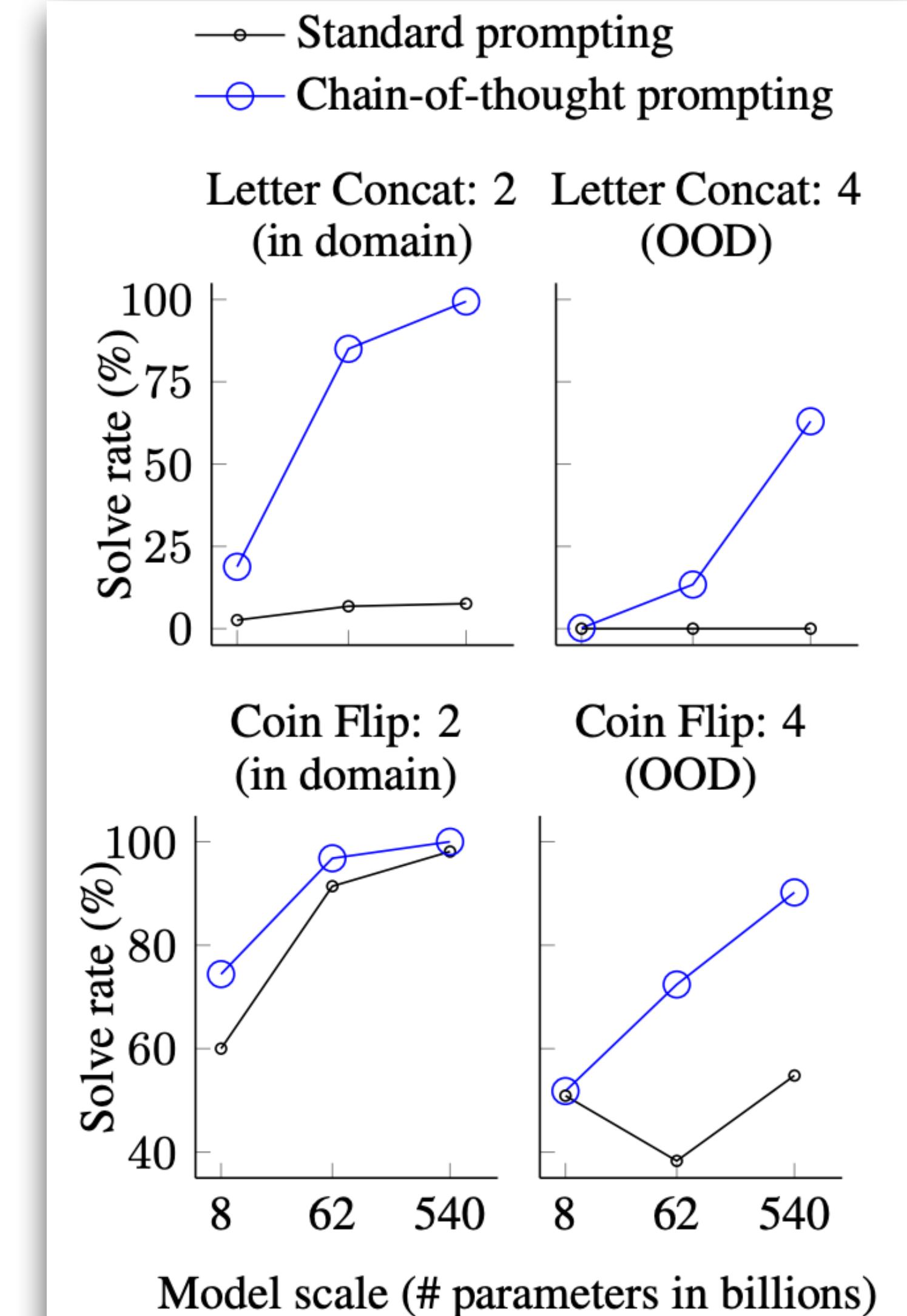


Figure 8: Using chain-of-thought prompting facilitates generalization to longer sequences in two symbolic reasoning tasks.

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*[†]**

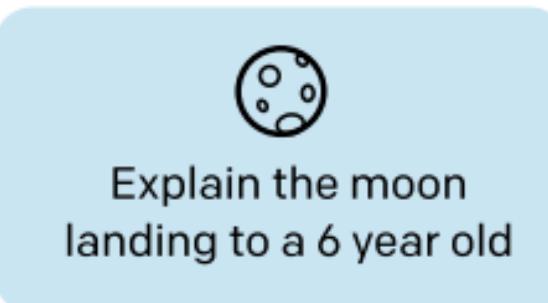
Jan Leike* **Ryan Lowe***

OpenAI

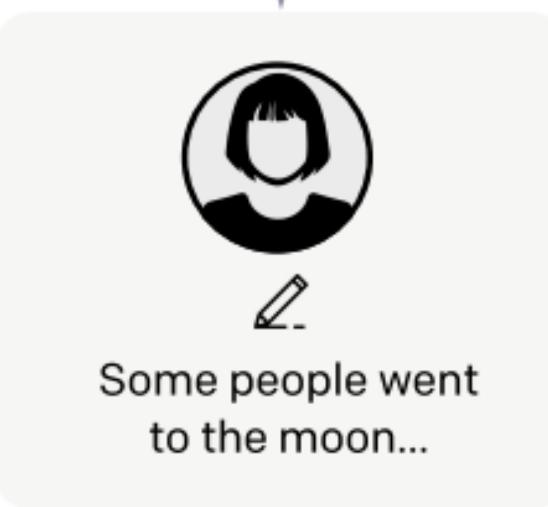
Step 1

Collect demonstration data, and train a supervised policy.

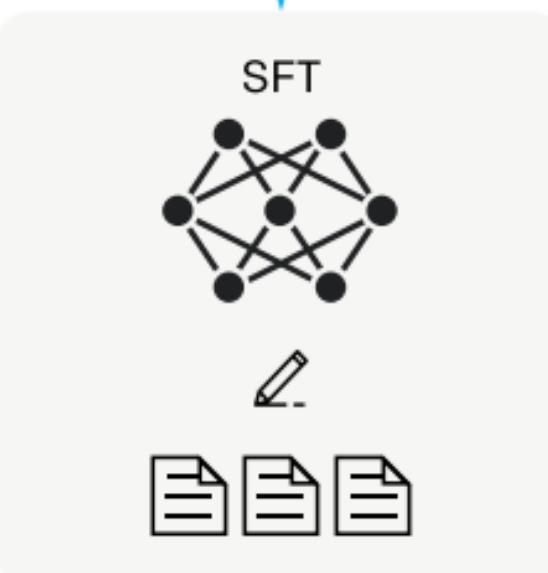
A prompt is sampled from our prompt dataset.



A labeler demonstrates the desired output behavior.



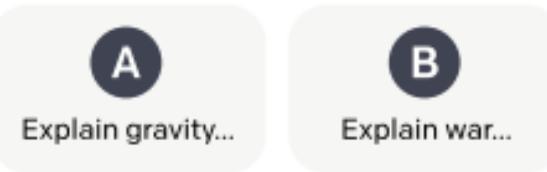
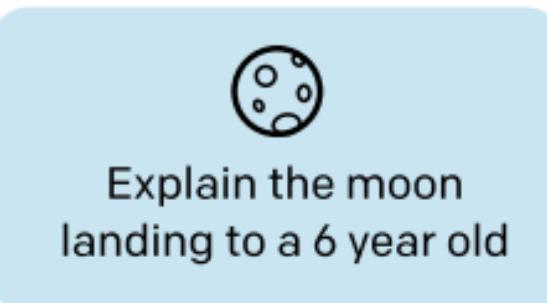
This data is used to fine-tune GPT-3 with supervised learning.



Step 2

Collect comparison data, and train a reward model.

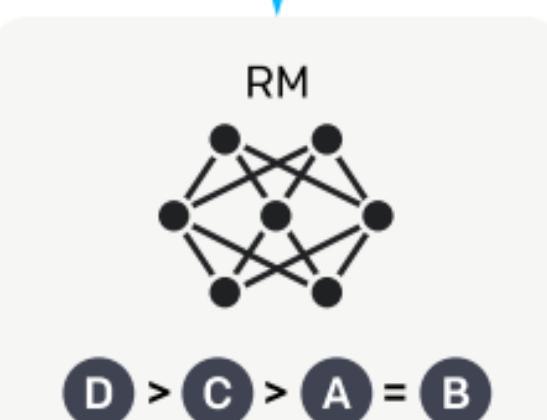
A prompt and several model outputs are sampled.



A labeler ranks the outputs from best to worst.



This data is used to train our reward model.



D > C > A = B

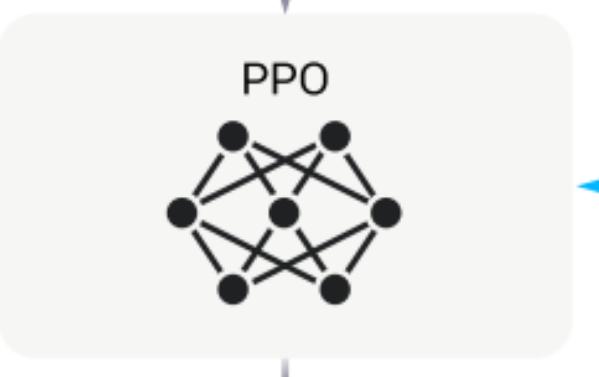
Step 3

Optimize a policy against the reward model using reinforcement learning.

A new prompt is sampled from the dataset.



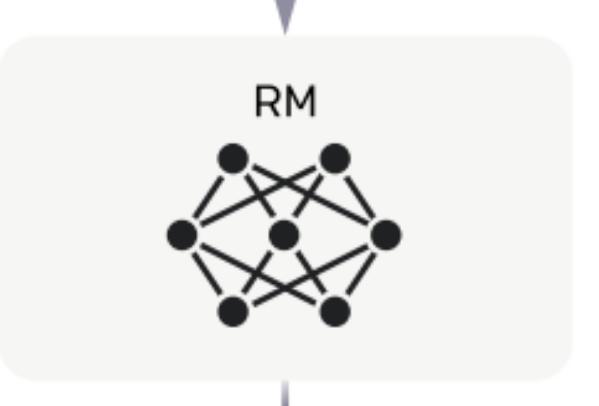
The policy generates an output.



Once upon a time...



The reward model calculates a reward for the output.



r_k

The reward is used to update the policy using PPO.

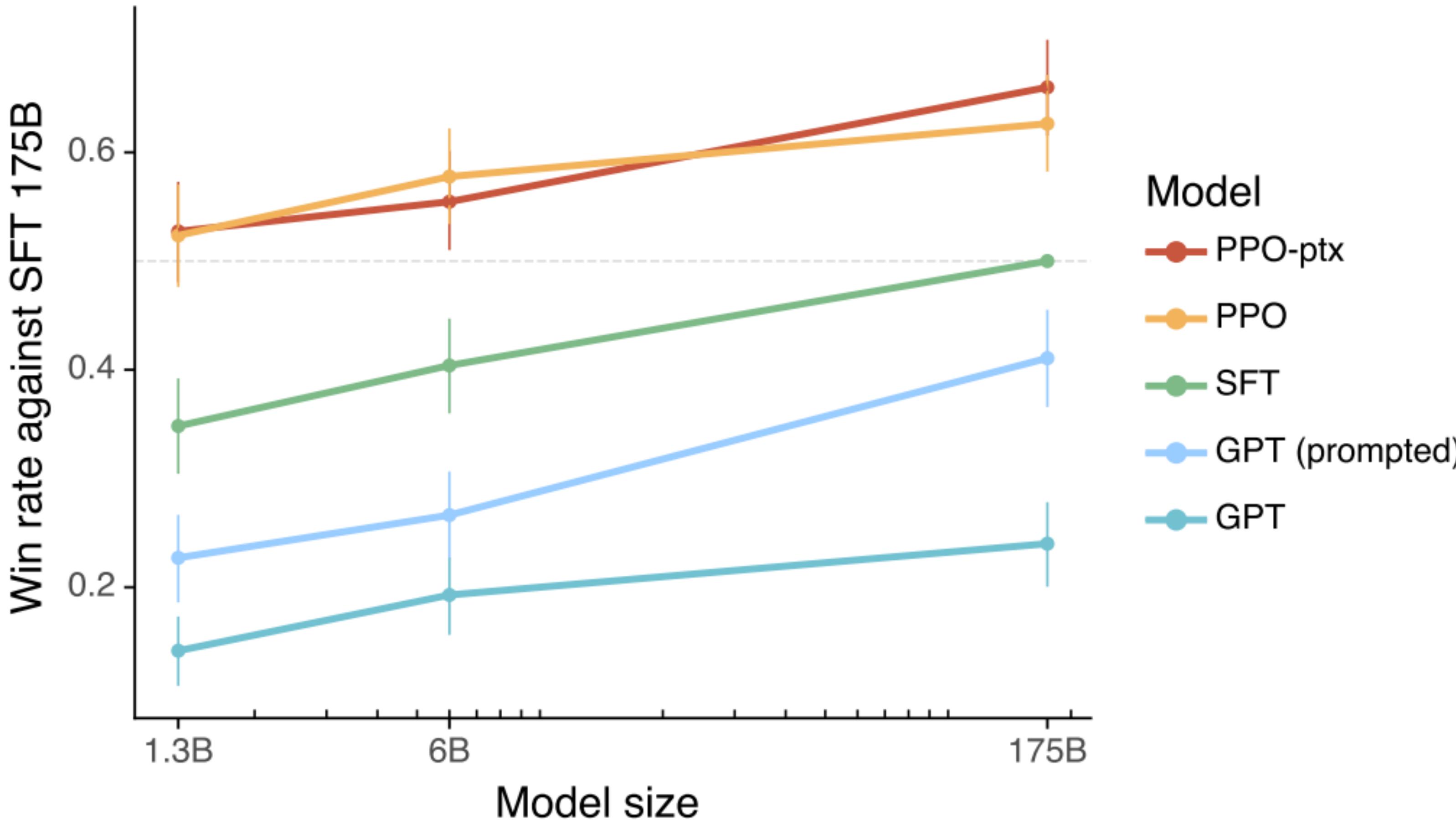


Figure 1: Human evaluations of various models on our API prompt distribution, evaluated by how often outputs from each model were preferred to those from the 175B SFT model. Our InstructGPT models (PPO-ptx) as well as its variant trained without pretraining mix (PPO) significantly outperform the GPT-3 baselines (GPT, GPT prompted); outputs from our 1.3B PPO-ptx model are preferred to those from the 175B GPT-3. Error bars throughout the paper are 95% confidence intervals.

Training data

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: """

Key point: **real instruction prompts** from the GPT-3 playground interface

Table 6: Dataset sizes, in terms of number of prompts.

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

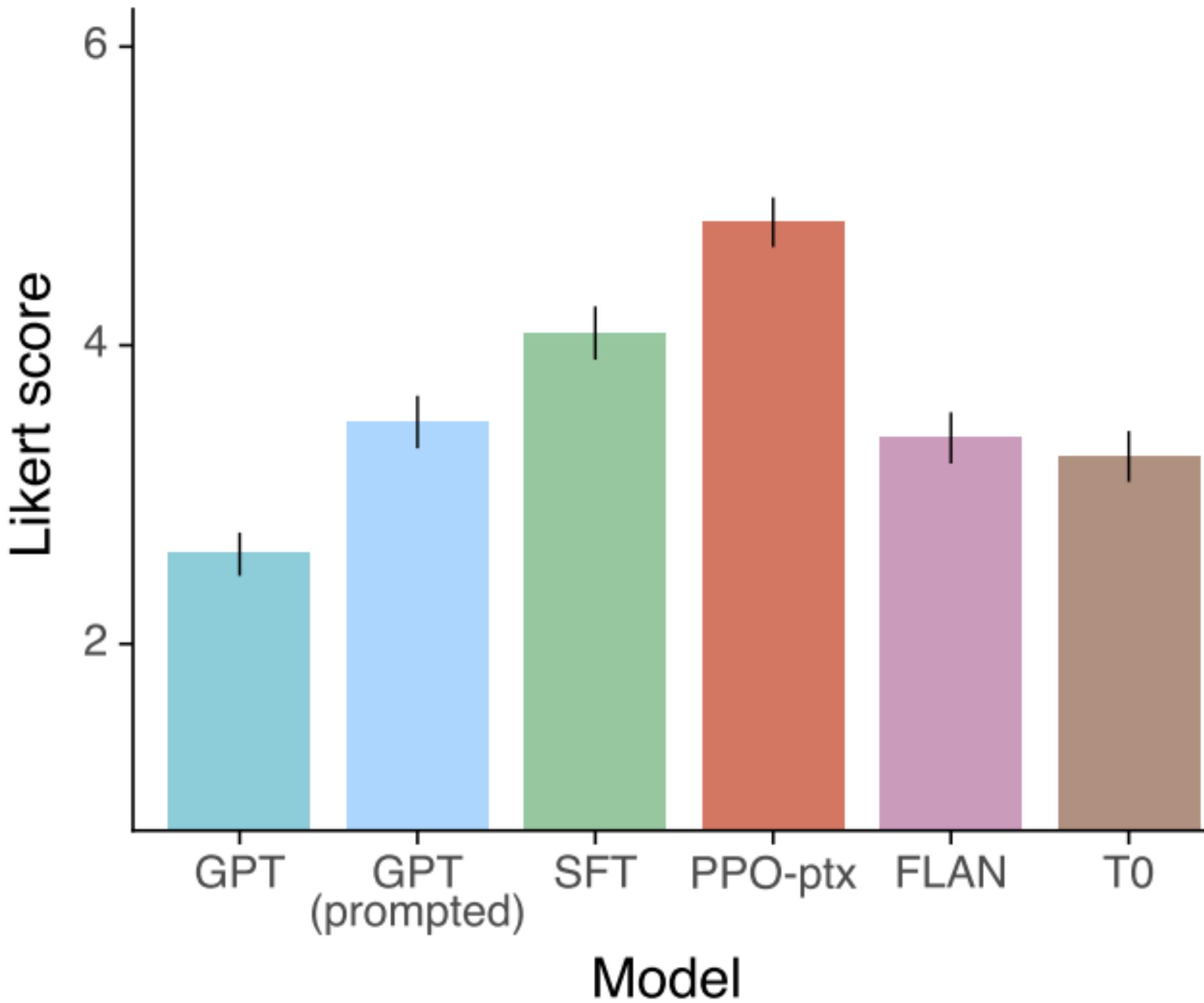
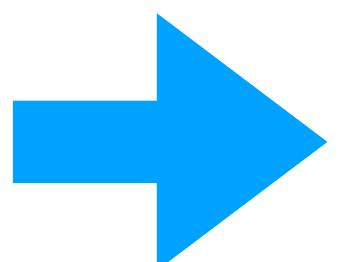
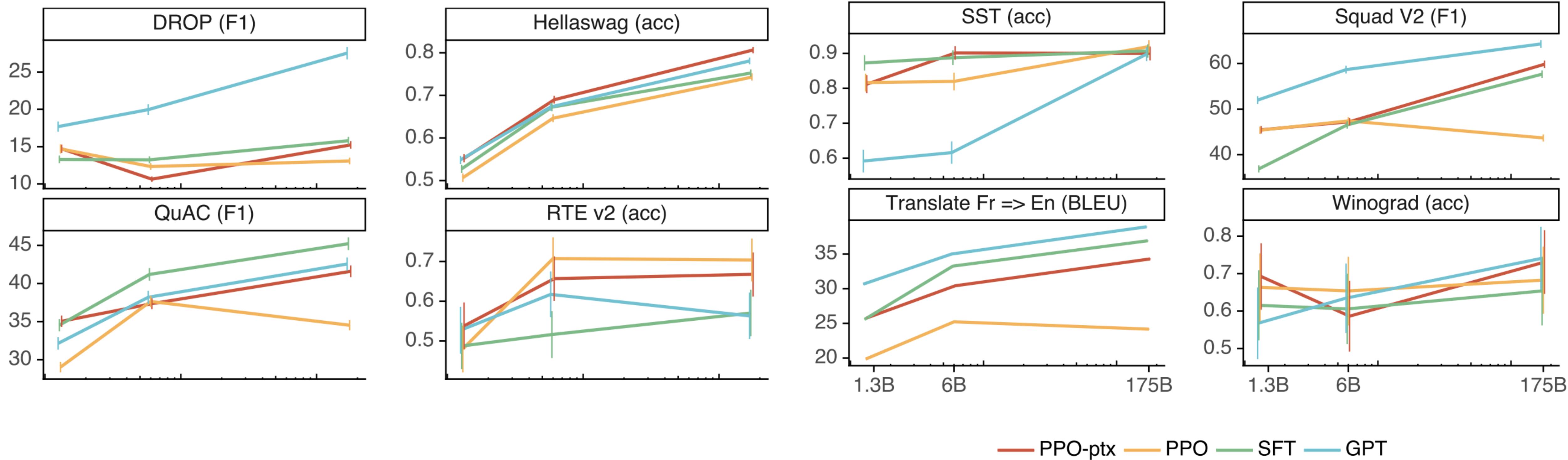


Figure 5: Comparing our models with FLAN and T0 in terms of Likert scores on a 1-7 scale, on the InstructGPT prompt distribution. FLAN and T0 perform better than default GPT-3, and comparably with a few-shot GPT-3 model placed into ‘instruction-following’ mode.

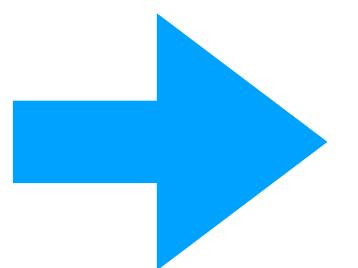
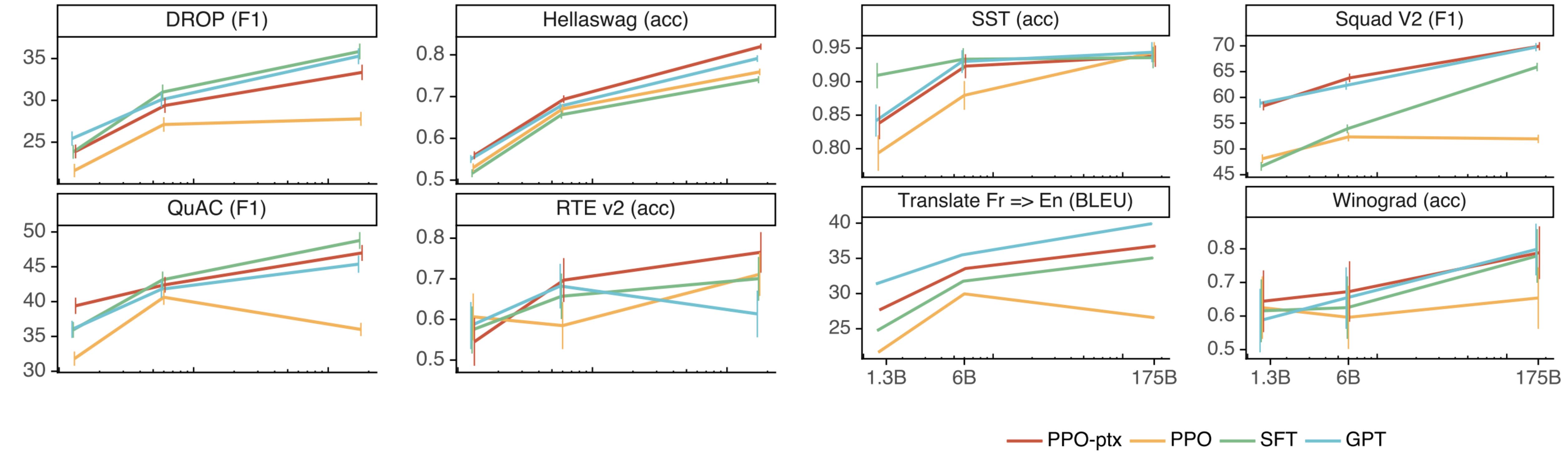


Fine-tuning on NLP datasets does not help “in the wild” LLM use.

Performance on standard NLP tasks (0-shot)



Performance on standard NLP tasks (few-shot)



Instruction tuning can decrease performance on standard NLP tasks

Alpaca: A Strong, Replicable Instruction-Following Model

Authors: Rohan Taori* and Ishaan Gulrajani* and Tianyi Zhang* and Yann Dubois* and Xuechen Li* and Carlos Guestrin and Percy Liang and Tatsunori B. Hashimoto

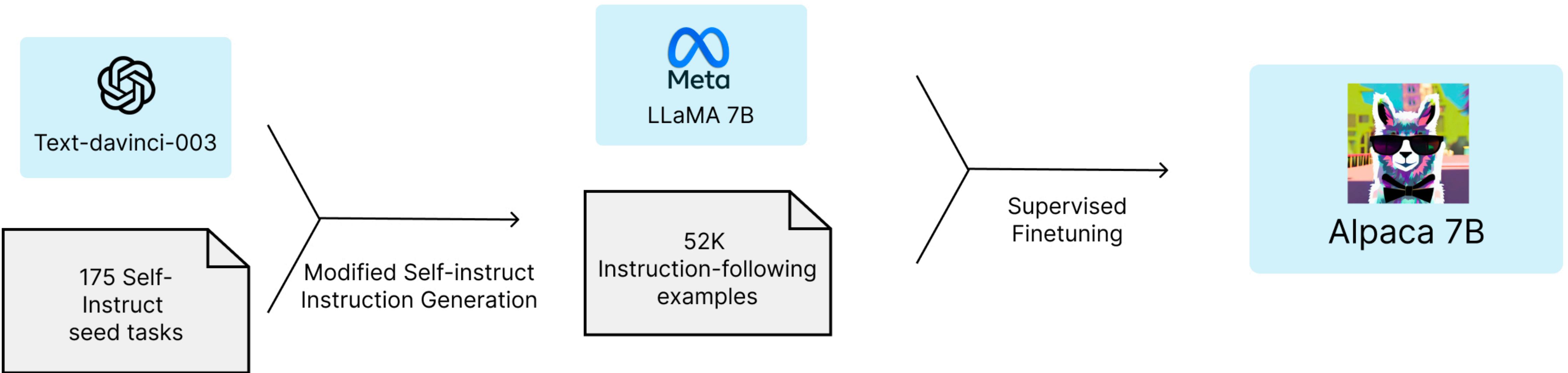
We introduce Alpaca 7B, a model fine-tuned from the LLaMA 7B model on 52K instruction-following demonstrations. On our preliminary evaluation of single-turn instruction following, Alpaca behaves qualitatively similarly to OpenAI's text-davinci-003, while being surprisingly small and easy/cheap to reproduce (<600\$).

[GitHub](#)

Stanford
Alpaca



First instruction-tuned LLaMA model (about 2 weeks after the LLaMA publication).



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 ...

Preliminary evaluation

To evaluate Alpaca, we conduct human evaluation (by the 5 student authors) on the inputs from the self-instruct evaluation set. This evaluation set was collected by the self-instruct authors and covers a diverse list of user-oriented instructions including email writing, social media, and productivity tools. We performed a blind pairwise comparison between text-davinci-003 and Alpaca 7B, and we found that these two models have very similar performance: Alpaca wins 90 versus 89 comparisons against text-davinci-003.

Self-instruction evaluation examples (252 total)

Instruction: "Change the first person to the third person in the given sentence. The meaning should be kept, but you can paraphrase it or expand it in order to have a better pose."

Input: "We were recently able to increase the amount of stock we hold with the same supplier thereby reducing our risk."

Output: "The company was recently able to increase the amount of stock it holds with the same supplier thereby reducing its risk."

Self-instruction evaluation examples (252 total)

Instruction: "Please let me know your thoughts on the given place and why you think it deserves to be visited."

Input: "Barcelona, Spain"

Output: "There's nowhere in the world like Barcelona. From days spent strolling the Gothic Quarter to afternoons at the beach and nights partying until dawn, the city is magnetic. While tapas and bustling mercados are worth every bit of attention they get in guidebooks, Barcelona's true crown jewel is the work of Antoni Gaudi. His iconic creations, like the jaw-dropping Basilica of La Sagrada Familia, are instantly recognizable and have fueled millions of travel dreams. You won't be the only tourist here (especially during the summer), but even amid the crowds you'll find your own Mediterranean magic any time of year."

Self-instruction evaluation examples (252 total)

Instruction: "Prepare an email signature template for the given person or company."

Input: "a graduate student of computer science"

Output: "[Your name]\n[Your degree] in Computer Science\n[Your university]\n\nWebsite: [Your website]\nORCID ID: [Your ORCID ID]"

SELF-INSTRUCT: Aligning Language Models with Self-Generated Instructions

Yizhong Wang[♦] **Yeganeh Kordi[◊]** **Swaroop Mishra[♡]** **Alisa Liu[♣]**
Noah A. Smith^{♣+*} **Daniel Khashabi[♣]** **Hannaneh Hajishirzi^{♣+}**

[♦]University of Washington [◊]Tehran Polytechnic [♡]Arizona State University
[♣]Johns Hopkins University ⁺Allen Institute for AI
yizhongw@cs.washington.edu

Abstract

Large “instruction-tuned” language models (i.e., finetuned to respond to instructions) have demonstrated a remarkable ability to generalize zero-shot to new tasks. Nevertheless, they depend heavily on human-written instruction data that is often limited in quantity, diversity, and creativity, therefore hindering the generality of the tuned model. We introduce SELF-INSTRUCT, a framework for improving the instruction-following capabilities of pre-trained language models by bootstrapping off

Instruction: Given an address and city, come up with the zip code.

Input:

Address: 123 Main Street, City: San Francisco 

Output: 94105

Instruction: I am looking for a job and I need to fill out an application form. Can you please help me complete it?

Input:

Application Form:

Name: _____ Age: _____ Sex: _____

Phone Number: _____ Email Address: _____

Education: _____ ...

Output:

Name: John Doe Age: 25 Sex: Male

Phone Number: ...



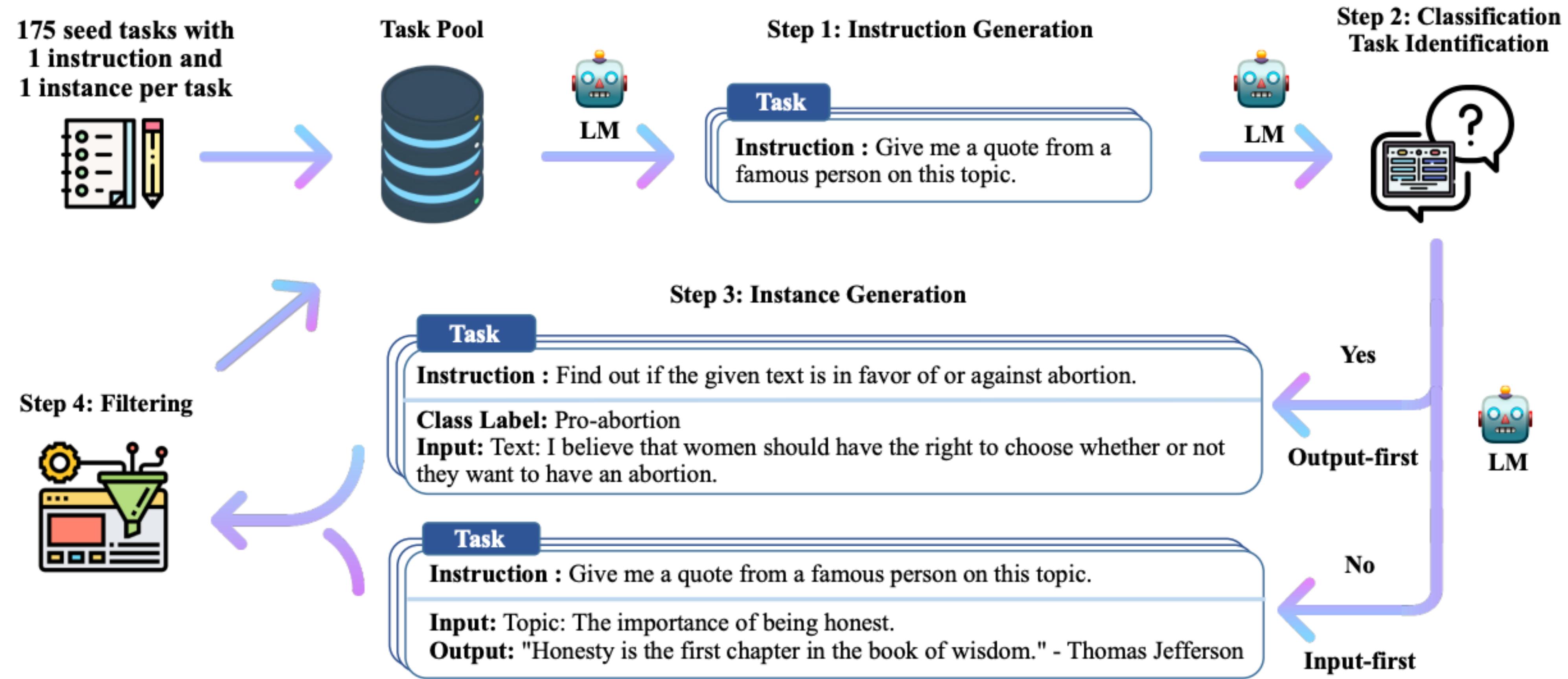


Figure 2: A high-level overview of SELF-INSTRUCT. The process starts with a small seed set of tasks as the task pool. Random tasks are sampled from the task pool, and used to prompt an off-the-shelf LM to generate both new instructions and corresponding instances, followed by filtering low-quality or similar generations, and then added back to the initial repository of tasks. The resulting data can be used for the instruction tuning of the language model itself later to follow instructions better. Tasks shown in the figure are generated by GPT3.

statistic	
# of instructions	52,445
- # of classification instructions	11,584
- # of non-classification instructions	40,861
# of instances	82,439
- # of instances with empty input	35,878
ave. instruction length (in words)	15.9
ave. non-empty input length (in words)	12.7
ave. output length (in words)	18.9

Table 1: Statistics of the generated data by applying SELF-INSTRUCT to GPT3.



Figure 3: The top 20 most common root verbs (inner circle) and their top 4 direct noun objects (outer circle) in the generated instructions. Despite their diversity, the instructions shown here only account for 14% of all the generated instructions because many instructions (e.g., “Classify whether the user is satisfied with the service.”) do not contain such a verb-noun structure.

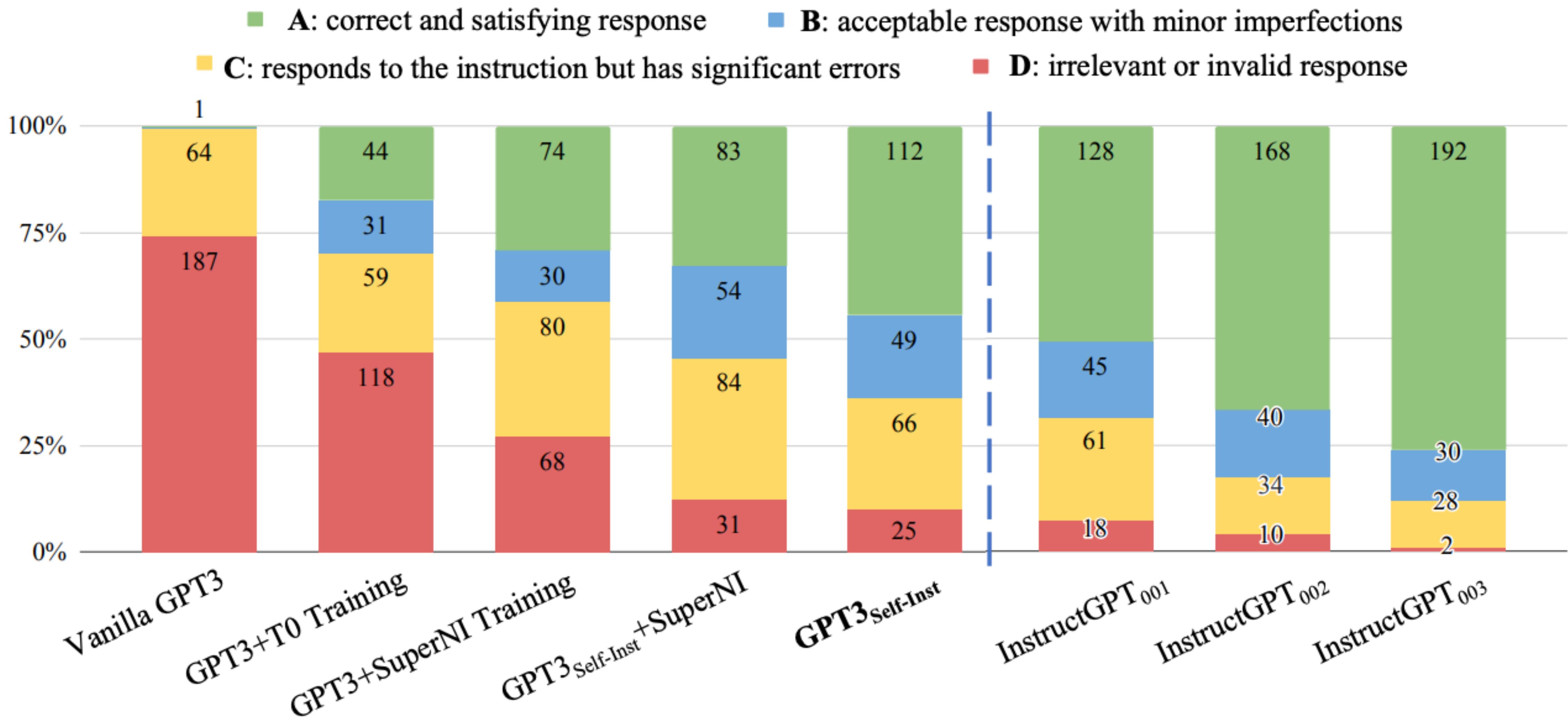


Figure 6: Performance of GPT3 model and its instruction-tuned variants, evaluated by human experts on our 252 user-oriented instructions (§4.4). Human evaluators are instructed to rate the models’ responses into four levels. The results indicate that GPT3_{SELF-INST} outperforms all the other GPT3 variants trained on publicly available instruction datasets. Additionally, GPT3_{SELF-INST} scores nearly as good as InstructGPT₀₀₁ (cf. footnote 1).

Vicuna: An Open-Source Chatbot Impressing GPT-4 with 90%* ChatGPT Quality

by: The Vicuna Team, Mar 30, 2023

We introduce Vicuna-13B, an open-source chatbot trained by fine-tuning LLaMA on user-shared conversations collected from ShareGPT. Preliminary evaluation using GPT-4 as a judge shows Vicuna-13B achieves more than 90%* quality of OpenAI ChatGPT and Google Bard while outperforming other models like LLaMA and Stanford Alpaca in more than 90%* of cases. The cost of training Vicuna-13B is around \$300. The [code](#) and [weights](#), along with an online [demo](#), are publicly available for non-commercial use.



Vicuna (generated by stable diffusion 2.1)

*According to a fun and non-scientific evaluation with GPT-4. Further rigorous evaluation is needed.

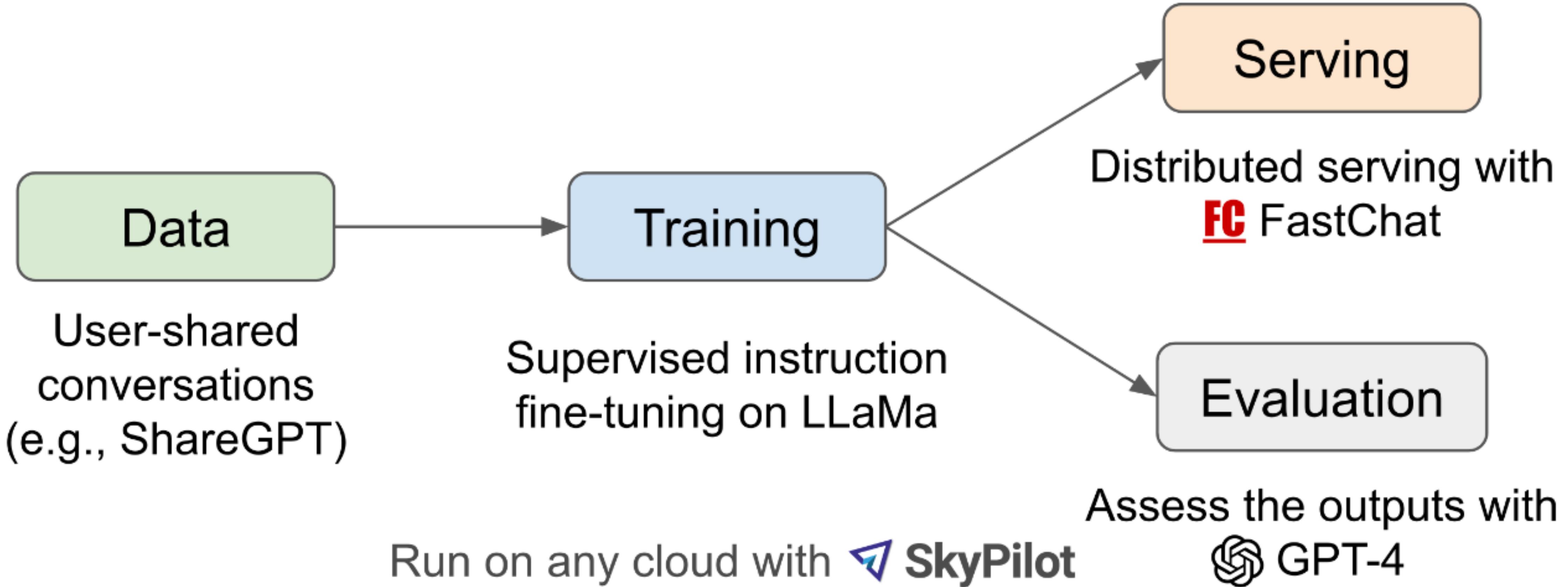


Figure 2. Workflow Overview

Table 1. Comparison between several notable models

Model Name	LLaMA	Alpaca	Vicuna	Bard/ChatGPT
Dataset	Publicly available datasets (1T token)	Self-instruct from davinci-003 API (52K samples)	User-shared conversations (70K samples)	N/A
Training code	N/A	Available	Available	N/A
Evaluation metrics	Academic benchmark	Author evaluation	GPT-4 assessment	Mixed
Training cost (7B)	82K GPU-hours	\$500 (data) + \$100 (training)	\$140 (training)	N/A
Training cost (13B)	135K GPU-hours	N/A	\$300 (training)	N/A

Preliminary evaluation

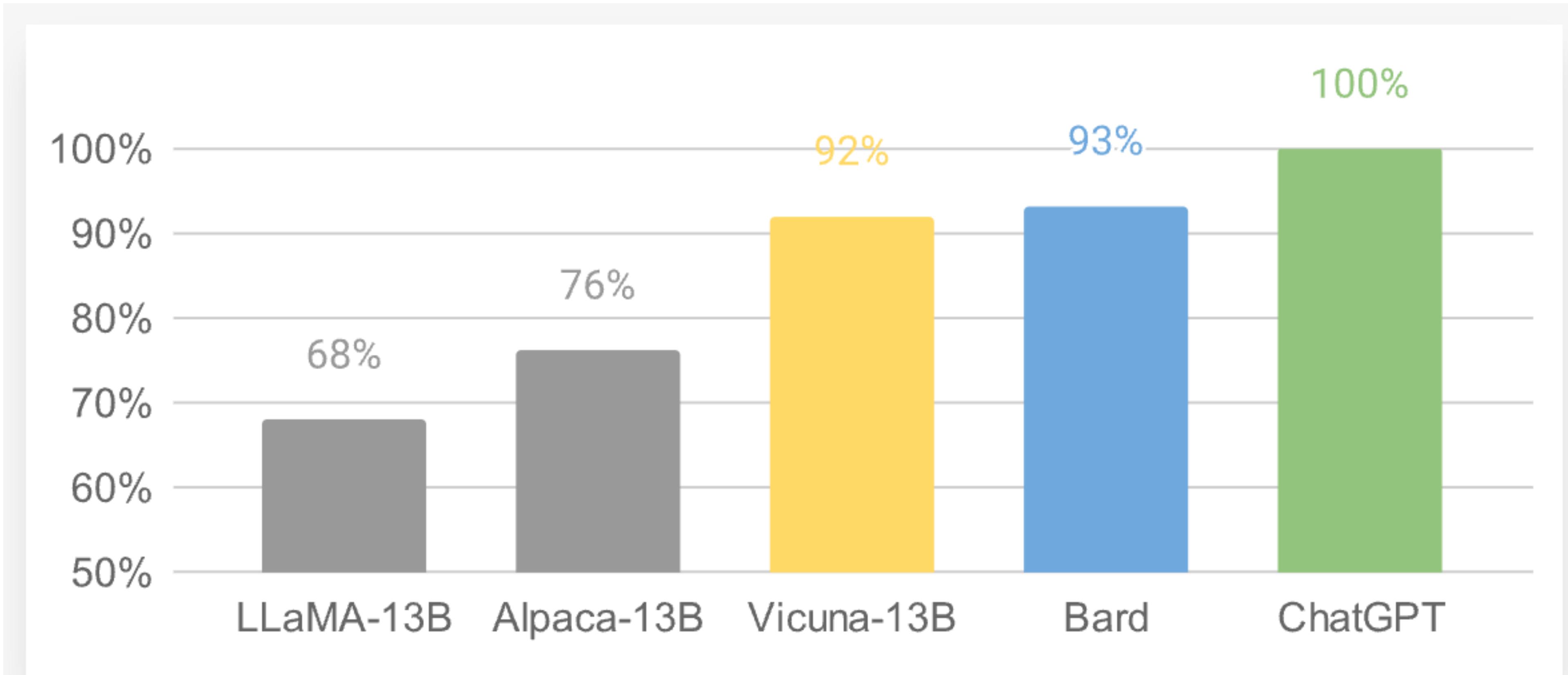


Figure 1. Relative Response Quality Assessed by GPT-4*

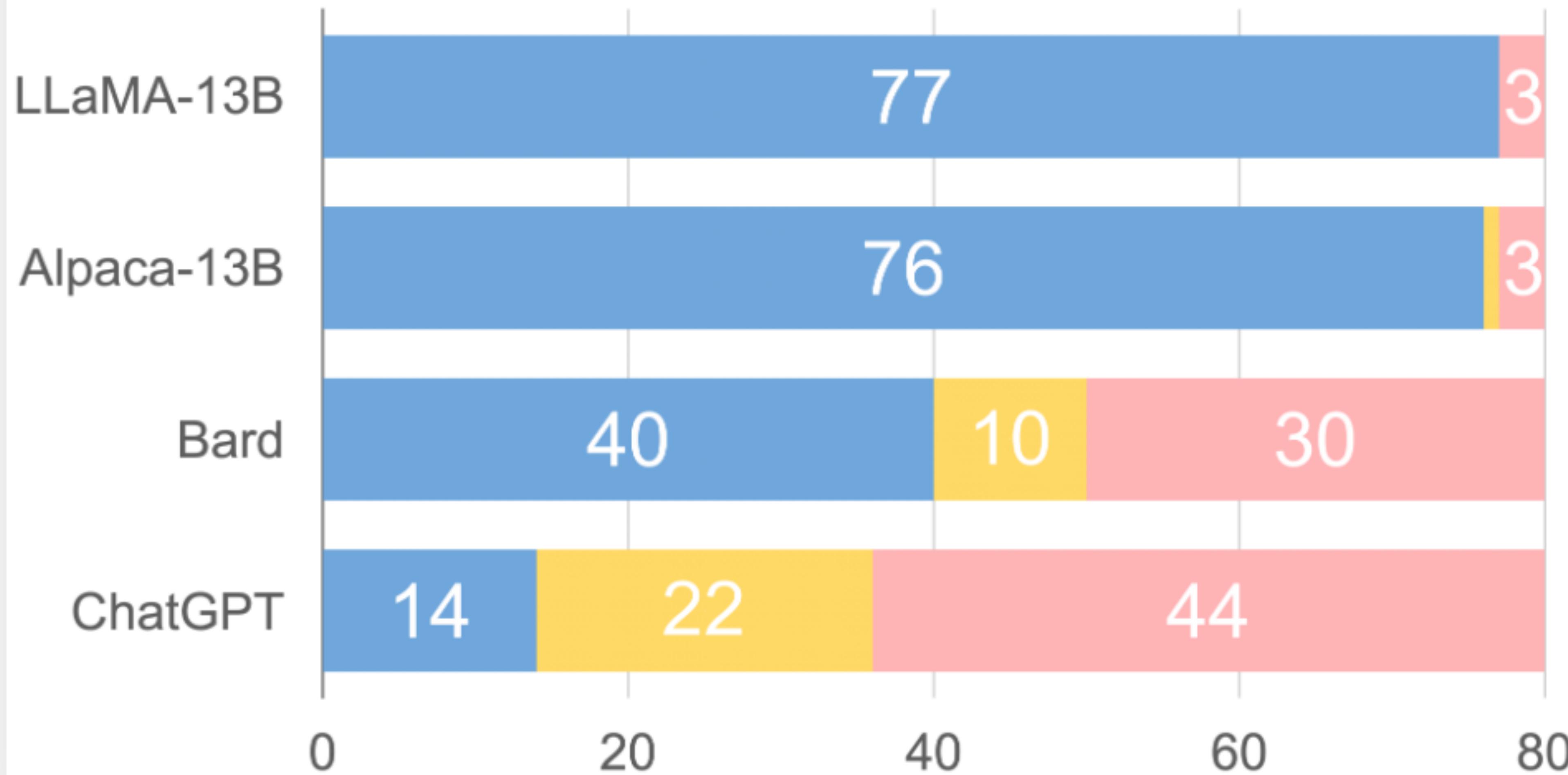


Figure 3. Response Comparison Assessed by GPT-4

Test set: 80 examples

- Draft an apology email to a customer who experienced a delay in their order, and provide reassurance that the issue has been resolved.
- How do language and cultural barriers affect the way people communicate and form relationships in multicultural societies?
- Given that $f(x) = 5x^3 - 2x + 3$, find the value of $f(2)$.
- How many snowflakes fall during a typical winter? Try to explain your answer. Your explanation should take the reader through your reasoning step-by-step.
- Implement a binary search algorithm to find a specific element in a sorted array.

LIMA: Less Is More for Alignment

Chunting Zhou^{μ*} Pengfei Liu^{π*} Puxin Xu^μ Srini Iyer^μ Jiao Sun^λ

Yuning Mao^μ Xuezhe Ma^λ Avia Efrat^τ Ping Yu^μ Lili Yu^μ Susan Zhang^μ

Gargi Ghosh^μ Mike Lewis^μ Luke Zettlemoyer^μ Omer Levy^μ

^μ Meta AI

^π Carnegie Mellon University

^λ University of Southern California

^τ Tel Aviv University

Abstract

Large language models are trained in two stages: (1) unsupervised pretraining from raw text, to learn general-purpose representations, and (2) large scale instruction tuning and reinforcement learning, to better align to end tasks and user preferences. We measure the relative importance of these two stages by training LIMA, a 65B parameter LLaMa language model fine-tuned with the standard supervised loss on only 1,000 carefully curated prompts and responses, without any reinforcement learning or human preference modeling. LIMA demonstrates remarkably strong performance, learning to follow specific response formats from only a handful of examples in the training data, including complex queries that range from planning trip itineraries to speculating about alternate history. Moreover, the model tends to generalize well to unseen tasks that did not appear in the training data. In a controlled human study, responses from LIMA are either equivalent or strictly preferred to GPT-4 in 43% of cases; this statistic is as high as 58% when compared to Bard and 65% versus DaVinci003, which was trained with human feedback. Taken together, these results strongly suggest that almost all knowledge in large language models is learned during pretraining, and only limited instruction tuning data is necessary to teach models to produce high quality output.

Source	#Examples	Avg Input Len.	Avg Output Len.
Training			
Stack Exchange (STEM)	200	117	523
Stack Exchange (Other)	200	119	530
wikiHow	200	12	1,811
Pushshift r/WritingPrompts	150	34	274
Natural Instructions	50	236	92
Paper Authors (Group A)	200	40	334
Dev			
Paper Authors (Group A)	50	36	N/A
Test			
Pushshift r/AskReddit	70	30	N/A
Paper Authors (Group B)	230	31	N/A

Table 1: Sources of training prompts (inputs) and responses (outputs), and test prompts. The total amount of training data is roughly 750,000 tokens, split over exactly 1,000 sequences.

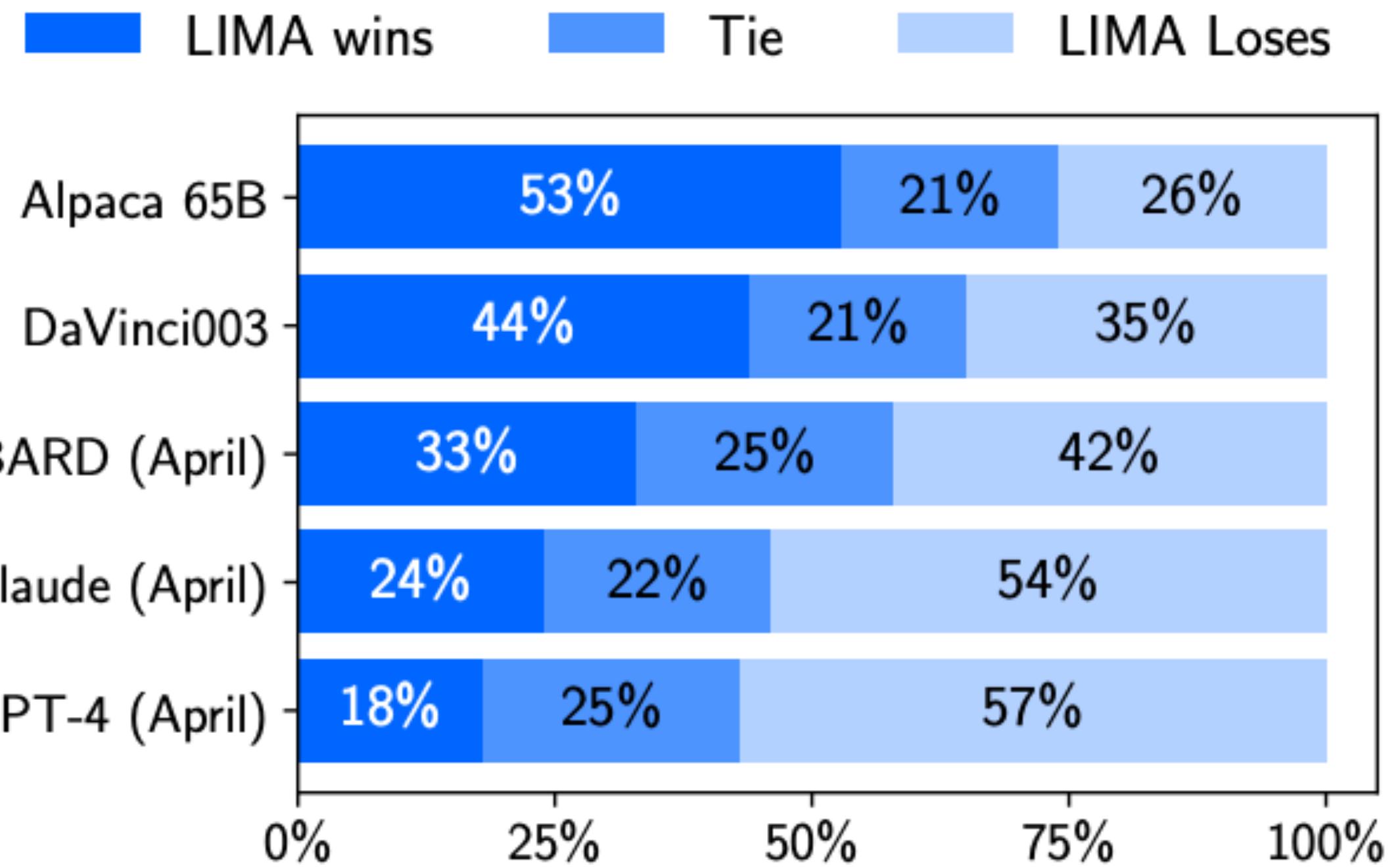


Figure 1: Human preference evaluation, comparing LIMA to 5 different baselines across 300 test prompts.

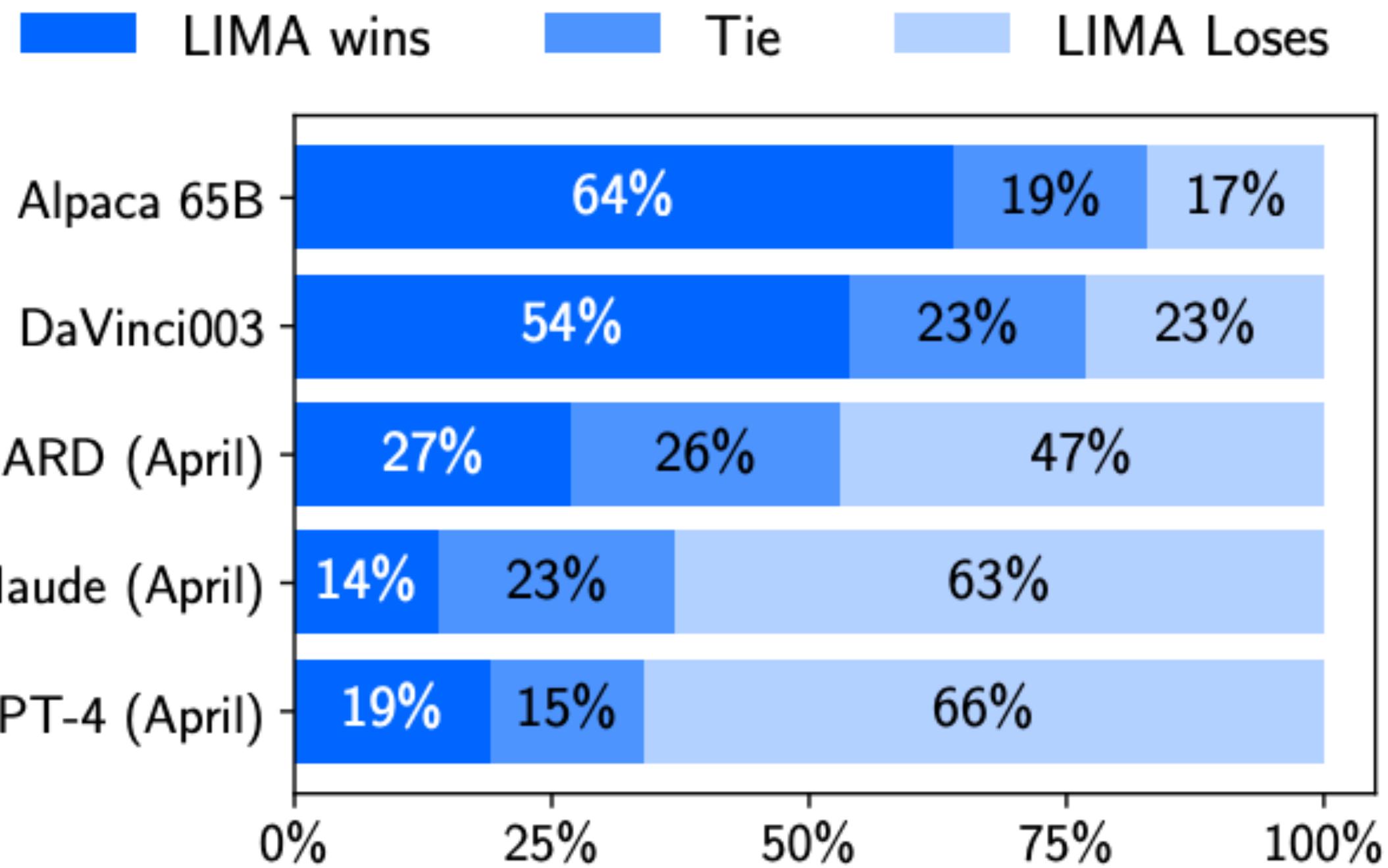


Figure 2: Preference evaluation using GPT-4 as the annotator, given the same instructions provided to humans.

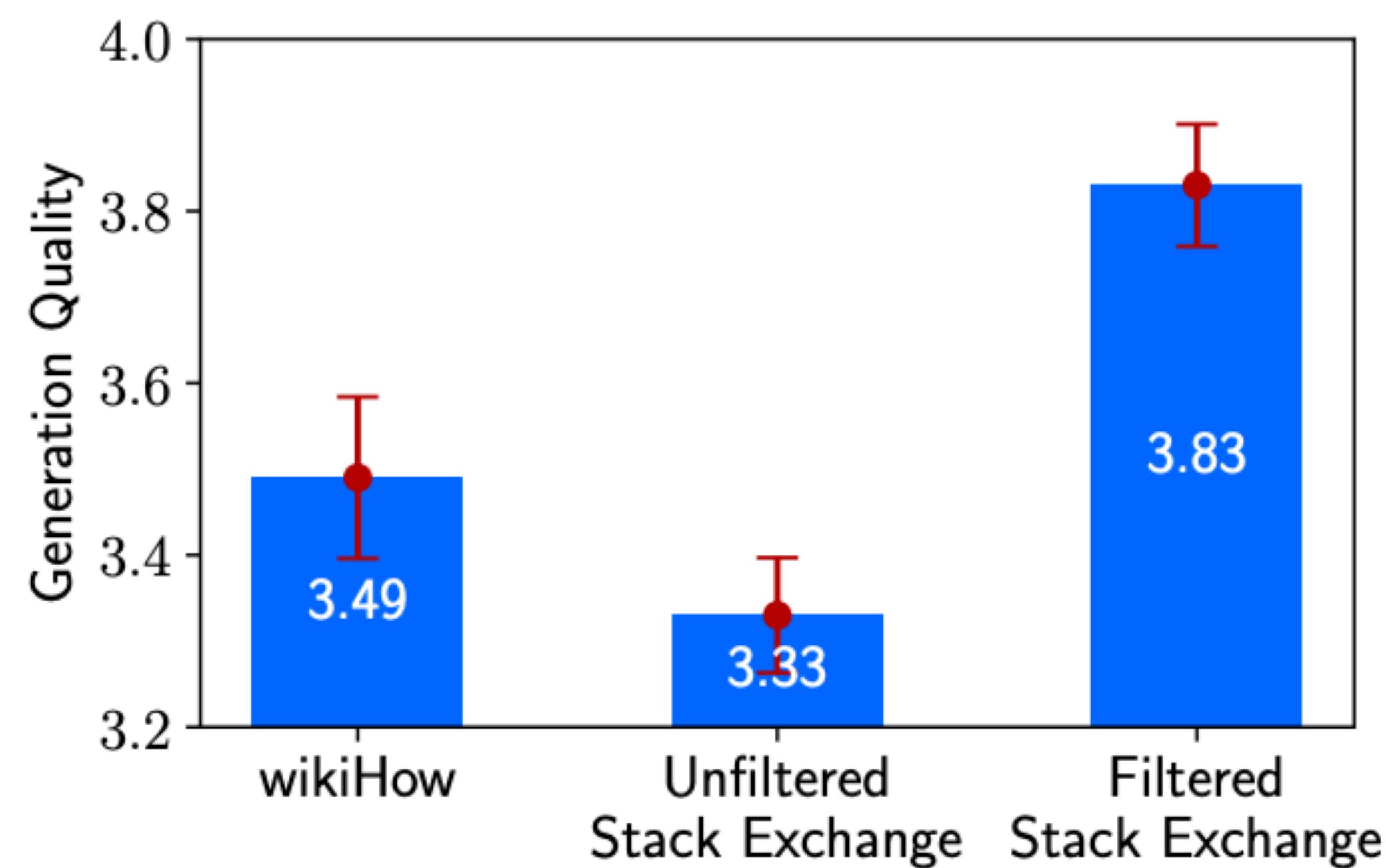


Figure 5: Performance of 7B models trained with 2,000 examples from different sources. **Filtered Stack Exchange** contains diverse prompts and high quality responses; **Unfiltered Stack Exchange** is diverse, but does not have any quality filters; **wikiHow** has high quality responses, but all of its prompts are “how to” questions.

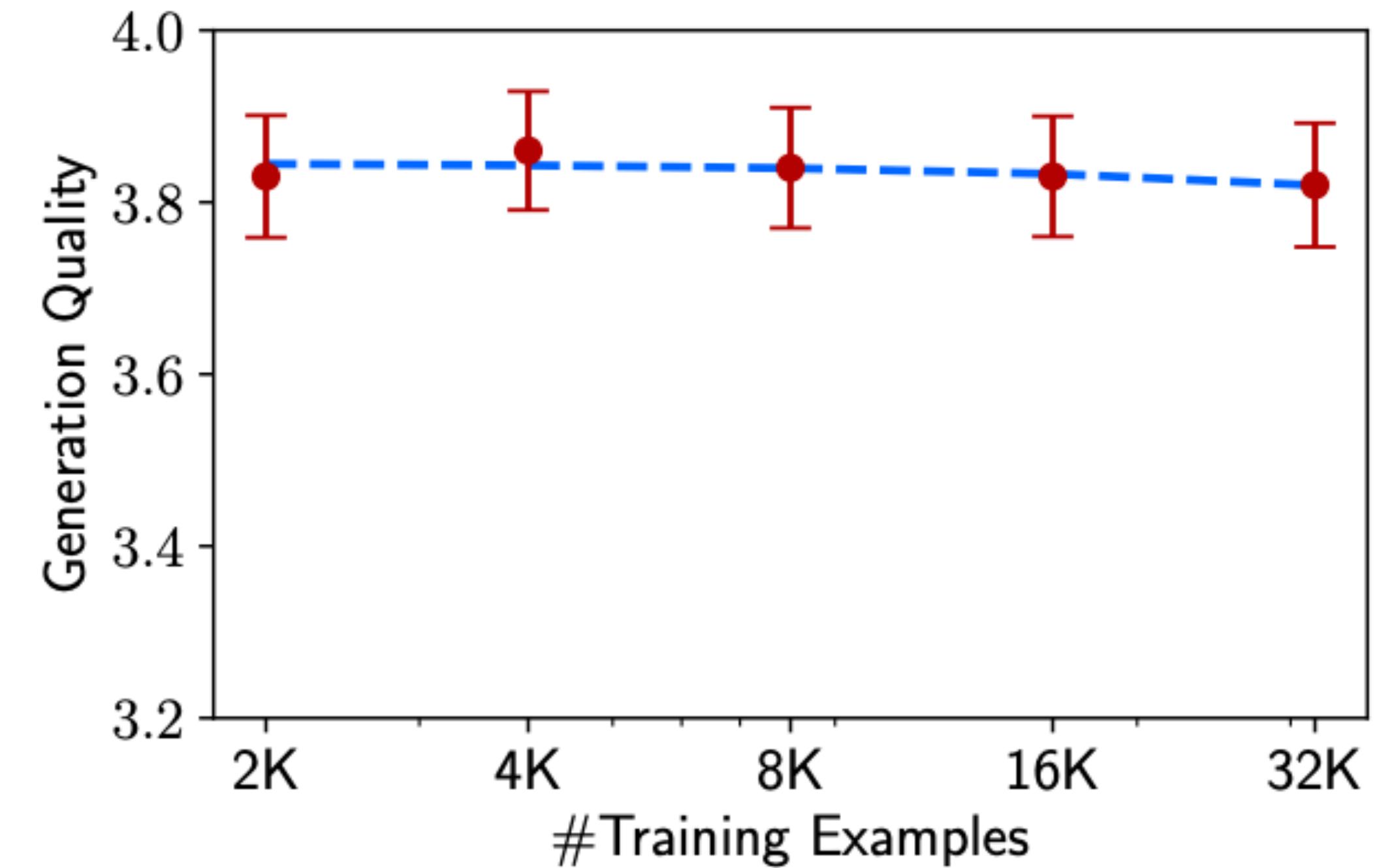


Figure 6: Performance of 7B models trained with exponentially increasing amounts of data, sampled from (quality-filtered) Stack Exchange. Despite an up to 16-fold increase in data size, performance as measured by ChatGPT plateaus.

*We define the **Superficial Alignment Hypothesis**: A model's knowledge and capabilities are learnt almost entirely during pretraining, while alignment teaches it which subdistribution of formats should be used when interacting with users. If this hypothesis is correct, and alignment is largely about learning style, then a corollary of the Superficial Alignment Hypothesis is that one could sufficiently tune a pretrained language model with a rather small set of examples [Kirstain et al., 2021].*

The False Promise of Imitating Proprietary LLMs

Arnav Gudibande*
UC Berkeley
arnavg@berkeley.edu

Xinyang Geng
UC Berkeley
young.geng@berkeley.edu

Sergey Levine
UC Berkeley
svlevine@berkeley.edu

Eric Wallace*
UC Berkeley
ericwallace@berkeley.edu

Hao Liu
UC Berkeley
hao.liu@berkeley.edu

Dawn Song
UC Berkeley
dawnsong@berkeley.edu

Charlie Snell*
UC Berkeley
csnell22@berkeley.edu

Pieter Abbeel
UC Berkeley
pabbeel@berkeley.edu

Abstract

An emerging method to cheaply improve a weaker language model is to finetune it on outputs from a stronger model, such as a proprietary system like ChatGPT (e.g., Alpaca, Self-Instruct, and others). This approach looks to cheaply imitate the

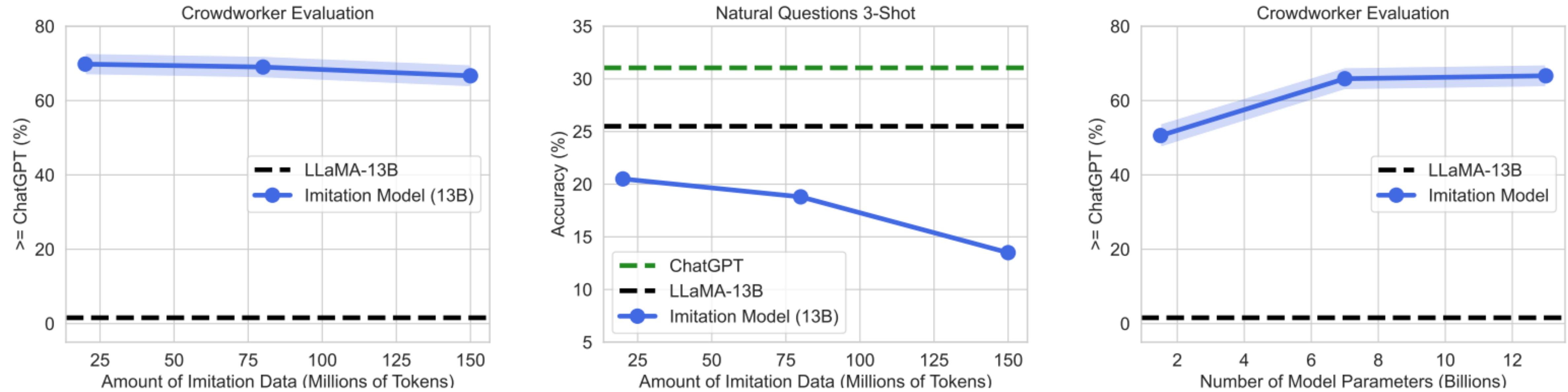
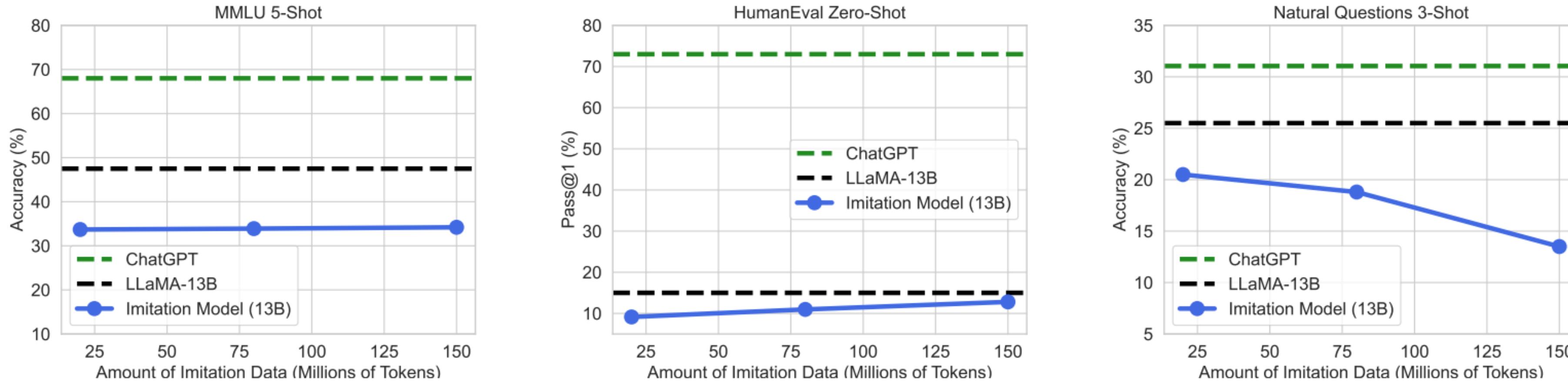


Figure 1: Crowdworkers initially rate the quality of our imitation models highly, as $\sim 70\%$ of their outputs are rated as equal or better than those of ChatGPT (*left*). However, as we train on more imitation data, our models fail to further close the gap, and even begin to regress along other axes, e.g. factual knowledge according to Natural Questions (*center*). Our main conclusion is that the biggest limitation of current open-source LMs is their weaker base capabilities. In turn, the best way for the open-source community to improve models is by increasing these capabilities (e.g., via scaling, better pretraining data, etc.,) rather than fine-tuning on more and more imitation data (*right*).

Increasing Amount of Imitation Data



Increasing Size of Imitation LM

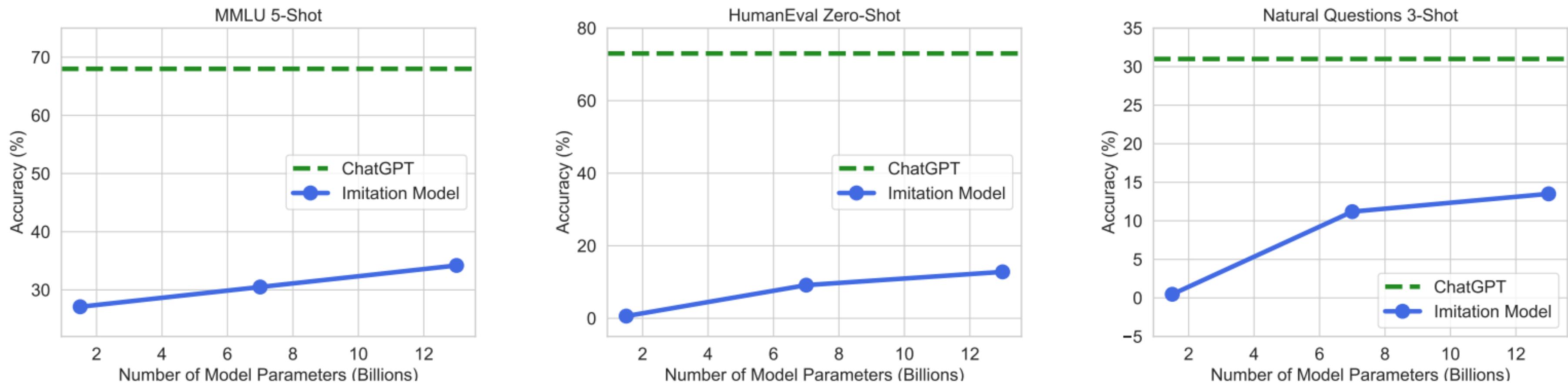


Figure 4: Automatic evaluations. As we increase the amount of imitation data, there is little improvement on various benchmarks, or even performance regressions (*top*). On the other hand, scaling up the base LM steadily improves results (*bottom*), suggesting that the key difference between open-source and closed-source LMs is a raw capabilities gap, rather than the finetuning data used.