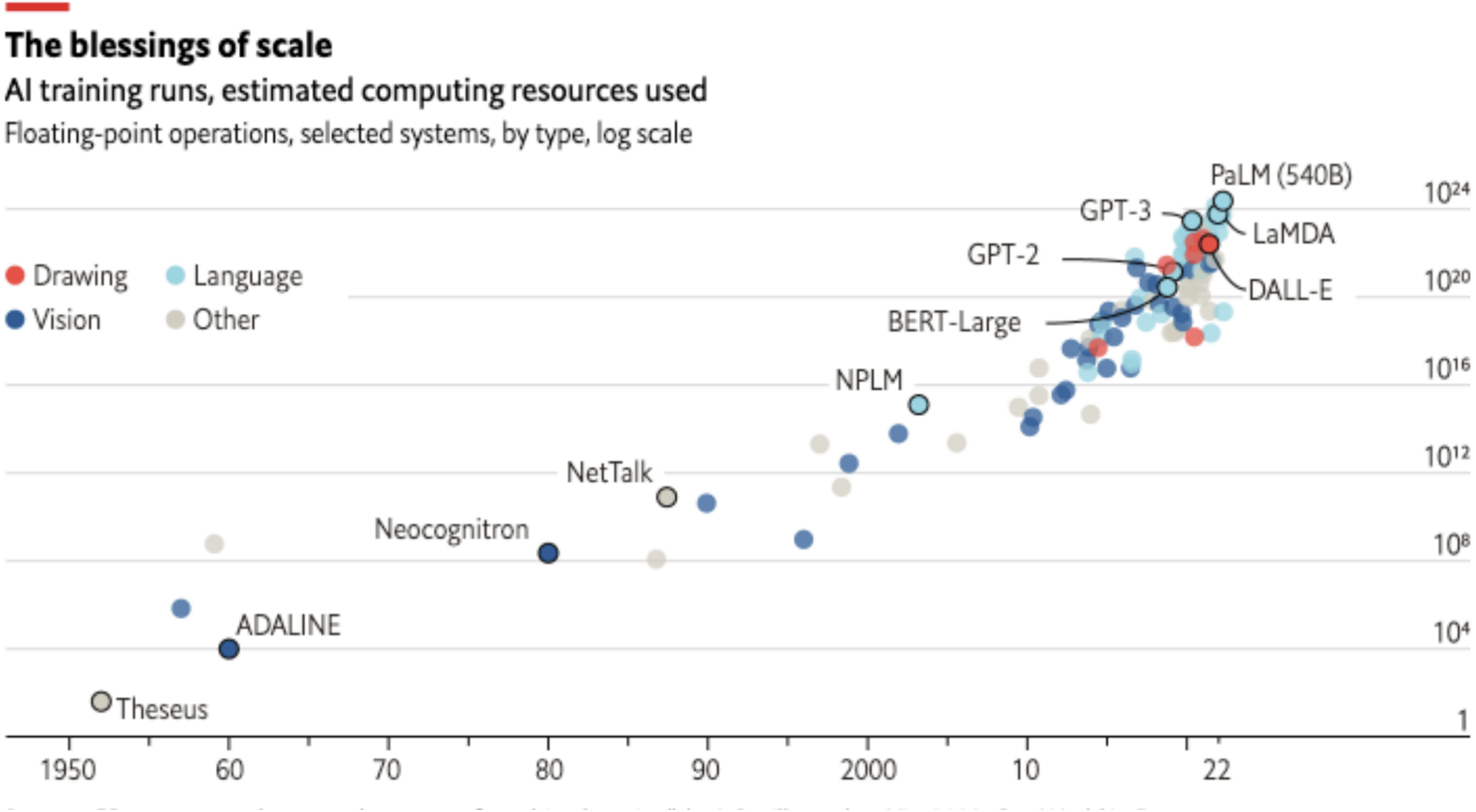
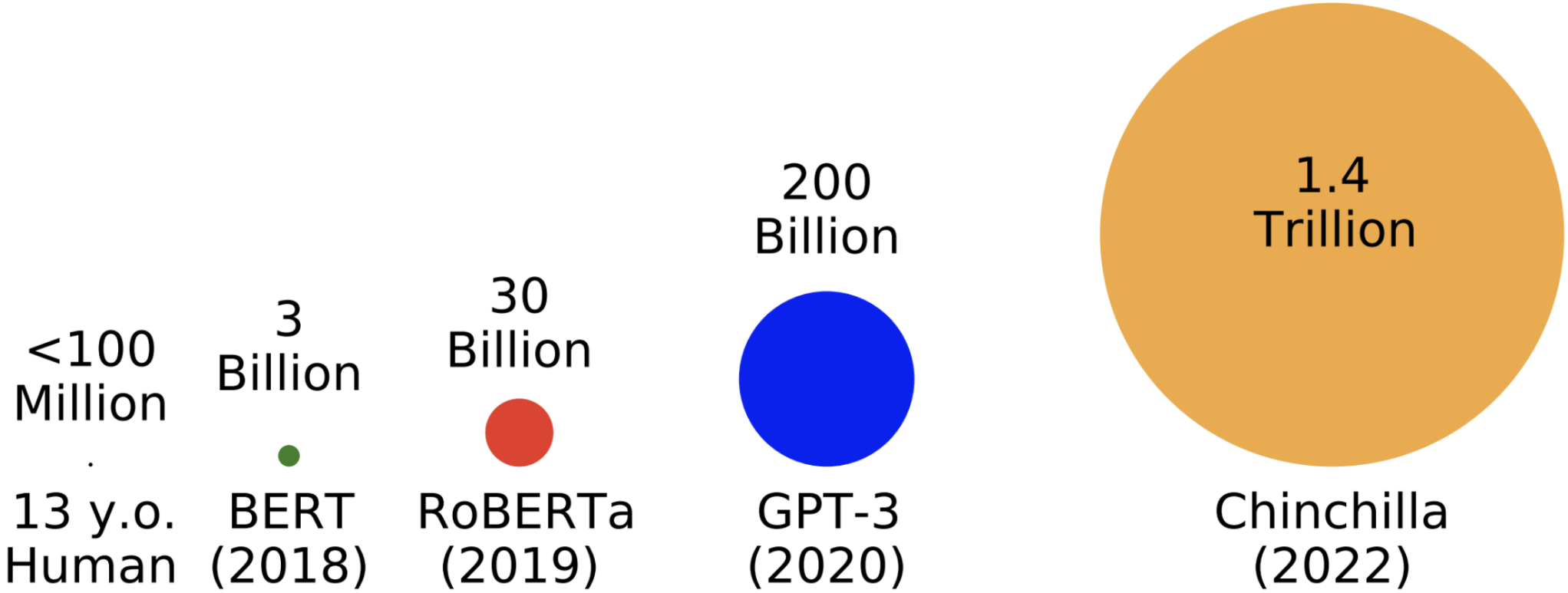
CIS 6930 Special Topics in Large Language Models

LLM Post-Training

Larger and Larger Models



More and More Data



tokens seen during training

https://babylm.github.io/

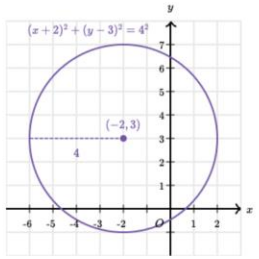
What does LLM learn?

- *I put ___ fork down on the table.* [syntax]
- *The woman walked across the street, checking for traffic over ___ shoulder.* [coreference]
- *I went to the ocean to see the fish, turtles, seals, and _____.* [lexical semantics/topic]
- *Overall, the value I got from the two hours watching it was the sum total of the popcorn and the drink. The movie was ___.* [sentiment]
- Iroh went into the kitchen to make some tea. Standing next to Iroh, Zuko pondered his destiny. Zuko left the _____. [some reasoning – this is harder]
- I was thinking about the sequence that goes 1, 1, 2, 3, 5, 8, 13, 21, ____ [some basic arithmetic; they don't learn the Fibonnaci sequence]

LLM as Multi-task Helper?

We can describe circles in the xy-plane using equations in terms of x and y. Circle equations questions require us to understand the connection between these equations and the features of circles.

For example, the equation $(x+2)^2+(y-3)^2=4^2$ is graphed in the xy-plane below. It is a circle with a center at $(-2, 3)$ and a radius of 4.



Math

https://www.khanacademy.org/test-prep/sat/x0a8c2e5f:untitled-652

Rapid and chronic ethanol tolerance are composed of distinct memory-like states in Drosophila

Abstract

Ethanol tolerance is the first type of behavioral plasticity and neural plasticity that is induced by ethanol intake, and yet its molecular and circuit bases remain largely unexplored. Here, we characterize three distinct forms of ethanol tolerance in male Drosophila: rapid, chronic, and repeated. Rapid tolerance is composed of two short-lived memory-like states, one that is labile and one that is consolidated. Chronic tolerance, induced by continuous exposure, lasts for two days, induces ethanol preference, and hinders the development of rapid tolerance through the activity of

```
// Determine whether the sentiment of text is positive
// Use a web service
async function isPositive(text: string): Promise<boolean> {
  const response = await fetch(`http://text-processing.com/api/sentiment/`, {
    method: "POST",
    body: `text=${text}`,
    headers: {
      "Content-Type": "application/x-www-form-urlencoded",
    },
  });
  const json = await response.json();
  return json.label === "pos";
}
```

Code

Medicine

What happens after LLM pre-training?

Outline

- Zero-shot and Few-shot in In-context Learning
- Instruction Tuning
- Optimization for Human Preference

1. Reinforcement Learning from Human Feedback (RLHF)
2. Direct Preference Optimization (DPO)

In-Context Learning

Emergent Capability of LLM

GPT (117M parameters; Radford et al., 2018)

- Transformer decoder with 12 layers.
- Trained on BooksCorpus: over 7000 unique books (4.6GB text).

GPT-2 (1.5B parameters; Radford et al., 2019)

- Same architecture as GPT, just bigger (117M -> 1.5B)
- But trained on **much more data**: 4GB -> 40GB of internet text data (WebText)

GPT-3 (175B parameters; Brown et al., 2020)

- Another increase in size (1.5B -> **175B**)
- and data (40GB -> **over 600GB**)

Emergent Zero-Shot Learning

One key emergent ability in GPT-2 is **zero-shot learning**: the ability to do many tasks with **no examples,** and **no gradient updates,** by simply:

- Specifying the right sequence prediction problem (e.g. question answering):

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

- Comparing probabilities of sequences (e.g. Winograd Schema Challenge [Levesque, 2011]):

  `The cat couldn't fit into the hat because it was too big.`
  Does `it = the cat` **or** `the hat?`

  ≡ Is `P(...because` **`the cat`** `was too big) >=`
  `P(...because` **`the hat`** `was too big)?`

Emergent Zero-Shot Learning

GPT-2 beats SoTA on language modeling benchmarks with no task-specific fine-tuning

Context: "Why?" "I would have thought you'd find him rather dry," she said. "I don't know about that," said Gabriel. "He was a great craftsman," said Heather. "That he was," said Flannery.
Target sentence: "And Polish, to boot," said _____.
Target word: Gabriel

LAMBADA (language modeling w/ long discourse dependencies) [Paperno et al., 2016]

	LAMBADA (PPL)	LAMBADA (ACC)	CBT-CN (ACC)	CBT-NE (ACC)	WikiText2 (PPL)
SOTA	99.8	59.23	85.7	82.3	39.14
117M	**35.13**	45.99	**87.65**	**83.4**	**29.41**
345M	**15.60**	55.48	**92.35**	**87.1**	**22.76**
762M	**10.87**	**60.12**	**93.45**	**88.0**	**19.93**
1542M	**8.63**	**63.24**	**93.30**	**89.05**	**18.34**

Emergent Zero-Shot Learning

You can get interesting zero-shot behavior if you're creative enough with how you specify your task!

Summarization on CNN/DailyMail dataset [See et al., 2017]:

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

Select from article

"Too Long, Didn't Read"

"Prompting"?

		ROUGE		
		R-1	R-2	R-L
2018 SoTA	Bottom-Up Sum	**41.22**	**18.68**	**38.34**
	Lede-3	40.38	17.66	36.62
Supervised (287K)	Seq2Seq + Attn	31.33	11.81	28.83
	GPT-2 TL;DR:	29.34	8.27	26.58
	Random-3	28.78	8.63	25.52

[Radford et al., 2019]

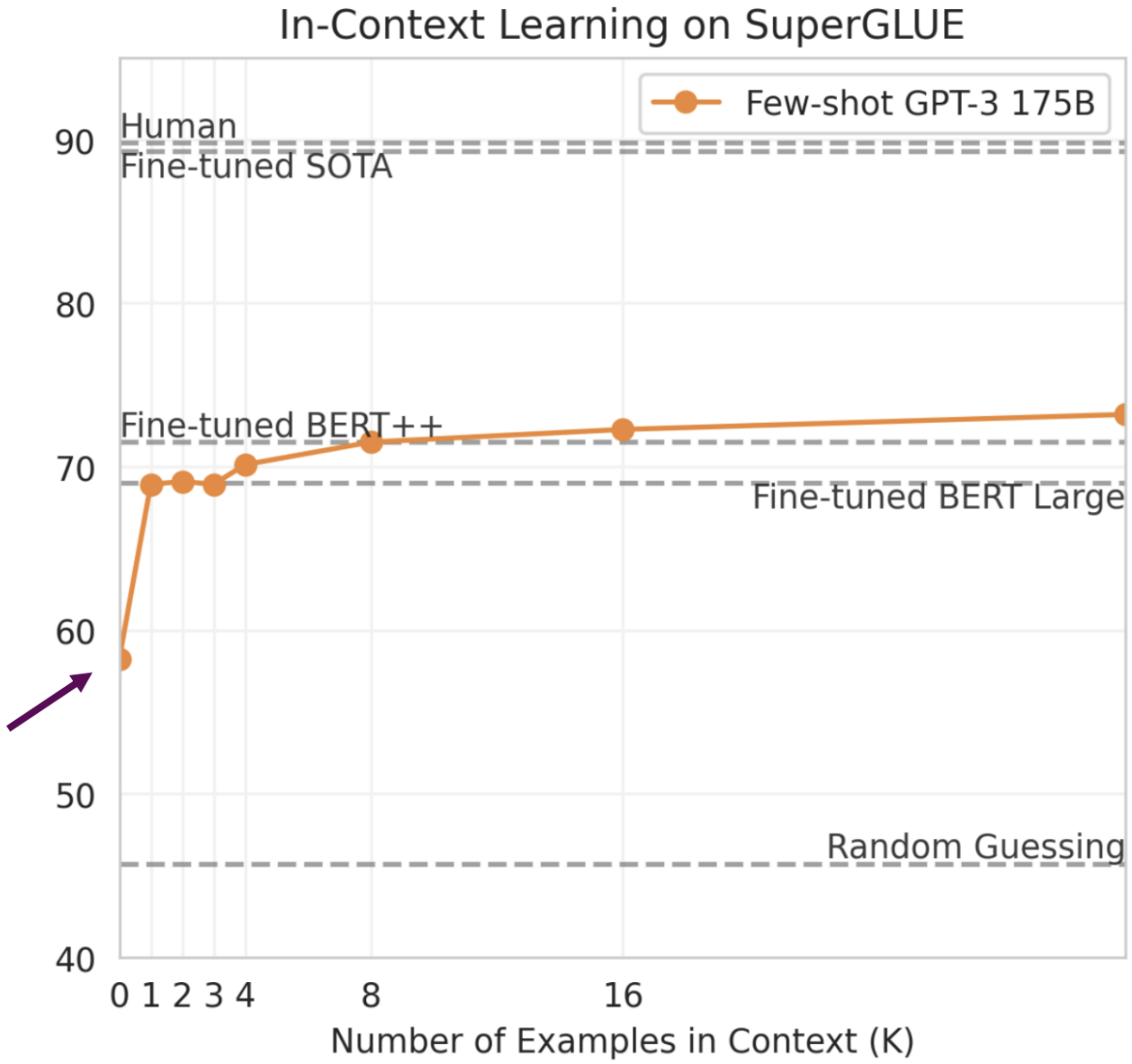
Emergent Few-Shot Learning

- Specify a task by simply **prepending examples of the task before your example**
- Also called **in-context learning**, to stress that *no gradient updates* are performed when learning a new task (there is a separate literature on few-shot learning with gradient updates)

Emergent Few-Shot Learning

Zero-shot

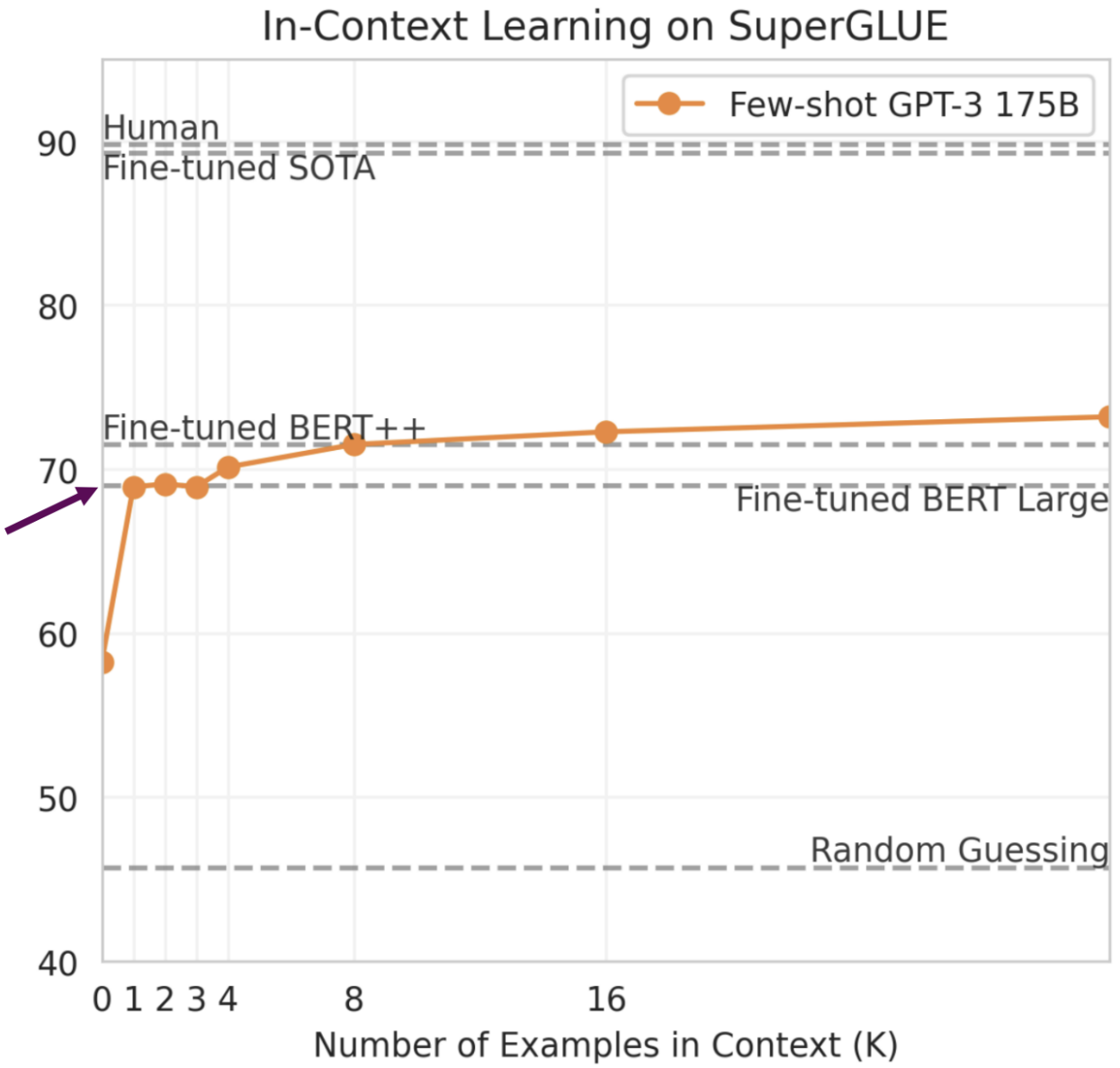
```
1  Translate English to French:
2  cheese =>
```



Emergent Few-Shot Learning

One-shot

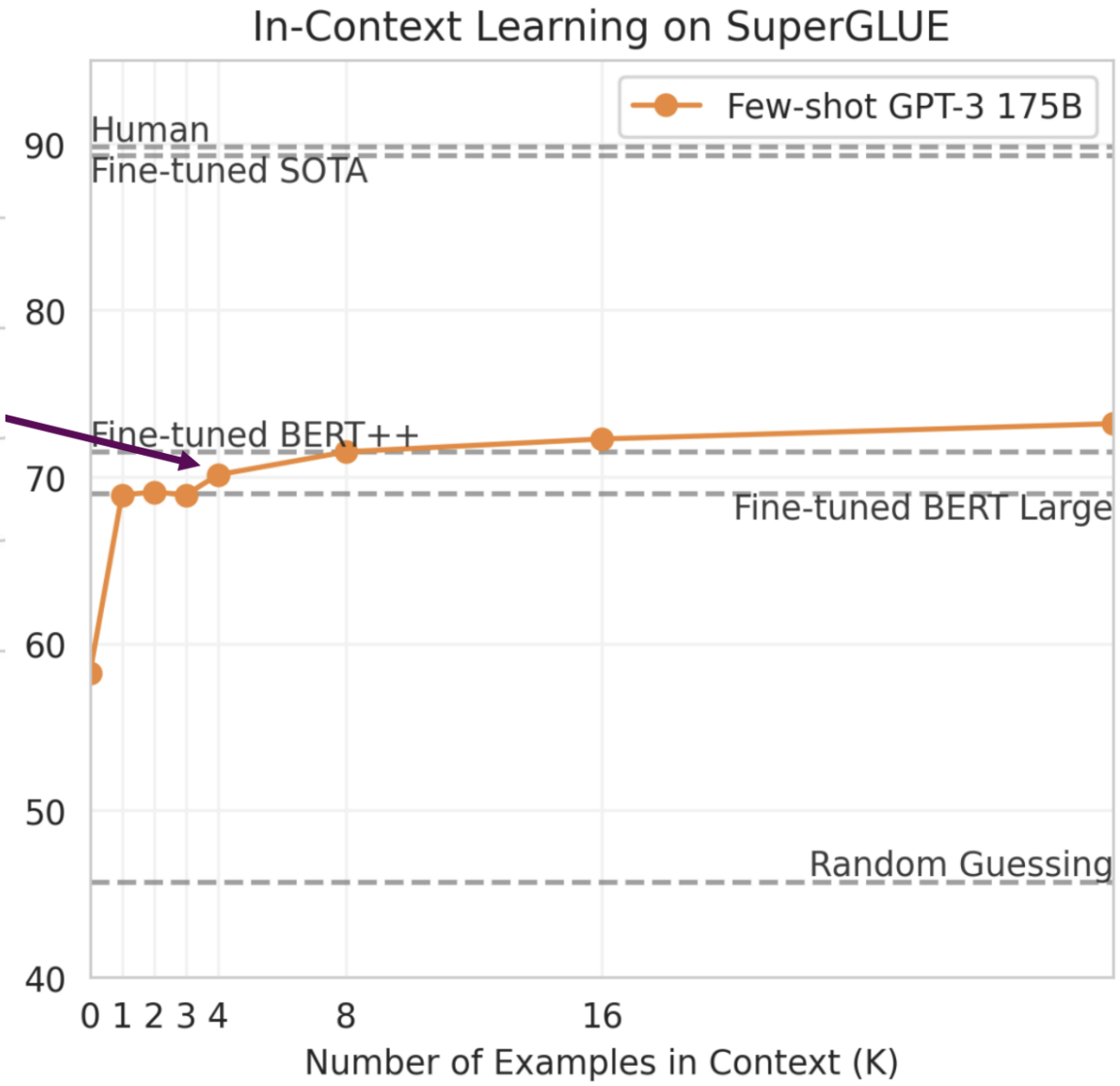
```
1  Translate English to French:
2  sea otter => loutre de mer
3  cheese =>
```



Emergent Few-Shot Learning

Few-shot

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



New Methods of Prompting LM

Zero/few-shot prompting

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

Traditional fine-tuning

```
1  sea otter => loutre de mer
```

gradient update

```
1  peppermint => menthe poivrée
```

gradient update

```
1  cheese =>
```

Limits of Prompting for harder tasks

Some tasks seem too hard for even large LMs to learn through prompting alone.

Especially tasks involving **richer, multi-step reasoning.**

(Humans struggle at these tasks too!)

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

Solution: change the prompt!

Chain-of-Thought Prompting

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.

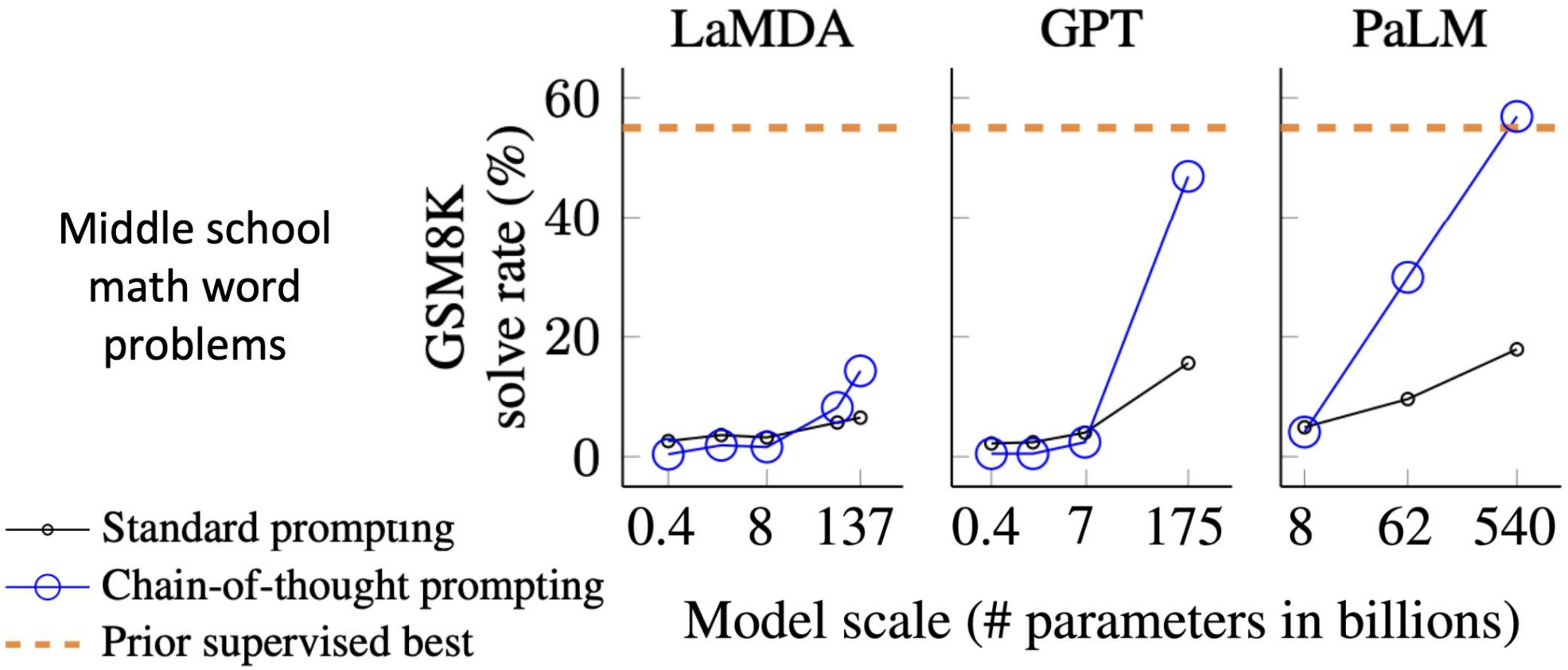
Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

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. ✓

Chain-of-Thought Prompting

Chain-of-Thought prompting is an emergent property of model scale



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. ✓

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

Zero-shot Chain-of-Thought Prompting

Model Input

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

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

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

Model Output

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

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

A: ***Let's think step by step.*** *There are 16 balls in total. Half of the balls are golf balls. That means there are 8 golf balls. Half of the golf balls are blue. That means there are 4 blue golf balls.* ✓

Zero-shot Chain-of-Thought Prompting

	MultiArith	GSM8K
Zero-Shot	**17.7**	**10.4**
Few-Shot (2 samples)	33.7	15.6
Few-Shot (8 samples)	33.8	15.6
Zero-Shot-CoT	**78.7**	**40.7**
Few-Shot-CoT (2 samples)	84.8	41.3
Few-Shot-CoT (4 samples : First) (*1)	89.2	-
Few-Shot-CoT (4 samples : Second) (*1)	90.5	-
Few-Shot-CoT (8 samples)	93.0	48.7

Greatly outperforms zero-shot →

Manual CoT still better →

Zero-shot Chain-of-Thought Prompting

No.	Category	Zero-shot CoT Trigger Prompt	Accuracy
1	LM-Designed	Let's work this out in a step by step way to be sure we have the right answer.	**82.0**
2	Human-Designed	Let's think step by step. (*1)	78.7
3		First, (*2)	77.3
4		Let's think about this logically.	74.5
5		Let's solve this problem by splitting it into steps. (*3)	72.2
6		Let's be realistic and think step by step.	70.8
7		Let's think like a detective step by step.	70.3
8		Let's think	57.5
9		Before we dive into the answer,	55.7
10		The answer is after the proof.	45.7
-		(Zero-shot)	17.7

Recap

Zero-shot and Few-shot in In-context Learning

- Pro:

No finetuning needed, prompt engineering (e.g. CoT) is helpful

- Cons:

Limits to what you can fit in context

Complex tasks will probably need gradient steps

Instruction Tuning

LM may not align with user intent

PROMPT *Explain the moon landing to a 6 year old in a few sentences.*

COMPLETION GPT-3

```
Explain the theory of gravity to a 6 year old.

Explain the theory of relativity to a 6 year old in a few sentences.

Explain the big bang theory to a 6 year old.

Explain evolution to a 6 year old.
```

Language models are not *aligned* with user intent [Ouyang et al., 2022].

LM may not align with user intent

PROMPT *Explain the moon landing to a 6 year old in a few sentences.*

COMPLETION **Human**
A giant rocket ship blasted off from Earth carrying astronauts to the moon. The astronauts landed their spaceship on the moon and walked around exploring the lunar surface. Then they returned safely back to Earth, bringing home moon rocks to show everyone.

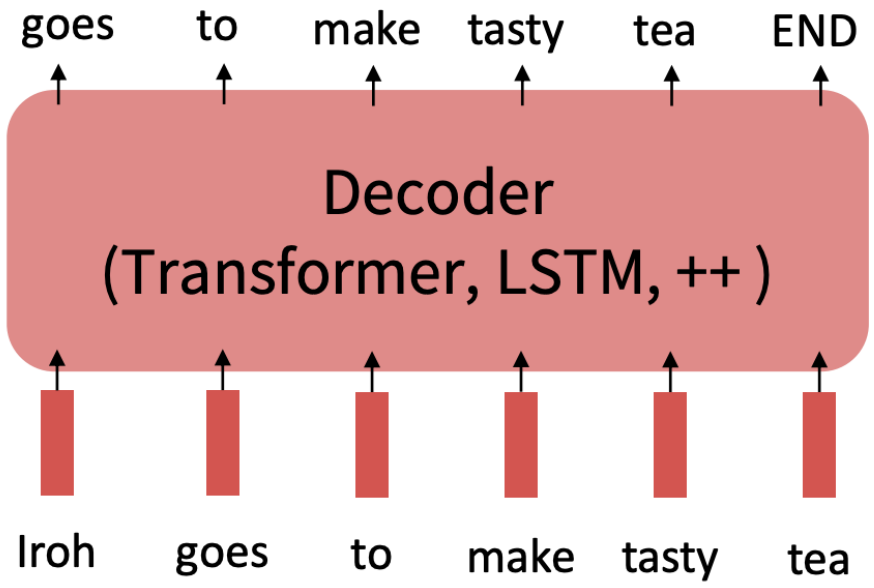
Language models are not *aligned* with user intent [Ouyang et al., 2022].
Finetuning to the rescue!

Recall Pretraining / Finetuning paradigm

Pretraining can improve NLP applications by serving as parameter initialization.

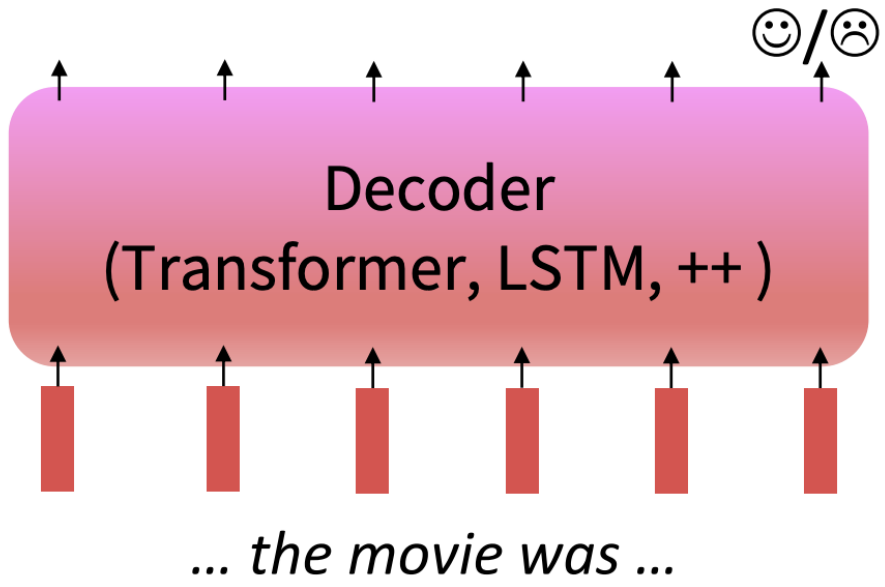
Step 1: Pretrain (on language modeling)

Lots of text; learn general things!



Step 2: Finetune (on your task)

Not many labels; adapt to the task!

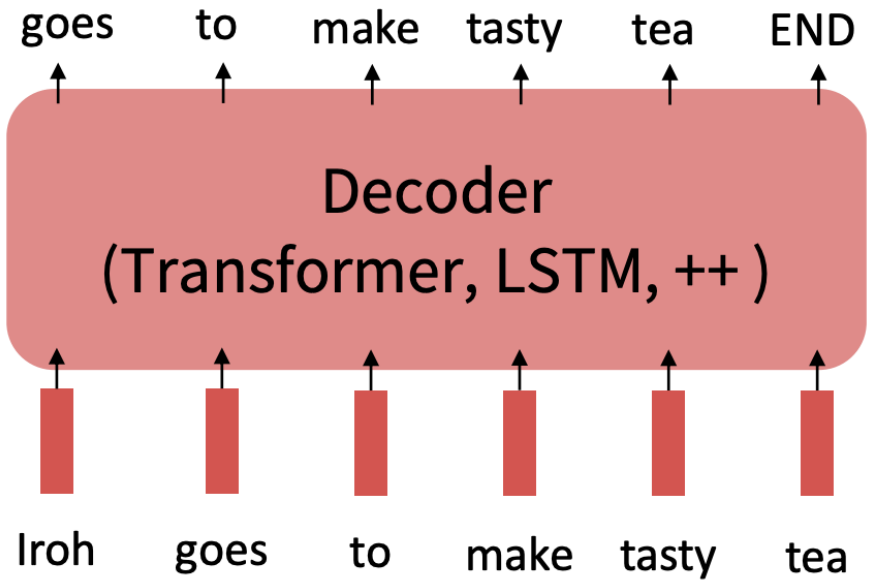


Scaling up Finetuning with Instructions

Pretraining can improve NLP applications by serving as parameter initialization.

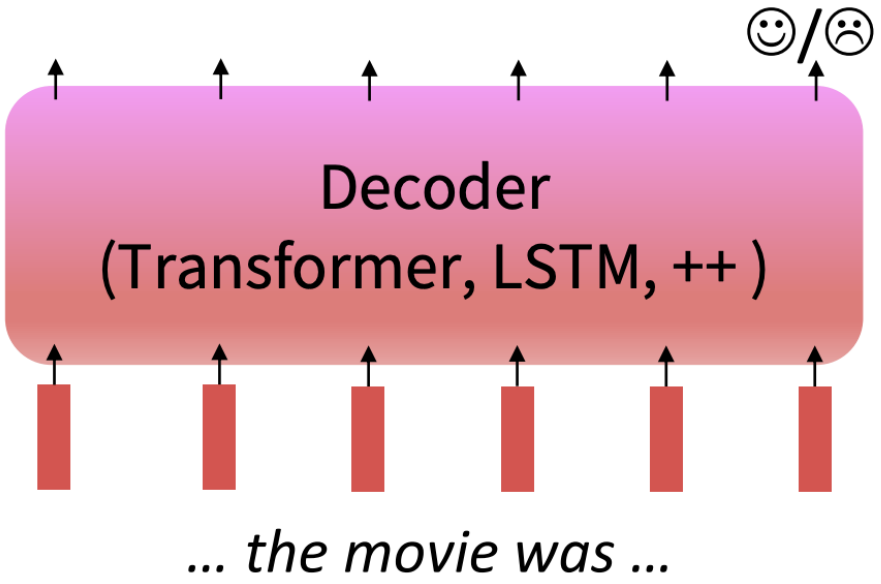
Step 1: Pretrain (on language modeling)

Lots of text; learn general things!



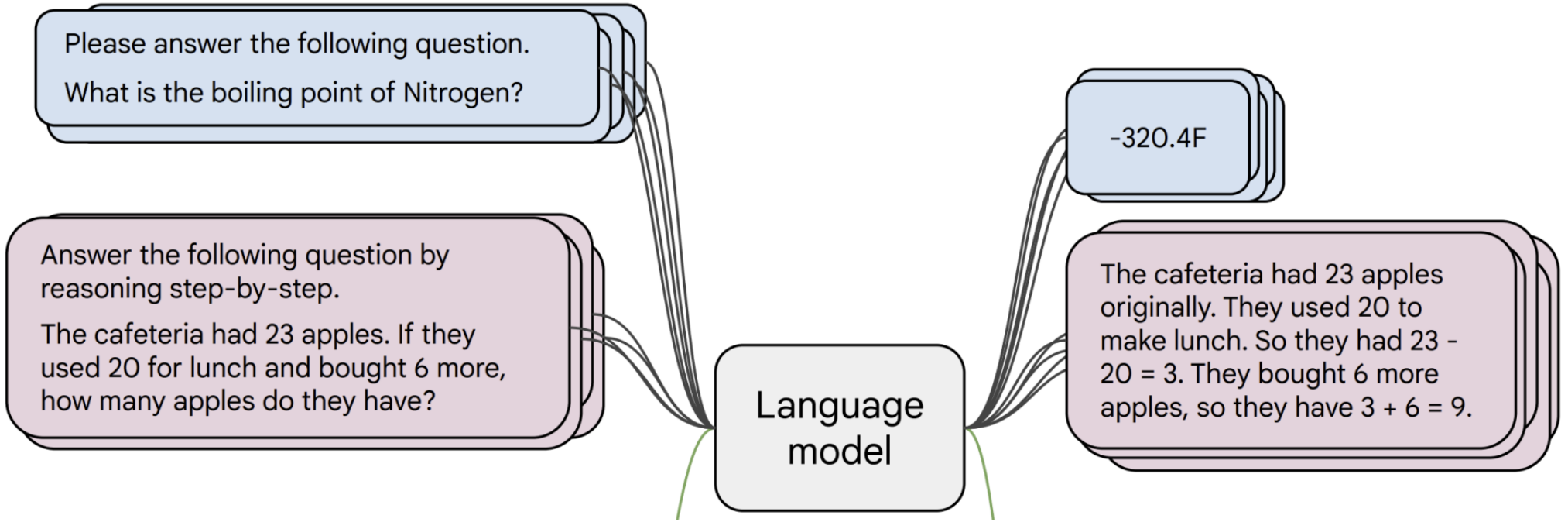
Step 2: Finetune (on many tasks)

~~Not~~ many labels; adapt to the tasks!



Instruction Tuning

- **Collect examples** of (instruction, output) pairs across many tasks and finetune an LM

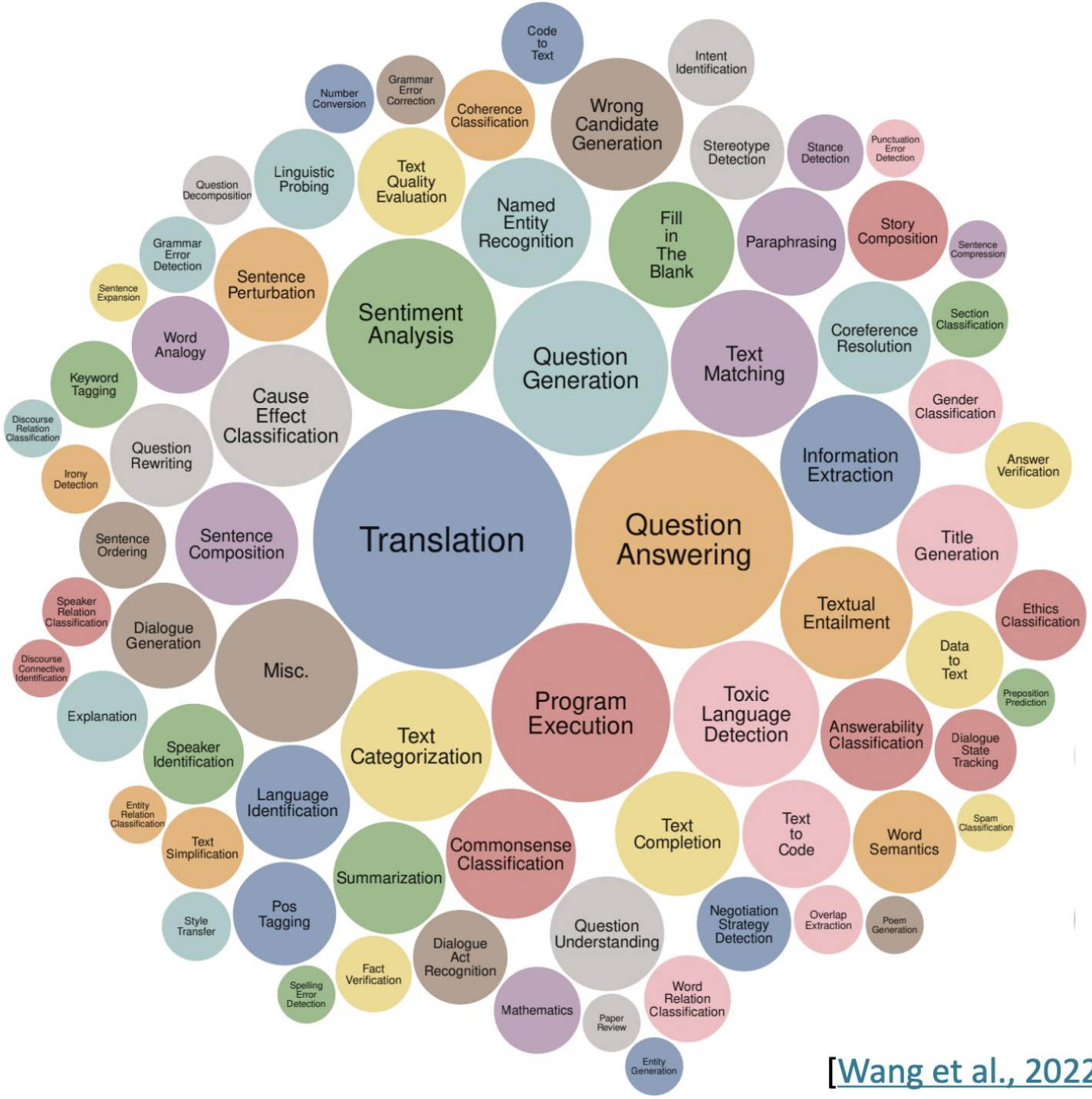


- Evaluate on **unseen tasks**

Instruction Tuning

For example, **Super-Natural Instructions** dataset contains over **1.6K tasks**, **3M+** examples

Tasks include classification, sequence tagging, rewriting, translation, QA...

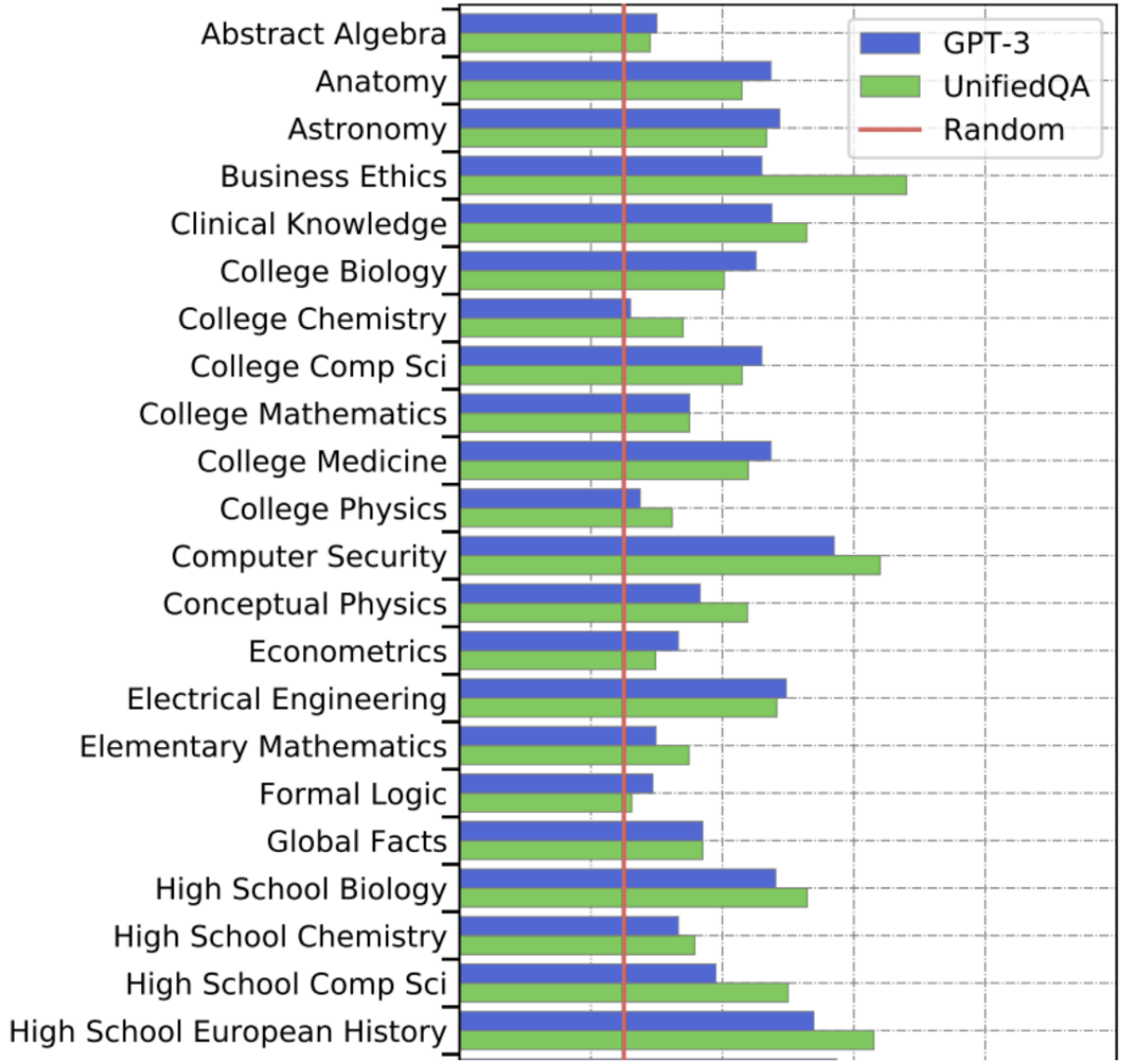


[Wang et al., 2022]

Evaluate LLM after Instruction Tuning

Massive Multitask Language Understanding (MMLU)
[Hendrycks et al., 2021]

New benchmarks for measuring LM performance on 57 diverse *knowledge intensive* tasks



Evaluate LLM after Instruction Tuning

BIG-Bench [Srivastava et al., 2022]

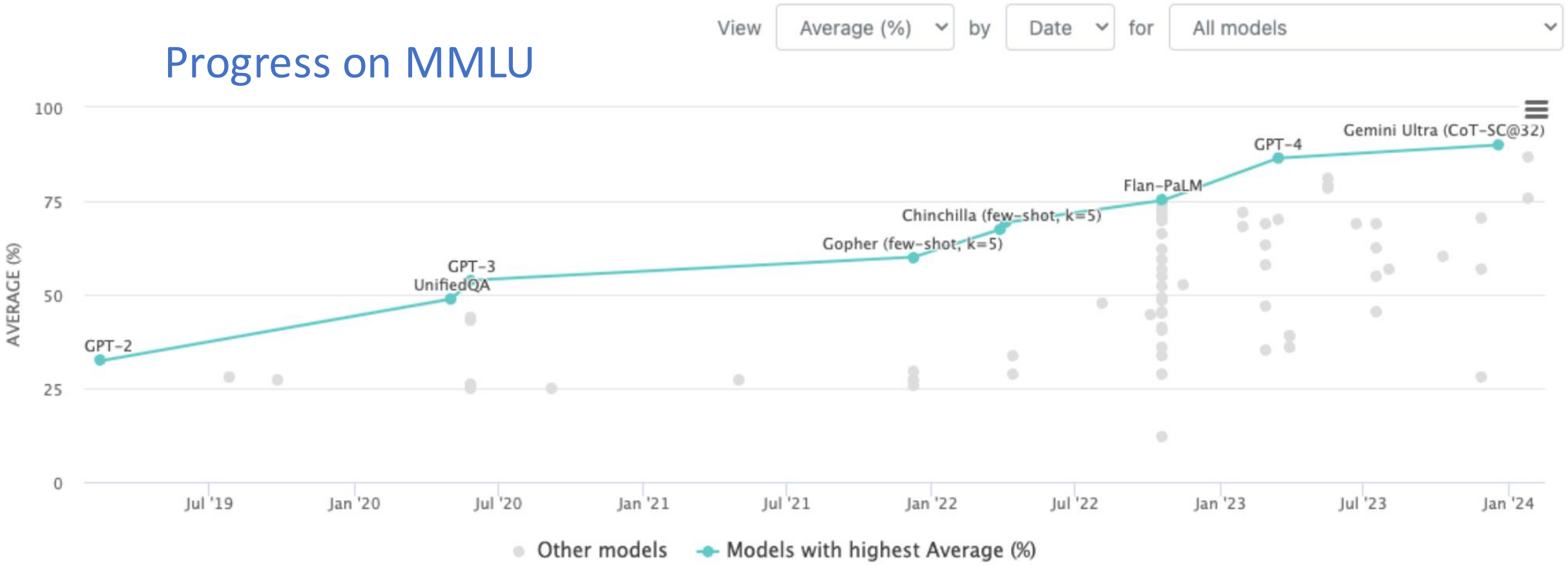
200+ tasks, spanning:

https://github.com/google/BIG-bench/blob/main/bigbench/benchmark_tasks/README.md

BEYOND THE IMITATION GAME: QUANTIFYING AND EXTRAPOLATING THE CAPABILITIES OF LANGUAGE MODELS

Alphabetic author list:*
Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R. Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, Agnieszka Kluska, Aitor Lewkowycz, Akshat Agarwal, Alethea Power, Alex Ray, Alex Warstadt, Alexander W. Kocurek, Ali Safaya, Ali Tazarv, Alice Xiang, Alicia Parrish, Allen Nie, Aman Hussain, Amanda Askell, Amanda Dsouza, Ambrose Slone, Ameet Rahane, Anantharaman S. Iyer, Anders Andreassen, Andrea Madotto, Andrea Santilli, Andreas Stuhlmüller, Andrew Dai, Andrew La, Andrew Lampinen, Andy Zou, Angela Jiang, Angelica Chen, Anh Vuong, Animesh Gupta, Anna Gottardi, Antonio Norelli, Anu Venkatesh, Arash Gholamidavoodi, Arfa Tabassum, Arul Menezes, Arun Kirubarajan, Asher Mullokandov, Ashish Sabharwal, Austin Herrick, Avia Efrat, Aykut Erdem, Ayla Karakaş, B. Ryan Roberts, Bao Sheng Loe, Barret Zoph, Bartłomiej Bojanowski, Batuhan Özyurt, Behnam Hedayatnia, Behnam Neyshabur, Benjamin Inden, Benno Stein, Berk Ekmekci, Bill Yuchen Lin, Blake Howald, Cameron Diao, Cameron Dour, Catherine Stinson, Cedrick Argueta, César Ferri Ramírez, Chandan Singh, Charles Rathkopf, Chenlin Meng, Chitta Baral, Chiyu Wu, Chris Callison-Burch, Chris Waites, Christian Voigt, Christopher D. Manning, Christopher Potts, Cindy Ramirez, Clara E. Rivera, Clemencia Siro, Colin Raffel, Courtney Ashcraft, Cristina Garbacea, Damien Sileo, Dan Garrette, Dan Hendrycks, Dan Kilman, Dan Roth, Daniel Freeman, Daniel Khashabi, Daniel Levy, Daniel Moseguí González, Danielle Perszyk, Danny Hernandez, Danqi Chen, Daphne Ippolito, Dar Gilboa, David Dohan, David Drakard, David Jurgens, Debajyoti Datta, Deep Ganguli, Denis Emelin, Denis Kleyko, Deniz Yuret, Derek Chen, Derek Tam, Dieuwke Hupkes, Diganta Misra, Dilyar Buzan, Dimitri Coelho Mollo, Diyi Yang, Dong-Ho Lee, Ekaterina Shutova, Ekin Dogus Cubuk, Elad Segal, Eleanor Hagerman, Elizabeth Barnes, Elizabeth Donoway, Ellie Pavlick, Emanuele Rodola, Emma Lam, Eric Chu, Eric Tang, Erkut Erdem, Ernie Chang, Ethan A. Chi, Ethan Dyer, Ethan Jerzak, Ethan Kim, Eunice Engefu Manyasi, Evgenii Zheltonozhskii, Fanyue Xia, Fatemeh Siar, Fernando Martínez-Plumed, Francesca Happé, Francois Chollet, Frieda Rong, Gaurav Mishra, Genta Indra Winata, Gerard de Melo, Germán Kruszewski, Giambattista Parascandolo, Giorgio Mariani, Gloria Wang, Gonzalo Jaimovitch-López, Gregor Betz, Guy Gur-Ari, Hana Galijasevic, Hannah Kim, Hannah Rashkin, Hannaneh Hajishirzi, Harsh Mehta, Hayden Bogar, Henry Shevlin, Hinrich Schütze, Hiromu Yakura, Hongming Zhang, Hugh Mee Wong, Ian Ng, Isaac Noble, Jaap Jumelet, Jack Geissinger, Jackson Kernion, Jacob Hilton, Jaehoon Lee, Jaime Fernández Fisac, James B. Simon, James Koppel, James Zheng, James Zou, Jan Kocoń, Jana Thompson, Jared Kaplan, Jarema Radom, Jascha Sohl-Dickstein, Jason Phang, Jason Wei, Jason Yosinski, Jekaterina Novikova, Jelle Bosscher, Jennifer Marsh, Jeremy Kim, Jeroen Taal, Jesse Engel, Jesujoba Alabi, Jiacheng Xu, Jiaming Song, Jillian Tang, Joan Waweru, John Burden, John Miller, John U. Balis, Jonathan Berant, Jörg Frohberg, Jos Rozen, Jose Hernandez-Orallo, Joseph Boudeman, Joseph Jones, Joshua B. Tenenbaum, Joshua S. Rule, Joyce Chua, Kamil Kanclerz, Karen Livescu, Karl Krauth, Karthik Gopalakrishnan, Katerina Ignatyeva, Katja Markert, Kaustubh D. Dhole, Kevin Gimpel, Kevin Omondi, Kory Mathewson, Kristen Chiafullo, Ksenia Shkaruta, Kumar Shridhar, Kyle McDonell, Kyle Richardson, Laria Reynolds, Leo Gao, Li Zhang, Liam Dugan, Lianhui Qin, Lidia Contreras-Ochando, Louis-Philippe Morency, Luca Moschella, Lucas Lam, Lucy Noble, Ludwig Schmidt, Luheng He, Luis Oliveros Colón, Luke Metz, Lütfi Kerem Şenel, Maarten Bosma, Maarten Sap, Maartje ter Hoeve, Maheen Farooqi, Manaal Faruqui, Mantas Mazeika, Marco Baturan, Marco Marelli, Marco Maru, Maria Jose Ramírez Quintana, Marie Tolkiehn, Mario Giulianelli, Martha Lewis, Martin Potthast, Matthew L. Leavitt, Matthias Hagen, Mátyás Schubert, Medina Orduna Baitemirova, Melody Arnaud, Melvin McElrath, Michael A. Yee, Michael Cohen, Michael Gu, Michael Ivanitskiy, Michael Starritt, Michael Strube, Michał Swędrowski, Michele Bevilacqua, Michihiro Yasunaga, Mihir Kale, Mike Cain, Mimee Xu, Mirac Suzgun, Mo Tiwari, Mohit Bansal, Moin Aminnaseri, Mor Geva, Mozhdeh Gheini, Mukund Varma T, Nanyun Peng, Nathan Chi, Nayeon Lee, Neta Gur-Ari Krakover, Nicholas Cameron, Nicholas Roberts, Nick Doiron, Nikita Nangia, Niklas Deckers, Niklas Muennighoff, Nitish Shirish Keskar, Niveditha S. Iyer, Noah Constant, Noah Fiedel, Nuan Wen, Oliver Zhang, Omar Agha, Omar Elbaghdadi, Omer Levy, Owain Evans, Pablo Antonio Moreno Casares, Parth Doshi, Pascale Fung, Paul Pu Liang, Paul Vicol, Pegah Alipoormolabashi, Peiyuan Liao, Percy Liang, Peter Chang, Peter Eckersley, Phu Mon Htut, Pinyu Hwang, Piotr Miłkowski, Piyush Patil, Pouya Pezeshkpour, Priti Oli, Qiaozhu Mei, Qing Lyu, Qinlang Chen, Rabin Banjade, Rachel Etta Rudolph, Raefer Gabriel, Rahel Habacker, Ramón Risco Delgado, Raphaël Millière, Rhythm Garg, Richard Barnes, Rif A. Saurous, Riku Arakawa, Robbe Raymaekers, Robert Frank, Rohan Sikand, Roman Novak, Roman Sitelew, Ronan LeBras, Rosanne Liu, Rowan Jacobs, Rui Zhang, Ruslan Salakhutdinov, Ryan Chi, Ryan Lee, Ryan Stovall, Ryan Teehan, Rylan Yang, Sahib Singh, Saif M. Mohammad, Sajant Anand, Sam Dillavou, Sam Shleifer, Sam Wiseman, Samuel Gruetter, Samuel R. Bowman, Samuel S. Schoenholz, Sanghyun Han, Sanjeev Kwatra, Sarah A. Rous, Sarik Ghazarian, Sayan Ghosh, Sean Casey, Sebastian Bischoff, Sebastian Gehrmann, Sebastian Schuster, Sepideh Sadeghi, Shadi Hamdan, Sharon Zhou, Shashank Srivastava, Sherry Shi, Shikhar Singh, Shima Asaadi, Shixiang Shane Gu, Shubh Pachchigar, Shubham Toshniwal, Shyam Upadhyay, Shyamolima (Shammie) Debnath, Siamak Shakeri, Simon Thormeyer, Simone Melzi, Siva Reddy, Sneha Priscilla Makini, Soo-Hwan Lee, Spencer Torene, Sriharsha Hatwar, Stanislas Dehaene, Stefan Divic, Stefano Ermon, Stella Biderman, Stephanie Lin, Stephen Prasad, Steven T. Piantadosi, Stuart M. Shieber, Summer Misherghi, Svetlana Kiritchenko, Swaroop Mishra, Tal Linzen, Tal Schuster, Tao Li, Tao Yu, Tariq Ali, Tatsu Hashimoto, Te-Lin Wu, Théo Desbordes, Theodore Rothschild, Thomas Phan, Tianle Wang, Tiberius Nkinyili, Timo Schick, Timofei Kornev, Timothy Telleen-Lawton, Titus Tunduny, Tobias Gerstenberg, Trenton Chang, Trishala Neeraj, Tushar Khot, Tyler Shultz, Uri Shaham, Vedant Misra, Vera Demberg, Victoria Nyamai, Vikas Raunak, Vinay Ramasesh, Vinay Uday Prabhu, Vishakh Padmakumar, Vivek Srikumar, William Fedus, William Saunders, William Zhang, Wout Vossen, Xiang Ren, Xiaoyu Tong, Xinran Zhao, Xinyi Wu, Xudong Shen, Yadollah Yaghoobzadeh, Yair Lakretz, Yangqiu Song, Yasaman Bahri, Yejin Choi, Yichi Yang, Yiding Hao, Yifu Chen, Yonatan Belinkov, Yu Hou, Yufang Hou, Yuntao Bai, Zachary Seid, Zhuoye Zhao, Zijian Wang, Zijie J. Wang, Zirui Wang, Ziyi Wu

Evaluate LLM after Instruction Tuning



impressive progress on challenging knowledge-intensive benchmarks

Instruction Tuning

T5 pretrained on span corruption task

Flan-T5: instruction tuning T5 models on 1.8k additional tasks

Params	Model	BIG-bench + MMLU avg (normalized)
80M	T5-Small	-9.2
	Flan-T5-Small	-3.1 (+6.1)
250M	T5-Base	-5.1
	Flan-T5-Base	6.5 (+11.6)
780M	T5-Large	-5.0
	Flan-T5-Large	13.8 (+18.8)
3B	T5-XL	-4.1
	Flan-T5-XL	19.1 (+23.2)
11B	T5-XXL	-2.9
	Flan-T5-XXL	23.7 (+26.6)

Bigger model = bigger Δ

[Chung et al., 2022]

Instruction Tuning

Model input (Disambiguation QA)

Q: In the following sentences, explain the antecedent of the pronoun (which thing the pronoun refers to), or state that it is ambiguous.

Sentence: The reporter and the chef will discuss their favorite dishes.

Options:
(A) They will discuss the reporter's favorite dishes
(B) They will discuss the chef's favorite dishes
(C) Ambiguous

A: Let's think step by step.

Before instruction finetuning

The reporter and the chef will discuss their favorite dishes.
The reporter and the chef will discuss the reporter's favorite dishes.
The reporter and the chef will discuss the chef's favorite dishes.
The reporter and the chef will discuss the reporter's and the chef's favorite dishes.

✖ **(doesn't answer question)**

Instruction Tuning

Model input (Disambiguation QA)

Q: In the following sentences, explain the antecedent of the pronoun (which thing the pronoun refers to), or state that it is ambiguous.

Sentence: The reporter and the chef will discuss their favorite dishes.

Options:
(A) They will discuss the reporter's favorite dishes
(B) They will discuss the chef's favorite dishes
(C) Ambiguous

A: Let's think step by step.

After instruction finetuning

The reporter and the chef will discuss their favorite dishes does not indicate whose favorite dishes they will discuss. So, the answer is (C).

Limitations of Instruction Tuning

- It's expensive to collect ground truth data for tasks
- Tasks like open-ended creative generation have no right answer

e.g. Write me a story about a dog and her pet grasshopper.

- Humans generate suboptimal answers
- Even with instruction finetuning, there a mismatch between the LM objective and the objective of "satisfying human preferences"

Recap

Instruction Tuning

- Pro:

Simple and straightforward, generalize to unseen tasks

- Cons:

Collecting demonstrations for so many tasks is expensive

Mismatch between LM objective and human preferences

Optimization for Human Preference

Optimization for Human Preference

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

x

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

y_1

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

$R(x, y_1) = 8.0$

y_2

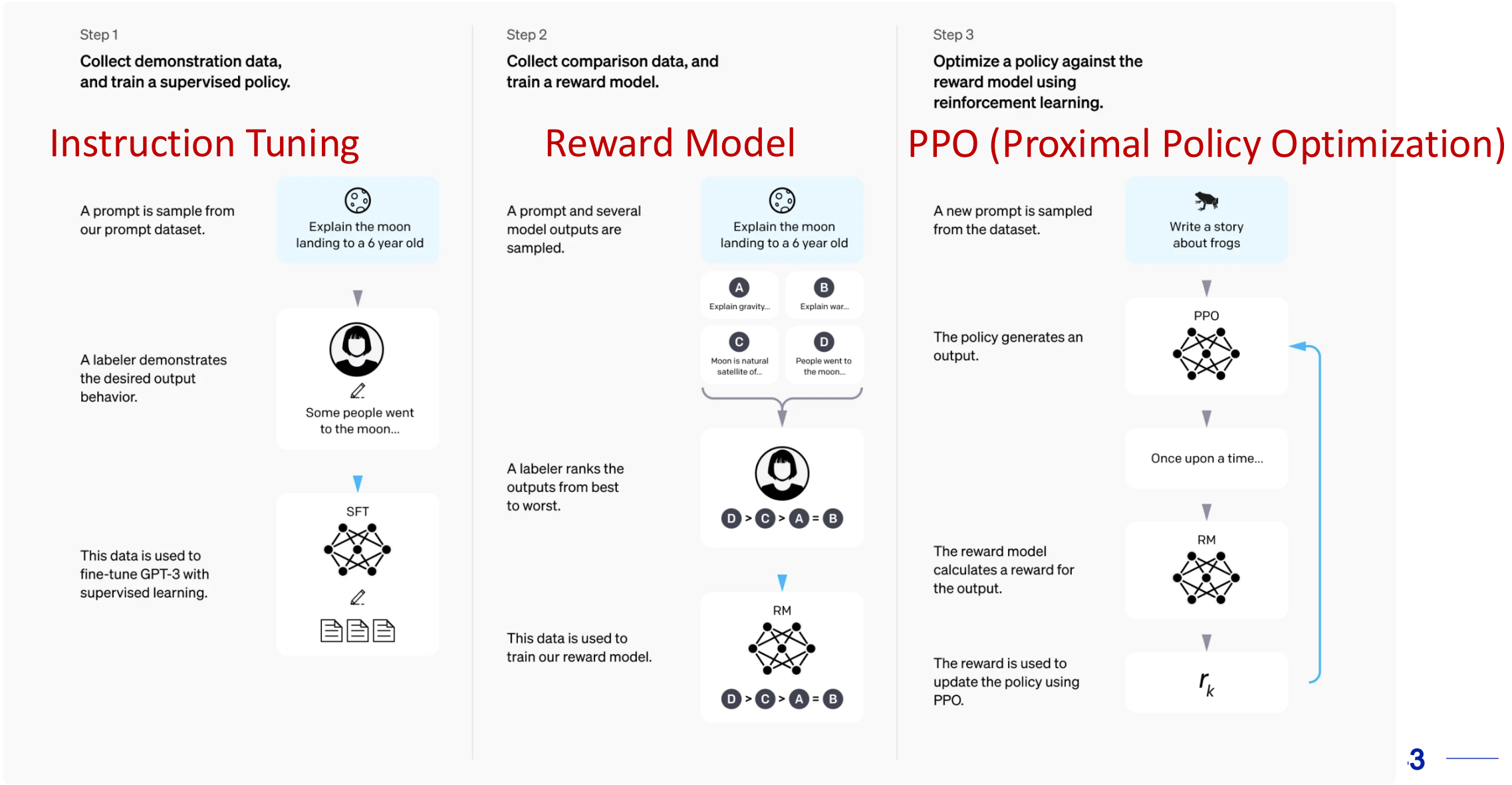
```
The Bay Area has
good weather but is
prone to
earthquakes and
wildfires.
```

$R(x, y_2) = 1.2$

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

$$\mathbb{E}_{\hat{y} \sim p_\theta(y \mid x)}[R(x, \hat{y})]$$

RLHF (Reinforcement Learning from Human Feedback)



RLHF – Reward Model

- **Problem 1:** human-in-the-loop is expensive!
  - **Solution:** instead of directly asking humans for preferences, **model their preferences** as a separate (NLP) problem! [Knox and Stone, 2009]

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

$R(x, y_1) = 8.0$

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

$R(x, y_2) = 1.2$

Train a $RM_\phi(x, y)$ to predict human reward from an annotated dataset, then optimize for RM_ϕ instead.

RLHF – Reward Model

- **Problem 2:** human judgments are noisy and miscalibrated!
- **Solution:** instead of asking for direct ratings, ask for **pairwise comparisons**, which can be more reliable [Phelps et al., 2015; Clark et al., 2018]

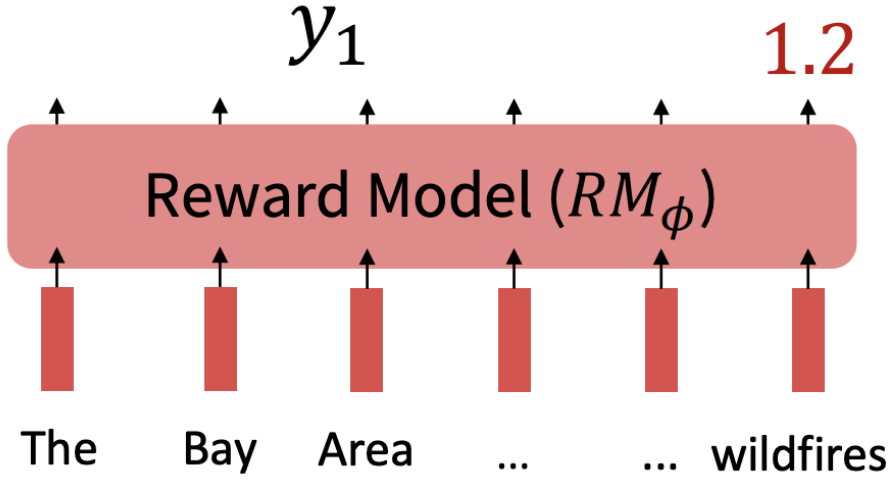
```
A 4.2 magnitude
earthquake hit
San Francisco,
resulting in
massive damage.
```

$$R(x, y_3) = \overset{y_3}{\underset{}{\begin{matrix} 4.1? & 6.6? & 3.2? \end{matrix}}}$$

RLHF – Reward Model

- **Problem 2:** human judgments are noisy and miscalibrated!
- **Solution:** instead of asking for direct ratings, ask for **pairwise comparisons**, which can be more reliable [Phelps et al., 2015; Clark et al., 2018]

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



y_3 y_2

Bradley-Terry [1952] paired comparison model

$$J_{RM}(\phi) = -\mathbb{E}_{(x, y^w, y^l)\sim D}\left[\log \sigma(RM_\phi(x, y^w) - RM_\phi(x, y^l))\right]$$

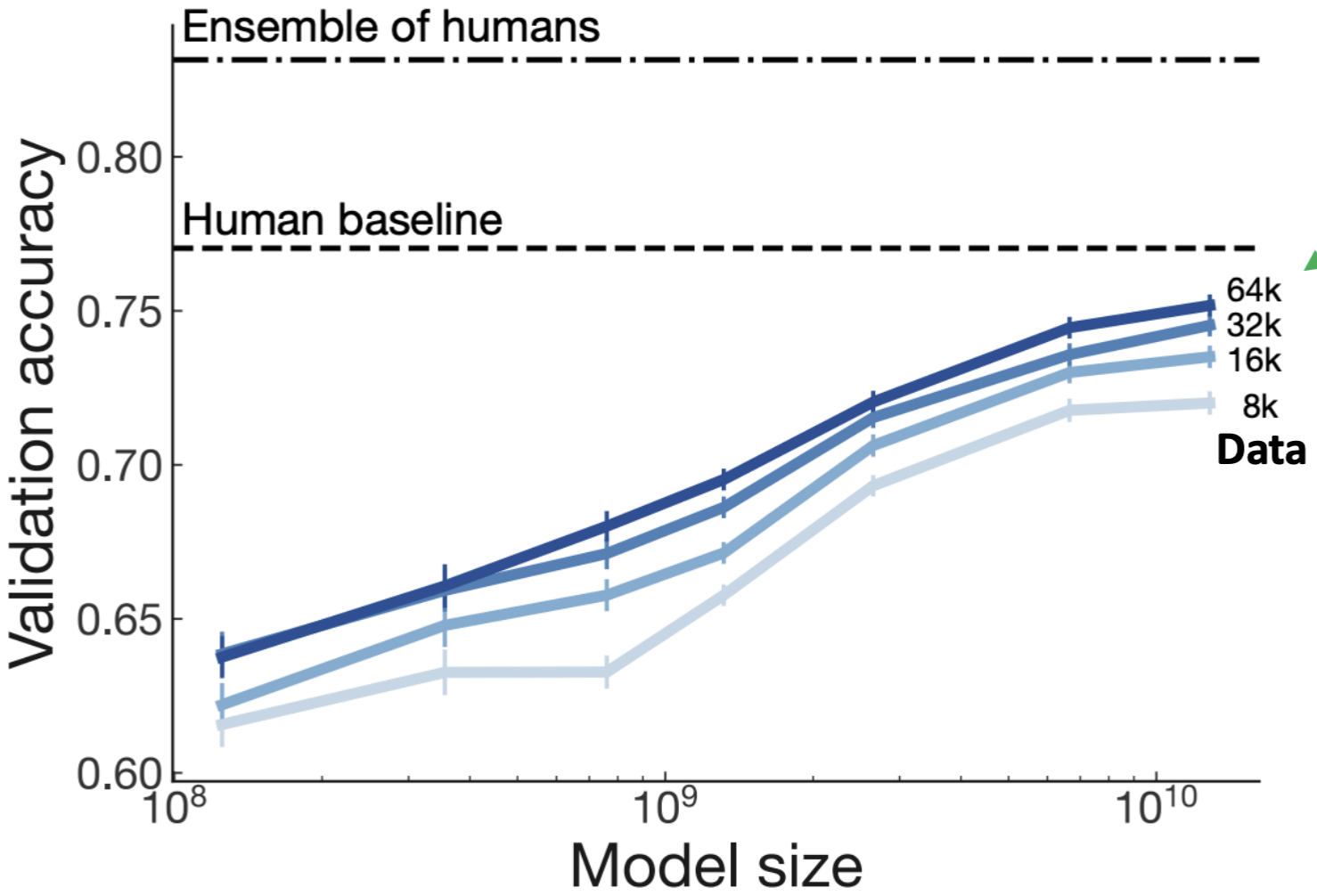
"winning" sample — y^w

"losing" sample — y^l

y^w should score higher than y^l

RLHF – Reward Model

Evaluate RM on predicting outcome of held-out human judgments



Large enough RM trained on enough data approaching single human perf

[Stiennon et al., 2020]

RLHF – Reward Model

- We have the following:
  - A pretrained (possibly instruction-finetuned) LM $p^{PT}(y \mid x)$
  - A reward model $RM_\phi(x, y)$ that produces scalar rewards for LM outputs, trained on a dataset of human comparisons
- Now to do RLHF:
  - Copy the model $p_\theta^{RL}(y \mid x)$, with parameters θ we would like to optimize
  - We want to optimize:

$$\mathbb{E}_{\hat{y} \sim p_\theta^{RL}(\hat{y}|x)} [RM_\phi(x, \hat{y})]$$

RLHF – Reward Model

- We want to optimize:

$$\mathbb{E}_{\hat{y}\sim p_\theta^{RL}(\hat{y}|x)}\left[RM_\phi(x,\hat{y})\right]$$

- Do you see any problems?
  - Learned rewards are imperfect; this quantity can be imperfectly optimized
- Add a penalty for drifting too for from the initialization:

$$\mathbb{E}_{\hat{y}\sim p_\theta^{RL}(\hat{y}|x)}\left[RM_\phi(x,\hat{y}) - \beta \underbrace{\log\left(\frac{p_\theta^{RL}(\hat{y}\mid x)}{p^{PT}(\hat{y}\mid x)}\right)}\right]$$

Pay a price when $p_\theta^{RL}(\hat{y}\mid x) > p^{PT}(\hat{y}\mid x)$

This penalty which prevents us from diverging too far from the pretrained model. In expectation, it is known as the **Kullback-Leibler (KL)** divergence between $p_\theta^{RL}(\hat{y}\mid x)$ and $p^{PT}(\hat{y}\mid x)$.

RLHF – Proximal Policy Optimization (PPO)

- How do we actually change our LM parameters θ to maximize this?

$$\mathbb{E}_{\hat{s}\sim p_\theta(s)}[R(\hat{s})]$$

- Let's try doing gradient ascent!

$$\theta_{t+1} := \theta_t + \alpha \nabla_{\theta_t} \mathbb{E}_{\hat{s}\sim p_{\theta_t}(s)}[R(\hat{s})]$$

How do we estimate this expectation??

What if our reward function is non-differentiable??

- **Policy gradient** methods in RL (e.g., REINFORCE; [Williams, 1992]) give us tools for estimating and optimizing this objective.

RLHF – Proximal Policy Optimization (PPO)

- We want to obtain

(defn. of expectation) (linearity of gradient)

$$\nabla_\theta \mathbb{E}_{\hat{s}\sim p_\theta(s)}[R(\hat{s})] = \nabla_\theta \sum_s R(s) p_\theta(s) = \sum_s R(s)\, \nabla_\theta p_\theta(s)$$

- Here we'll use a very handy trick known as the **log-derivative trick**. Let's try taking the gradient of $\log p_\theta(s)$

$$\nabla_\theta \log p_\theta(s) = \frac{1}{p_\theta(s)} \nabla_\theta p_\theta(s) \quad \Longrightarrow \quad \nabla_\theta p_\theta(s) = \nabla_\theta \log p_\theta(s)\, p_\theta(s)$$

(chain rule)

- Plug back in:

This is an expectation of this

$$\sum_s R(s)\, \nabla_\theta p_\theta(s) = \sum_s p_\theta(s) R(s)\, \nabla_\theta \log p_\theta(s)$$

$$= \mathbb{E}_{\hat{s}\sim p_\theta(s)}[R(\hat{s})\, \nabla_\theta \log p_\theta(\hat{s})]$$

RLHF – Proximal Policy Optimization (PPO)

- Now we have put the gradient "inside" the expectation, we can approximate this objective with Monte Carlo samples:

$$\nabla_\theta \mathbb{E}_{\hat{s}\sim p_\theta(s)}[R(\hat{s})] = \mathbb{E}_{\hat{s}\sim p_\theta(s)}[R(\hat{s})\,\nabla_\theta \log p_\theta(\hat{s})] \approx \frac{1}{m}\sum_{i=1}^{m} R(s_i)\,\nabla_\theta \log p_\theta(s_i)$$

This is why it's called **"reinforcement learning"**: we **reinforce** good actions, increasing the chance they happen again.

- Giving us the update rule:

$$\theta_{t+1} := \theta_t + \alpha \frac{1}{m}\sum_{i=1}^{m} R(s_i)\,\nabla_{\theta_t} \log p_{\theta_t}(s_i)$$

If R is +++

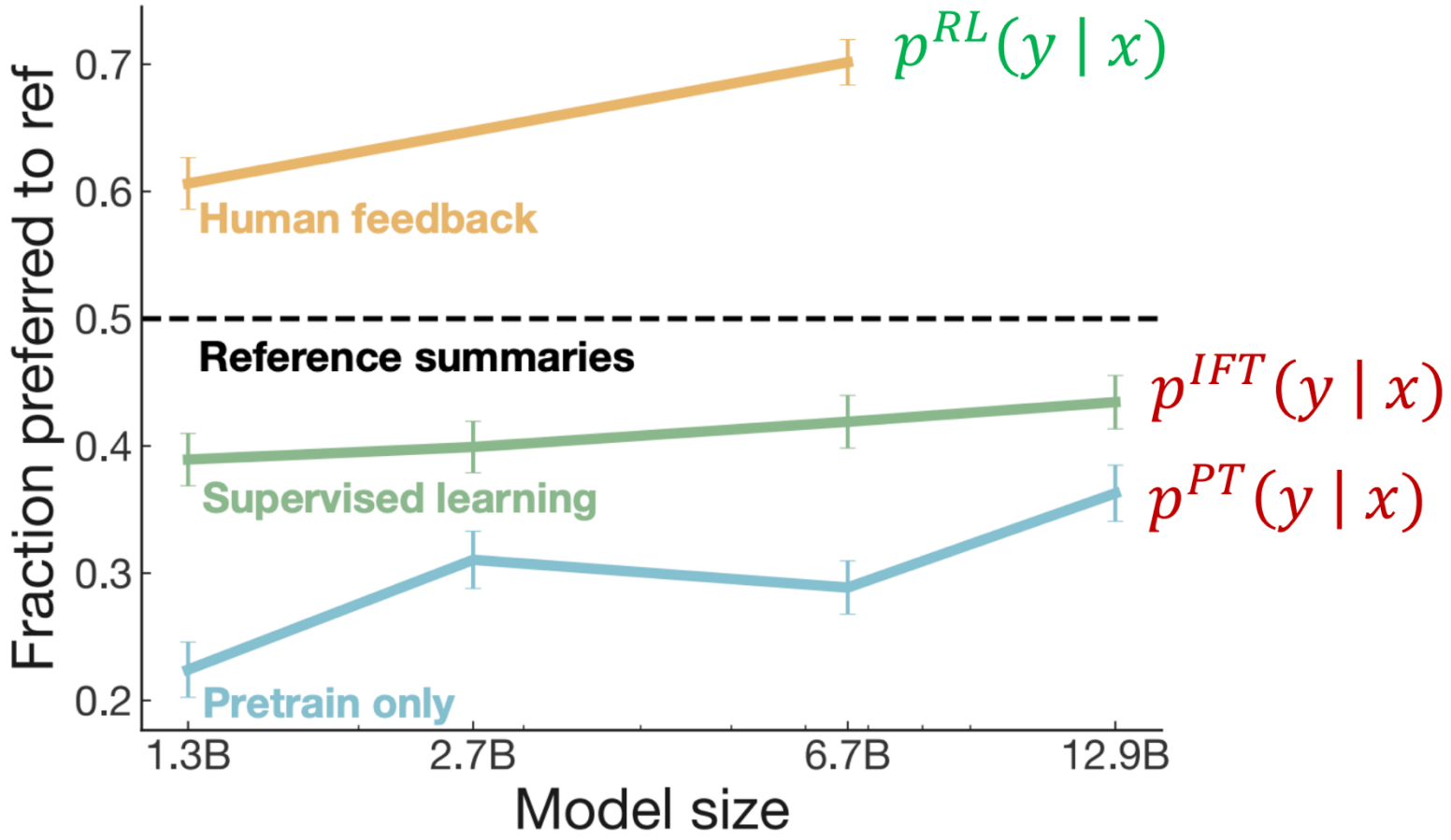
Take gradient steps to maximize $p_\theta(s_i)$

If R is ---

Take steps to minimize $p_\theta(s_i)$

RLHF

RLHF provides gains over pre-training + instruction fine-tuning



RLHF – clear stylistic changes

- Significantly more detailed, nicer/clearer list like formatting

Instruction: What are the five most common causes of stress among people?

Alpaca (Finetuned w/ 10k data): The five most common causes of stress among people are work, money, relationships, health, and family

Alpaca + PPO: The five most common causes of stress among people are:

1. Pressure at work: Whether it is due to tight deadlines, high expectations, or an overwhelming workload, pressure at work can be a major source of stress
2. Money problems: Financial issues can be a significant source of stress, from not being able to make ends meet to struggling with debt.
3. Unhealthy lifestyle: Poor diet, lack of exercise, and sleep deprivation can all contribute to stress.
4. Unhappiness at home: Problems in relationships, conflicts with family members, and other domestic issues can be a major source of stress.
5. Uncertainty: Excessive worry and anxiety about the future, health issues, or other unknowns can be a major cause of stress

InstructGPT: scaling up RLHF to thousands of tasks

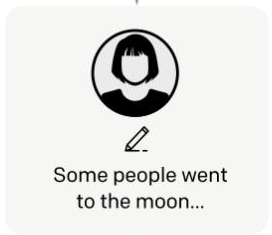
30k tasks!

Step 1

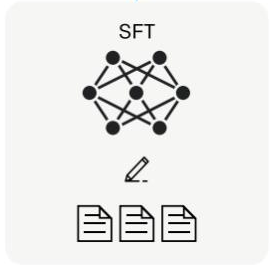
Collect demonstration data, and train a supervised policy.

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.

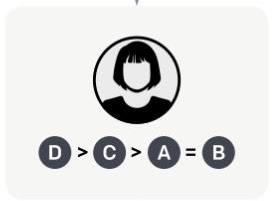


Step 2

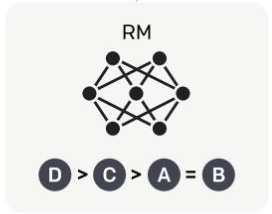
Collect comparison data, and train a reward model.

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

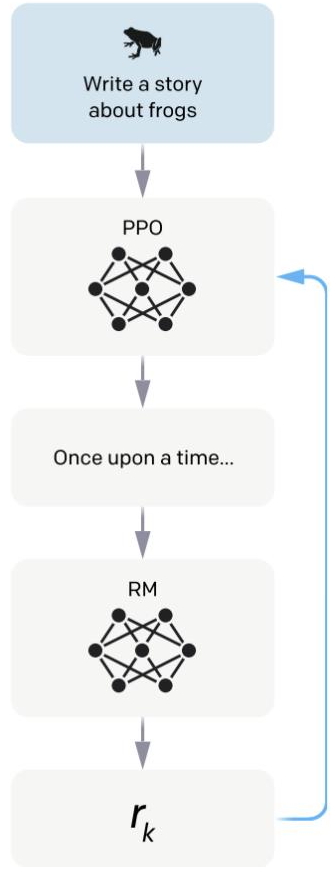
Optimize a policy against the reward model using reinforcement learning.

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.



InstructGPT

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.
```

InstructGPT

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

COMPLETION

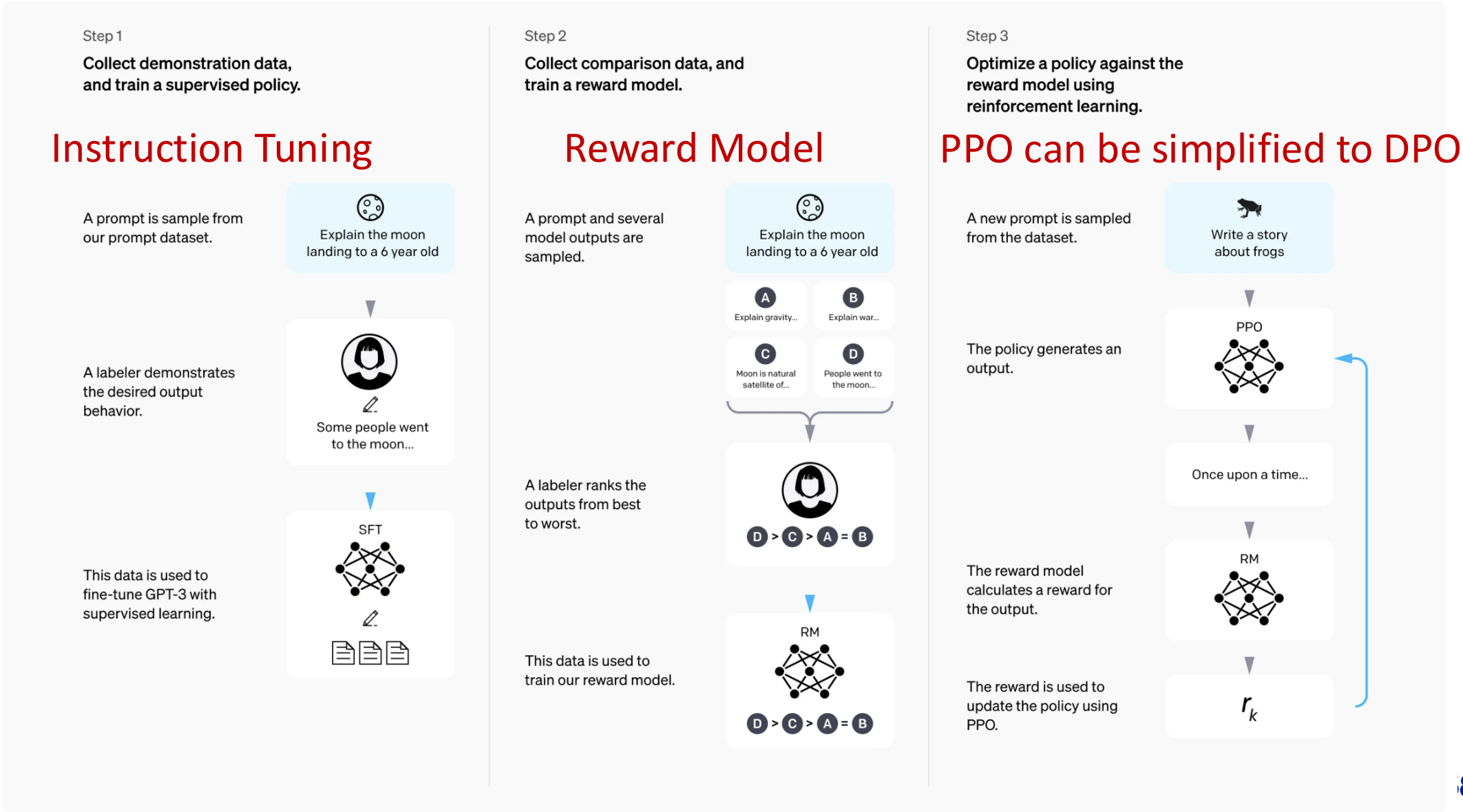
GPT-3

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

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

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

InstructGPT

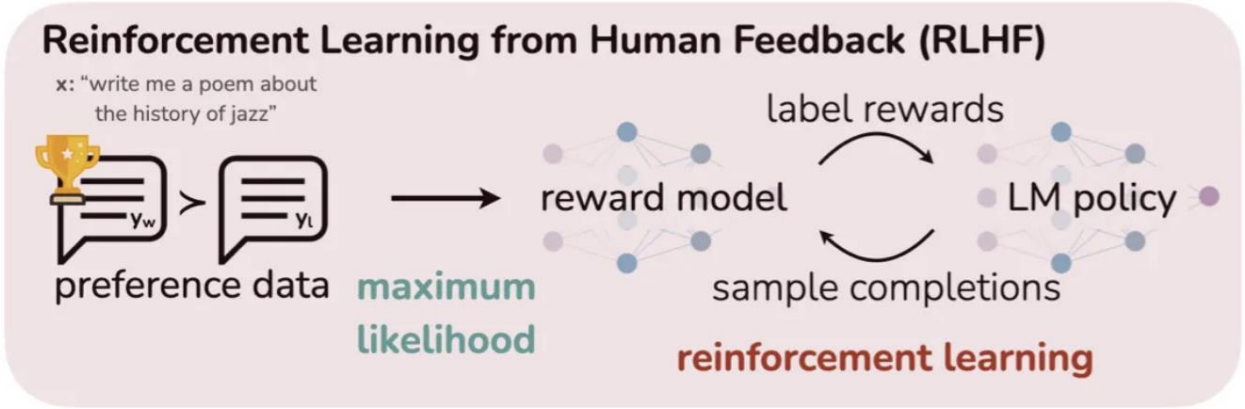
```
The frog is a symbol of wisdom
He knows all the secrets of the world
He is a master of disguise
And a great teacher of life
He is a symbol of transformation
And the bringer of change
He is the frog who has seen it all
And knows the meaning of it all
```

RLHF (Reinforcement Learning from Human Feedback)



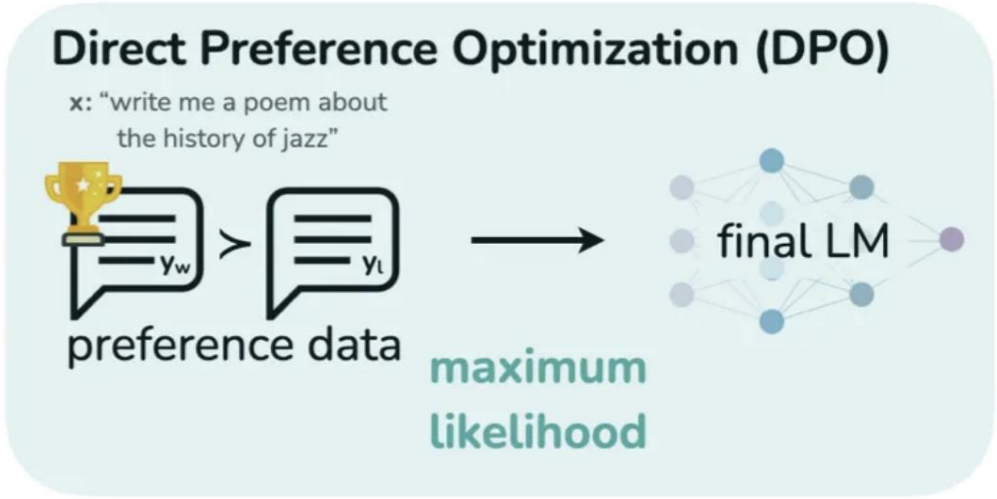
RLHF – Direct Preference Optimization (DPO)

Preference data: (prompt, winning response, losing response) $(x, y_w, y_l) \sim D$



Drawbacks:

- Involve multiple models SFT, RM, policy models
- Involve multiple stages of training
- Complex, hard to get it right!



1. Optimize **reward model** over **preference data**

2. Optimize **policy model** according to the **reward model**

Why not directly learn the **policy model** from **preference data**?

RLHF – Direct Preference Optimization (DPO)

- Recall, we want to maximize the following objective:

$$\mathbb{E}_{\hat{y}\sim p_\theta^{RL}(\hat{y}|x)}\left[RM(x,\hat{y}) - \beta \log\left(\frac{p_\theta^{RL}(\hat{y}\mid x)}{p^{PT}(\hat{y}\mid x)}\right)\right]$$

- There is a closed form solution to this:

$$p^*(\hat{y}\mid x) = \frac{1}{Z(x)} p^{PT}(\hat{y}\mid x)\exp\left(\frac{1}{\beta}RM(x,\hat{y})\right)$$

- Rearrange the terms:

$$RM(x,\hat{y}) = \beta \log\frac{p^*(\hat{y}\mid x)}{p^{PT}(\hat{y}\mid x)} + \beta \log Z(x)$$

- This holds true for arbitrary LMs

Derive $RM_\theta(x,y)$ in terms of $p_\theta^{RL}(\hat{y}\mid x)$

$$RM_\theta(x,\hat{y}) = \beta \log\frac{p_\theta^{RL}(\hat{y}\mid x)}{p^{PT}(\hat{y}\mid x)} + \beta \log Z(x)$$

RLHF – Direct Preference Optimization (DPO)

- Recall, how we fit the reward model $RM_\phi(x, y)$:

$$J_{RM}(\phi) = -\mathbb{E}_{(x, y^w, y^l)\sim D}\left[\log \sigma(RM_\phi(x, y^w) - RM_\phi(x, y^l))\right]$$

- Notice that we only need the **difference** between the rewards for y^w and y^l. Simplify for $RM_\theta(x, y)$:

$$RM_\theta(x, y^w) - RM_\theta(x, y^l) = \beta \log \frac{p_\theta^{RL}(y^w \mid x)}{p^{PT}(y^w \mid x)} - \beta \log \frac{p_\theta^{RL}(y^l \mid x)}{p^{PT}(y^l \mid x)}$$

- The final DPO loss function is:

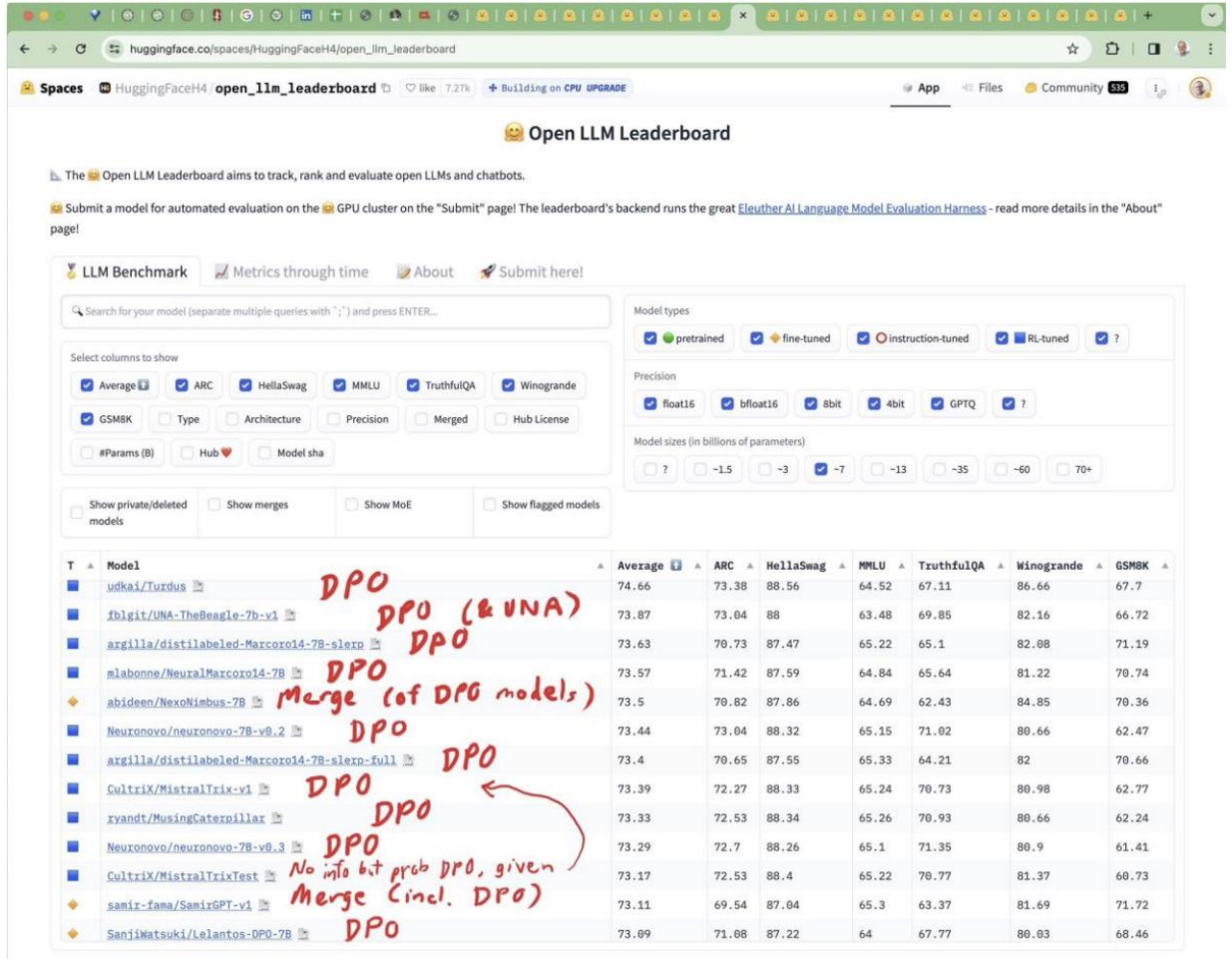
$$J_{\text{DPO}}(\theta) = -\mathbb{E}_{(x, y^w, y^l)\sim D}\left[\log \sigma\left(RM_\theta(x, y^w) - RM_\theta(x, y^l)\right)\right]$$

We have a *simple classification loss* function that connects **preference data** to **language model parameters** directly!

Summary of PPO and DPO

- We want to optimize for human preferences
  - Instead of humans writing the answers or giving uncalibrated scores, we get humans to rank different LM generated answers
- Reinforcement learning from human feedback
  - Train an explicit reward model on comparison data to predict a score for a given completion
  - Optimize the LM to maximize the predicted score (under KL-constraint)
  - Very effective when tuned well, computationally expensive and tricky to get right
- Direct Preference Optimization
  - Optimize LM parameters directly on preference data by solving a binary classification problem
  - Simple and effective, similar properties to RLHF, does not leverage online data

Open source LLMs use DPO more



Open source LLMs now almost all just use DPO (and it works well!)

MISTRAL AI_ 8X7

	GPT - 3.5	Mistral Small	Mistral Medium
MT Bench (for Instruct models)	8.32	8.30	**8.61**

https://mistral.ai/news/mixtral-of-experts/

Instruction fine-tuning

pretrained models in chat use cases, we innovated on our well. Our approach to post-training is a combination of tion sampling, proximal policy optimization (PPO), and O). The quality of the prompts that are used in SFT and sed in PPO and DPO has an outsized influence on the me of our biggest improvements in model quality came nd performing multiple rounds of quality assurance on notators.

Learning from preference rankings via PPO and DPO also greatly improved the performance of Llama 3 on reasoning and coding tasks. We found that if you ask a model a reasoning question that it struggles to answer, the model will sometimes produce the right reasoning trace: The model knows how to produce the right answer, but it does not know how to select it. Training on preference rankings enables the model to learn how to select it.

Limitations of RL + Reward Modeling

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

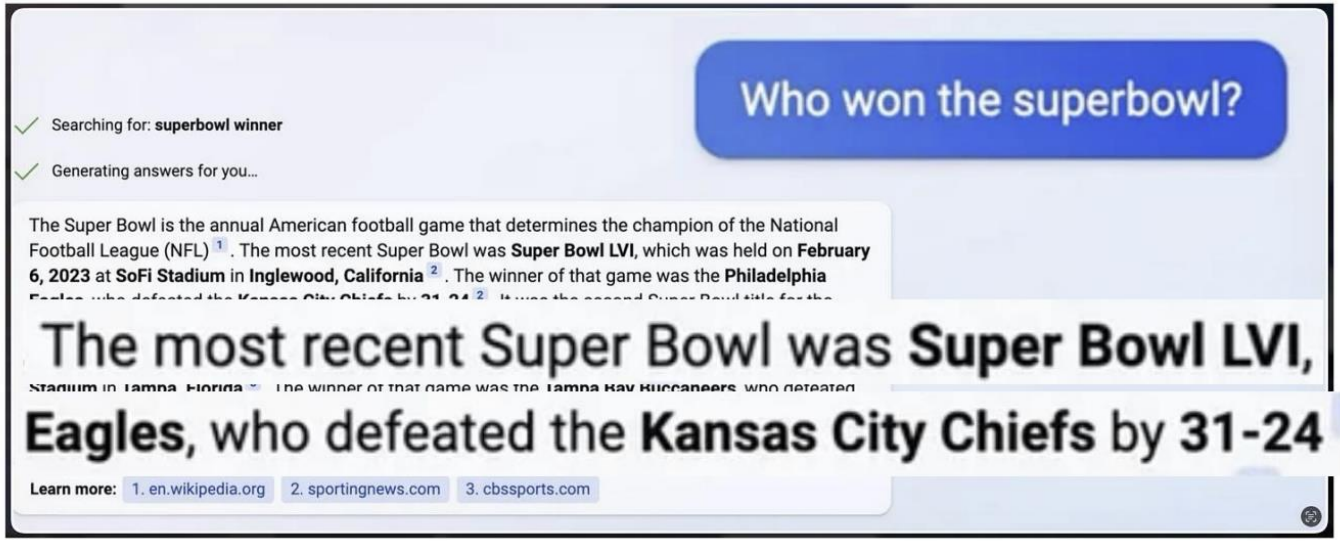
TECHNOLOGY

Google shares drop $100 billion after its new AI chatbot makes a mistake

February 9, 2023 · 10:15 AM ET

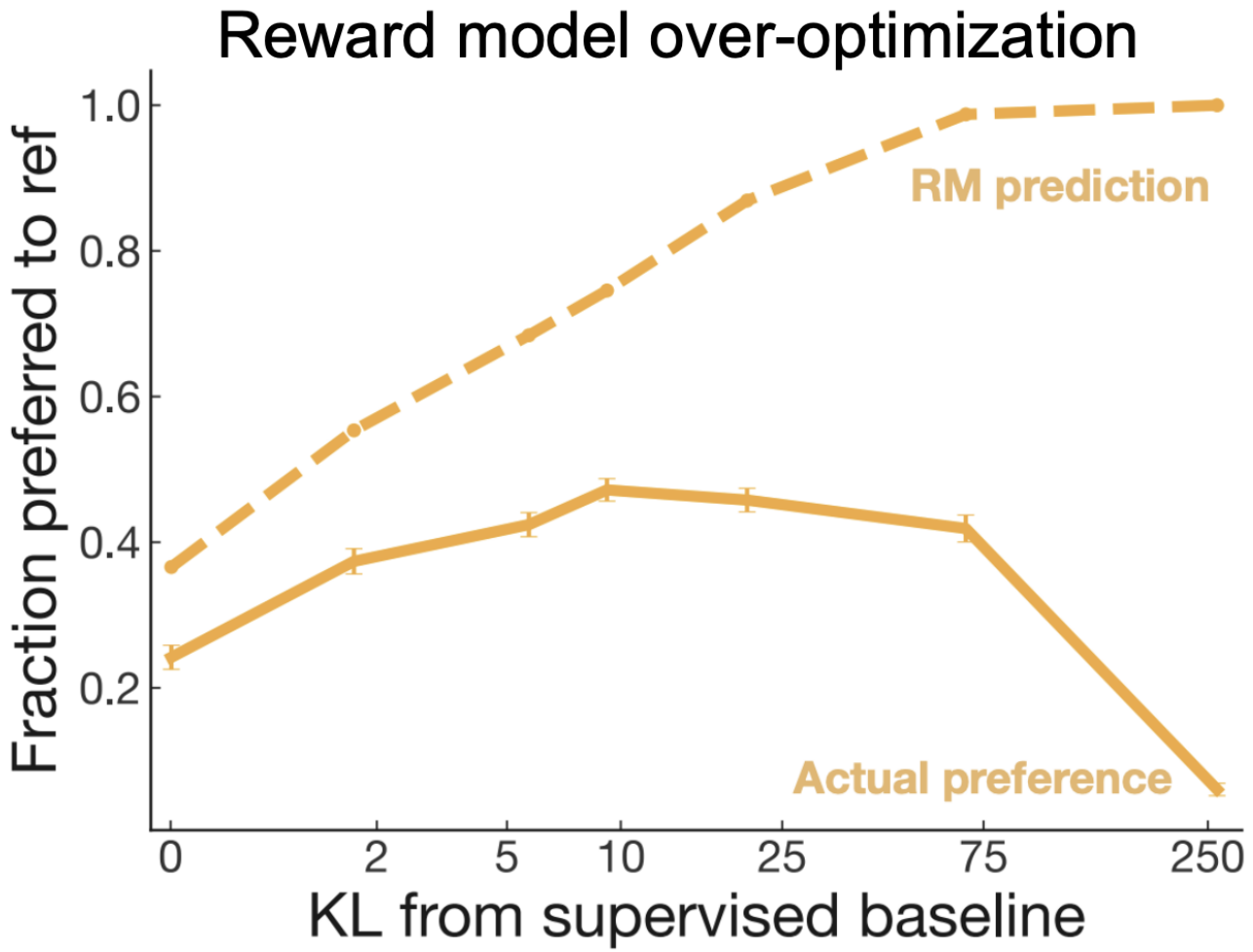
https://www.npr.org/2023/02/09/1155650909/google-chatbot--error-bard-shares

Bing AI hallucinates the Super Bowl



Limitations of RL + Reward Modeling

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



$$R(s) = RM_\phi(s) - \beta \log \left(\frac{p_\theta^{RL}(s)}{p^{PT}(s)} \right)$$

Recap

Optimization for Human Preference (RLHF PPO / DPO)

- Pro:

Directly model preferences, generalize beyond labeled data

- Cons:

RL is very tricky to get right

Human preferences are unreliable

models of human preferences are even more unreliable


Thank you!
UF
Herbert Wertheim
College of Engineering
UNIVERSITY of FLORIDA
LEADING THE CHARGE, CHARGING AHEAD