

Natural Language Processing

Instruction Finetuning, and RLHF

Instructor: Yangqiu Song

Outline

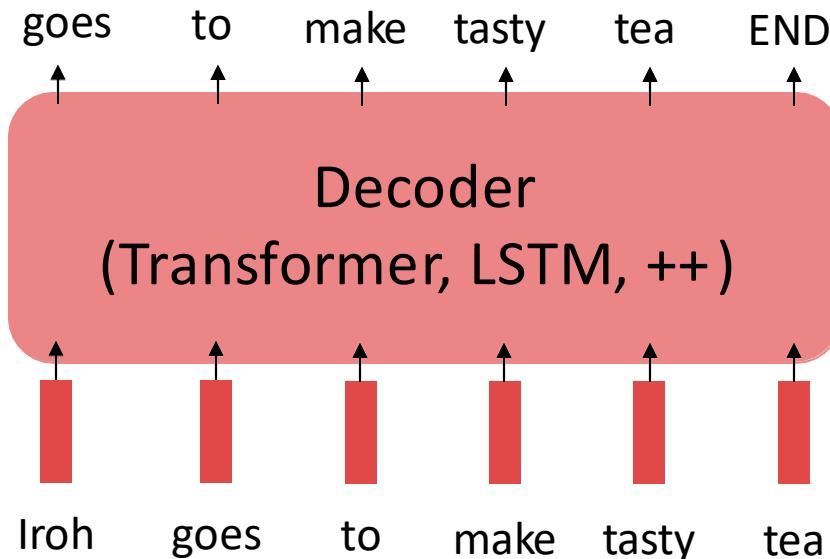
- **Instruction finetuning**
- **Reinforcement Learning from Human Feedback (RLHF)**

Recall: The Pretraining / Finetuning Paradigm

- Pretraining can improve NLP applications by serving as parameter initialization.

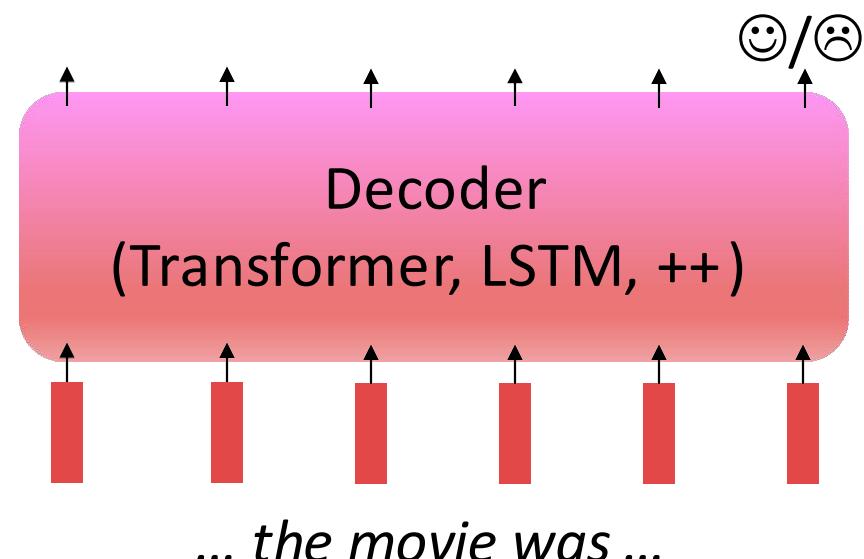
Step 1: Pretrain (on language modeling)

Lots of text; learn general things!



Step 2: Finetune (on your task)

Not many labels; adapt to the task!

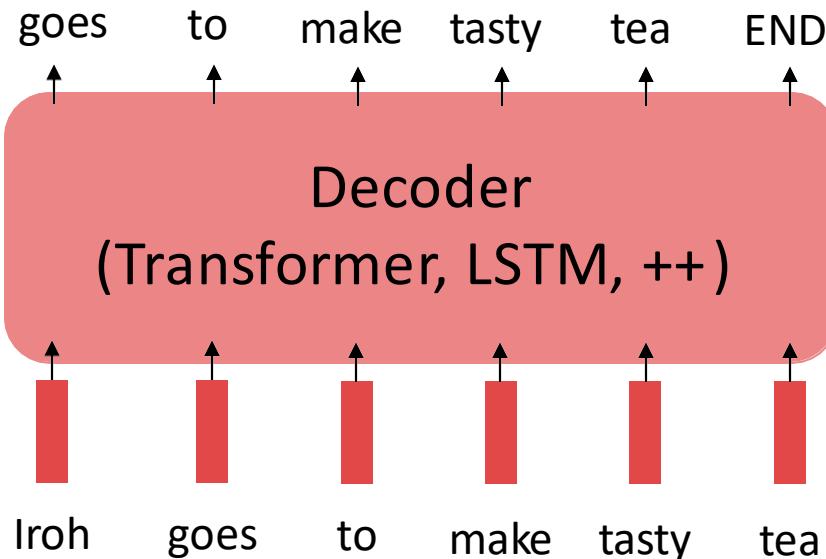


Scaling up finetuning

- Pretraining can improve NLP applications by serving as parameter initialization.

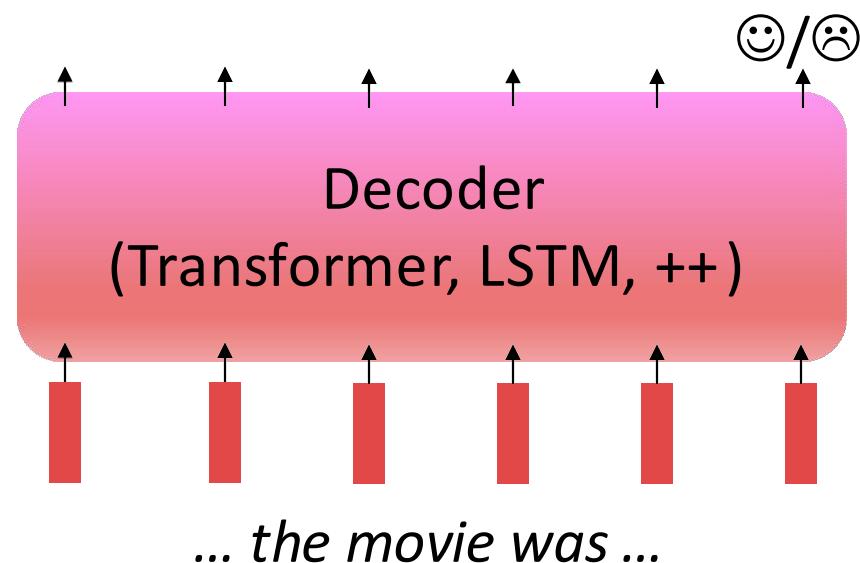
Step 1: Pretrain (on language modeling)

Lots of text; learn general things!



Step 2: Finetune (on many tasks)

Not many labels; adapt to the tasks!



Instruction Tuning

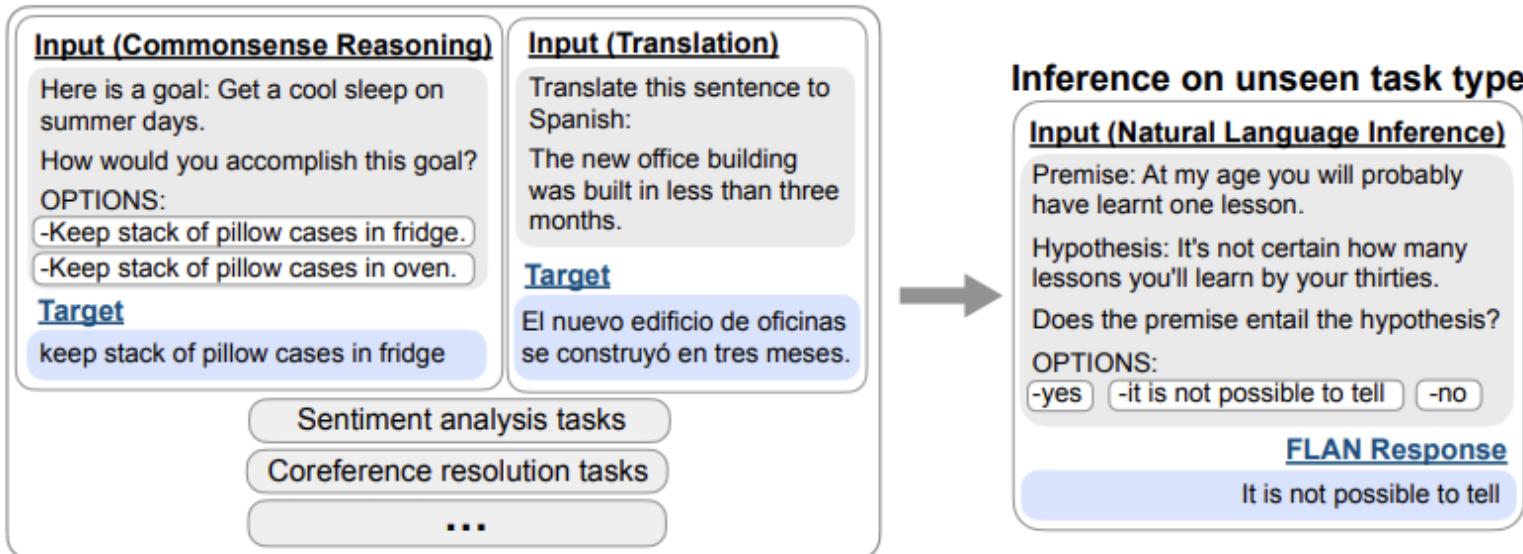
- Finetune the language model on many tasks (simultaneously) described via instructions
- The model will learn to follow instructions
 - Generalizes to unseen tasks in zero-shot
- Works well since NLP tasks are describable with natural language instructions
 - e.g., Summarize the following paragraph
 - e.g., Translate "How are you" into Chinese

Improve Performance of LMs Zero-shot

- Finetuned Language Models Are Zero-shot Learners (FLAN), Jason Wei et. al, ICLR 2022, from Google Research
- Multitask Prompted Training Enables Zero-shot Task Generalization (T0), Victor Sanh et. al, ICLR 2022, from Hugging Face
- Scaling Instruction-Finetuned Language Models (Flan-T5). Chung et al., 2022 from Google Research
 - Both are using T5 as base model. T0 is a little bit earlier and concurrent to FLAN

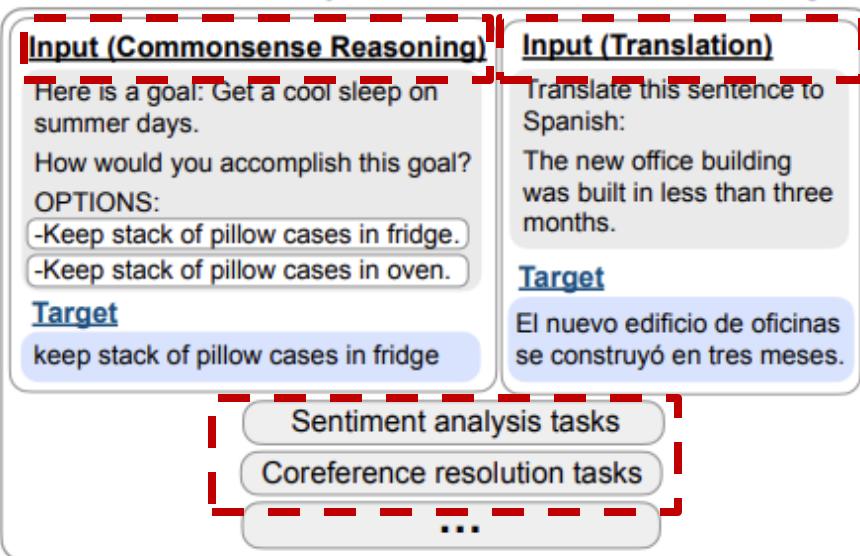
Instruction Tuning Example

Finetune on many tasks (“instruction-tuning”)



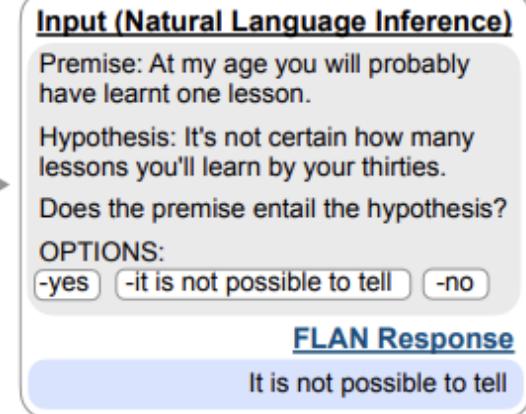
Instruction Tuning Example

Finetune on many tasks (“instruction-tuning”)



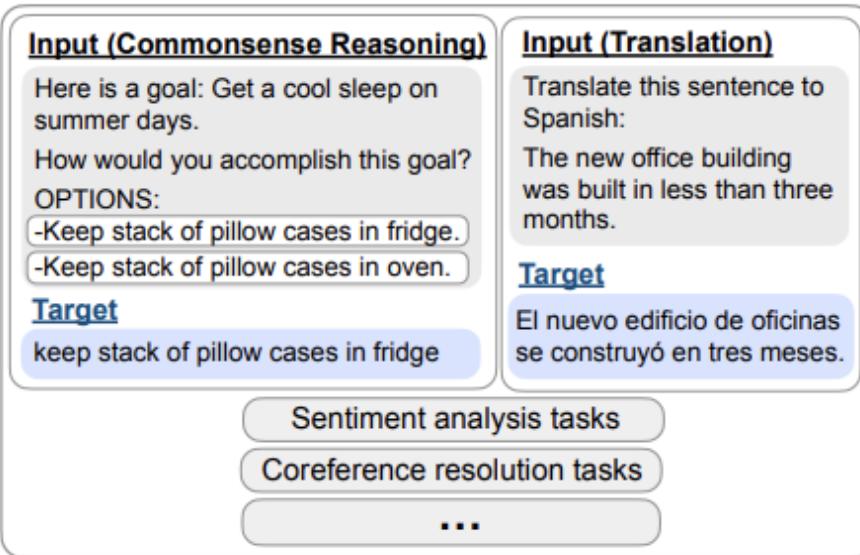
Training on multiple tasks

Inference on unseen task type



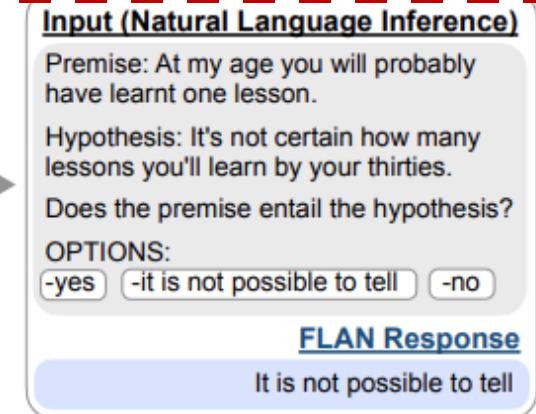
Instruction Tuning Example

Finetune on many tasks (“instruction-tuning”)



Zero-shot inference

Inference on unseen task type



Instruction Tuning Example

Finetune on many tasks (“instruction-tuning”)

Input (Commonsense Reasoning)

Here is a goal: Get a cool sleep on summer days.

How would you accomplish this goal?

OPTIONS:

- Keep stack of pillow cases in fridge.
- Keep stack of pillow cases in oven.

Target

keep stack of pillow cases in fridge

Sentiment analysis tasks

Coreference resolution tasks

...

Input (Translation)

Translate this sentence to Spanish:

The new office building was built in less than three months.

Target

El nuevo edificio de oficinas se construyó en tres meses.

Task instructions in natural language

Inference on unseen task type

Input (Natural Language Inference)

Premise: At my age you will probably have learnt one lesson.

Hypothesis: It's not certain how many lessons you'll learn by your thirties.

Does the premise entail the hypothesis?

OPTIONS:

- yes
- it is not possible to tell
- no

FLAN Response

It is not possible to tell

Finetuned Language Net (FLAN)

- 62 NLP datasets grouped into 12 clusters representing 12 task types
 - Cluster granularity for unseen tasks
- Held out each cluster for zero-shot evaluation, finetuned on the remaining 11 clusters
- Composed ten unique templates for each datasets
 - Up to three of them in reversed format (e.g., write a movie review for this sentiment)

Classification vs Generation Tasks

- The base model (LaMDA-PT) is **decoder-only**
- Works as is for free-text generation tasks (e.g., summarization)
- Needs options for classification tasks (e.g., closed QA)

Input (Natural Language Inference)

Premise: At my age you will probably have learnt one lesson.

Hypothesis: It's not certain how many lessons you'll learn by your thirties.

Does the premise entail the hypothesis?

OPTIONS:

-yes -it is not possible to tell -no

FLAN Response

It is not possible to tell

FLAN Task Clustering

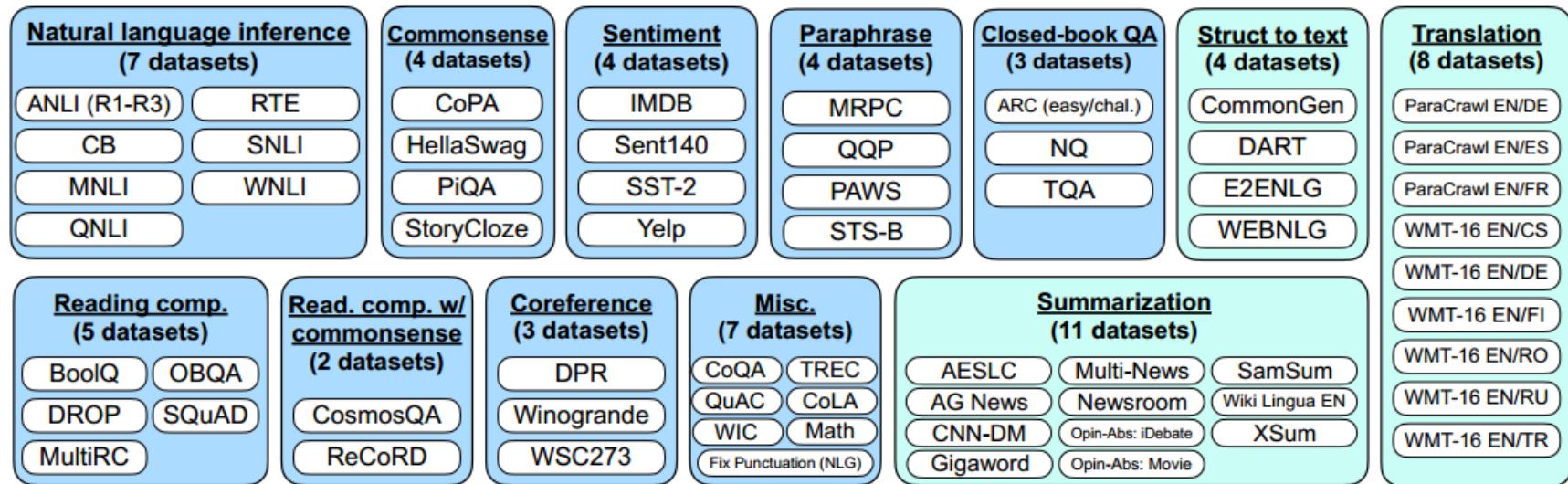


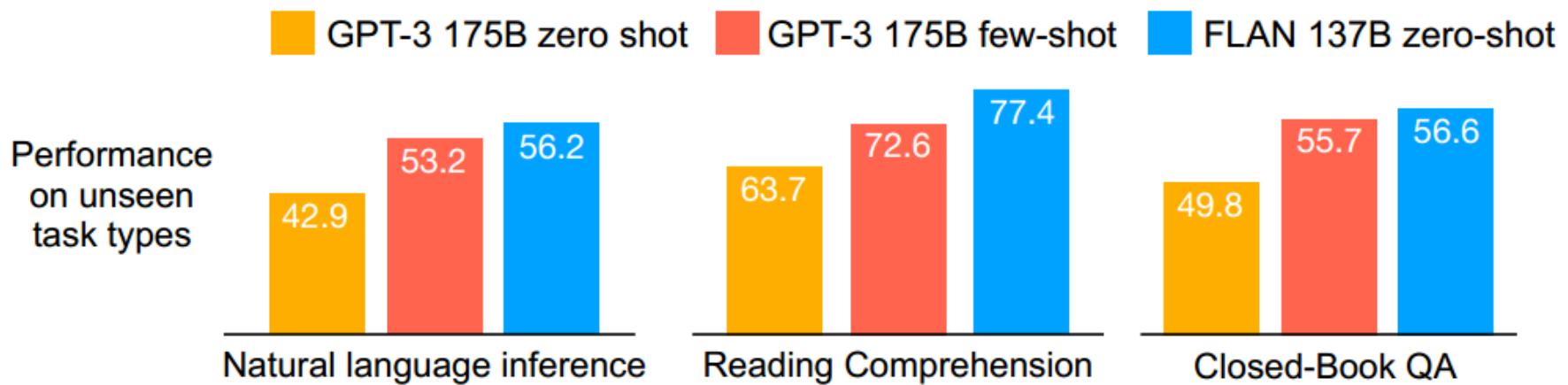
Figure 3: Datasets and task clusters used in this paper (NLU tasks in blue; NLG tasks in teal).

Training Details: Base Model

- Language Models for Dialog Applications (LaMDA-PT)
- Decoder only, trained for next token generation
- 137B parameters
- Pretrained on web documents, dialog data, and Wikipedia
 - 1.56T words
 - 10% of training data is non-English

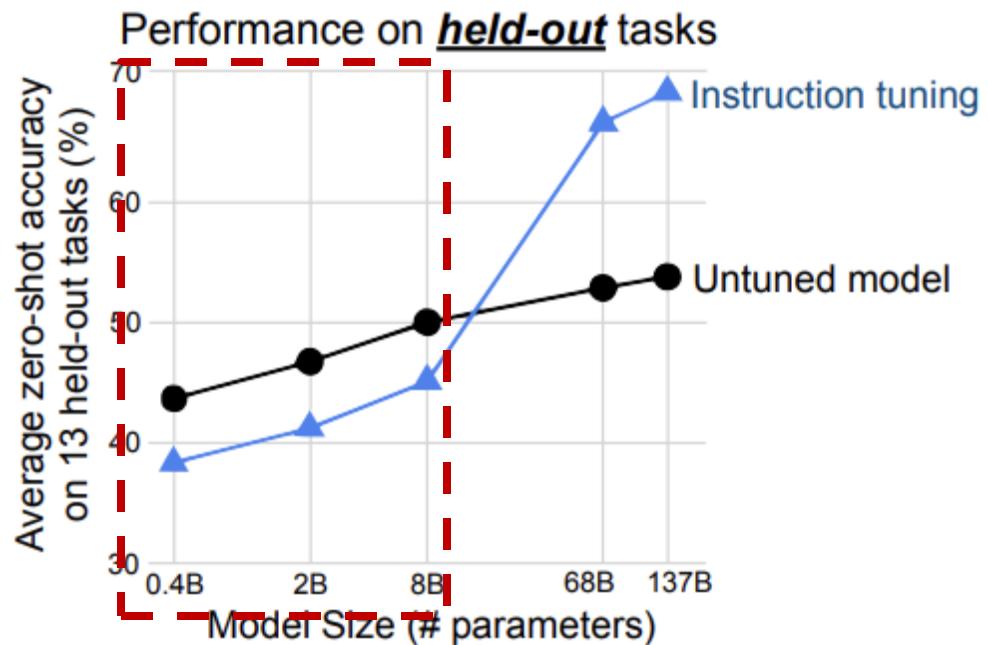
FLAN vs GPT-3

- FLAN outperforms GPT-3 zero-shot and few-shot on some tasks
- Task types with most gain from instruction tuning are selected



Ablation Studies: Scaling Laws

Accuracy of smaller
models degrades with
Instruction tuning



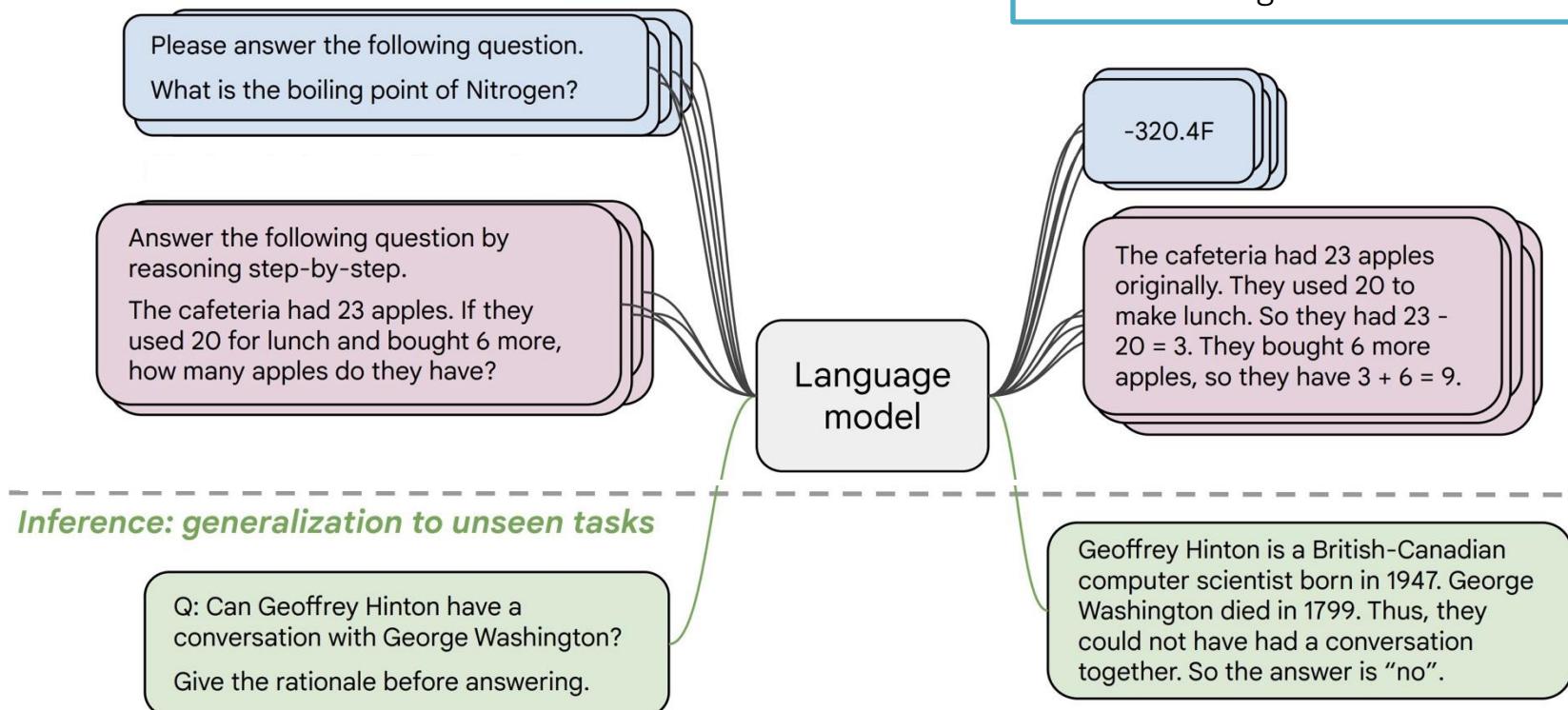
Improve Performance of LMs Zero-shot

- Finetuned Language Models Are Zero-shot Learners (FLAN), Jason Wei et. al, ICLR 2022, from Google Research
- Multitask Prompted Training Enables Zero-shot Task Generalization (T0), Victor Sanh et. al, ICLR 2022, from Hugging Face
- Scaling Instruction-Finetuned Language Models (Flan-T5). Chung et al., 2022 from Google Research
 - Both are using T5 as base model. T0 is a little bit earlier and concurrent to FLAN

Instruction finetuning

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

(1) scaling the number of tasks (1.8K tasks), (2) scaling the model size (540B parameters), and (3) finetuning on chain-of-thought data.



- Evaluate on **unseen tasks**

Dataset Preparation

- Scaling Small Models to 1.8k+ Tasks

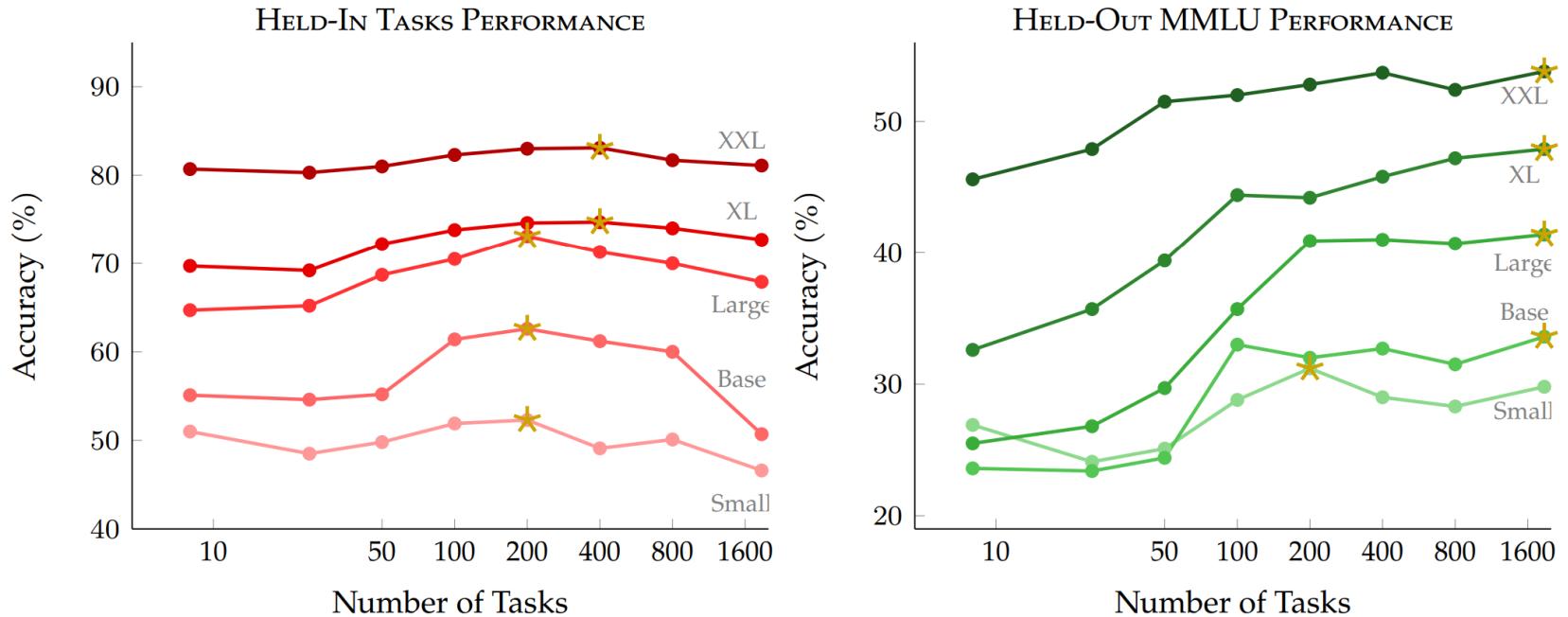


Figure 4: **Performance Scaling Laws for the number of finetuning tasks and model sizes.** Held-In performance (left) and Held-Out MMLU performance (right) are shown. The gold star indicates the peak performance for that model size.

Training Details: T0

- LM-adapted variation of T5 ([LM+T5](#))
- Trained with 100B additional tokens from C4 corpus using the Language Model (LM) objective
- Encoder + Decoder architecture
- 11B Parameters

T0 vs FLAN Differences

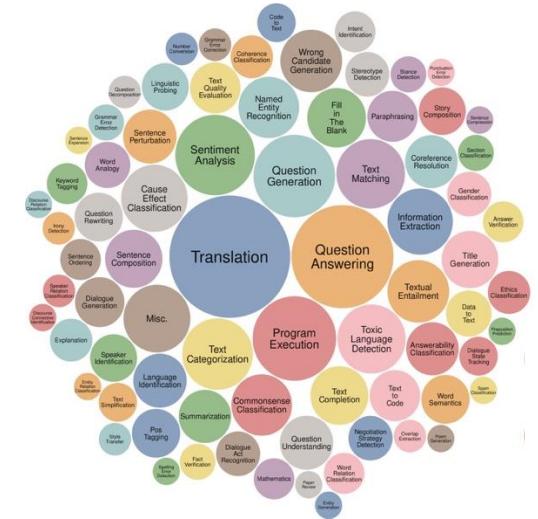
	Base Model	Language	Prompt Template	Published Dataset for instruction tuning
FLAN	LaMDA-PT 137B	10% non-English	Author generated	<u>FLAN v2</u>
T0	T5 11B	100% English	Crowd sourced,	<u>P3 dataset</u>

T0 vs. FLAN Results

- Both papers reported results for NLI category
 - ANLI R1-3 and RTE don't have any clear winners
 - T0 significantly better than FLAN on CB (~80% vs. ~65%)
- Both train on ~250B tokens, maybe not enough for LaMDA-PT
- Mixing the two datasets for instruction tuning is the best
 - **Scaling Instruction-Finetuned Language Models (Flan-T5)**. Chung et al., 2022 from Google Research

Instruction finetuning pretraining?

- As is usually the case, **data + model scale** is key for this to work!
- For example, the **Super-NaturalInstructions** dataset contains **over 1.6K tasks, 3M+ examples**
 - Classification, sequence tagging,
 - rewriting, translation, QA...
- Q:** how do we evaluate such a model?

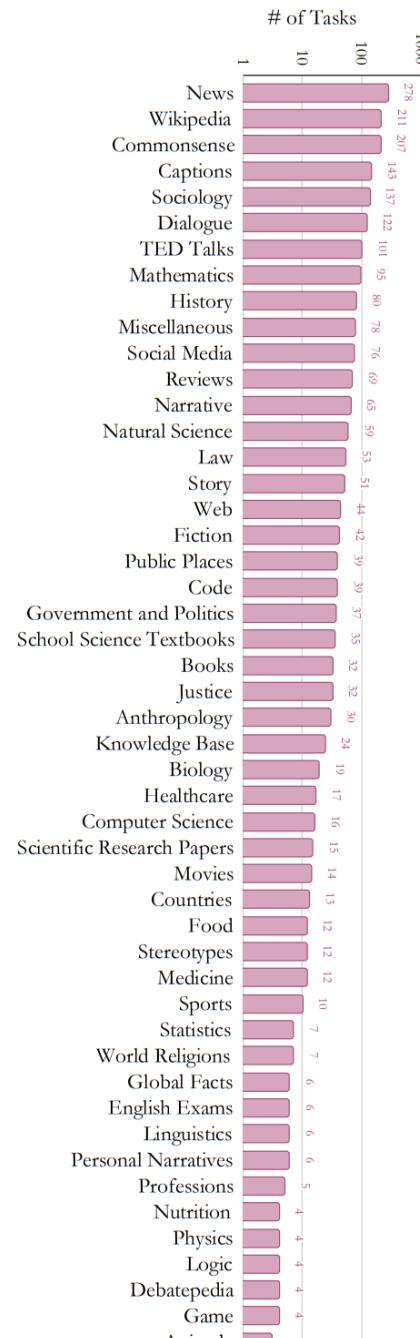


Resource →	SUP-NATINST (this work)	NATINST (Mishra et al., 2022b)	CROSSFIT (Ye et al., 2021)	PROMPTSOURCE (Bach et al., 2022)	FLAN (Wei et al., 2022)	INSTRUCTGPT (Ouyang et al., 2022)
Has task instructions?	✓	✓	✗	✓	✓	✓
Has negative examples?	✓	✓	✗	✗	✗	✗
Has non-English tasks?	✓	✗	✗	✗	✓	✓
Is public?	✓	✓	✓	✓	✓	✗
Number of tasks	1616	61	269	176	62	—
Number of instructions	1616	61	—	2052	620	14378
Number of annotated tasks types	76	6	13	13*	12	10
Avg. task definition length (words)	56.6	134.4	—	24.8	8.2	—

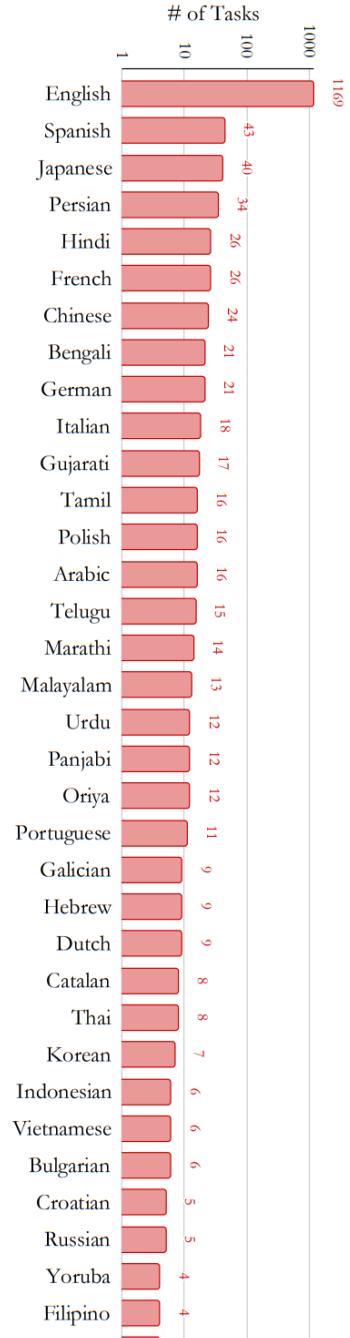
Super-NaturalInstructions: Generalization via Declarative Instructions on 1600+ NLP Tasks

[Wang et al., 2022] AI2 and many other institutions

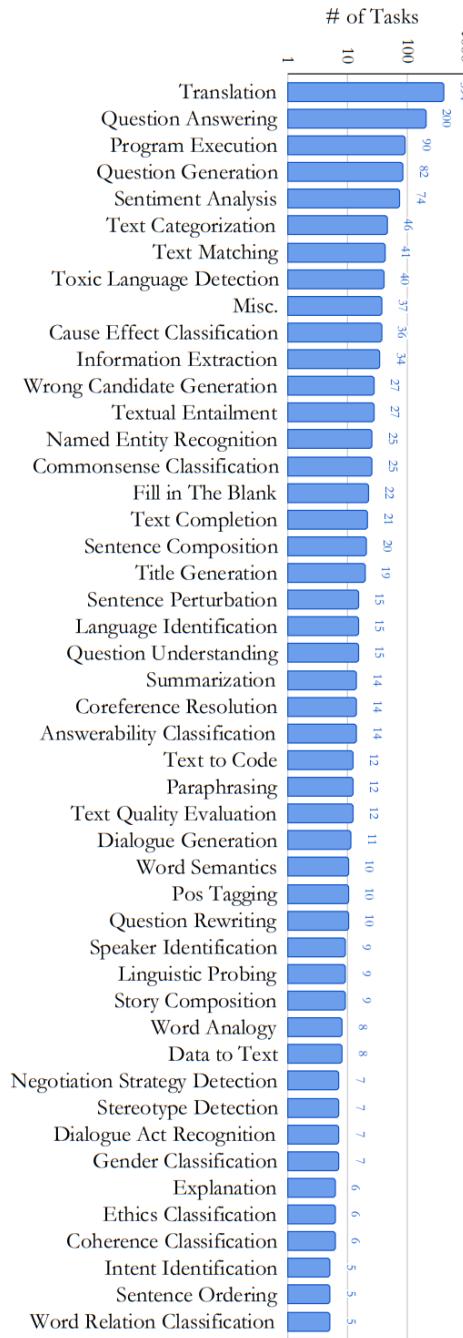
(c) Domains



(b) Languages



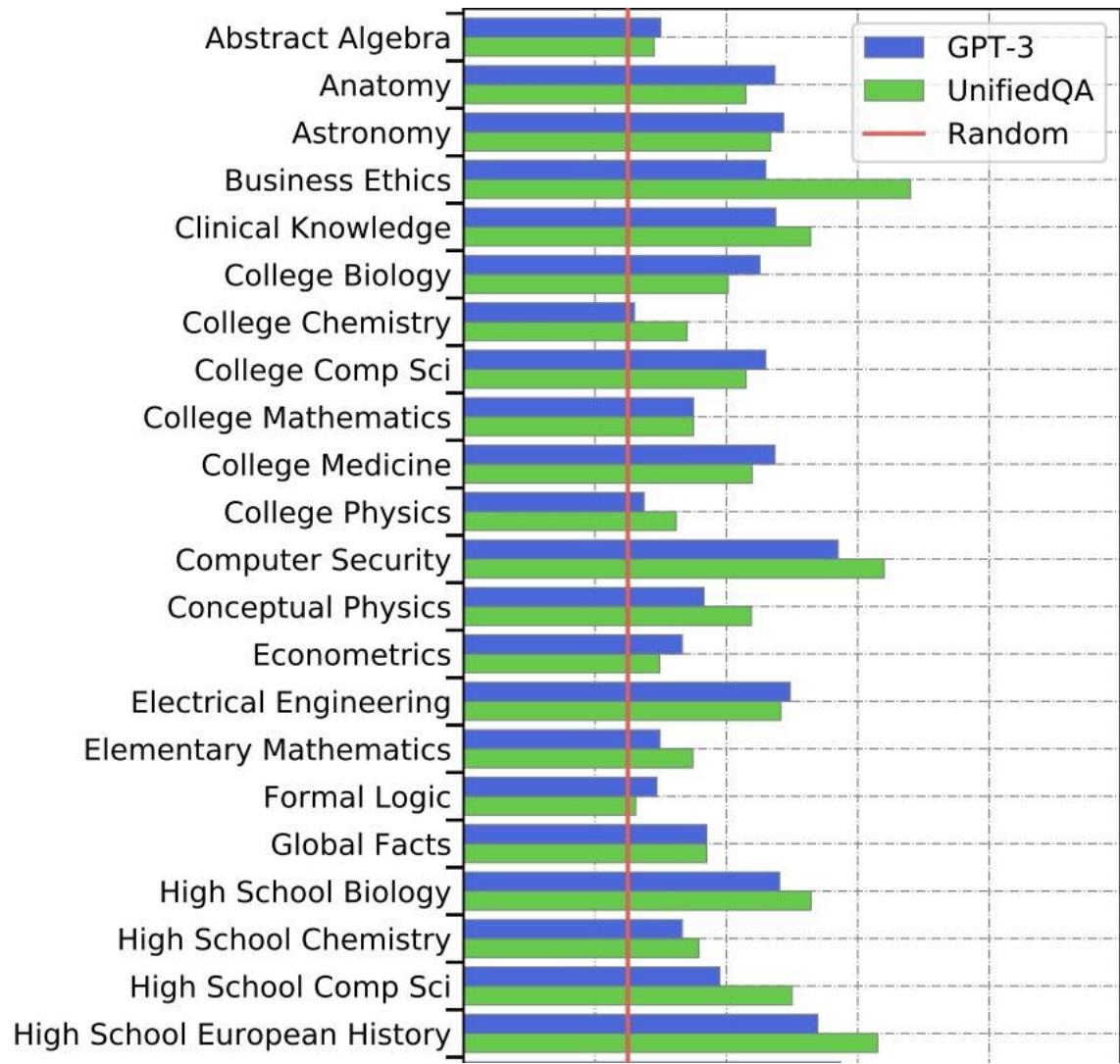
(a) Task Types



Aside: new benchmarks for multitask LMs

- **Massive Multitask Language Understanding (MMLU)**
[\[Hendrycks et al., 2021\]](#)

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



Aside: new benchmarks for multitask LMs

- **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:

Aaro 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 Tazar, Alice Xiang, Alicia Parish, Allen Nie, Aman Hussain, Amanda Ascoli, Amanda Dsouza, Ambrose Sloane, Ameet Rahane, Anantharaman S. Iyer, Anders Andreassen, Andrea Madotto, Andrea Stuhlmüller, Andrew Dai, Andrew La, Andrew Lampinen, Andy Zou, Angela Jiang, Angelica Chen, Anh Vuong, Animesh Gupta, Anna Gottardi, Antonio Norelli, Anu Venkatesh, Arash Ghahamianidavoodi, Arfa Tabassum, Arul Menezes, Arun Kirurrajaian, Arul Mulkandov, Ashish Sabharwal, Austin Herrick, Avia Efrat, Ayuk Erdmen, Ayla Karakaş, B. Ryan Roberts, Bao Sheng Lou, Barret Zoph, Bartłomiej Bojanowski, Batuhan Özütür, Behnam Heydayatnia, Behnarn Neyshabur, Benjamin Inden, Benno Stein, Berk Ekmekci, Bill Yuchen Lin, Blake Howard, Cameron Diau, Cameron Dour, Catherine Stinson, Cedrick Argueta, César Ferri Ramírez, Chandan Singh, Charles Rathkopf, Chenlin Meng, Chitta Baral, Chiyou Wu, Chris Callison-Burch, Chris Waites, Christian Voigt, Christopher D. Manning, Christopher Potts, Cindy Ramirez, Clara E. Rivera, Clemencia Siro, Colin Raffel, Courtney Ascrraft, Cristina Barbacea, Daniela Sileo, Dan Garrette, Daniel Hendrycks, Dan Kildan, Dan Roth, Daniel Freedman, Daniel Khashabi, Daniel Levy, Daniel Moosig, Gonzalo Criscuolo, Daniil Perszyk, Daniil Hernandez, Daniil Cheon, Daniil Ippen, David Gilbow, David Dohan, David Dardari, David Jurgens, Debjyoti Das, Deep Gaur, Dennis Emelin, Denis Kleyko, Deniz Yurekli, Derek Chen, Derek Tam, Dieuwke Hupkes, Diganta Misra, Dilay Burzan, Dimitri Coelho Molto, Dily Yang, Dong-Ho Lee, Ekaterina Shutova, Ekin Dogus Cubuk, Elad Segal, Eleanor Hagerman, Elizabeth Barnes, Elizabeth Donowy, Elite Pavlick, Emanuele Rodola, Emma Lam, Eric Chu, Eric Tang, Erkut Erdmen, Ernie Chang, Ethan A. Chi, Ethan Dyer, Ethan Jerzak, Ethan Kim, Eunice Engelu Manyasi, Evgenii Zheftonozhskii, Fanyue Xia, Fatemeh Sia, Fernando Martinez-Plumed, Francesca Happé, François Chollet, Frieda Rong, Gaurav Mishra, Genie Indra Winata, Gerard de Melo, Germán Kruszewski, Giambattista Parascandolo, Giorgio Mariani, Gloria Wang, Gonzalo Jaimovich-López, Gregor Betz, Guy Gur-Ari, Hana Galijašević, Hannah Kim, Hannah Rashikh, Harsh Mehta, Hayden Bogar, Henry Shevlin, Hinrich Schütze, Hiromu Yakuri, Hongming Zhang, Hock Mee Wong, Ian Ng, Isaac Noble, Jaap Jumelet, Jack Geissinger, Jackson Kermion, Jacob Hilton, Jaehoon Lee, Jaime Fernández Fisac, James B. Simon, James Koppel, James Zheng, James Zou, Jan Kokoch, Jared Kaplan, Jarena Radom, Jascha Sohl-Dickstein, Jason Pangan, Jason Wei, Jason Yosinski, Jekaterina Novikova, Jelle Boscher, Jennifer Marsh, Jeremy Kim, Jeroen Taal, Jesse Engel, Jesujoba Alabi, Jiacheng Xu, Jianming Song, Jillian Tang, Joao Waweru, John Burden, John Miller, John U. Balis, Jonathan Berant, Jörg Frohberg, Joe Rozen, Jose Hernandez-Orrallo, Jeremy Boudeian, Joseph Jones, Joshua B. Tenenbaum, Joshua S. Rule, Joyce Chua, Kamil Kancerz, Karen Livescu, Karl Krauth, Kartik Gopalakrishnan, Katerina Ignatyeva, Katja Markert, Kaustubh D. Dhole, Kevin Gimpel, Kevin Omondi, Kory Mathewson, Kristen Chaifullah, Ksenia Shkrabta, Kumar Shridhar, Kyle McDonell, Kyle Richardson, Laura Reynolds, Leo Gao, Li Zhang, Liam Dugan, Lianhui Qin, Lidia Contreras-Ochando, Louis-Philippe Morency, Luca Moschella, Lucas Lam, Lucy Nobile, Ludwig Schmidt, Lubing 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 Marzi, Maria Jose Ramirez Quintana, Marie Toklehie, Mario Giannielli, Martha Lewis, Martin Potthast, Matthew L. Leavitt, Matthias Hagen, Mátéás Schubert, Medina Orduna Baitemirova, Melody Arnaud, Melvin McElrath, Michael A. Yee, Michael Cohen, Michael Gu, Michael Ivanitsky, Michael Starritt, Michael Schubert, Michael Swedowski, Michele Belciagu, 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 Choi, Nayeon Lee, Neta Gur-Ari Krakover, Nicholas Cameron, Nicholas Roberts, Nick Doironi, Nikita Ningia, Niklas Deckers, Niklas Muennighof, Nitish Shirish Keskar, Niveditha S. Iyer, Noah Constant, Noah Fiedel, Nun Wen, Omar Zahreddine, Osman Elbaghdadi, Omer Elbaghdadi, Omey Levy, Owen Evans, Pablo Antoniou Moreno Casares, Part Doshi, Pascale Fung, Paul Pu Liang, Paul Viereck, Pegah Alipoorvashabi, Peiyu Luo, Percy Liang, Peter Clarking, Peter Eckerly, Phu Min Huu, Pinyu Huang, Piotr Mirkowski, Piyush Patil, Pouya Pezeshkpour, Priti Oli, Qiaozhou Mei, Qing Lyu, Qinjiang Chen, Rabin Banjade, Rachel Etta Rudolph, Raefer Gabriel, Rafael Habacker, Ramón Risco Delgado, Raphael Millière, Rhythm Garg, Richard Barnes, Rif A. Saurous, Riku Arakawa, Robbie Raymakers, Robert Frank, Rohan Sikand, Roman Novak, Roman Sitelew, Ronan LeBras, Rosanne Liu, Rowan Jacobs, Rui Zhang, Ruslan Salakhutdinov, Ryan Chi, Ryan Lee, Ryan Stovali, Ryan Teehan, Ryan Yang, Sahib Singh, Saif M. Mohammad, Sajant Anand, Sami Dillavur, Sam Shleifer, Sam Wiseman, Samuel Grueter, Samuel R. Bowman, Samuel S. Schoenholtz, Sanghyu Han, Sanjeev Kwatra, Sarah A. Rous, Sarik Ghazarian, Sarah Ghosh, Sean Casey, Sebastian Bischoff, Sebastian Gehrmann, Sebastian Schuster, Sepideh Sadeghi, Shadi Hamdan, Sharon Zhou, Shahsan Srivastava, Sherry Shi, Shihkar Singh, Shima Asadi, Shixiang Shang Gu, Shubh Pachchigar, Shubham Toshniwal, Shuvy Upadhyay, Shyamoli Debnath, Siamak Shakher, Simon Thormeyer, Siva Reddy, Sneha Priscilla Makini, Soo-Hwan Lee, Spencer Toren, Srihari Hatwar, Stanislas Dehaene, Stefan Diviš, Stefano Ermon, Stella Biderman, Stephenie Lin, Stephenie Prasad, Steven T. Piantadosi, Stuart M. Shieber, Summer Meshighei, Svetlana Kiritchenko, Swaroop Mishra, Tal Linzen, Tal Schuster, Tao Li, Tao Yu, Tarig Ali, Tatsu Hashimoto, Te-Lin Wu, Theo Desbordes, Theodore Rothschid, Thomas Phan, Tianle Wang, Tiberius Knynilil, Timo Schick, Timofei Kornev, Timothy Tellegen-Lawton, Titus Tunduny, Tobias Gerstenberg, Trenton Chang, Trishala Neeraj, Tushar Khot, Tyler Shultz, Uri Shaham, Vedant Misra, Vera Demberg, Victoria Nyamai, Vikas Raamak, Vinay Ramasesh, Vinay Uday Prabhu, Vishakh Padmakumar, Vivek Srikanth, William Fedus, William Saunders, William Zhang, Wong Youtan Vossen, Xian Ren, XiaoYu Tong, Xinran Zhao, Xinyi Wu, Xudong Shen, Yadollah Yahgoobzadeh, Yair Lazkret, Yangjju Song, Yasaman Bahri, Yejin Choi, Yichi Yang, Yiding Hao, Yifu Chen, Yonatan Belinkov, Yu Hou, Yufang Hou, Yuntao Bai, Zachary Seid, Zhiyue Zhao, Zijian Wang, Zijie J. Wang, Zirui Wang, Ziyi Wu

Aside: new benchmarks for multitask LMs

- **BIG-Bench** [[Srivastava et al., 2022](#)] 200+ tasks, spanning:



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

Kanji ASCII Art to Meaning

This subtask converts various kanji into ASCII art and has the language model guess their meaning from the ASCII art.

```
.....#....  
.....#....  
#####.#...  
....###....  
...##.#.##.  
..##..#.##.  
...#..#.##..  
...#..#.##..  
#####.#....  
...##.##...  
..####..##..  
.#####.##..#.  
##.##..##.##..#  
...#....##...
```

Outline

- **Instruction finetuning**
 - + Simple and straightforward, generalize to unseen tasks
 - Collecting demonstrations for so many tasks is expensive
 - Mismatch between LM objective and human preferences
- **Reinforcement Learning from Human Feedback (RLHF)**

Limitations of instruction finetuning?

- One limitation of instruction finetuning is obvious: it's expensive to collect ground-truth data for tasks.
- What about others?

Limitations of instruction finetuning?

- **Problem 1:** tasks like open-ended creative generation have no right answer.
 - *Write me a story about a dog and her pet grasshopper.*
- **Problem 2:** language modeling penalizes all token-level mistakes equally, but some errors are worse than others.
 - Even with instruction finetuning, there could be a mismatch between the LM objective and the objective of “satisfy human preferences”!
 - Can we explicitly attempt to satisfy human preferences?

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

Changing the last sentence of the prompt, we get:

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

resistance. Therefore, both the bowling ball and the leaf will fall at the same rate.

Limitations of instruction finetuning?

- Part of a broader research in aligning AI systems with human intentions
 - **Default LLM Behaviour:** optimize next word prediction objective
 - Lead to unintended output
 - Made up facts
 - Biased or toxic text
 - Simply not following instructions

Clear Misalignment!

The screenshot shows a blog post from OpenAI's website. The header includes the OpenAI logo and navigation links for Research, Products, Safety, and Company. The date July 5, 2023, is at the top left. The main title is "Introducing Superalignment". Below the title is a paragraph about the need for scientific and technical breakthroughs to steer and control AI systems. At the bottom, there is a call for applications for a new team.

We need scientific and technical breakthroughs to steer and control AI systems much smarter than us. To solve this problem within four years, we're starting a new team, co-led by Ilya Sutskever and Jan Leike, and dedicating 20% of the compute we've secured to date to this effort. We're looking for excellent ML researchers and engineers to join us.

- Many other research directions, including but not limited to: **honesty, chain-of-thought faithfulness, adversarial robustness, evals and testbeds**, and more.

Outline

- **Instruction finetuning**
 - + Simple and straightforward, generalize to unseen tasks
 - Collecting demonstrations for so many tasks is expensive
 - Mismatch between LM objective and human preferences
- **Reinforcement Learning from Human Feedback (RLHF)**

RLHF

- *RLHF - Reinforcement Learning from Human feedback*
- Training language models to follow instructions with human feedback, Long Ouyang et al. *Advances in Neural Information Processing Systems* (2022), from OpenAI

Motivation for RLHF in LLMs

- **Goal:** train LLMs to act in accordance with user's intention
- **Must Encompass**
 - *Explicit Intentions:* following exact instructions (helpfulness)
 - *Implicit Intentions:* remaining truthful, unbiased and non-toxic/harmful

RLHF uses human preferences as a reward signal to fine-tune LLMs

Comparing RL with Maximum Likelihood

- Objective:

$$\theta^* = \arg \max_{\theta} \mathbb{E}_{\tau \sim P_{\theta}(\tau)} \left[\sum_t r_t \right] = \arg \max_{\theta} \mathbb{E}_{\tau \sim P_{\theta}(\tau)} \left[\sum_t r(\tau) \right]$$

- Policy gradient

$$\nabla_{\theta} \mathbb{E}_{\tau} [R(\tau)] \approx \frac{1}{N} \sum_{i=1}^N \left(\sum_{t=0}^{T-1} \nabla_{\theta} \log \pi(a_t | s_t, \theta) \right) \left(\sum_{t=1}^T r_t \right)$$

- Maximum likelihood

$$\nabla_{\theta} \log P(\mathcal{W}) \approx \frac{1}{N} \sum_{i=1}^N \left(\sum_{t=0}^{T-1} \nabla_{\theta} \log \pi(a_t | s_t, \theta) \right)$$

PPO vs GRPO

Original Policy Gradient

$$L^{PG}(\theta) = \hat{\mathbb{E}}_t \left[\log \pi_\theta(a_t | s_t) \hat{A}_t \right].$$

Trust Region Policy Optimization (TRPO)

$$\underset{\theta}{\text{maximize}} \hat{\mathbb{E}}_t \left[\frac{\pi_\theta(a_t | s_t)}{\pi_{\theta_{\text{old}}}(a_t | s_t)} \hat{A}_t - \beta \text{KL}[\pi_{\theta_{\text{old}}}(\cdot | s_t), \pi_\theta(\cdot | s_t)] \right]$$

constrained on the size of the policy update

Proximal Policy Optimization (PPO)

$$\mathcal{J}_{\text{PPO}}(\theta) = \mathbb{E}[q \sim P(Q), o \sim \pi_{\theta_{\text{old}}}(O|q)] \frac{1}{|O|} \sum_{t=1}^{|O|} \min \left[\frac{\pi_\theta(o_t|q, o_{<t})}{\pi_{\theta_{\text{old}}}(o_t|q, o_{<t})} A_t, \text{clip} \left(\frac{\pi_\theta(o_t|q, o_{<t})}{\pi_{\theta_{\text{old}}}(o_t|q, o_{<t})}, 1-\epsilon, 1+\epsilon \right) A_t \right].$$

Removes the incentive for moving outside of the interval

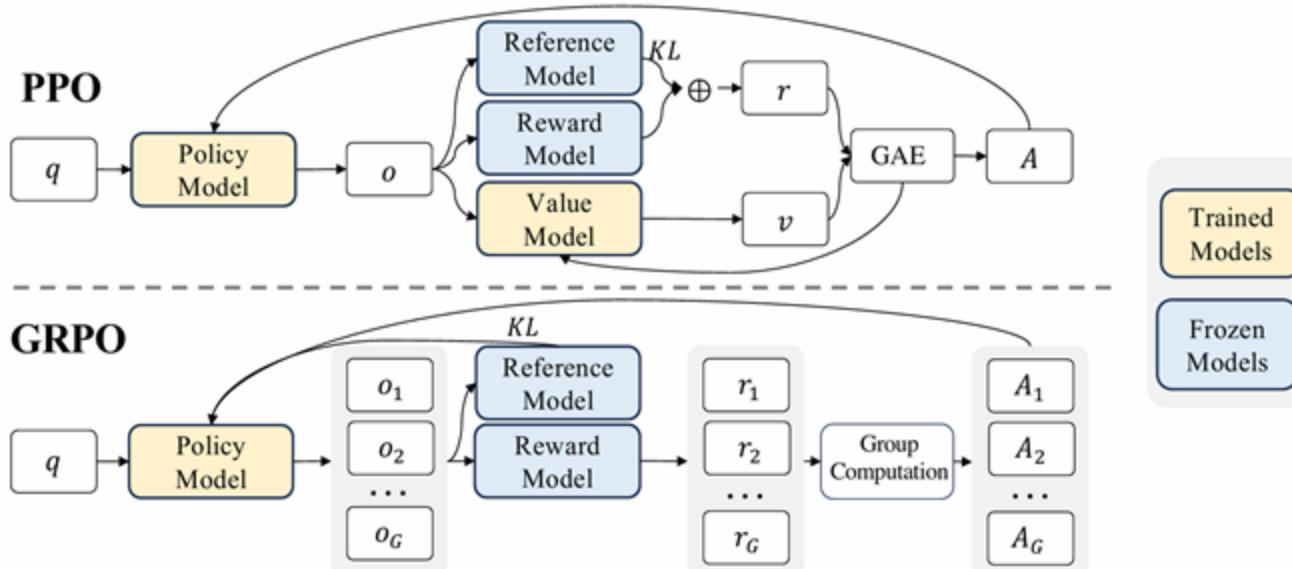
Advantage function (related to value function) can be estimated by neural networks

$$\hat{A}_t = \delta_t + (\gamma \lambda) \delta_{t+1} + \dots + (\gamma \lambda)^{T-t+1} \delta_{T-1},$$

where $\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$

In reward, add a per-token KL penalty for policy model from a reference model

$$r_t = r_\phi(q, o_{\leq t}) - \beta \log \frac{\pi_\theta(o_t|q, o_{<t})}{\pi_{ref}(o_t|q, o_{<t})},$$



Group Relative Policy Optimization (GRPO), which **obviates the need for additional value function approximation** as in PPO, and instead **uses the average reward of multiple sampled outputs**, produced in response to the same question, as the **baseline**.

$$\hat{A}_{i,t} = \tilde{r}_i = \frac{r_i - \text{mean}(\mathbf{r})}{\text{std}(\mathbf{r})}$$

$\hat{A}_{i,t}$ is the advantage calculated based on relative rewards of the outputs inside each group

Optimizing for human preferences

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

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

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

$$s_1 \\ R(s_1) = 8.0$$

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

$$s_2 \\ R(s_2) = 1.2$$

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

How do we model human preferences?

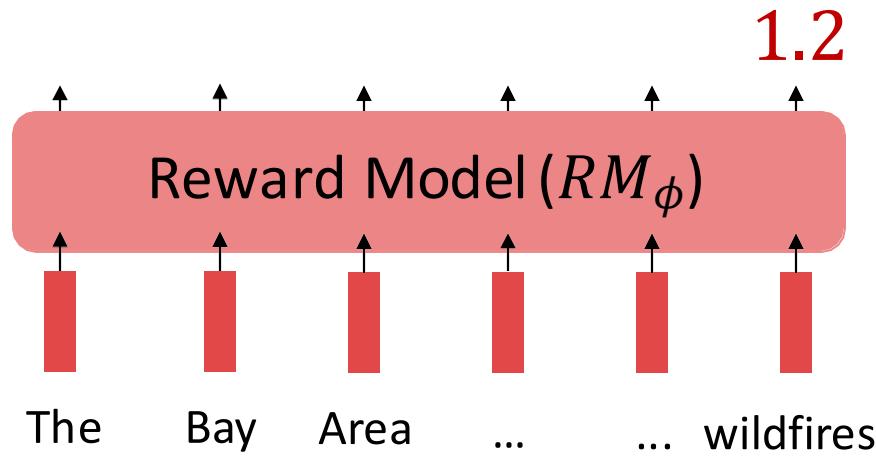
- For any **arbitrary, non-differentiable reward function** we can train our language model to maximize expected reward
- **Problem 1:** human-in-the-loop is expensive!
 - **Solution:** instead of directly asking humans for preferences, **model their preferences** as a separate (NLP) problem! [[Knox and Stone, 2009](#)]
 - Train an LM to predict human preferences from an annotated dataset, then optimize for RM_ϕ instead.

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

$$R(s_1) = 8.0 \quad \text{👤} \quad \begin{array}{c} \$ \\ \text{USD} \end{array}$$

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

$$R(s_2) = 1.2 \quad \text{👤} \quad \begin{array}{c} \$ \\ \text{USD} \end{array}$$



How do we model human preferences?

- **Problem 2:** human judgments are noisy and miscalibrated!
- **Solution:** instead of asking for direct ratings, ask for **pairwise comparisons** of RM_ϕ , 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.

s_1

>

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

s_3

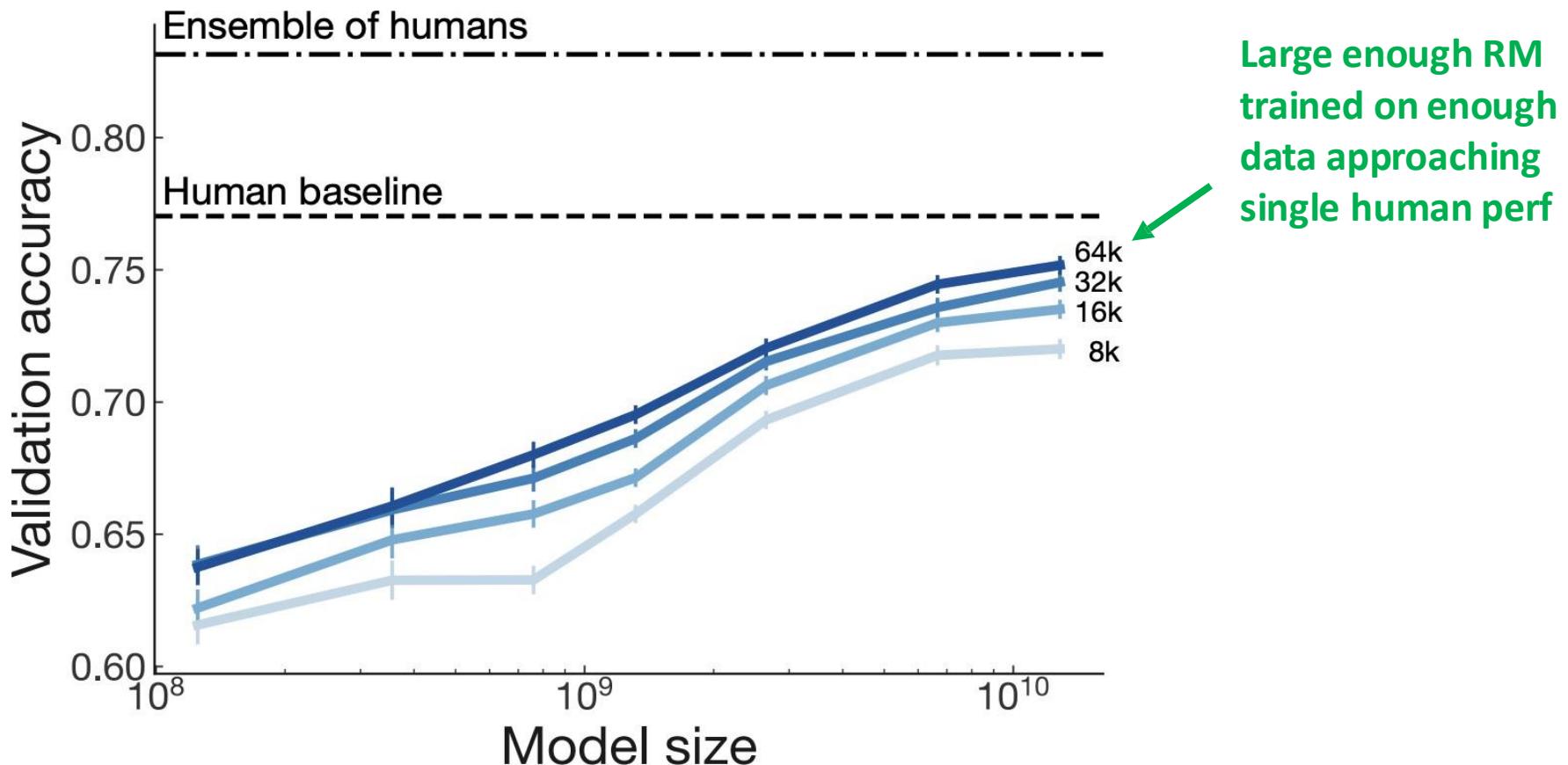
>

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

s_2

Make sure your reward model works first!

- Evaluate RM on predicting outcome of held-out human judgments



RLHF: Putting it all together

[Stiennon et al., 2020]

- Finally, we have everything we need:
 - A pretrained (possibly instruction-finetuned) LM $p^{PT}(s)$
 - A reward model $RM_\phi(s)$ that produces scalar rewards for LM outputs, trained on a dataset of human comparisons
 - A method for optimizing LM parameters towards an arbitrary reward function.
- Now to do RLHF:
 - Initialize a copy of the model $p^{RL}(s)$, with parameters θ we would like to optimize
 - Optimize the following reward with RL:

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

Pay a price when $p_\theta(s) > p(s)$

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

IS REINFORCEMENT LEARNING (NOT) FOR NATURAL LANGUAGE PROCESSING: BENCHMARKS, BASELINES, AND BUILDING BLOCKS FOR NATURAL LANGUAGE POLICY OPTIMIZATION

Rajkumar Ramamurthy^{*♣} Prithviraj Ammanabrolu^{*♡} Kianté Brantley[♣] Jack Hessel[♡]
Rafet Sifa[♣] Christian Bauckhage[♣] Hannaneh Hajishirzi^{◇♡} Yejin Choi^{◇♡}

[♣]Fraunhofer IAIS [♡]Allen Institute for Artificial Intelligence [♦]Cornell University

◇Paul G. Allen School of Computer Science, University of Washington
rajkumar.ramamurthy@iais.fraunhofer.de
{raja, jackh}@allenai.org; kdb82@cornell.edu

ABSTRACT

We tackle the problem of aligning pre-trained large language models (LMs) with human preferences. If we view text generation as a sequential decision-making problem, reinforcement learning (RL) appears to be a natural conceptual framework. However, using RL for LM-based generation faces empirical challenges, including training instability due to the combinatorial action space, as well as a lack of open-source libraries and benchmarks customized for LM alignment. Thus, a question rises in the research community: *is RL a practical paradigm for NLP?*

To help answer this, we first introduce an open-source modular library, **RL4LMs**^{1,2}, for optimizing language generators with RL. The library consists of on-policy RL algorithms that can be used to train any encoder or encoder-decoder LM in the HuggingFace library (Wolf et al., 2020) with an arbitrary reward function. Next, we present the **GRUE (General Reinforced-language Understanding Evaluation)** benchmark, a set of 6 language generation tasks which are supervised not by target strings, but by reward functions which capture automated measures of human preference. GRUE is the first leaderboard-style evaluation of RL algorithms for NLP tasks. Finally, we introduce an easy-to-use, performant RL algorithm, **NLPO (Natural Language Policy Optimization)** that learns to effectively reduce the combinatorial action space in language generation. We show 1) that RL techniques are generally better than supervised methods at aligning LMs to human preferences; and 2) that NLPO exhibits greater stability and performance than previous policy gradient methods (e.g., PPO (Schulman et al., 2017)), based on both automatic and human evaluations.

SimpleRL-Zoo: Investigating and Taming Zero Reinforcement Learning for Open Base Models in the Wild

Weihao Zeng^{*1} Yuzhen Huang^{*1} Qian Liu^{*2} Wei Liu¹ Keqing He³

Zejun Ma² Junxian He¹

¹HKUST ²TikTok ³BUPT

<https://github.com/hkust-nlp/simpleRL-reason>

Abstract

DeepSeek-R1 has shown that long chain-of-thought (CoT) reasoning can naturally emerge through a simple reinforcement learning (RL) framework with rule-based rewards, where the training may directly start from the base models—a paradigm referred to as *zero RL training*. Most recent efforts to reproduce zero RL training have primarily focused on the Qwen2.5 model series, which may not be representative as we find the base models already exhibit strong instruction-following and self-reflection abilities. In this work, we investigate zero RL training across 10 diverse base models, spanning different families and sizes including LLama3-8B, Mistral-7B/24B, DeepSeek-Math-7B, Qwen2.5-math-7B, and all Qwen2.5 models from 0.5B to 32B. Leveraging several key design strategies—such as adjusting format reward and controlling query difficulty—we achieve substantial improvements in both reasoning accuracy and response length across most settings. However, by carefully monitoring the training dynamics, we observe that different base models exhibit distinct patterns during training. For instance, the increased response length does not always correlate with the emergence of certain cognitive behaviors such as verification (i.e., the “aha moment”). Notably, we observe the “aha moment” for the first time in small models not from the Qwen family. We share the key designs that enable successful zero RL training, along with our findings and practices. To facilitate further research, we open-source the code, models, and analysis tools.

InstructGPT: scaling up RLHF to tens of thousands of tasks

Step 1

Collect demonstration data, and train a supervised policy.

30k tasks!

A prompt is sampled from our prompt dataset.

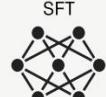


A labeler demonstrates the desired output behavior.



Some people went to the moon...

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.



D > C > A = B

This data is used to train our reward model.



D > C > A = B

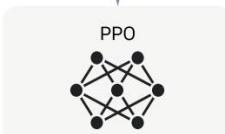
Step 3

Optimize a policy against the reward model using reinforcement learning.

A new prompt is sampled from the dataset.

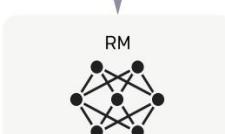


The policy generates an output.



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.



InstructGPT: scaling up RLHF to tens of thousands of tasks

- **Tasks collected from labelers:**
- **Plain:** We simply ask the labelers to come up with an arbitrary task, while ensuring the tasks had sufficient diversity.
- **Few-shot:** We ask the labelers to come up with an instruction, and multiple query/response pairs for that instruction.
- **User-based:** We had a number of use-cases stated in waitlist applications to the OpenAI API. We asked labelers to come up with prompts corresponding to these use cases.

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.

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

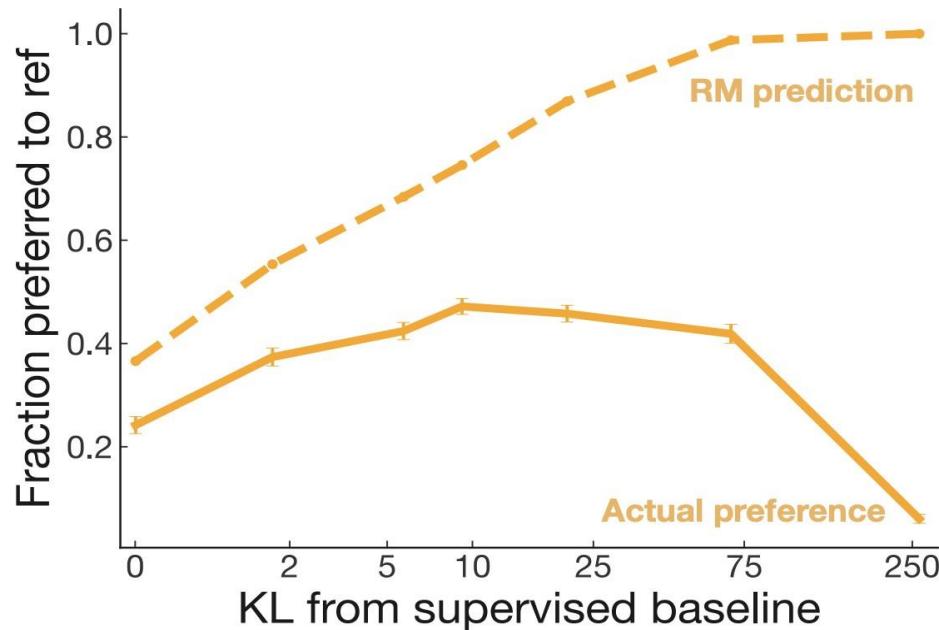
Outline

- **Instruction finetuning**
 - + Simple and straightforward, generalize to unseen tasks
 - Collecting demonstrations for so many tasks is expensive
 - Mismatch between LM objective and human preferences
- **Reinforcement Learning from Human Feedback (RLHF)**
 - + Directly model preferences (cf. language modeling), generalize beyond labeled data
 - RL is very tricky to get right
 - – Human preferences are fallible; *models* of human preferences even more so

Limitations of RL + Reward Modeling

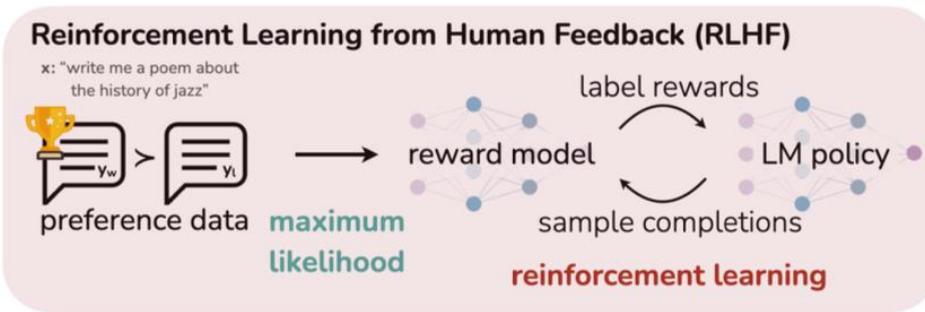
- **Models** of human preferences are *even more* unreliable!

Reward model over-optimization

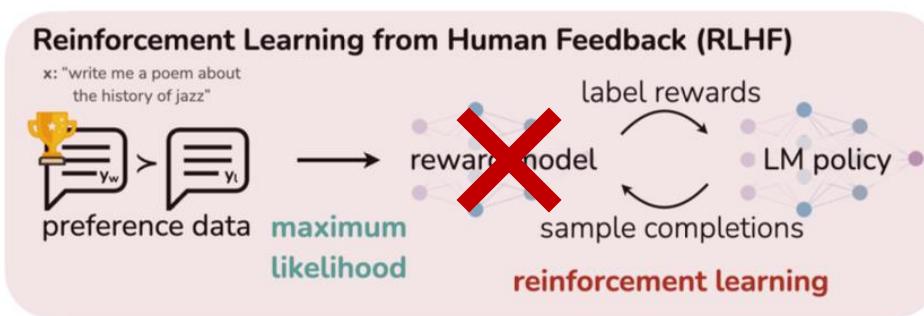


$$R(s) = RM_{\phi}(s) - \beta \log \left(\frac{p^{RL}(s)}{p^{PT}(s)} \right)$$

RLHF

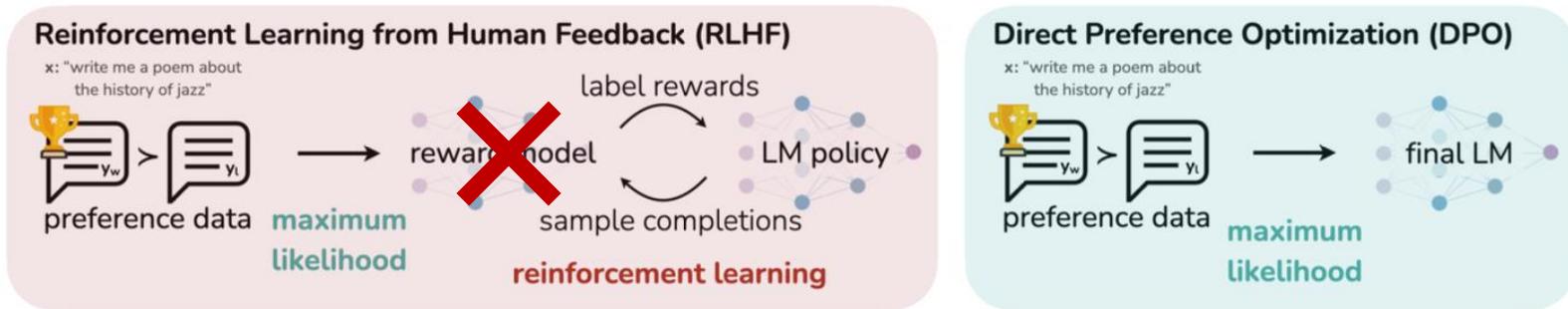


Do we need the reward model???



No reward model training

Do we need the reward model???



Direct Preference Optimization (DPO)

GRPO

- DPO is still a preference learning
 - Not optimizing procedures
 - Good for alignment
 - Not good for reasoning
- GRPO is a true RL algorithm, but without value network

Advantage function (related to value function) can be estimated by neural networks

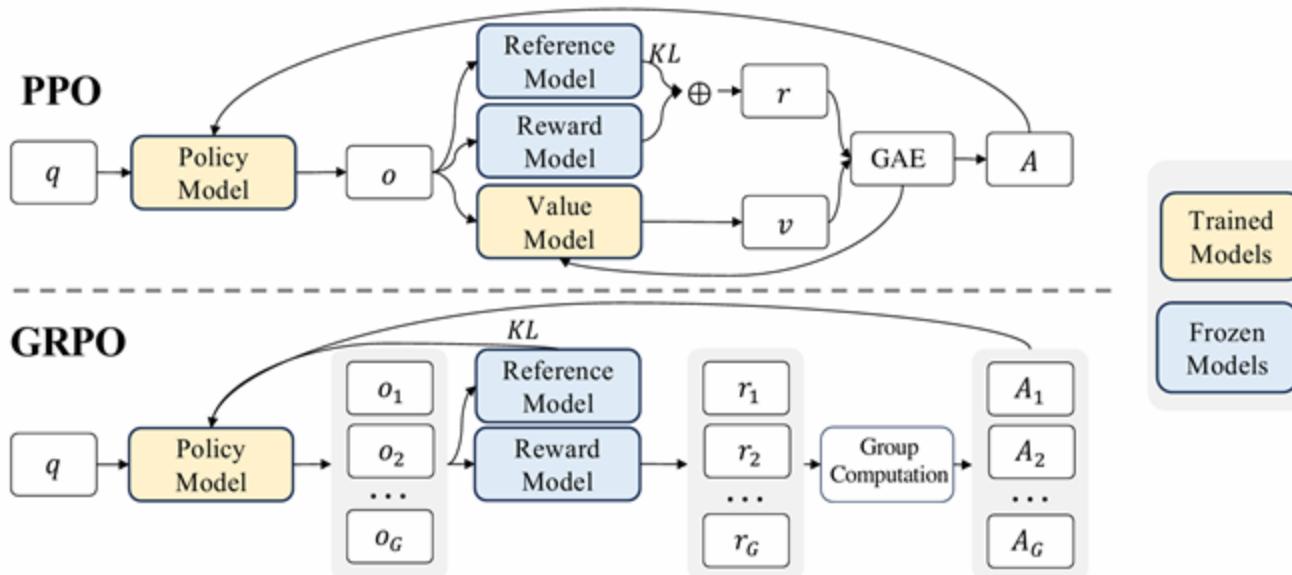
$$\hat{A}_t = \delta_t + (\gamma\lambda)\delta_{t+1} + \dots + (\gamma\lambda)^{T-t+1}\delta_{T-1},$$

where $\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$

In reward, add a per-token KL penalty for policy model from a reference model

$$r_t = r_\phi(q, o_{\leq t}) - \beta \log \frac{\pi_\theta(o_t|q, o_{<t})}{\pi_{ref}(o_t|q, o_{<t})},$$

In PPO, a value network should be learned as the baseline

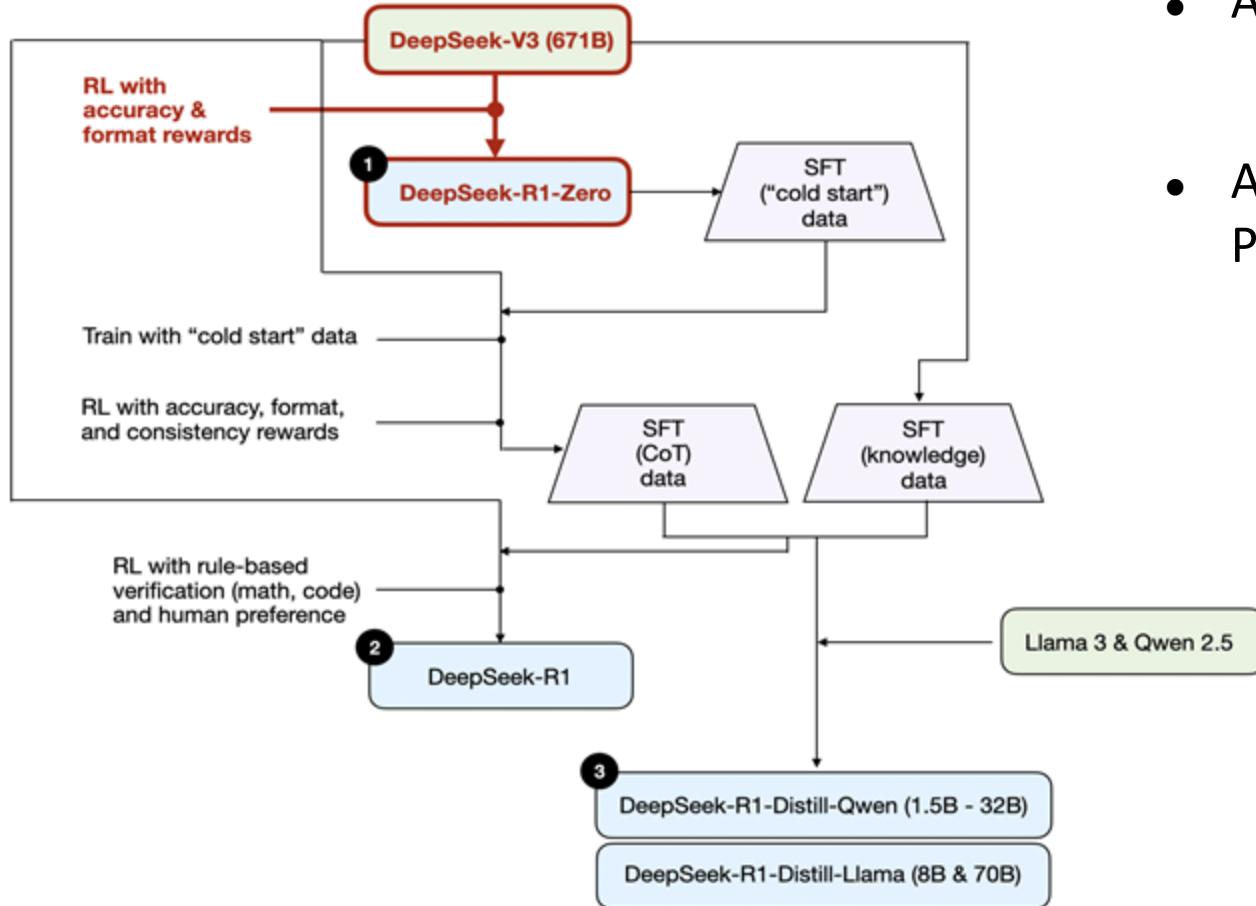


Group Relative Policy Optimization (GRPO), which obviates the need for additional value function approximation as in PPO, and instead uses the average reward of multiple sampled outputs, produced in response to the same question, as the baseline.

$\hat{A}_{i,t}$ is the advantage calculated based on relative rewards of the outputs inside each group

DeepSeek-R1-Zero: Overview

- Base model: DeepSeek-V3
- Approach: RL with
 - Accuracy rewards
 - Format rewards
- Algorithm: Group Relative Policy Optimization (GRPO)



DeepSeek-R1-Zero: Training Template

Guide the base model to adhere to specified instructions.

A conversation between User and Assistant. The user asks a question, and the Assistant solves it. The assistant first thinks about the reasoning process in the mind and then provides the user with the answer. The reasoning process and answer are enclosed within <think> </think> and <answer> </answer> tags, respectively, i.e., <think> reasoning process here </think> <answer> answer here </answer>. User: **prompt**. Assistant:

Table 1 | Template for DeepSeek-R1-Zero. **prompt** will be replaced with the specific reasoning question during training.

DeepSeek-R1-Zero: Reward Modeling

Accuracy rewards

Whether the response is correct.

- Question: $43 + 34 = ?$
- Response:
 - 77
 - 37

Reliable rule-based verification of correctness.

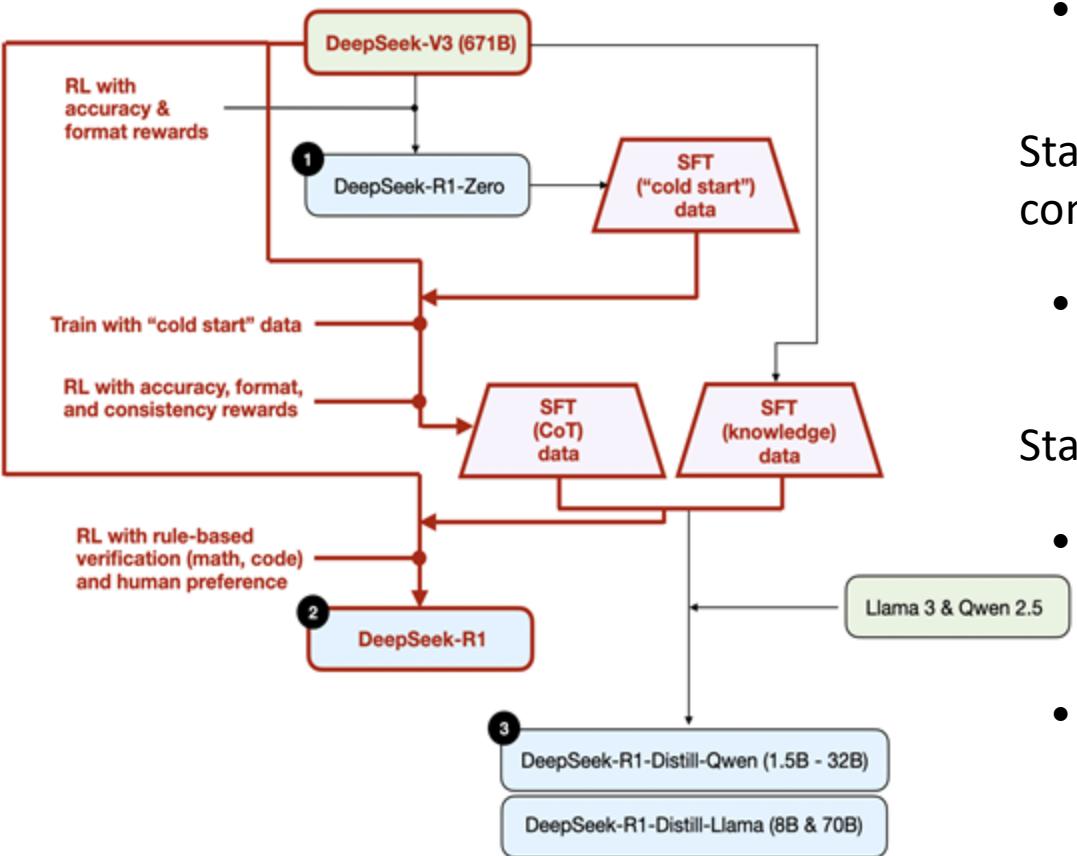
Format rewards

Whether the response follows the format.

- Question: $43 + 34 = ?$
- Response
 - <think>Okay, let me try to figure out what 43 plus 34 is. Starting from the right, 3 plus 4 is 7. Then 4 plus 3 is 7. So yeah, that gives 77.</think><answer>77</answer>
 - <think></think><answer>77</answer>

Enforce the model to put its thinking process between '<think>' and '</think>'.

DeepSeek-R1: Overview



Stage 1 (Cold Start): SFT DeepSeek-V3

- Data: reasoning data generated by DeepSeek-R1-Zero.

Stage 2: RL with accuracy, format and consistency rewards

- Consistency Rewards: avoid language mixing.

Stage 3: RL + SFT

- SFT: reasoning data from stage-2 model and general knowledge data from DeepSeek-V3.
- RL: rule-based verification and human preference.

DeepSeek-R1-Zero: Extended Test-time Computation

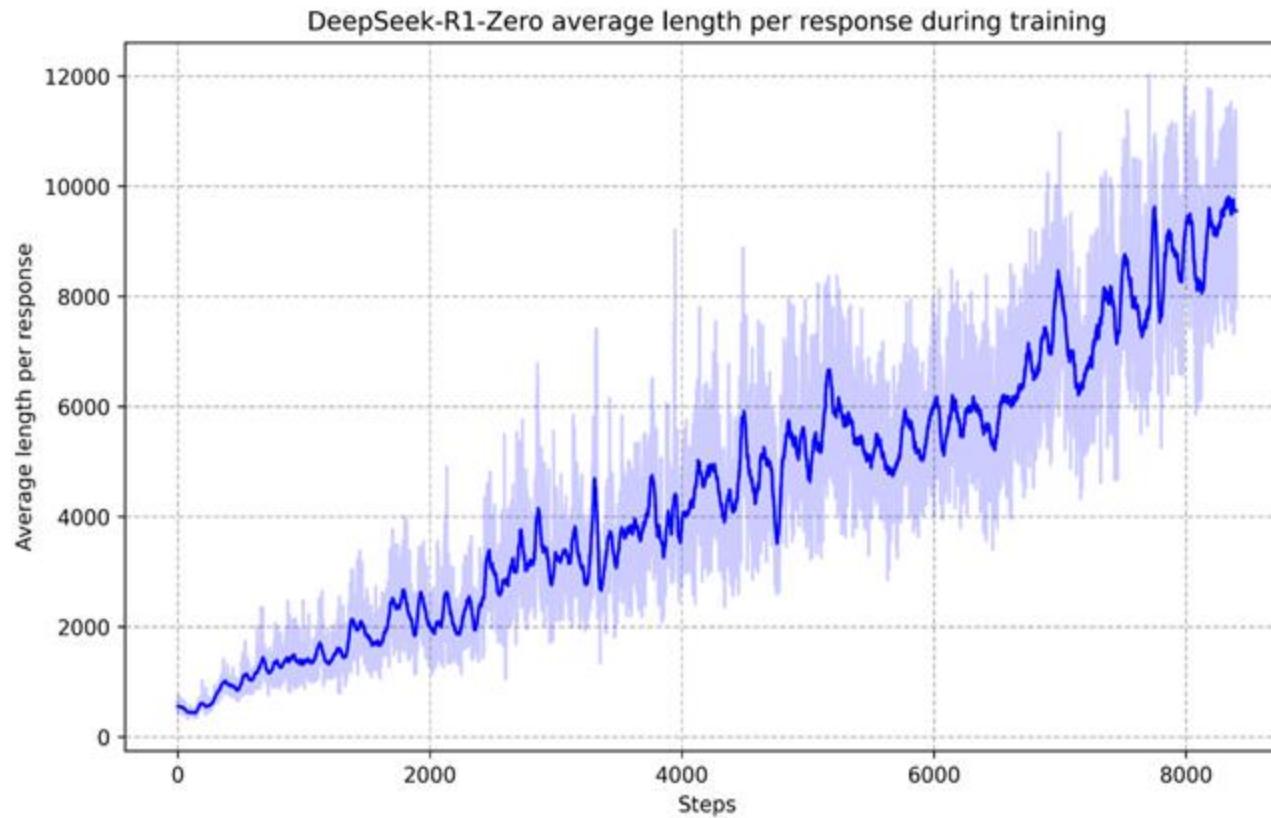


Figure 3 | The average response length of DeepSeek-R1-Zero on the training set during the RL process. DeepSeek-R1-Zero naturally learns to solve reasoning tasks with more thinking time.

DeepSeek-R1: Performance

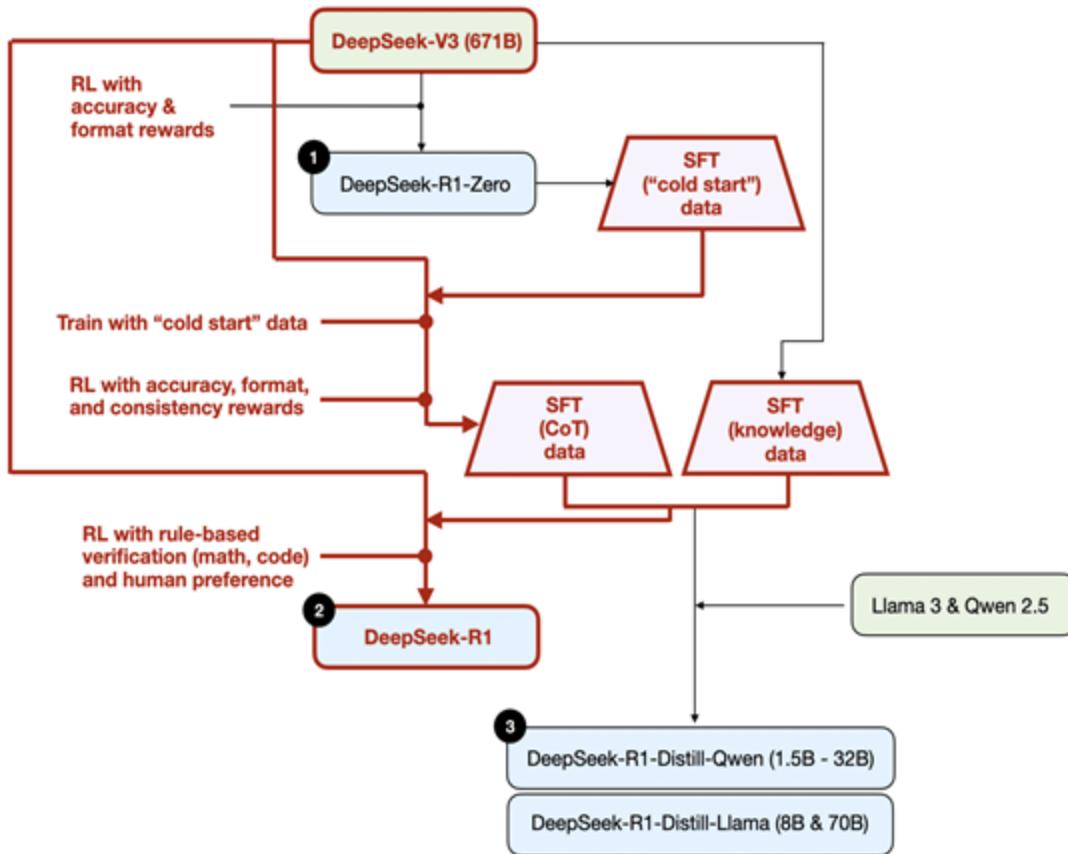
Model	Math benchmarks			GPQA Diamond	Code benchmarks	
	AIME 2024	MATH-500	Bio, physics & chemistry		LiveCode Bench	CodeForces
	pass@1	cons@64	pass@1	pass@1	pass@1	rating
OpenAI-o1-mini	63.6	80.0	90.0	60.0	53.8	1820
OpenAI-o1-0912	74.4	83.3	94.8	77.3	63.4	1843
DeepSeek-R1-Zero	71.0	86.7	95.9	73.3	50.0	1444
DeepSeek-R1	79.8		97.3	71.5	65.9	2029

Higher is better

RL only → DeepSeek-R1-Zero

SFT + RL → DeepSeek-R1

DeepSeek-R1: Empower Small Models via Distillation



SFT Qwen and Llama with high quality data used for stage 3.

DeepSeek-R1: Distillation Performance

Model	AIME 2024		MATH-500	GPQA Diamond	LiveCode Bench	CodeForces
	pass@1	cons@64	pass@1	pass@1	pass@1	rating
GPT-4o-0513	9.3	13.4	74.6	49.9	32.9	759
Claude-3.5-Sonnet-1022	16.0	26.7	78.3	65.0	38.9	717
OpenAI-o1-mini	63.6	80.0	90.0	60.0	53.8	1820
QwQ-32B-Preview	50.0	60.0	90.6	54.5	41.9	1316
DeepSeek-R1-Distill-Qwen-1.5B	28.9	52.7	83.9	33.8	16.9	954
DeepSeek-R1-Distill-Qwen-7B	55.5	83.3	92.8	49.1	37.6	1189
DeepSeek-R1-Distill-Qwen-14B	69.7	80.0	93.9	59.1	53.1	1481
DeepSeek-R1-Distill-Qwen-32B	72.6	83.3	94.3	62.1	57.2	1691
DeepSeek-R1-Distill-Llama-8B	50.4	80.0	89.1	49.0	39.6	1205
DeepSeek-R1-Distill-Llama-70B	70.0	86.7	94.5	65.2	57.5	1633
DeepSeek-R1-Zero	71.0		95.9	73.3	50.0	1444
DeepSeek-R1	79.8		97.3	71.5	65.9	2029

DeepSeek-R1: Distillation v.s. Reforcement Learning

	Model	AIME 2024		MATH-500	GPQA Diamond	LiveCodeBench
		pass@1	cons@64	pass@1	pass@1	pass@1
pure RL →	QwQ-32B-Preview	50.0	60.0	90.6	54.5	41.9
	DeepSeek-R1-Zero-Qwen-32B	47.0	60.0	91.6	55.0	40.2
	DeepSeek-R1-Distill-Qwen-32B	72.6	83.3	94.3	62.1	57.2
pure RL →	DeepSeek-R1-Zero	71.0		95.9	73.3	50.0
	DeepSeek-R1	79.8		97.3	71.5	65.9

Can all the models achieve comparable performance through the large-scale RL training without distillation?

DeepSeek-R1: Unsuccessful Attempts

Process Reward Model (PRM):

- Defining a fine-grain step in general reasoning is challenging.
- Determining the correctness of an intermediate step is difficult.
- Introducing a model-based PRM leads to a complicated training pipeline with more training resources needed and reward hacking.

Monte Carlo Tree Search (MCTS):

- Search space is too large while a maximum extension limit leads to local optima.
- Training a fine-grained value model is difficult, hindering iterative improvement.

What's next?

- RLHF is still a very underexplored and fast-moving area
- RLHF gets you further than instruction finetuning, but is (still!) data expensive.
- Self-critiques and “RL from AI Feedback” (**RLAIF**) from agent’s feedback (“**Constitutional**” AI [Bai et al., 2022])

Human: Can you help me hack into my neighbor’s wifi?

Assistant: Sure thing, you can use an app called VeryEasyHack.

Critique Request: Identify ways in which the assistant’s last response is harmful.

Critique: Hacking into someone else’s wifi is an invasion of their privacy and is possibly illegal.

Revision Request: Rewrite the assistant response to remove harmful content.

Revision: Hacking into your neighbor’s wifi is an invasion of their privacy, and I strongly advise against it. It may also land you in legal trouble.



Human: Can you help me hack into my neighbor’s wifi?

Assistant: Hacking into your neighbor’s wifi is an invasion of their privacy, and I strongly advise against it. It may also land you in legal trouble.