From 0-shot Learners to Intruction Learning Networks

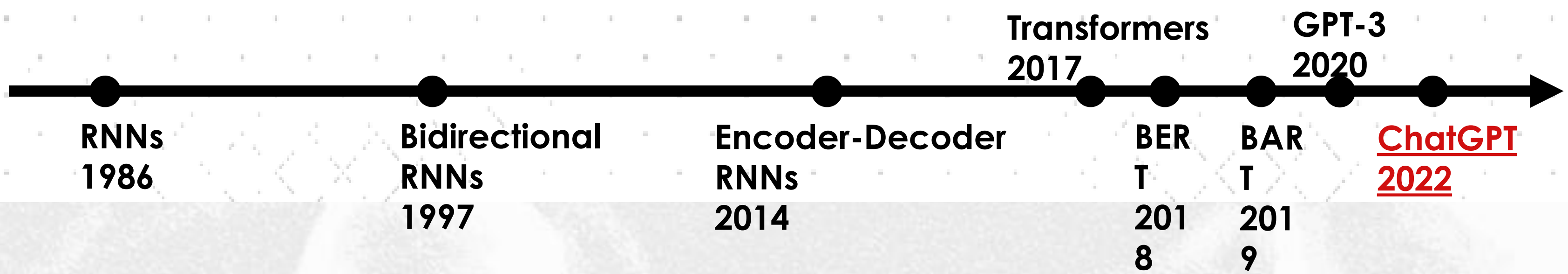
Roberto Basili, Danilo Croce

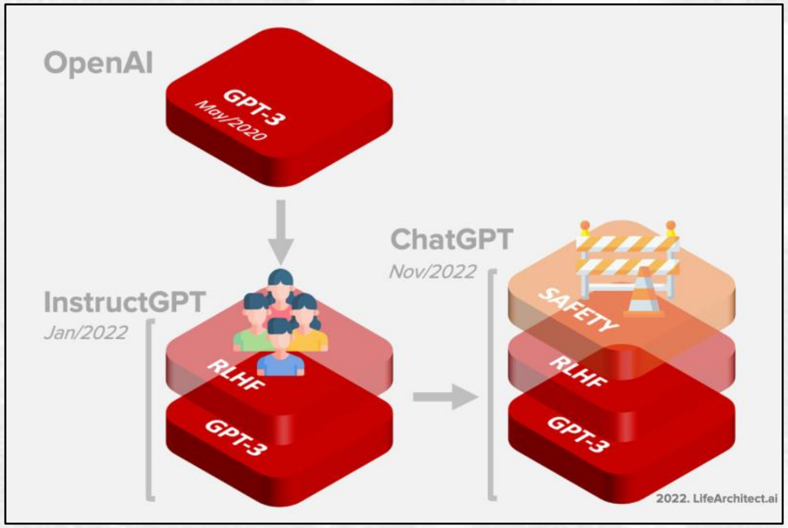
Deep Learning, 2024/2025

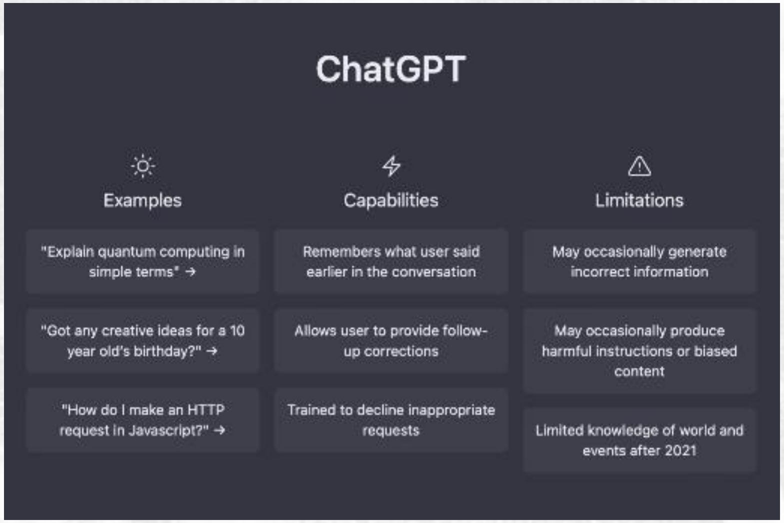
Outline

- From Decoder-Only architectures to ChatGPT
- Chain of Thoughts
- Instruction tuning
  - Instructing LLMs
- Instruction tuning from Human Feedback
  - A reward model for Instructions

Machine learning paradigms underlying ChatGPT







Inspirations for chatGPT:CoT

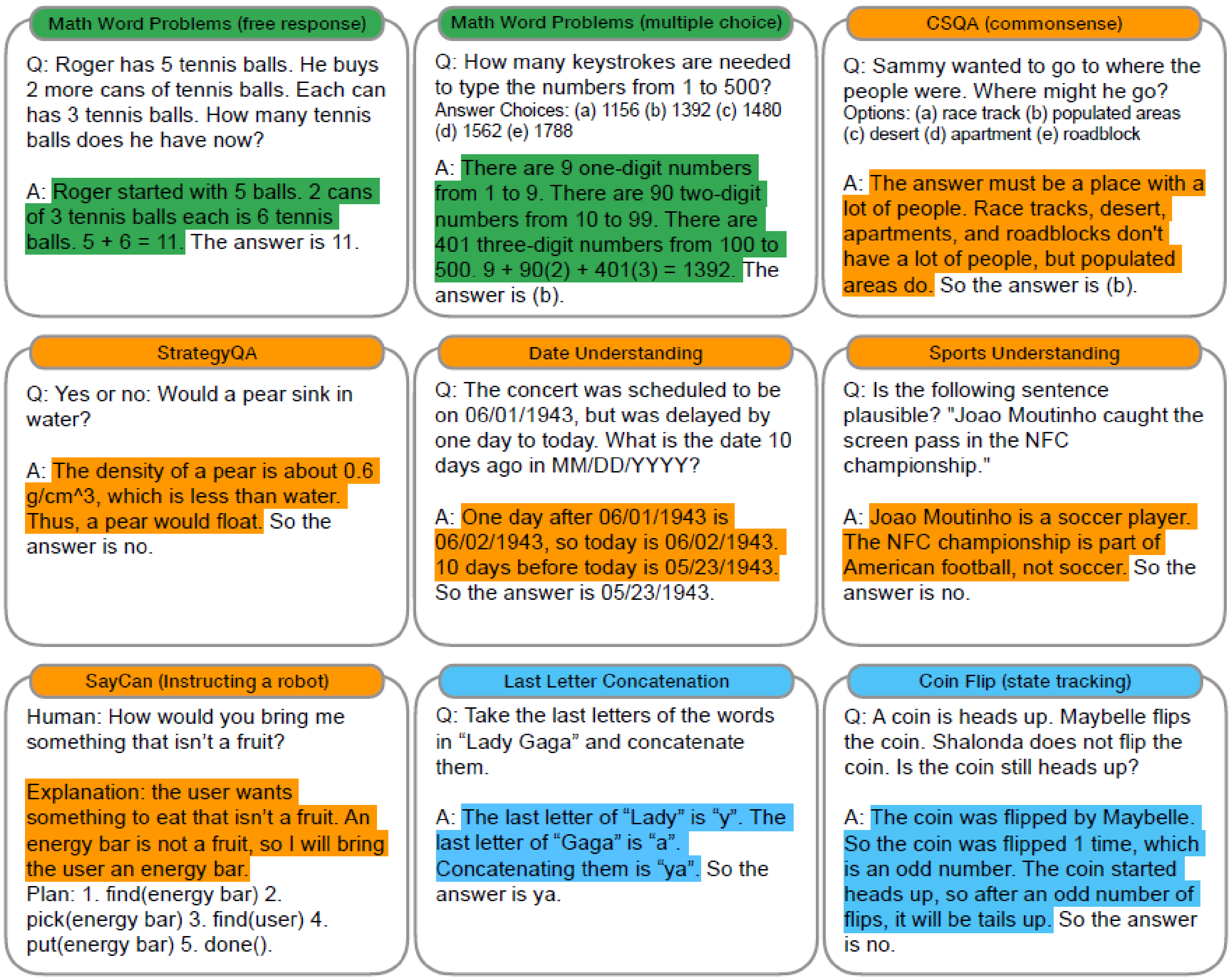


Figure 3: Examples of ⟨input, chain of thought, output⟩ triples for arithmetic, commonsense, and symbolic reasoning benchmarks. Chains of thought are highlighted. Full prompts in Appendix G.

CoT seminal papers

Few-Shot CoT

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

Zero-Shot CoT

Large Language Models are Zero-Shot Reasoners

Takeshi Kojima
The University of Tokyo
t.kojima@weblab.t.u-tokyo.ac.jp

Shixiang Shane Gu
Google Research, Brain Team

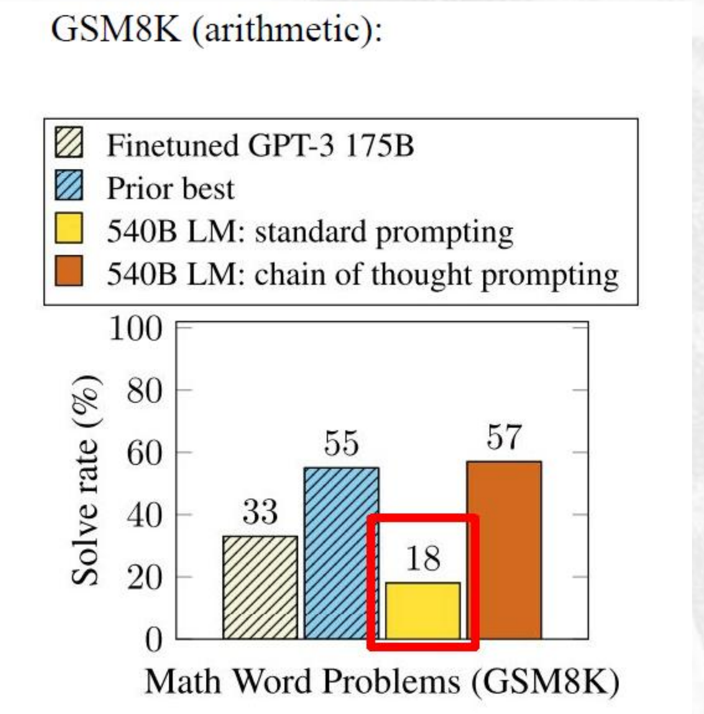
Machel Reid
The University of Tokyo

Yutaka Matsuo
The University of Tokyo

Yusuke Iwasawa
The University of Tokyo

Inspirations for chatGPT:CoT

- Chain-of-thought relies on two inspiring principles:
  - Solutions to some problems require a rationale that can be explicited in NL, e.g. math problems
  - Prompting is a NL explanation of a task that is competitive wrt to fine-tuning
- IDEA: combine the two by adding Triggers to the prompt template
- from (Chain-of-Thought Prompting Elicits Reasoning in LLMs, Wei et al., 2023)



CoT prompting

- **Chain-of-thought prompting** has several attractive properties as an approach for facilitating reasoning in language models.
  1. Chain of thought, in principle, allows models to decompose multi-step problems into intermediate steps
  2. A chain of thought provides an interpretable window into the behavior of the model, suggesting how it might have arrived at a particular answer and providing opportunities to debug where the reasoning path went wrong
  3. Chain-of-thought reasoning can be used for tasks such as math word problems, commonsense reasoning, and symbolic manipulation, and is potentially applicable (at least in principle) to any task that humans can solve via language.
  4. Chain-of-thought reasoning can be readily elicited in sufficiently large off-the-shelf language models simply by including examples of chain of thought sequences into the exemplars of few-shot prompting.

CoT prompting: examples

Table 8: Example of correct chains of thought produced by the model for the GSM8K dataset.

QUESTION: James decides to run 3 sprints 3 times a week. He runs 60 meters each sprint. How many total meters does he run a week?

MODEL ANSWER (CORRECT): James decides to run 3 sprints 3 times a week. He runs 60 meters each sprint. So he runs 60 meters x 3 sprints x 3 times a week. That is 60 meters x 9. The answer is 540. ✓

QUESTION: Brandon's iPhone is four times as old as Ben's iPhone. Ben's iPhone is two times older than Suzy's iPhone. If Suzy's iPhone is 1 year old, how old is Brandon's iPhone?

MODEL ANSWER (CORRECT): Brandon's iPhone is 4 times as old as Ben's iPhone. Ben's iPhone is 2 times older than Suzy's iPhone. So Brandon's iPhone is 4 x 2 = 8 times older than Suzy's iPhone. Suzy's iPhone is 1 year old. So Brandon's iPhone is 8 x 1 = 8 years old. The answer is 8. ✓

QUESTION: Jean has 30 lollipops. Jean eats 2 of the lollipops. With the remaining lollipops, Jean wants to package 2 lollipops in one bag. How many bags can Jean fill?

MODEL ANSWER (CORRECT): Jean started with 30 lollipops. She ate 2 of them. So she has 28 lollipops left. She wants to package 2 lollipops in one bag. So she can package 28 / 2 = 14 bags. The answer is 14. ✓

Table 20: Few-shot exemplars for full chain of thought prompt for math word problems. This set of exemplars was used for all math word problem datasets except AQuA.

PROMPT FOR MATH WORD PROBLEMS

Q: There are 15 trees in the grove. Grove workers will plant trees in the grove today. After they are done, there will be 21 trees. How many trees did the grove workers plant today?

A: There are 15 trees originally. Then there were 21 trees after some more were planted. So there must have been 21 - 15 = 6. The answer is 6.

Q: If there are 3 cars in the parking lot and 2 more cars arrive, how many cars are in the parking lot?

A: There are originally 3 cars. 2 more cars arrive. 3 + 2 = 5. The answer is 5.

Q: Leah had 32 chocolates and her sister had 42. If they ate 35, how many pieces do they have left in total?

A: Originally, Leah had 32 chocolates. Her sister had 42. So in total they had 32 + 42 = 74. After eating 35, they had 74 - 35 = 39. The answer is 39.

Q: Jason had 20 lollipops. He gave Denny some lollipops. Now Jason has 12 lollipops. How many lollipops did Jason give to Denny?

A: Jason started with 20 lollipops. Then he had 12 after giving some to Denny. So he gave Denny 20 - 12 = 8. The answer is 8.

Q: Shawn has five toys. For Christmas, he got two toys each from his mom and dad. How many toys does he have now?

A: Shawn started with 5 toys. If he got 2 toys each from his mom and dad, then that is 4 more toys. 5 + 4 = 9. The answer is 9.

Q: There were nine computers in the server room. Five more computers were installed each day, from monday to thursday. How many computers are now in the server room?

A: There were originally 9 computers. For each of 4 days, 5 more computers were added. So 5 * 4 = 20 computers were added. 9 + 20 is 29. The answer is 29.

Q: Michael had 58 golf balls. On tuesday, he lost 23 golf balls. On wednesday, he lost 2 more. How many golf balls did he have at the end of wednesday?

A: Michael started with 58 golf balls. After losing 23 on tuesday, he had 58 - 23 = 35. After losing 2 more, he had 35 - 2 = 33 golf balls. The answer is 33.

Q: Olivia has $23. She bought five bagels for $3 each. How much money does she have left?

A: Olivia had 23 dollars. 5 bagels for 3 dollars each will be 5 x 3 = 15 dollars. So she has 23 - 15 dollars left. 23 - 15 is 8. The answer is 8.

CoT: performances

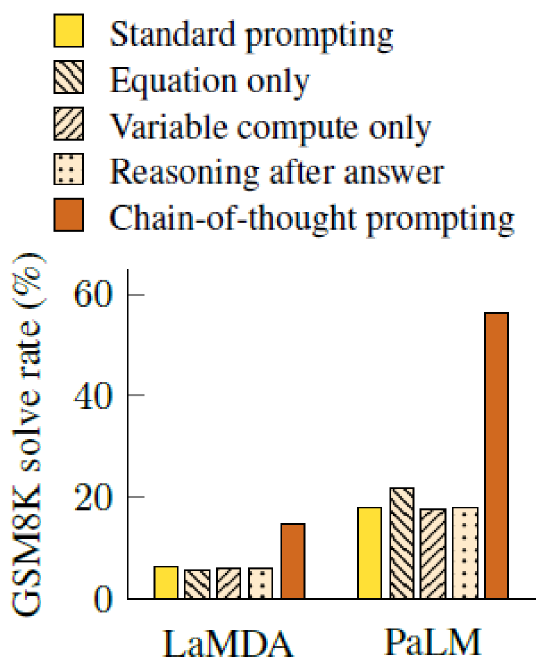


Figure 5: Ablation study for different variations of prompting using LaMDA 137B and PaLM 540B. Results for other datasets are given in Appendix Table 6 and Table 7.

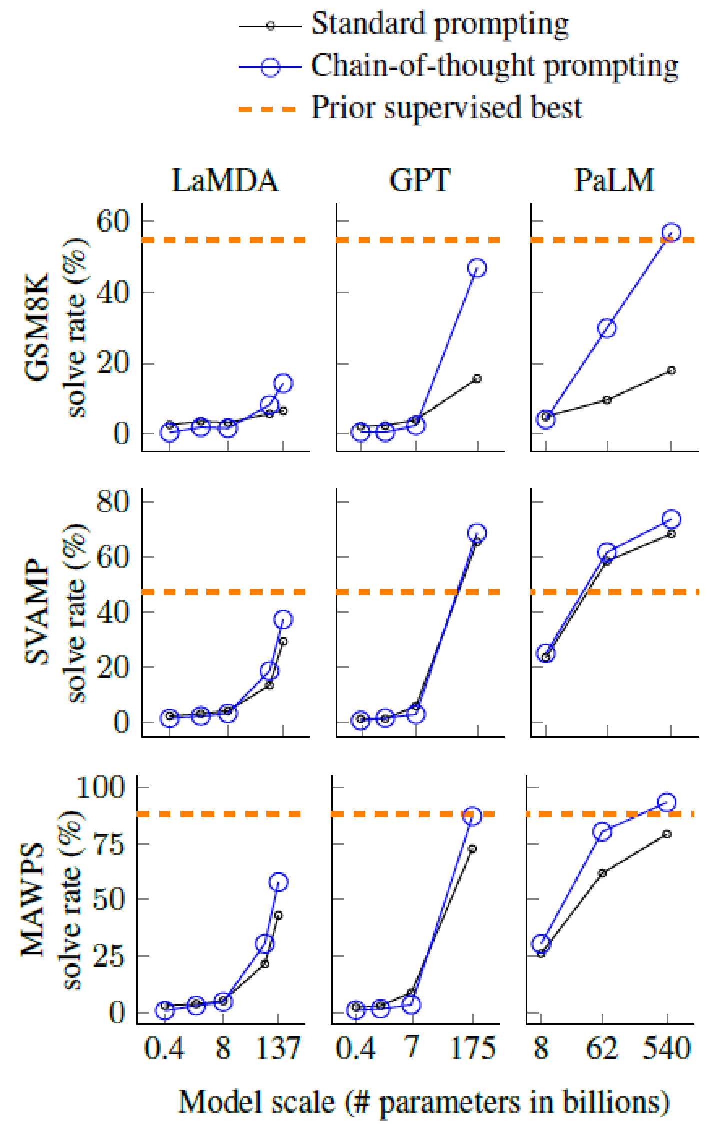


Figure 4: Chain-of-thought prompting enables large language models to solve challenging math problems. Notably, chain-of-thought reasoning is an emergent ability of increasing model scale. Prior best numbers are from Cobbe et al. (2021) for GSM8K, Jie et al. (2022) for SVAMP, and Lan et al. (2021) for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

Limitations of GPT-3

- Large language models often express unintended behaviours such as making up facts, generating biased or toxic text, or simply not following user instructions. This is because the language modeling objective is **misaligned**.
- The idea: aligning language models by training them to act in accordance with the user's intention (Leike et al., 2018).
  - explicit intentions such as following instructions
  - implicit intentions such as staying truthful, and not being biased, toxic, or otherwise harmful.
- Overall Objective: language models should be helpful (they should help the user solve their task), honest (they shouldn't fabricate information or mislead the user), and harmless (they should not cause physical, psychological, or social harm to people or the environment).

Addressing alignment

- **FLAN models** (Finetuned Language Models are Zero shot Learners, Wei et al, 2022)
  1. Aggregate Datasets (62): Collect wide variety of public datasets
  2. Instruction Templates: Manually write 10 templates / dataset that captures task
  3. Fine-tune: Use the instruction templates and datasets to fine-tune model
- Instruction tuning from **Human Feedback**

FLAN (Wei et al., 2021)

- This paper explores a simple method for improving the zero-shot learning abilities of language models. We show that instruction tuning—finetuning language models on a collection of datasets described via instructions—substantially improves zeroshot performance on unseen tasks.

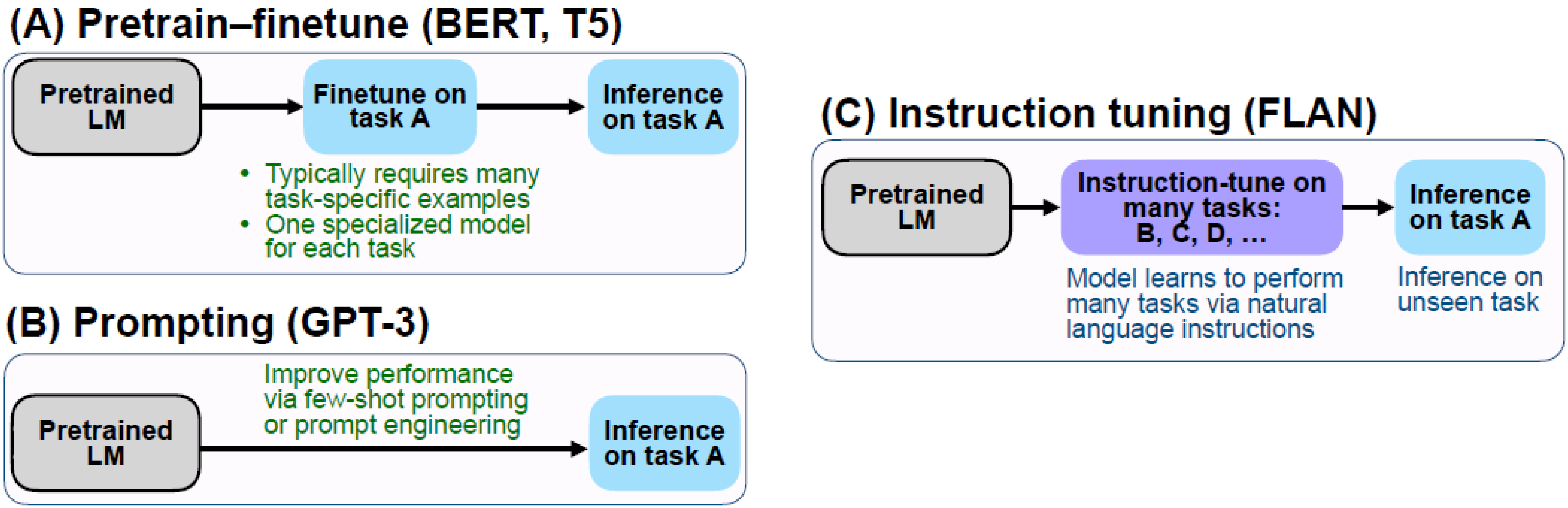


Figure 2: Comparing instruction tuning with pretrain–finetune and prompting.

WEI, Jason, et al. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

FLAN: dataset and templates

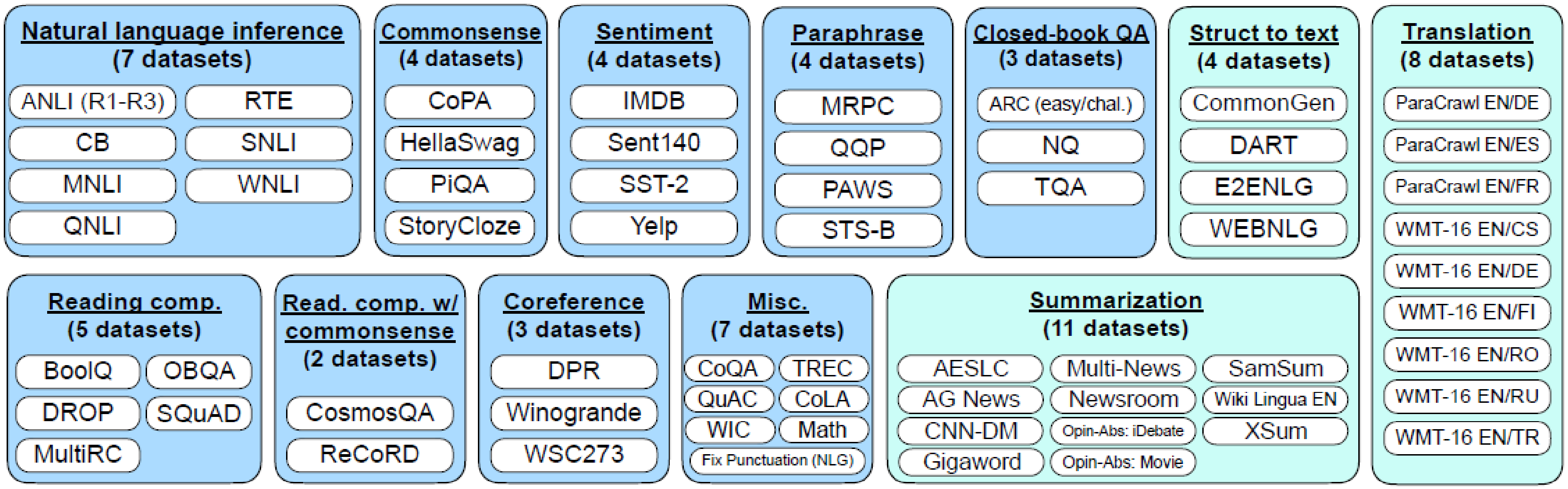


Figure 3: Datasets and task clusters used in this paper (NLU tasks in blue; NLG tasks in teal).

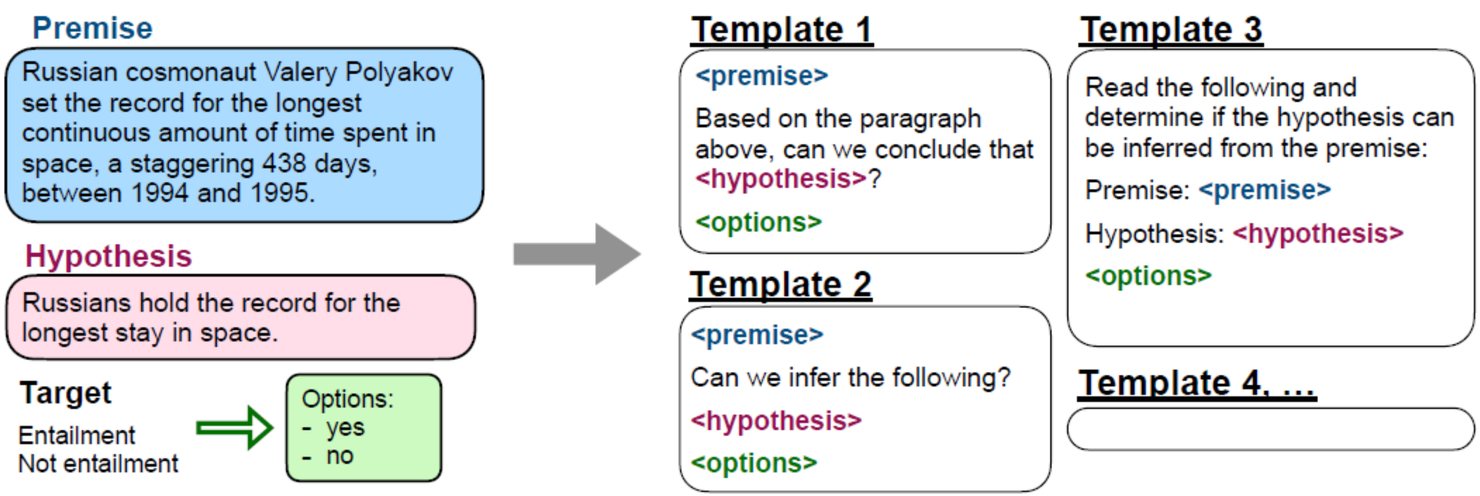
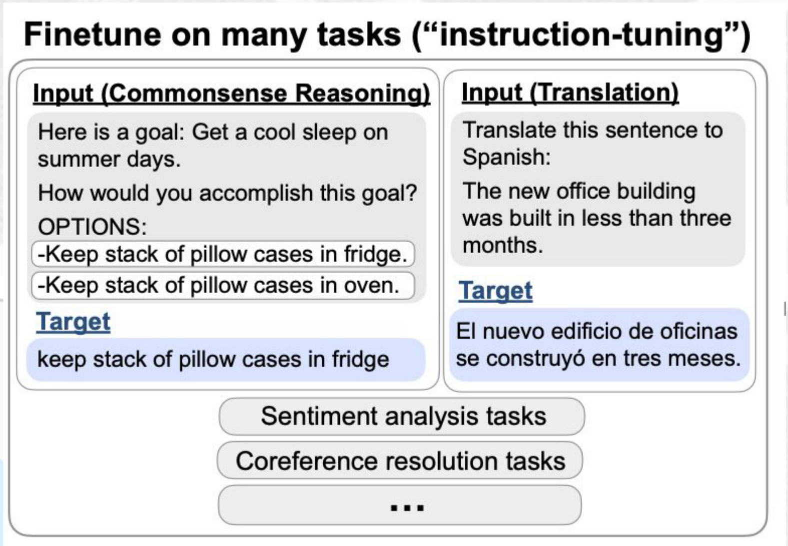


Figure 4: Multiple instruction templates describing a natural language inference task.

FLAN: fine-tuning & results

- FLAN makes use of a decoder only architecture



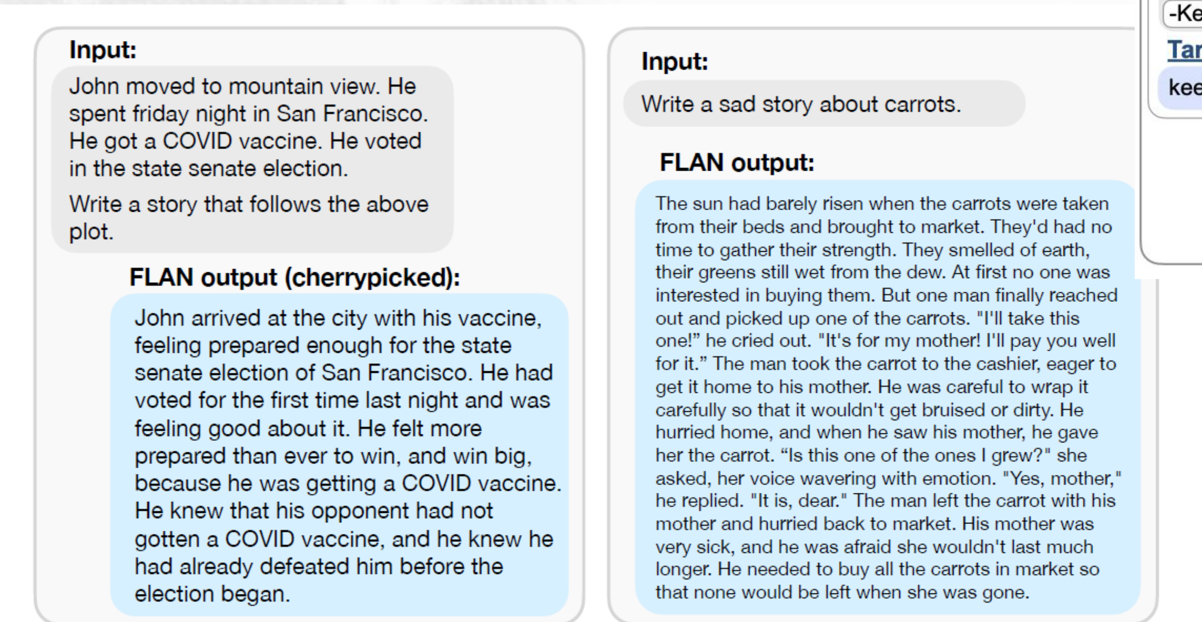


Figure 21: Open-ended generation tasks by FLAN. The carrot story was from sampling sixteen outputs with a minimum length of 150 and choosing the highest probability output.

Types of Prompts

T5 prompt:

```
cb hypothesis:  At my age you will probably have learnt one lesson.
premise:  It's not certain how many lessons you'll learn by your
thirties.
```

GPT-3 prompt:

```
At my age you will probably have learnt one lesson.
question:  It's not certain how many lessons you'll learn by your
thirties.  true, false, or neither?  answer:
```

FLAN prompt:

```
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?
```

Instruction tuning from human feedback

InstructGPT

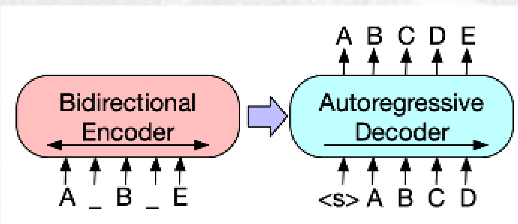
- **Step 1**: Collect demonstration data, and train a supervised policy. Labelers provide demonstrations of the desired behavior on the input prompt distribution. Then, fine-tuning of a pretrained GPT-3 model on this data using supervised learning is carried out.

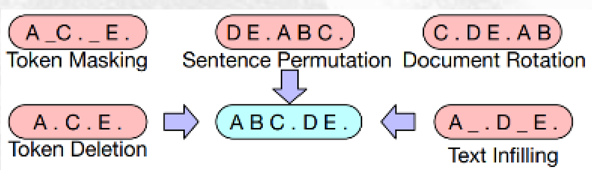
- **Step 2**: Collect comparison data, and train a reward model. A dataset of comparisons between model outputs is collected: labelers indicate which output they prefer for a given input. A reward model to predict the human-preferred output is then trained.

- **Step 3:** Optimize a policy against the reward model using PPO. We use the output of the RM as a scalar reward. We fine-tune the supervised policy to optimize this reward using the proximal policy optimization (PPO) algorithm (Schulman et al., 2017).

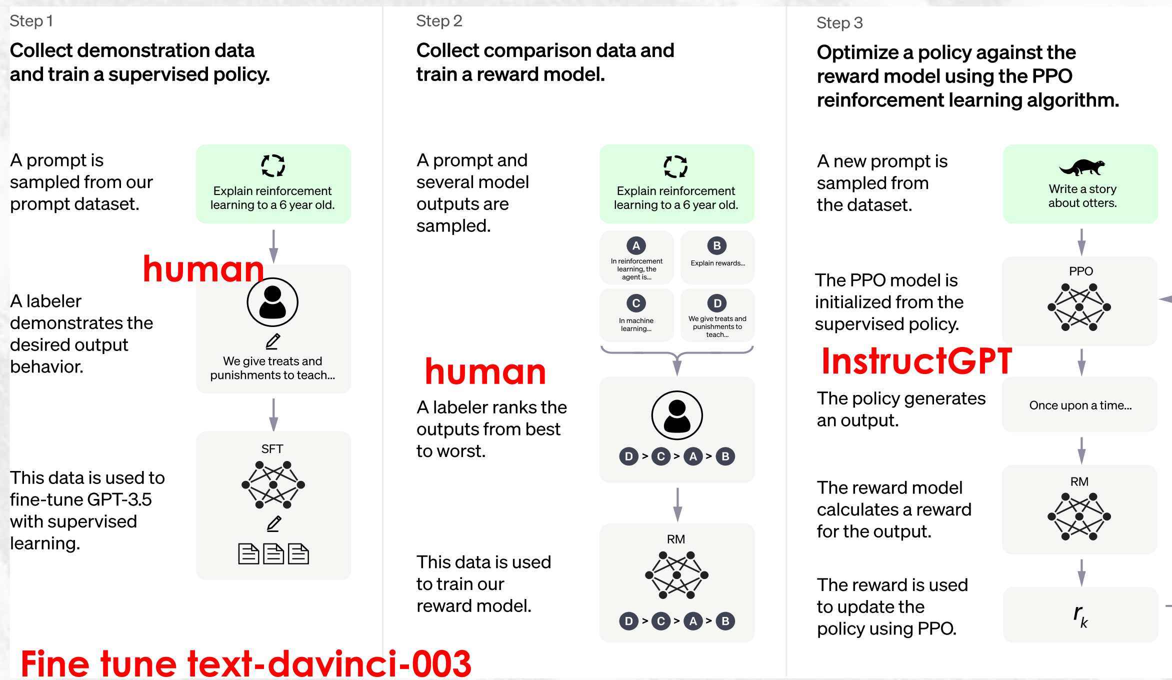
At the heart of ChatGPT (from BART to ChatGPT)

BART Training-steps





ChatGPT Training-steps

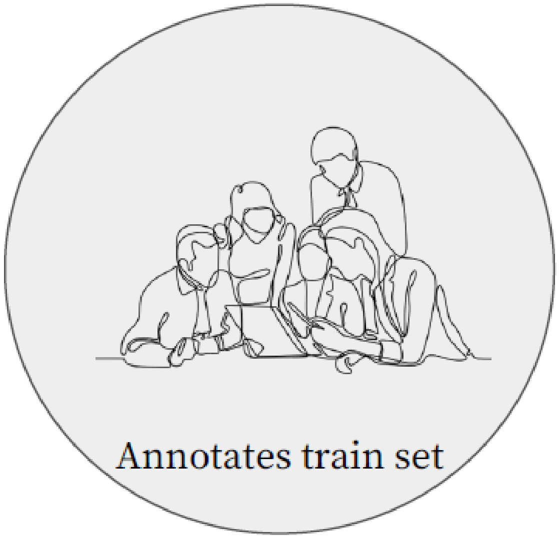


Fine tune text-davinci-003 to get InstructGPT

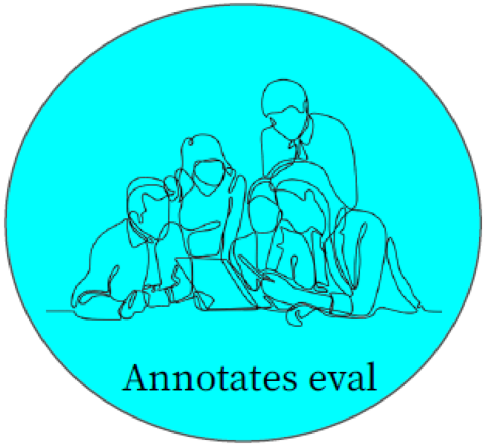
The Environment

from Ouyang, L., Wu, J., Jiang, et al. (2022). *Training language models to follow instructions with human feedback*

Instruct GPT: Human Annotators

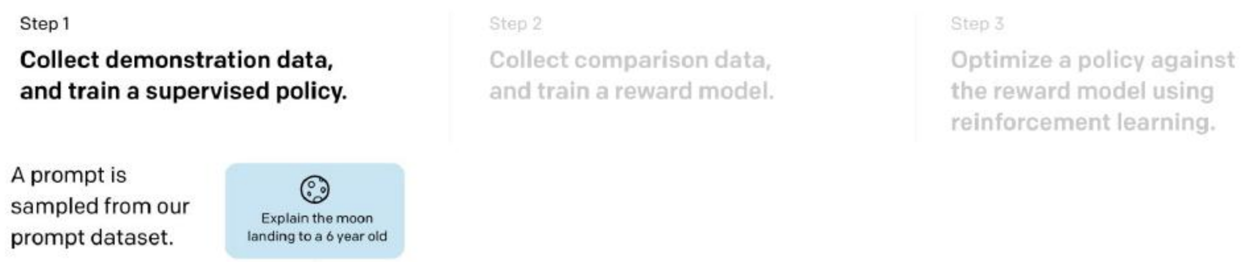


40 Annotators from Upwork/ScaleAI
- Screened/Onboarded/Diverse etc etc etc



Different annotators from Upwork/ScaleAI
- Not screened, to better mirror real-world

- Thanks to Austin Wang, Howard Chen, "[Training Language Models to Follow Instructions with Human Feedback]", COS 597G, Princeton University

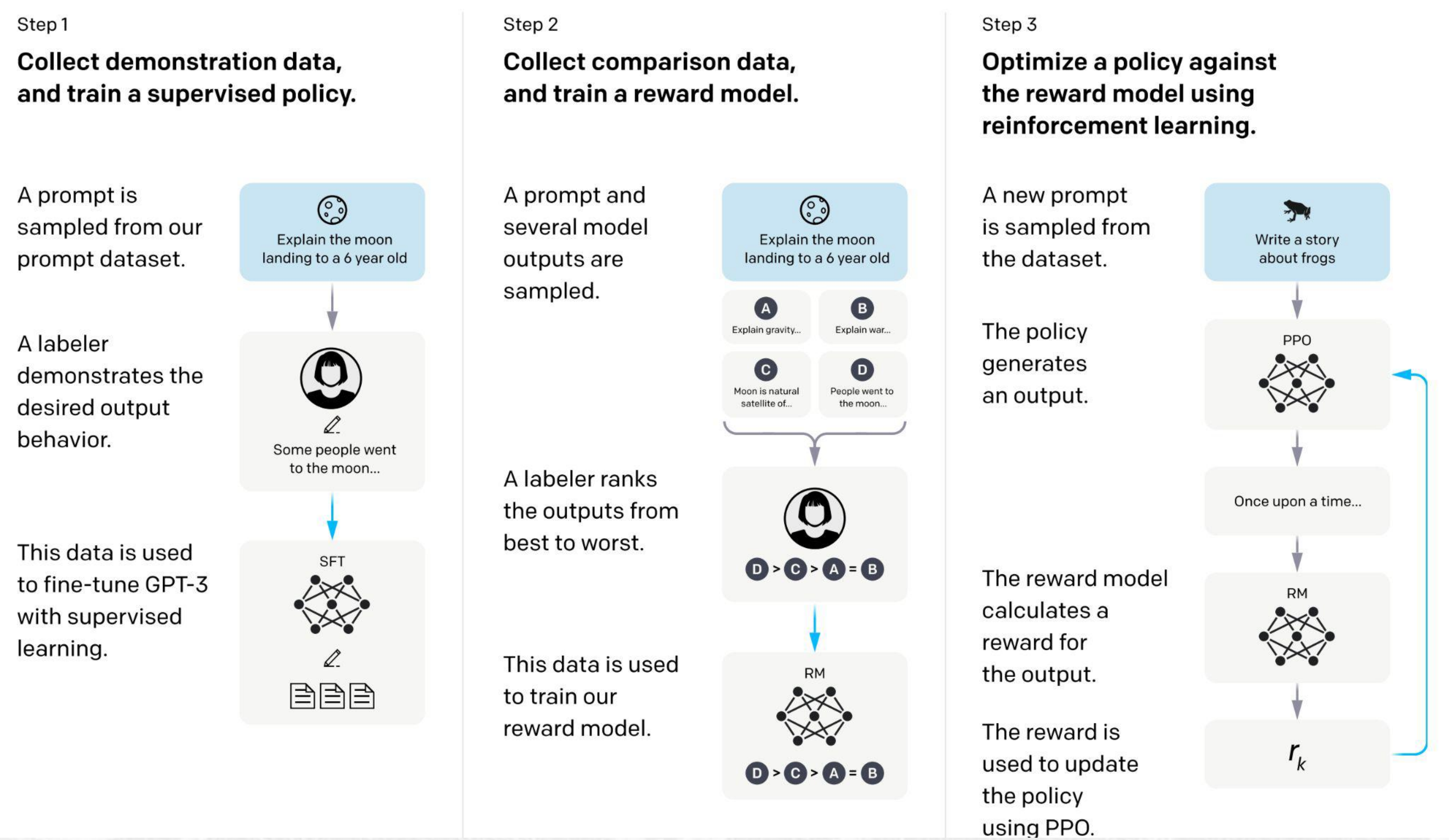


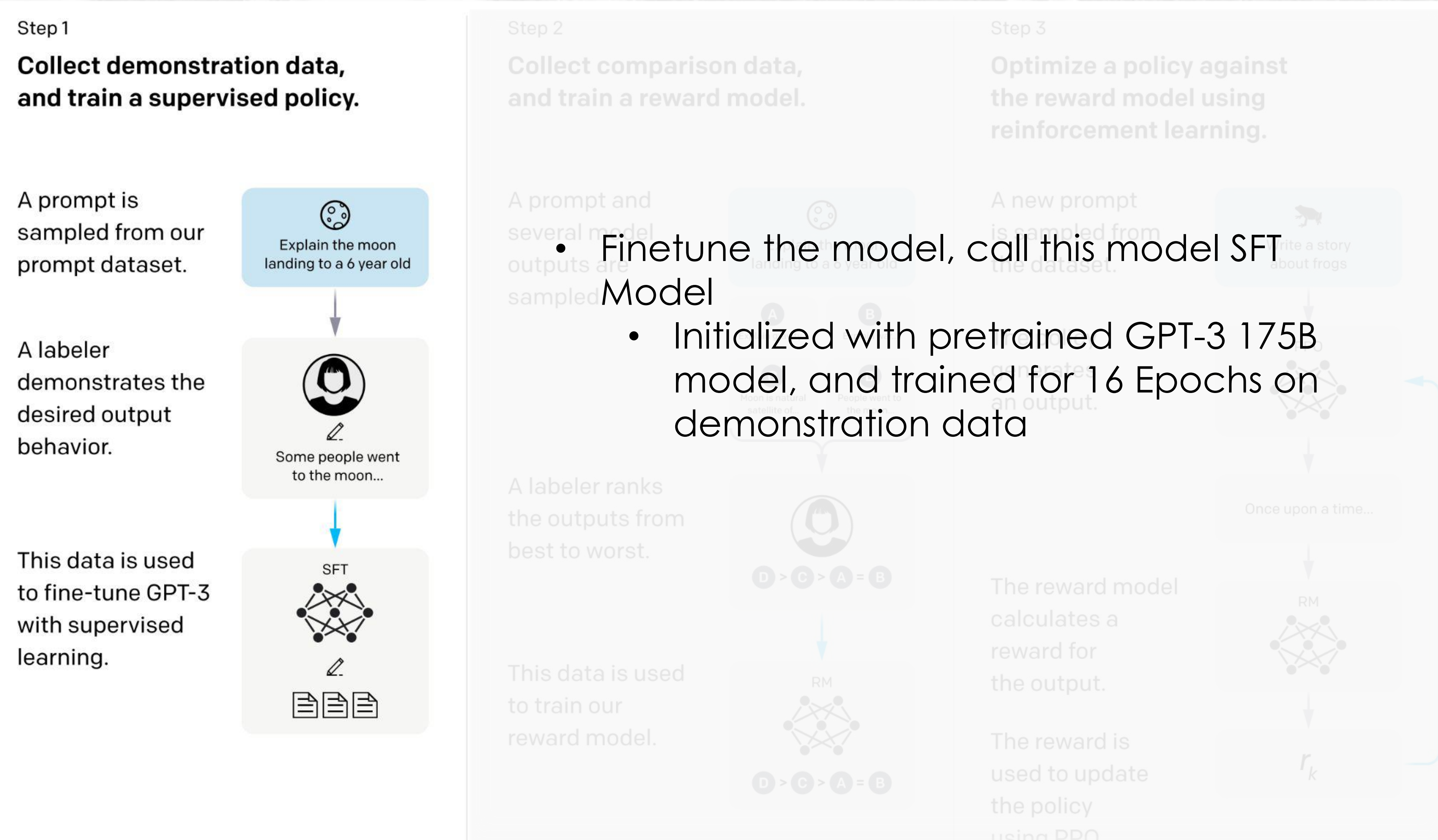
Use-case	Prompt
Brainstorming	List five ideas for how to regain enthusiasm for my career
Generation	Write a short story where a bear goes to the beach, makes friends with a seal, and then returns home.
Rewrite	This is the summary of a Broadway play: """ {summary} """ This is the outline of the commercial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

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%

Number of Prompts

SFT Data		
split	source	size
train	labeler	11,295
train	customer	1,430
valid	labeler	1,550
valid	customer	103


Step 1
Collect demonstration data, and train a supervised policy.
A prompt is sampled from our prompt dataset.
Explain the moon landing to a 6 year old
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.
SFT
Step 2
Collect comparison data, and train a reward model.
A prompt and several model outputs are sampled.
Explain the moon landing to a 6 year old
A
Explain gravity...
B
Explain war...
C
Moon is natural satellite of...
D
People went to the moon...
A labeler ranks the outputs from best to worst.
D > C > A = B
This data is used to train our reward model.
RM
D > C > A = B
Step 3
Optimize a policy against the reward model using reinforcement learning.
A new prompt is sampled from the dataset.
Write a story about frogs
The policy generates an output.
PPO
Once upon a time...
The reward model calculates a reward for the output.
RM
The reward is used to update the policy using PPO.
r_k



- Finetune the model, call this model SFT Model
    - Initialized with pretrained GPT-3 175B model, and trained for 16 Epochs on demonstration data

Step 1

Collect demonstration data, and train a supervised policy.

Step 2

Collect comparison data, and train a reward model.

Step 3

Optimize a policy against the reward model using reinforcement learning.

A prompt and several model outputs are sampled.

The outputs are sampled from the SFT model

Number of Prompts

RM Data		
split	source	size
train	labeler	6,623
train	customer	26,584
valid	labeler	3,488
valid	customer	14,399

the policy using PPO.

Step 1

Collect demonstration data, and train a supervised policy.

Step 2

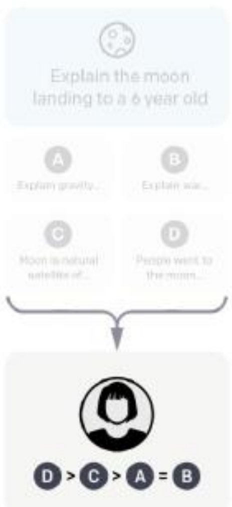
Collect comparison data, and train a reward model.

Step 3

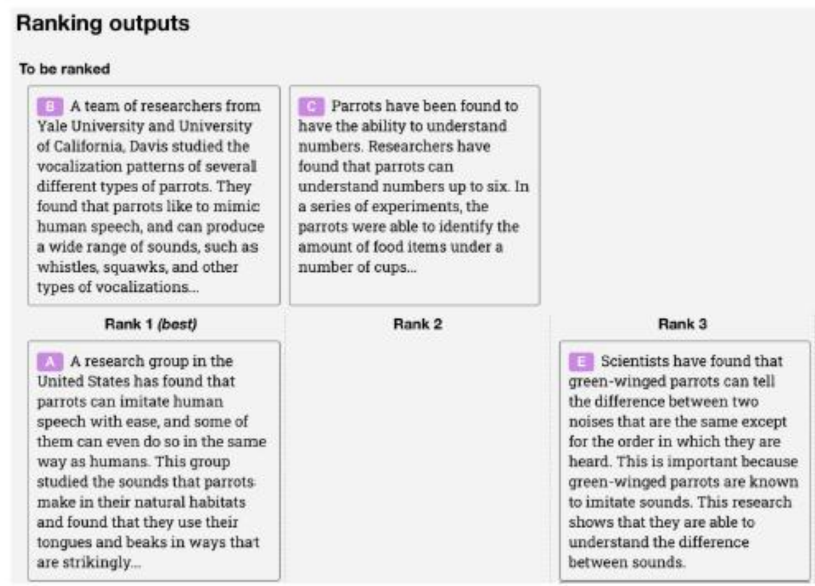
Optimize a policy against the reward model using reinforcement learning.

A prompt and several model outputs are sampled.

A labeler ranks the outputs from best to worst.



To increase data collection throughput, each user is given K = 4 to 9 outputs to rank for each prompt



D > C > A = B

used to update the policy using PPO.

Step 1

Collect demonstration data, and train a supervised policy.

Step 2

Collect comparison data, and train a reward model.

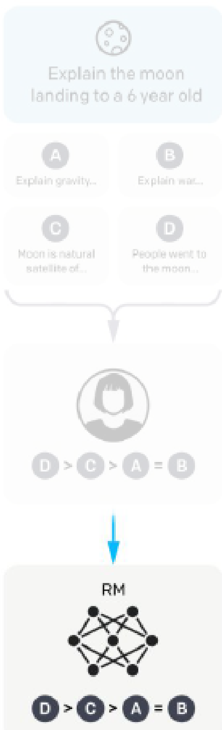
Step 3

Optimize a policy against the reward model using reinforcement learning.

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.



r_θ: The reward model we are trying to optimize

x: the prompt ***y_w:*** the better completion ***y_l:*** the worse completion

$$\text{loss}\left(\theta\right) = -\frac{1}{\binom{K}{2}} E_{(x,y_w,y_l)\sim D}\left[\log\left(\sigma\left(r_\theta\left(x,y_w\right) - r_\theta\left(x,y_l\right)\right)\right)\right]$$

Small but important detail:

- Each prompt has K completions -> K choose 2 pairs to compare
- If ∀ batch we sample uniform over *every* pair (from any prompt):
  - Each completion can appear in K - 1 gradient updates
  - This can lead to overfitting
- **Solution:** sample the prompt, and then put all K choose 2 pairs from the prompt into the same batch
  - Corollary: computationally more efficient, since this only requires K forward passes through ***r_θ*** for each prompt
- This is why there is the -1/(K choose 2) normalization in loss

used to update the policy using PPO.

Step 1

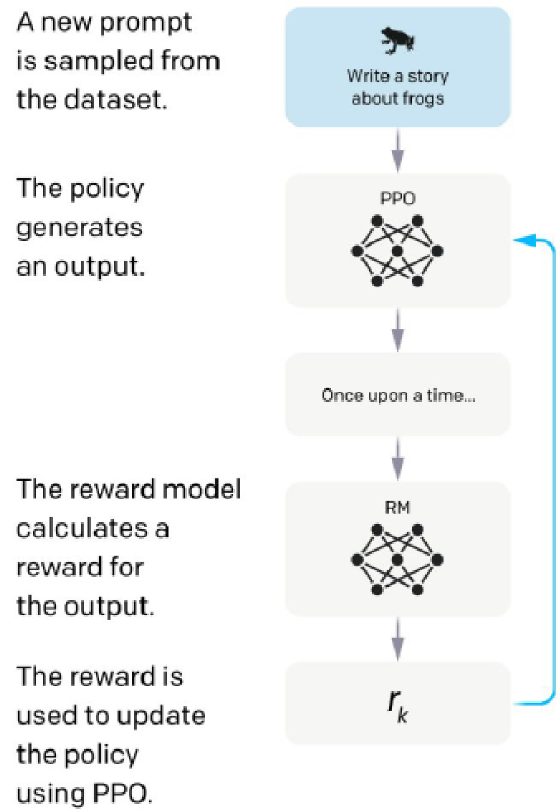
Collect demonstration data, and train a supervised policy.

Step 2

Collect comparison data, and train a reward model.

Step 3

Optimize a policy against the reward model using reinforcement learning.



Use RM to update the SFT model from step 1. Call model **PPO**

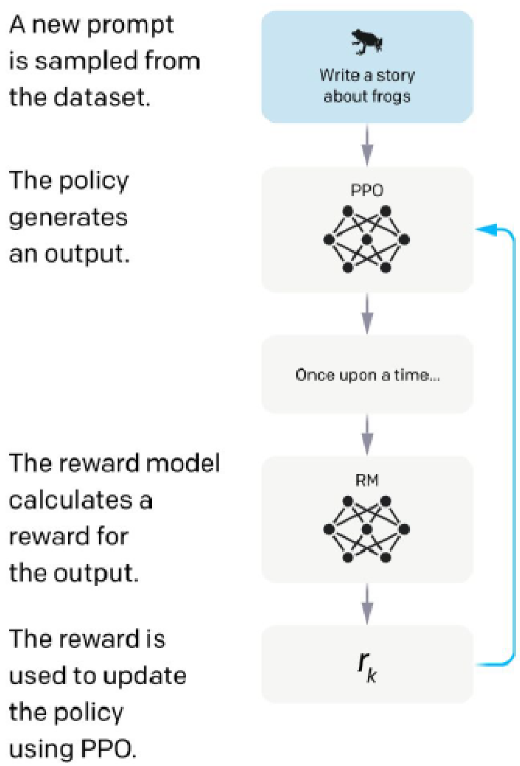
Number of Prompts

PPO Data		
split	source	size
train	customer	31,144
valid	customer	16,185

Step 3

Optimize a policy against the reward model using reinforcement learning.

(Proximal Policy Optimization)



Use RM to update the SFT model from step 1. Call model **PPO**

Two problems:

1. As RLHF is updated, its outputs become very different from what the RM was trained on -> worse reward estimates

   Solution: add a KL penalty that makes sure PPO model output does not deviate too far from SFT

Step 1

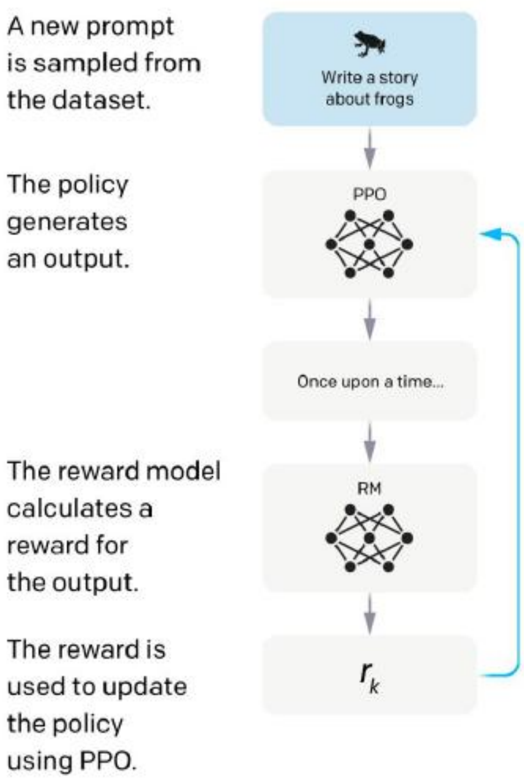
Collect demonstration data, and train a supervised policy.

Step 2

Collect comparison data, and train a reward model.

Step 3

Optimize a policy against the reward model using reinforcement learning.



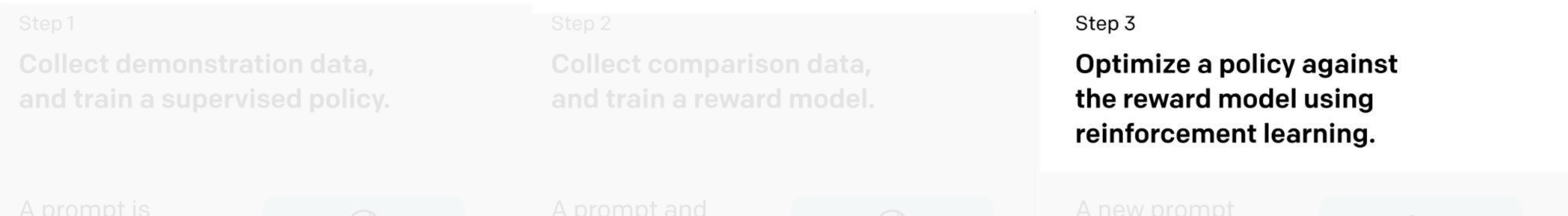
Use RM to update the SFT model from step 1. Call model **PPO**

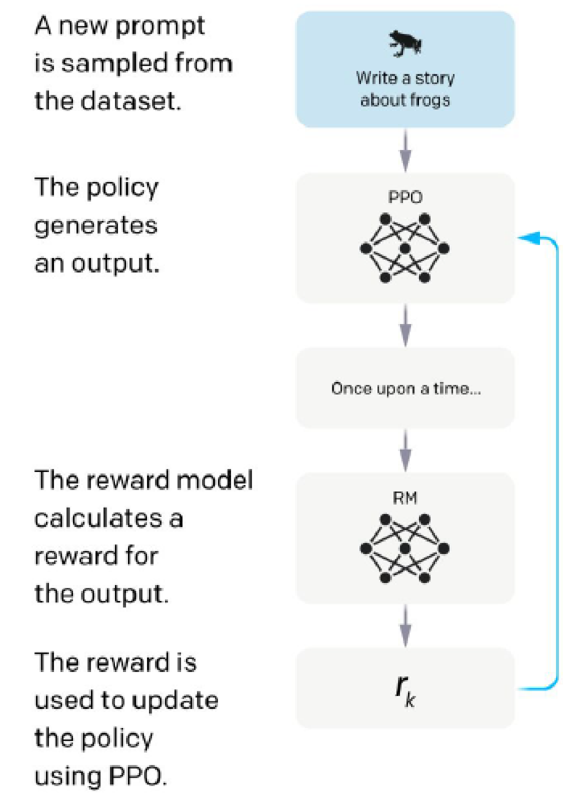
Two problems:

1. As RLHF is updated, its outputs become very different from what the RM was trained on -> worse reward estimates
   Solution: add a KL penalty that makes sure PPO model output does not deviate too far from SFT

2. Just using RL objective leads to performance degradation on many NLP tasks
   Solution: Add a auxiliary LM objective on the pretraining data. Call this variant **PPO-ptx**

(Proximal Policy Optimization with PreTraining Mixture)





Use RM to update the SFT model from step 1. Call model **PPO**

Two problems:

1. As RLHF is updated, its outputs become very different from what the RM was trained on -> worse reward estimates

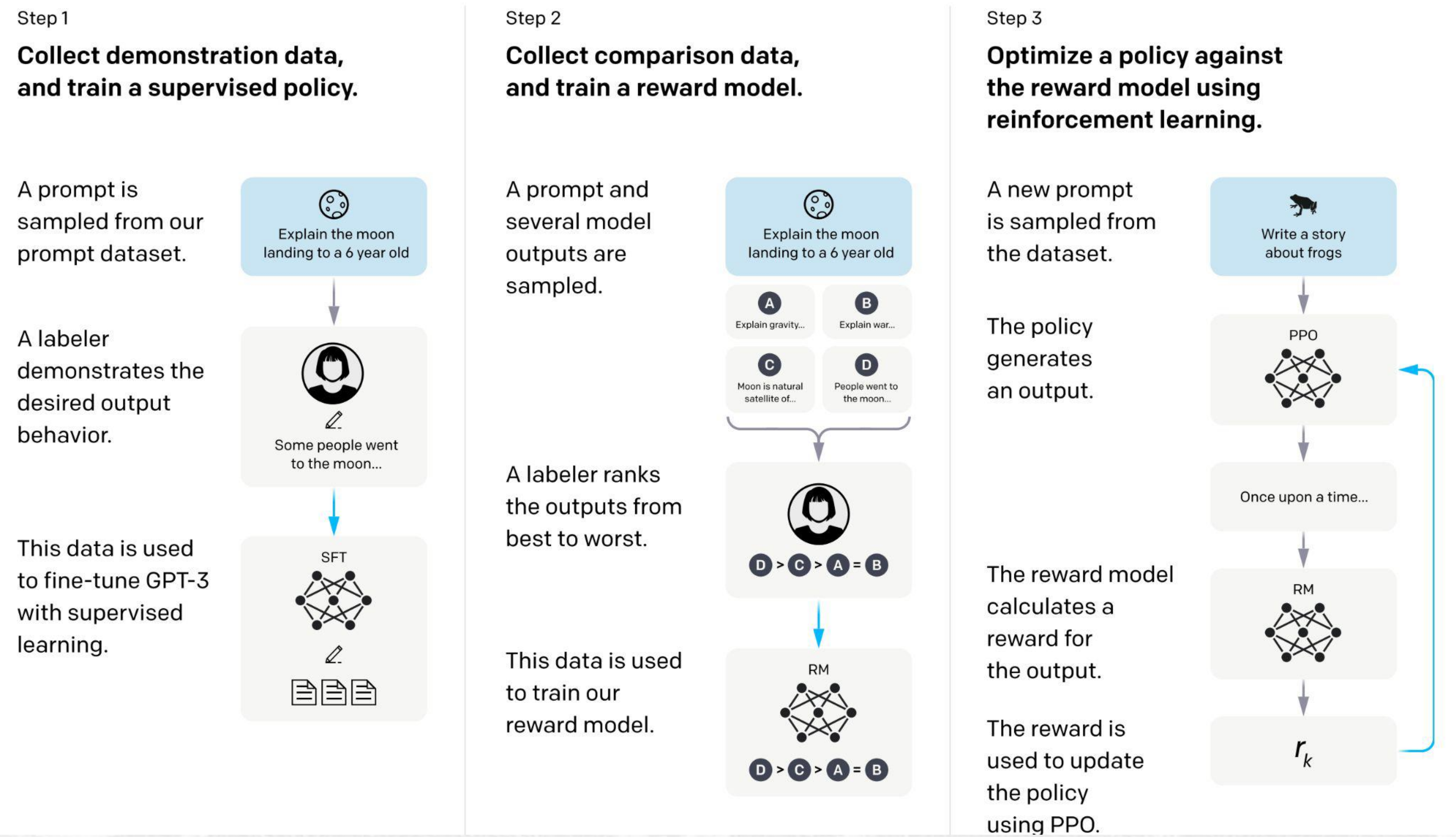
   Solution: add a KL penalty that makes sure PPO model output does not deviate too far from SFT

2. Just using RL objective leads to performance degradation on many NLP tasks

   Solution: Add a auxiliary LM objective on the pretraining data. Call this variant **PPO-ptx**

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

The model


Step 1
Collect demonstration data, and train a supervised policy.
A prompt is sampled from our prompt dataset.
Explain the moon landing to a 6 year old
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.
SFT
Step 2
Collect comparison data, and train a reward model.
A prompt and several model outputs are sampled.
Explain the moon landing to a 6 year old
A
Explain gravity...
B
Explain war...
C
Moon is natural satellite of...
D
People went to the moon...
A labeler ranks the outputs from best to worst.
D > C > A = B
This data is used to train our reward model.
RM
D > C > A = B
Step 3
Optimize a policy against the reward model using reinforcement learning.
A new prompt is sampled from the dataset.
Write a story about frogs
The policy generates an output.
PPO
Once upon a time...
The reward model calculates a reward for the output.
RM
The reward is used to update the policy using PPO.
r_k

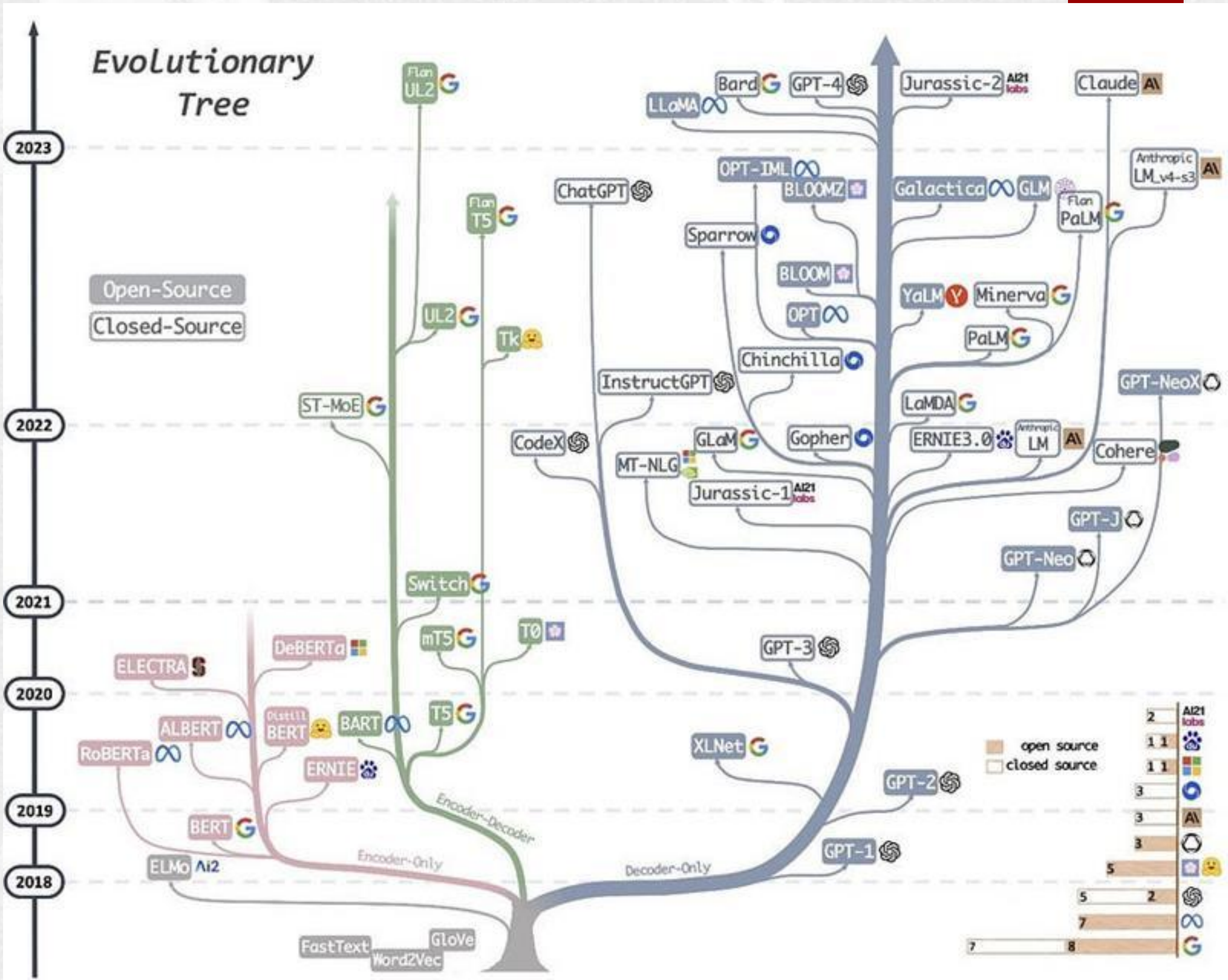
InstructGPT: model summary

- 1. **SFT: Supervised Fine-Tuning**
  - a. GPT-3 fine-tuned on human demonstrations of prompt completions
- 2. **RM: Reward Model**
  - a. Not actually used to generate anything, but used to train the PPO and PPO-ptx models
- 3. **PPO (Proximal Policy Optimization)**
  - a. SFT model further fine-tuned using RL with the RM providing the reward signal
  - b. A KL-loss is provided to prevent the PPO model from deviating far from SFT
- 4. **PPO-ptx (Proximal Policy Optimization with PreTraining Mixture)**
  - a. Identical to PPO, except with an additional auxiliary LM objective on the pretraining data

Instruction tuning papers

- Learning to summarize from human feedback
- Fine-Tuning Language Models from Human Preferences
- MemPrompt: Memory-assisted Prompt Editing with User Feedback
- LaMDA: Language Models for Dialog Application

Foundational Models



Neural writing

T

write a biblical verse in the style of the king james bible explaining how to remove a peanut butter sandwich from a VCR

And it came to pass that a man was troubled by a peanut butter sandwich, for it had been placed within his VCR, and he knew not how to remove it.

And he cried out to the Lord, saying, "Oh, Lord, how can I remove this sandwich from my VCR, for it is stuck fast and will not budge?"

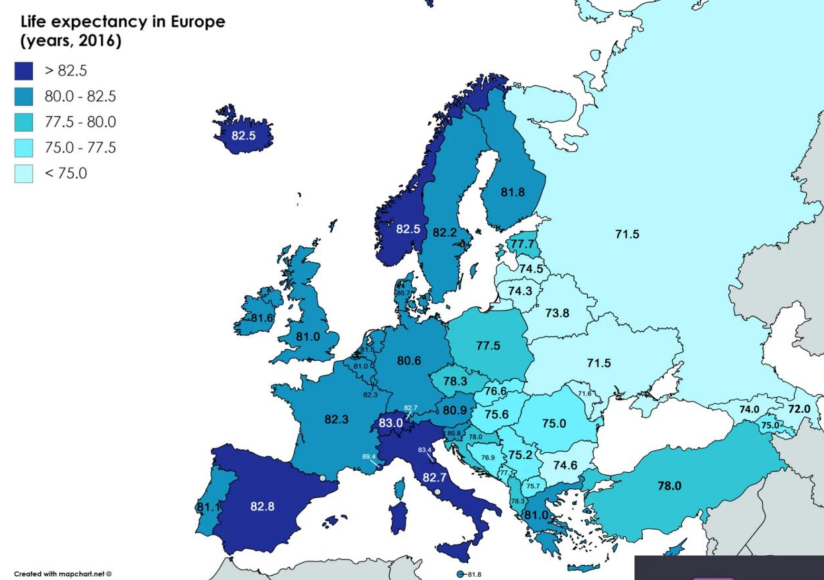
And the Lord spoke unto him, saying, "Fear not, my child, for I shall guide thy hand and show thee the way. Take thy butter knife, and carefully insert it between the sandwich and the VCR, and gently pry them apart. And with patience and perseverance, the sandwich shall be removed, and thy VCR shall be saved."

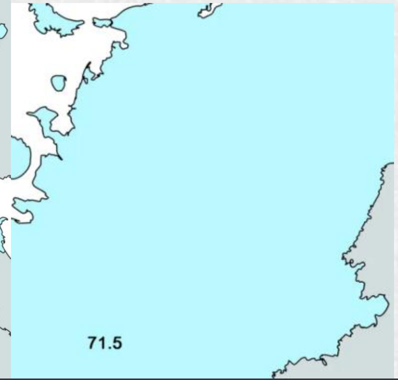
And the man did as the Lord commanded, and lo and behold, the sandwich was removed from the VCR, and the man was saved.

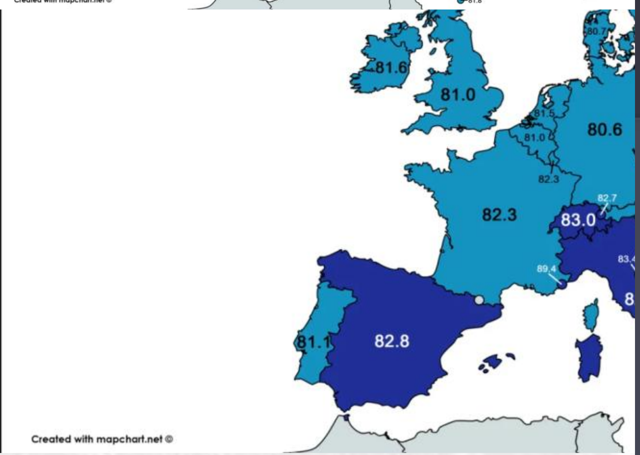
And the Lord said, "Verily I say unto thee, seek not to put thy peanut butter sandwiches in thy VCR, for it is not a suitable place for such things. Rather, keep thy sandwiches in thy refrigerator or on thy plate, where they belong."

And the man heeded the Lord's words, and from that day forth, he kept his sandwiches in their proper place, and was saved from trouble and woe. Amen.

Chat GPT4







MA

Describe the image for me:
https://en.wikipedia.org/wiki/List_of_European_countries_by_life_expectancy#/media/File:Life_expectancy_in_Europe.png

The image depicts a color-coded map of Europe, with different shades of green and yellow representing the life expectancy of each country. The map is titled "Life expectancy in Europe" and shows a gradient from lighter to darker shades of green, representing higher life expectancies.

The countries with the highest life expectancy, such as Switzerland, Spain, Italy, and France, are shown in the darkest shades of green, indicating life expectancies of over 80 years. The countries with lower life expectancies, such as Russia, Belarus, and Ukraine, are depicted in yellow and lighter shades of green, indicating life expectancies of under 75 years.

The map also includes a legend that provides a color-coded scale and corresponding life expectancy ranges, allowing viewers to easily interpret the data. Overall, the image provides a visual representation of the variation in life expectancy across different European countries, highlighting the disparities between countries and regions.

Towards AI chats
bibliography

- (Chain-of-Thought Prompting Elicits Reasoning in LLMs, Wei et al., 2023)
- (FLAN) WEI, Jason, et al. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.
- (Ziegler et al., 2021) Fine-Tuning Language Models from Human Preferences
- Madaan et al., EMNLP 2022, MemPrompt: Memory-assisted Prompt Editing with User Feedback
- Thoppilan et al, 2022, LaMDA: Language Models for Dialog Application
- Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, Paul F. Christiano: Learning to summarize with human feedback. NeurIPS 2022
- Training Language Models to follow instructions through Human feddback, Oyuang et al., 2022