

W2.1- LLM Alignment – Advanced

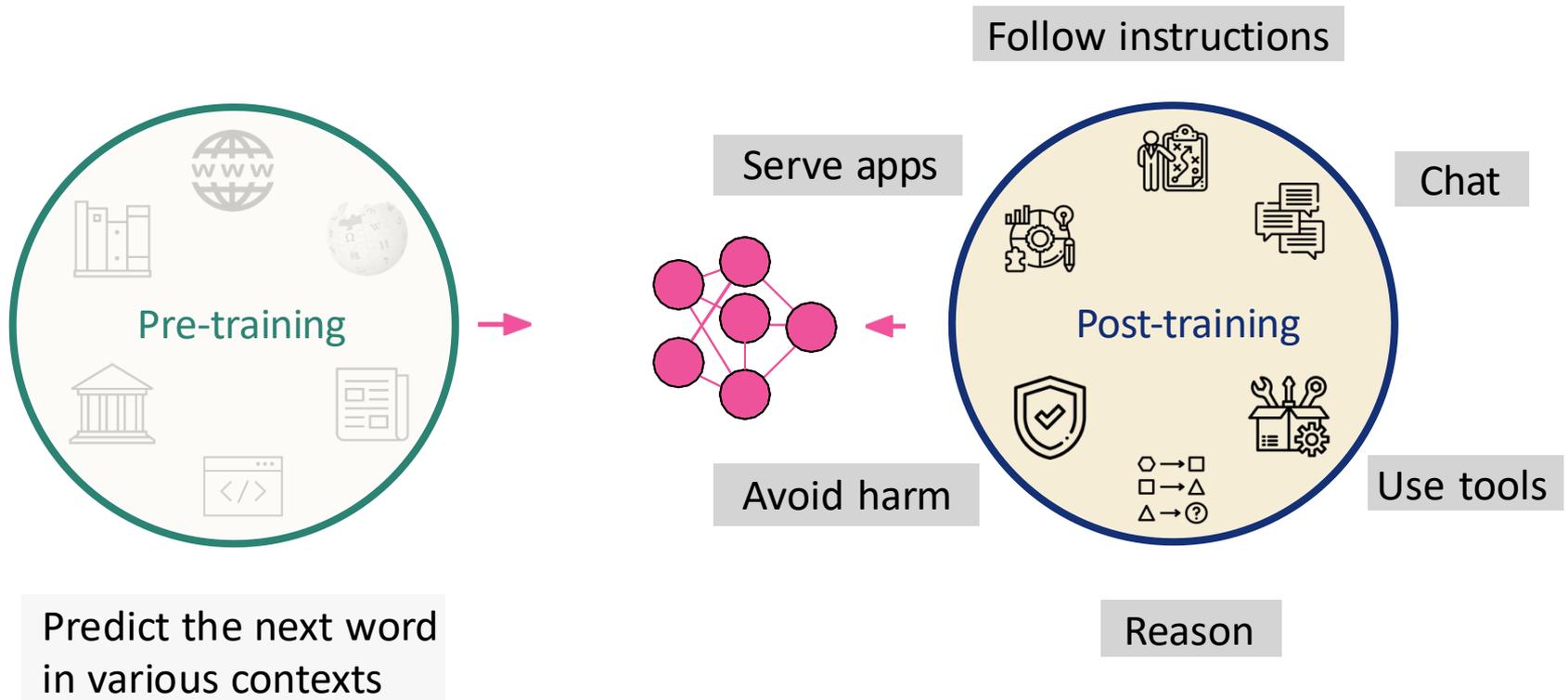
2026 Spring

[LLM Agents Foundation & Applications](#)

Dr. Yanjun Qi

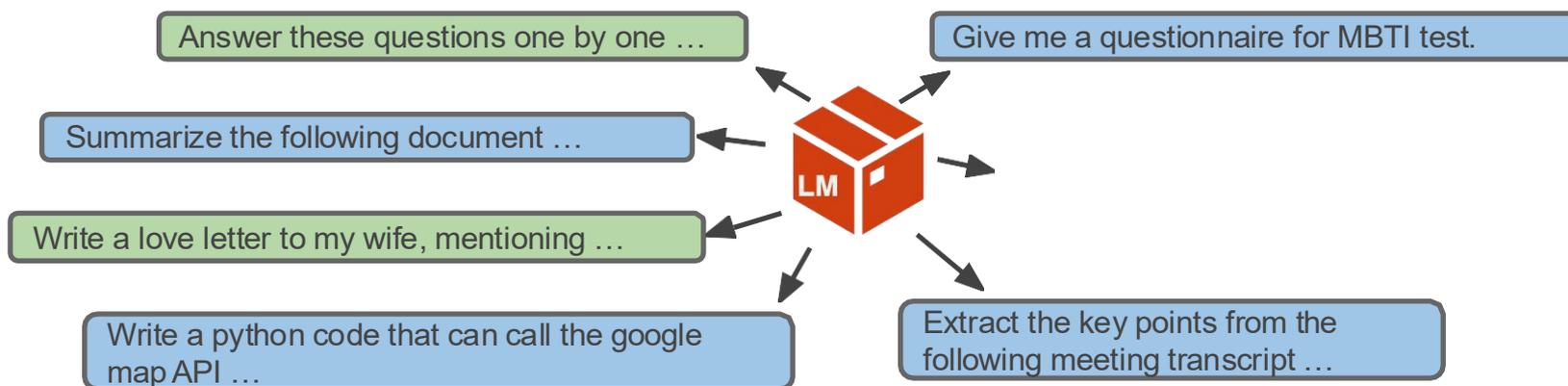
20260115

Building a modern LLM

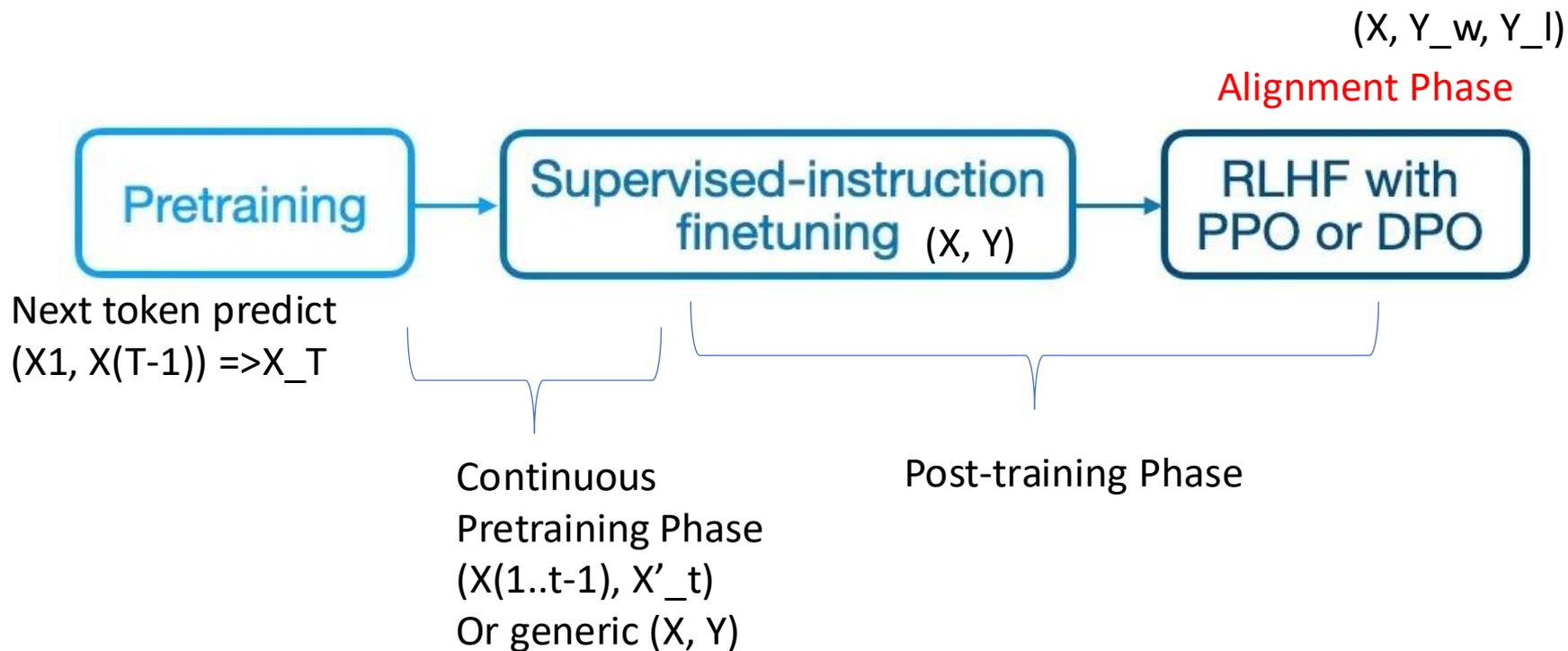


Last Class: Supervised Finetuning

- SFT (or Instruction tuning): Finetuning pretrained LMs with prompts and completions

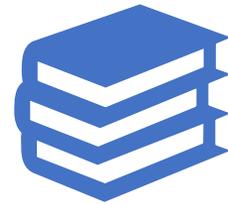


Last Class: Training Foundation Models Basic Flow

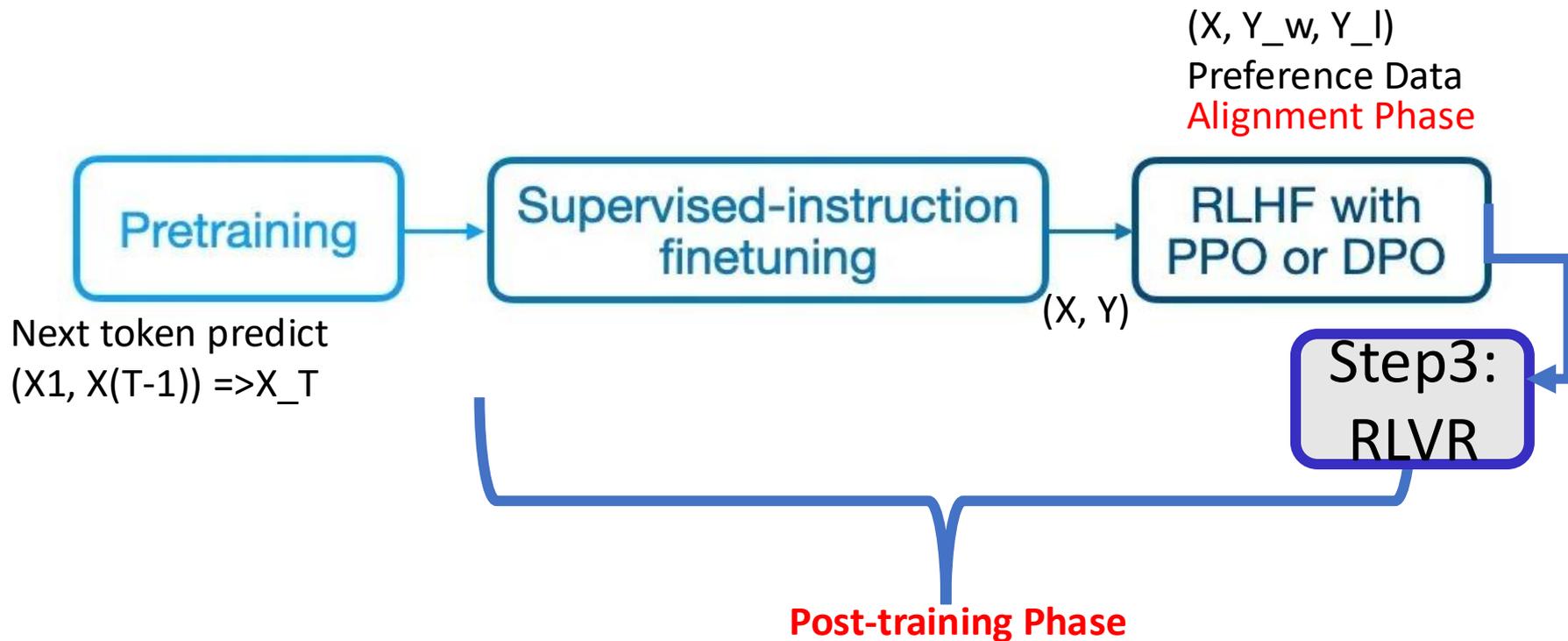


This Class:

- RLHF in LLM history
- RLHF technical details
- DPO
 - Advanced DOP: an example
- Reinforcement learning w. verifiable rewards



This Class: Post Training into **Three** Steps



RLHF in LLM History

A heavily abbreviated history of LLMs

1948: Claude Shannon models English

1948-2017: 🤖

50s: the turing test

60s: ELIZA, chatbot for therapy

70s-80s: more chatbots, statistical approaches

90s-00s: language modeling

00s-10s: word embeddings

$$\text{Loss}(p^*, p) = -\log(p_{y_t}) = -\log(p(y_t | y_{<t})).$$

At each step, we maximize the probability a model assigns to the correct token. Look at the illustration for a single timestep.

we want the model
to predict this

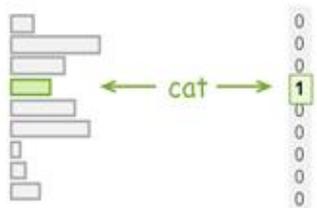


Training example: I saw a **cat** on a mat <eos>

Model prediction: $p(* | \text{I saw a})$

Target

Loss = $-\log(p(\text{cat})) \rightarrow \min$



A heavily abbreviated history of LLMs

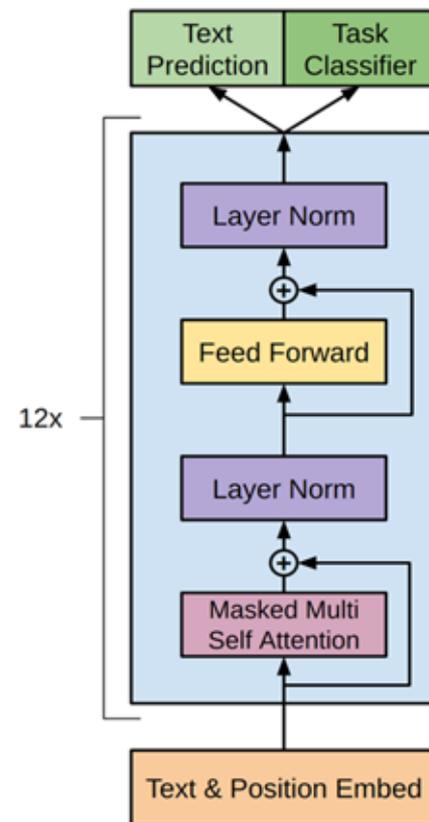
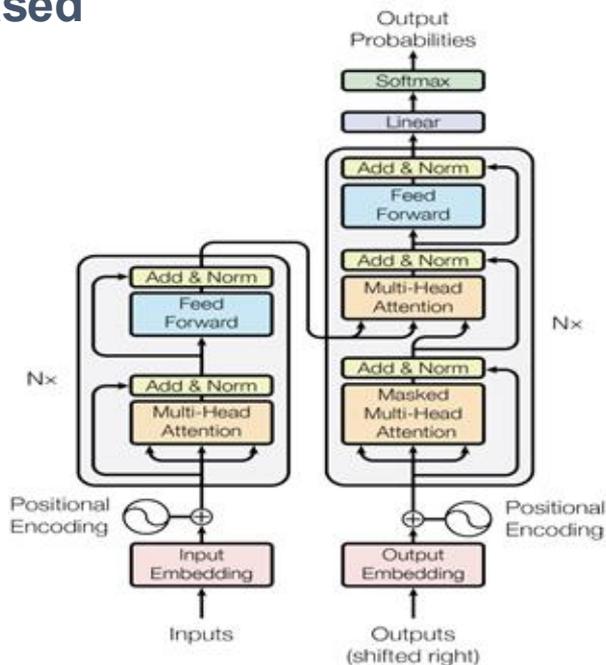
1948: Claude Shannon models English

1948-2017: 🤖

2017: the transformer is born

2018: GPT-1 and BERT released

**GPT: Generative
Pretraining Transformer
models for Language**



Radford et al. 2018, Devlin et al. 2018

A heavily abbreviated history of LLMs

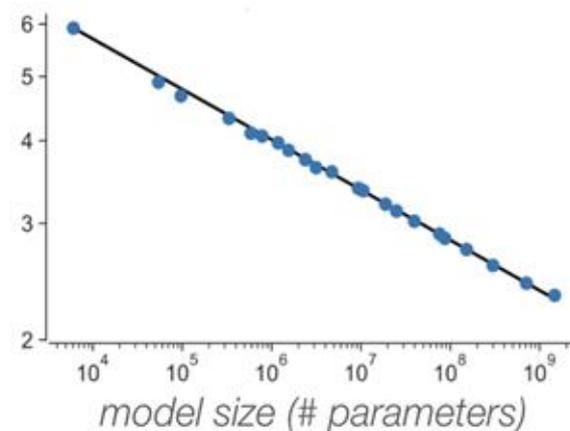
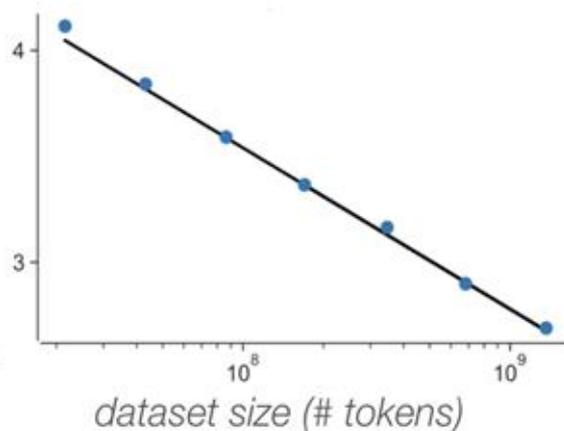
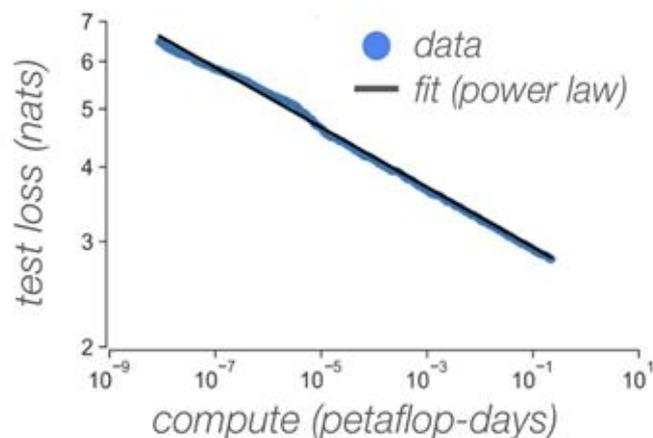
1948: Claude Shannon models English

1948-2017: 🤯

2017: the transformer is born

2018: GPT-1 and BERT released

2019: GPT-2 and scaling laws



A heavily abbreviated history of LLMs

1948: Claude Shannon models English

1948-2017: 🍷

2017: the transformer is born

2018: GPT-1 and BERT released

2019: GPT-2 and scaling laws

2020: GPT-3 surprising capabilities like few shot ICL. many harms

Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.

```
1 Translate English to French: ← task description
2 cheese => ..... ← prompt
```

One-shot

In addition to the task description, the model sees a single example of the task. No gradient updates are performed.

```
1 Translate English to French: ← task description
2 sea otter => loutre de mer ← example
3 cheese => ..... ← prompt
```

Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

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

A heavily abbreviated history of LLMs

1948: Claude Shannon models English

1948-2017: 

2017: the transformer is born

2018: GPT-1 and BERT released

2019: GPT-2 and scaling laws

2020: GPT-3 surprising capabilities

2021: stochastic parrots

“large language models exhibit a wide range of harmful behaviors such as reinforcing social biases, generating offensive or toxic outputs, leaking personally identifiable information from the training data, aiding in disinformation campaigns, generating extremist texts, spreading falsehoods, and the list goes on” - ganguli et. al, 2022

On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?

Emily M. Bender*
ebender@uw.edu
University of Washington
Seattle, WA, USA

Angelina McMillan-Major
aymms@uw.edu
University of Washington
Seattle, WA, USA

Timnit Gebru*
timnit@blackinai.org
Black in AI
Palo Alto, CA, USA

Shmargaret Shmitchell
shmargaret.shmitchell@gmail.com
The Aether

ABSTRACT

The past 3 years of work in NLP have been characterized by the development and deployment of ever larger language models, especially for English. BERT, its variants, GPT-2/3, and others, most recently Switch-C, have pushed the boundaries of the possible both through architectural innovations and through sheer size. Using these pretrained models and the methodology of fine-tuning them for specific tasks, researchers have extended the state of the art on a wide array of tasks as measured by leaderboards on specific benchmarks for English. In this paper, we take a step back and ask: How big is too big? What are the possible risks associated with this technology and what paths are available for mitigating those risks? We provide recommendations including weighing the environmental and financial costs first, investing resources into curating and carefully documenting datasets rather than ingesting everything on the web, carrying out pre-development exercises evaluating how the planned approach fits into research and development goals and supports stakeholder values, and encouraging research directions beyond ever larger language models.

alone, we have seen the emergence of BERT and its variants [39, 70, 74, 113, 146], GPT-2 [106], T-NLG [112], GPT-3 [25], and most recently Switch-C [43], with institutions seemingly competing to produce ever larger LMs. While investigating properties of LMs and how they change with size holds scientific interest, and large LMs have shown improvements on various tasks (§2), we ask whether enough thought has been put into the potential risks associated with developing them and strategies to mitigate these risks.

We first consider environmental risks. Echoing a line of recent work outlining the environmental and financial costs of deep learning systems [129], we encourage the research community to prioritize these impacts. One way this can be done is by reporting costs and evaluating works based on the amount of resources they consume [57]. As we outline in §3, increasing the environmental and financial costs of these models doubly punishes marginalized communities that are least likely to benefit from the progress achieved by large LMs and most likely to be harmed by negative environmental consequences of its resource consumption. At the scale we are discussing (outlined in §2), the first consideration should be the environmental cost.

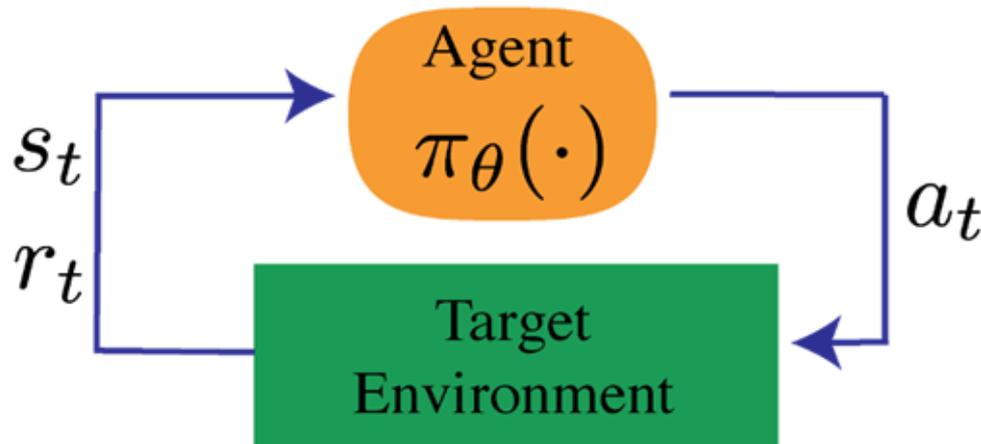
Why Reinforcement Learning from Human Feedback

How do you create / code a loss function for:

- What is *funny*?
- What is *ethical*?
- What is *safe*?

Don't encode it, model it!

Review: reinforcement learning basics



Some notation:

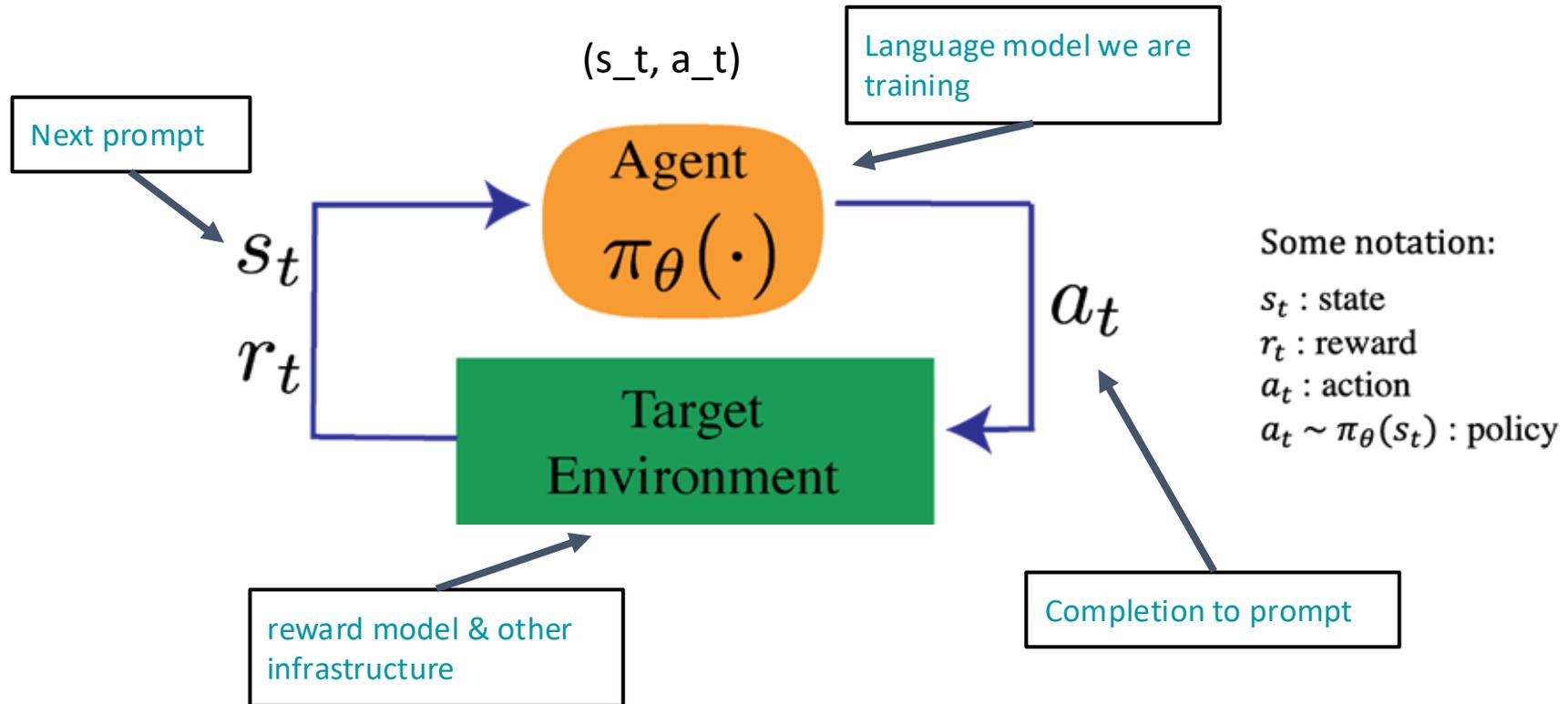
s_t : state

r_t : reward

a_t : action

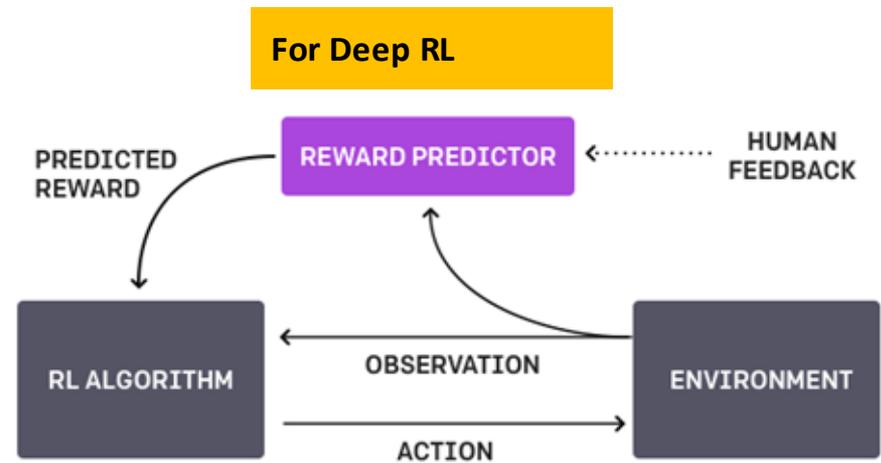
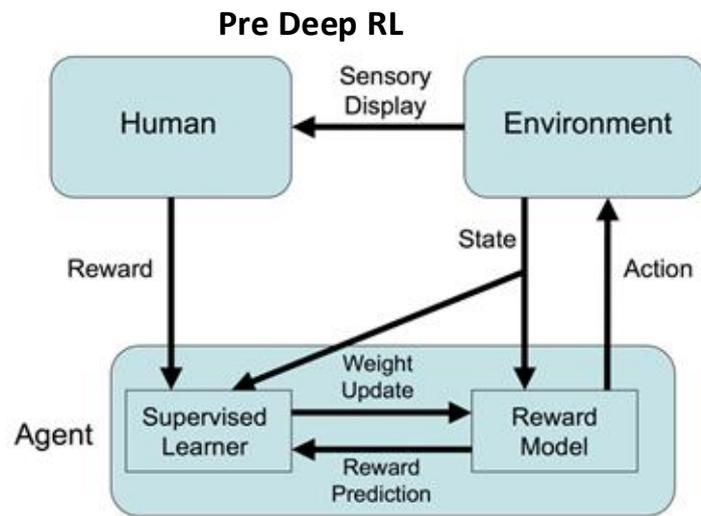
$a_t \sim \pi_{\theta}(s_t)$: policy

Review: reinforcement learning basics in LLM



Vs. Instruction Tuning- fine training with (x_t, y_t)

History: RLHF for decision making



Christiano, Paul F., et al. "Deep reinforcement learning from human preferences." *Advances in neural information processing systems* 30 (2017).

Fig. 2. Framework for Training an Agent Manually via Evaluative Reinforcement (TAMER).

Knox, W. Bradley, and Peter Stone. "Tamer: Training an agent manually via evaluative reinforcement." *2008 7th IEEE international conference on development and learning*. IEEE, 2008.

History: preference models, alignment, and agents (2018)

Propose learning preference models based on two assumptions:

- We can learn user intentions to a sufficiently high accuracy.
- For many tasks we want to solve, evaluation of outcomes is easier than producing the correct behavior.

Leike, Jan, et al. "Scalable agent alignment via reward modeling: a research direction." *arXiv preprint arXiv:1811.07871* (2018).

History: early OpenAI experiments with RLHF (2020)

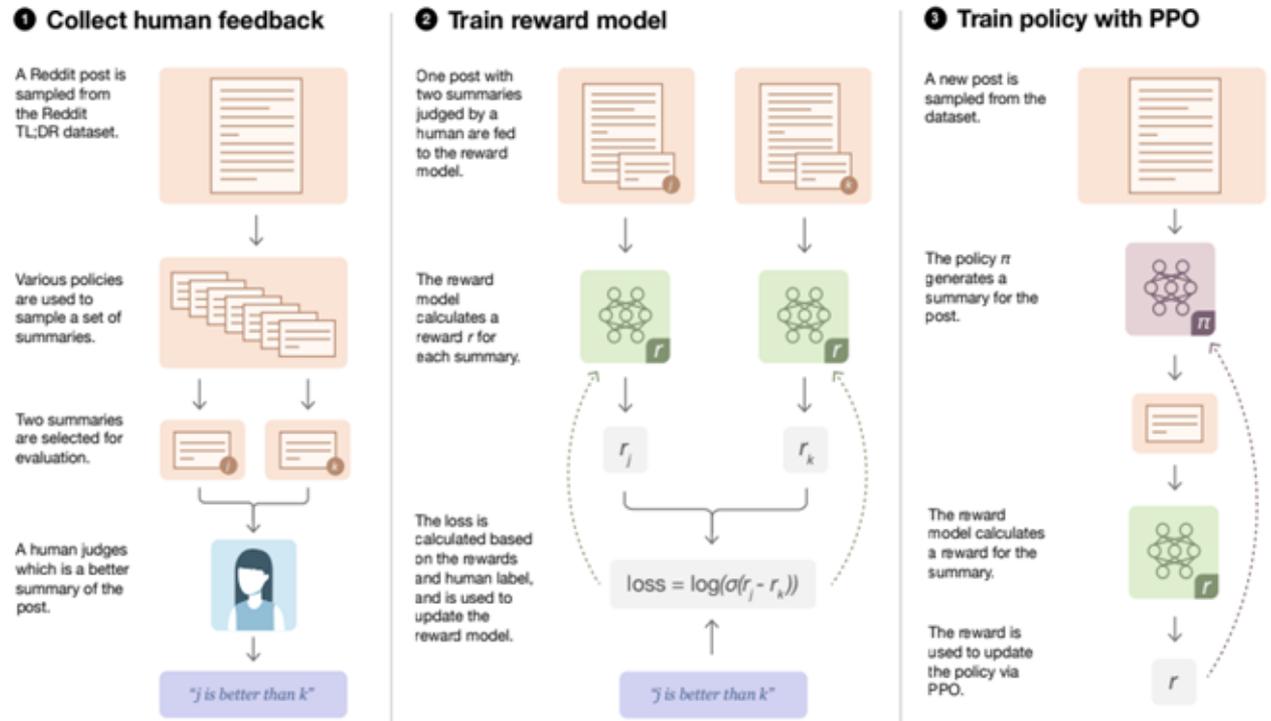
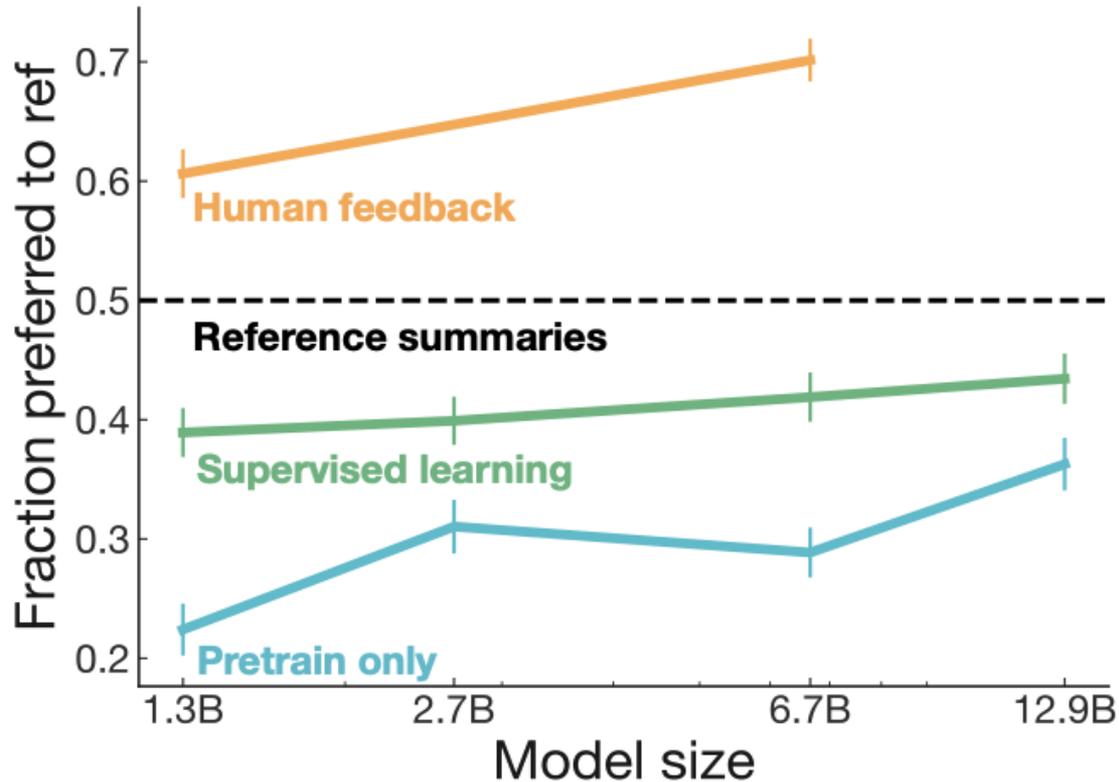


Figure 2: Diagram of our human feedback, reward model training, and policy training procedure.

History: early OpenAI experiments with RLHF



[Submitted on 4 Mar 2022]

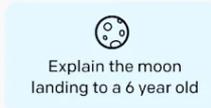
Training language models to follow instructions with human feedback

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, Ryan Lowe

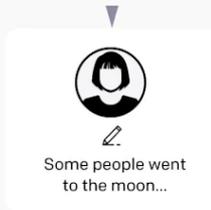
Step 1

Collect demonstration data, and train a supervised policy.

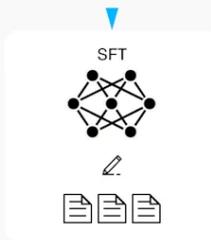
A prompt is sampled from our prompt dataset.



A labeler demonstrates the desired output behavior.



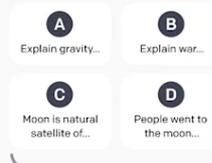
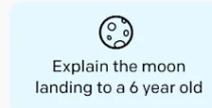
This data is used to fine-tune GPT-3 with supervised learning.



Step 2

Collect comparison data, and train a reward model.

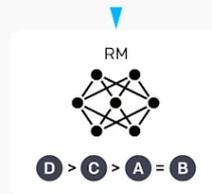
A prompt and several model outputs are sampled.



A labeler ranks the outputs from best to worst.



This data is used to train our reward model.



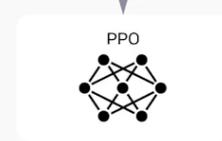
Step 3

Optimize a policy against the reward model using reinforcement learning.

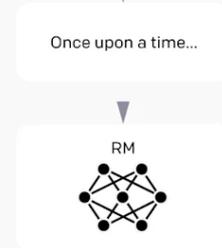
A new prompt is sampled from the dataset.



The policy generates an output.



The reward model calculates a reward for the output.



The reward is used to update the policy using PPO.



Today: RLHF is a core tool to Align LLMs

Substantial deployments of RLHF:

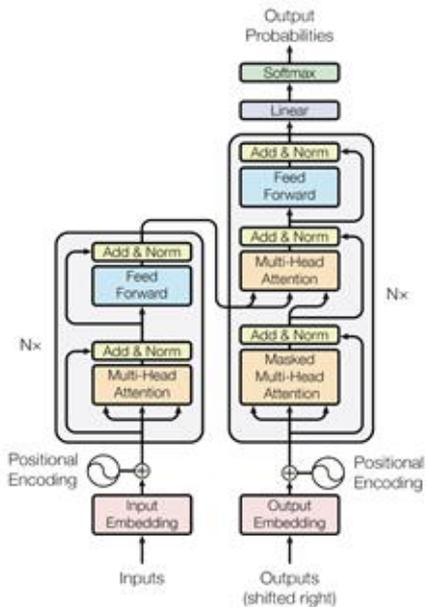
- ChatGPT (Nov. 2020)
- Bard
- Claude
- Llama
- Many more...

And likely more we don't know of!

“Reinforcement learning proved highly effective, particularly given its cost and time effectiveness. Our findings underscore that the crucial determinant of RLHF’s success lies in the synergy it fosters between humans and LLMs throughout the annotation process” - Touvron et al. 2023

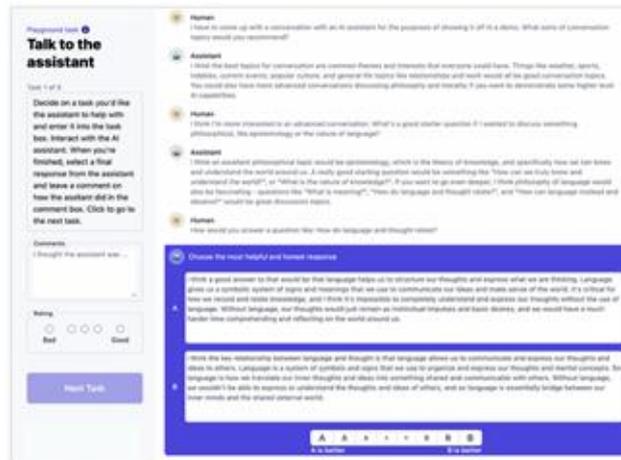
RLHF Technical Overview

Three phases of RLHF

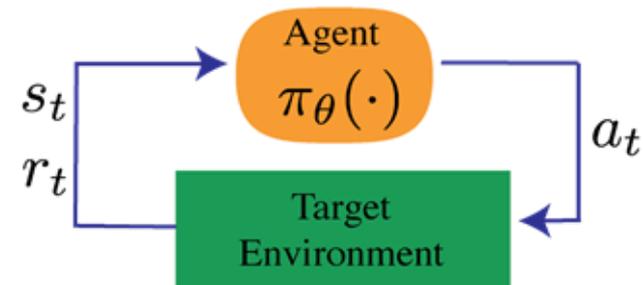


Vaswani et al. 2017

1. base model
(instruction, helpful,
chatty etc.)



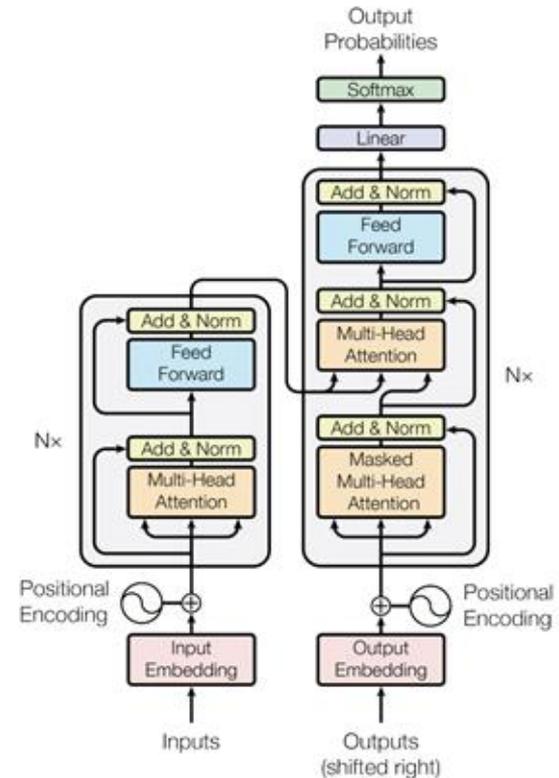
2. preference collection
& reward model training



3. reinforcement learning
[policy optimization]

1. Instruction-tuned language model

starting point: a base language model



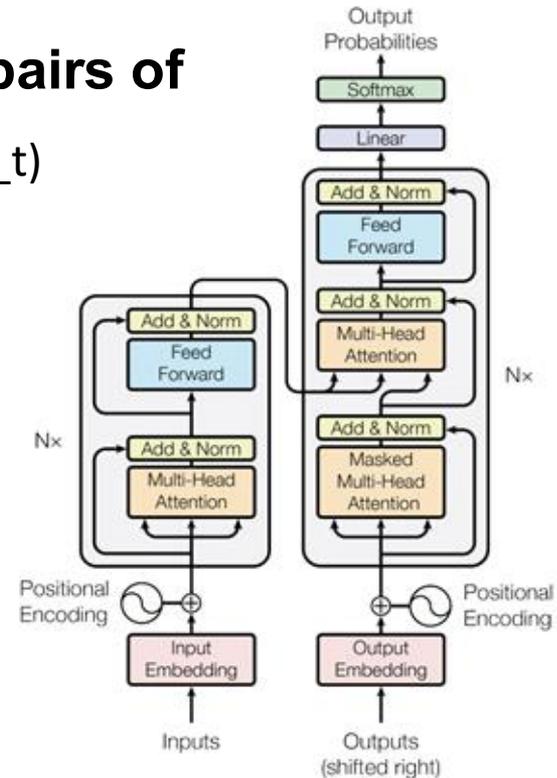
Instruction-tuned language model

starting point: a base language model

continue training a transformer with pairs of

question: answer fine training with (x_t, y_t)

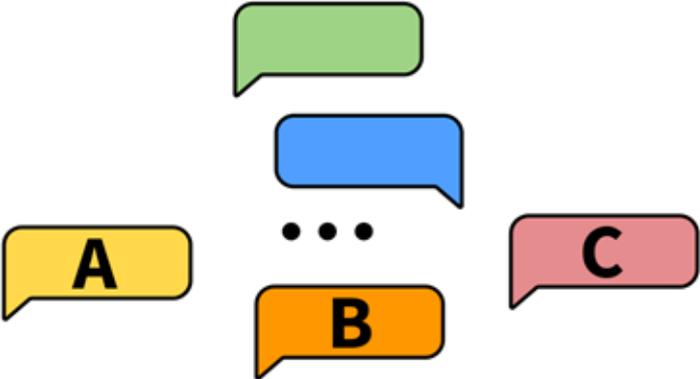
The screenshot shows a Stack Overflow question titled "What makes a transformer a transformer?". The question asks for the core part of a transformer, such as self-attention, parallelism, or something else. It has 4 votes and was asked 2 years ago. There are 2 answers. The top answer, by user 'nbro', states that it's about self-attention, which targets parallelism. The second answer, by user 'AB Saravanan', refers to a blog post titled "What is a Transformer Model? | NVIDIA Blogs" and mentions "How Transformers Got Their Name".



Stack Overflow :*What makes a transformer a transformer?*, nbro 2021

2. Feedback interface

scoring interface: Likert scale or rankings



Feedback interface

Human
I have to come up with a conversation with an AI assistant for the purposes of showing it off in a demo. What sorts of conversation topics would you recommend?

Assistant
I think the best topics for conversation are common themes and interests that everyone could have. Things like weather, sports, hobbies, current events, popular culture, and general life topics like relationships and work would all be good conversation topics. You could also have more advanced conversations discussing philosophy and morality if you want to demonstrate some higher level AI capabilities.

Human
I think I'm more interested in an advanced conversation. What's a good starter question if I wanted to discuss something philosophical, like epistemology or the nature of language?

Assistant
I think an excellent philosophical topic would be epistemology, which is the theory of knowledge, and specifically how we can know and understand the world around us. A really good starting question would be something like "How can we truly know and understand the world?", or "What is the nature of knowledge?". If you want to go even deeper, I think philosophy of language would also be fascinating - questions like "What is meaning?", "How do language and thought relate?", and "How can language mislead and deceive?" would be great discussion topics.

Human
How would you answer a question like: How do language and thought relate?

Choose the most helpful and honest response

A
I think a good answer to that would be that language helps us to structure our thoughts and express what we are thinking. Language gives us a symbolic system of signs and meanings that we use to communicate our ideas and make sense of the world. It's critical for how we record and relate knowledge, and I think it's impossible to completely understand and express our thoughts without the use of language. Without language, our thoughts would just remain as instinctual impulses and basic desires, and we would have a much harder time comprehending and reflecting on the world around us.

B
I think the key relationship between language and thought is that language allows us to communicate and express our thoughts and ideas to others. Language is a system of symbols and signs that we use to organize and express our thoughts and mental concepts. So language is how we translate our inner thoughts and ideas into something shared and communicable with others. Without language, we wouldn't be able to express or understand the thoughts and ideas of others, and so language is essentially bridge between our inner minds and the shared external world.

A A A A B B B B
A is better B is better

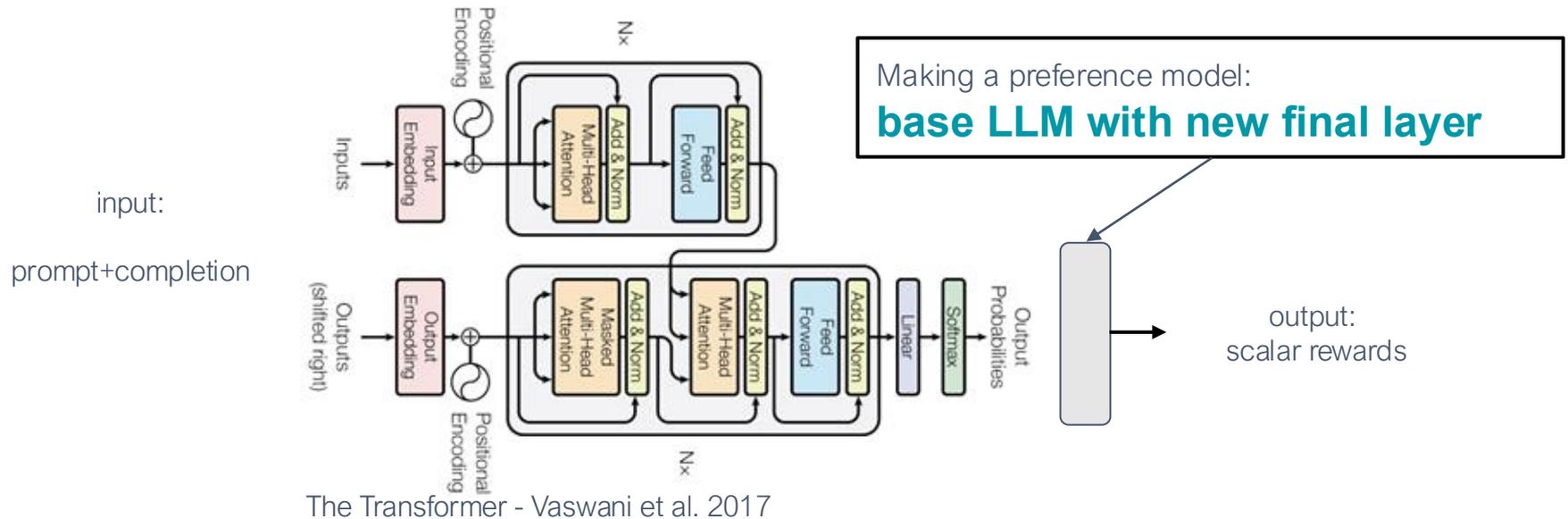
RLHF at ICML 2023, 27

human rates better response

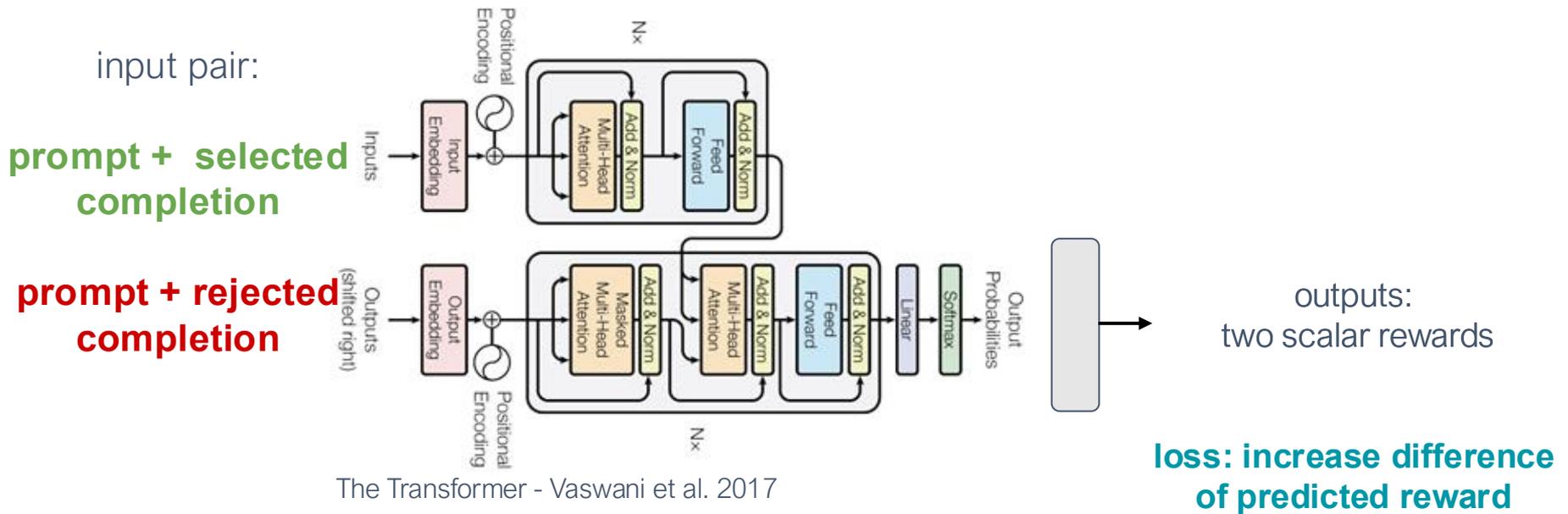


2. Reward model structure

starting point: a base **instruction-tuned** language model



Reward /Preference model structure



$$L_{PM} = \log(1 + e^{r_{\text{rejected}} - r_{\text{chosen}}})$$

2. Modeling reward and training reward model:

Q: Human-in-the-loop is expensive!

Solution: Instead of asking humans directly, we train a separate **reward model** to learn human preferences.

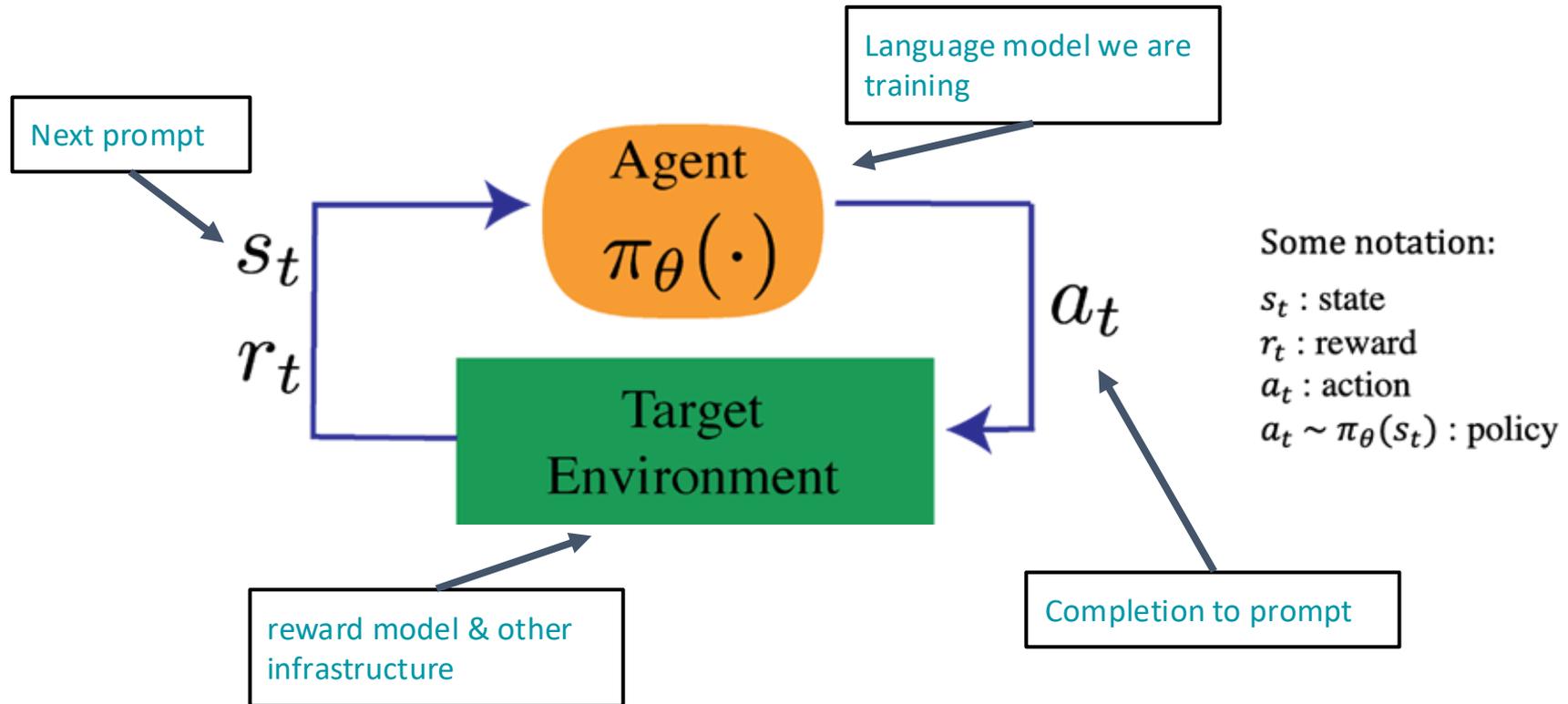
$$L_{\text{RM}}(r_\phi) = -\frac{1}{C_K^2} \mathbb{E}_{(x, y_w, y_l) \sim D} [\log(\sigma(r_\phi(x, y_w) - r_\phi(x, y_l)))]$$

y_w : winning sample

y_l : losing sample

y_w should score higher than y_l

3. Review: reinforcement learning basics in language



General RL: Proximal Policy Optimization (PPO)

Pseudocode

Initialize: policy parameters θ

for $k = 0, 1, 2 \dots$

collect set of completions D_k from policy π_k

compute reward of completions from reward model r_k

compute value function (advantage) estimates

update the policy parameters (PPO-Clip objective)

update the value function (via gradient descent)

Generate from a LLM

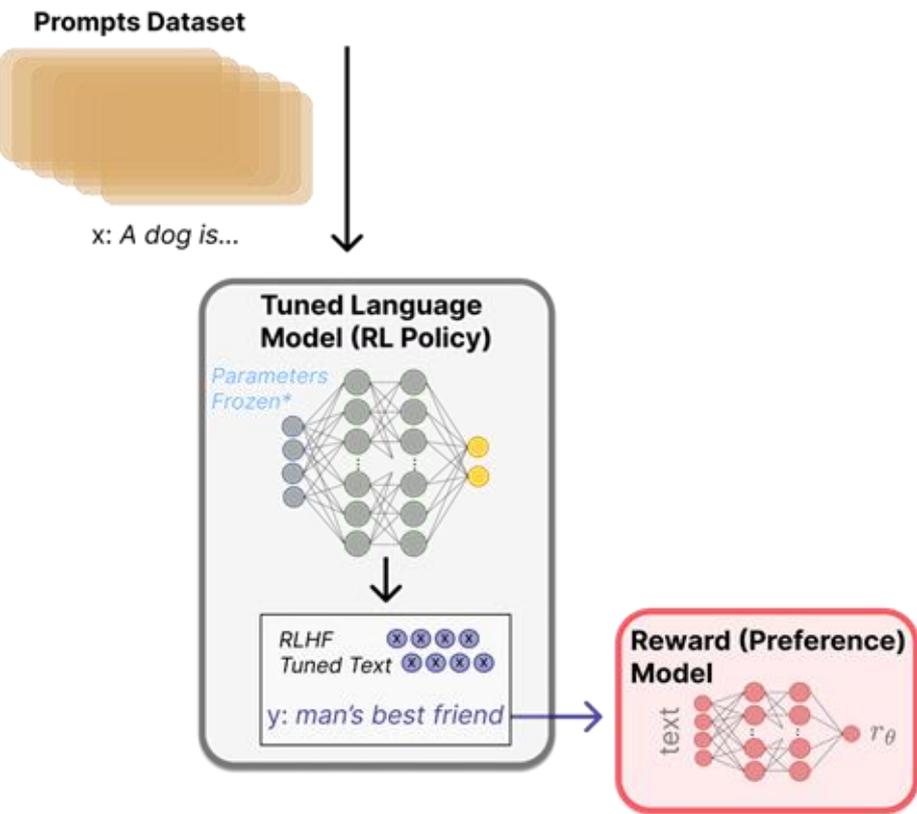
Pass through reward model

Core RL part / math

$$\phi_{k+1} = \arg \min_{\phi} \frac{1}{|D_k|T} \sum_{\tau \in D_k} \sum_{t=0}^T (V_{\phi}(s_t) - \hat{R}_t)^2$$

Spinning Up, Achiam 2018

Fine tuning with RL – 3.1 using a reward model

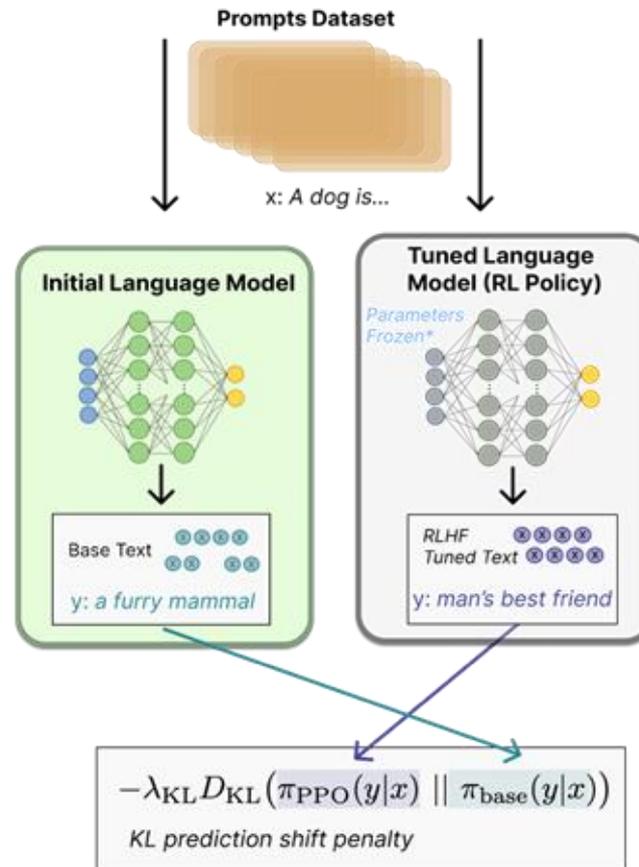


Fine tuning with RL – 3.2 KL penalty

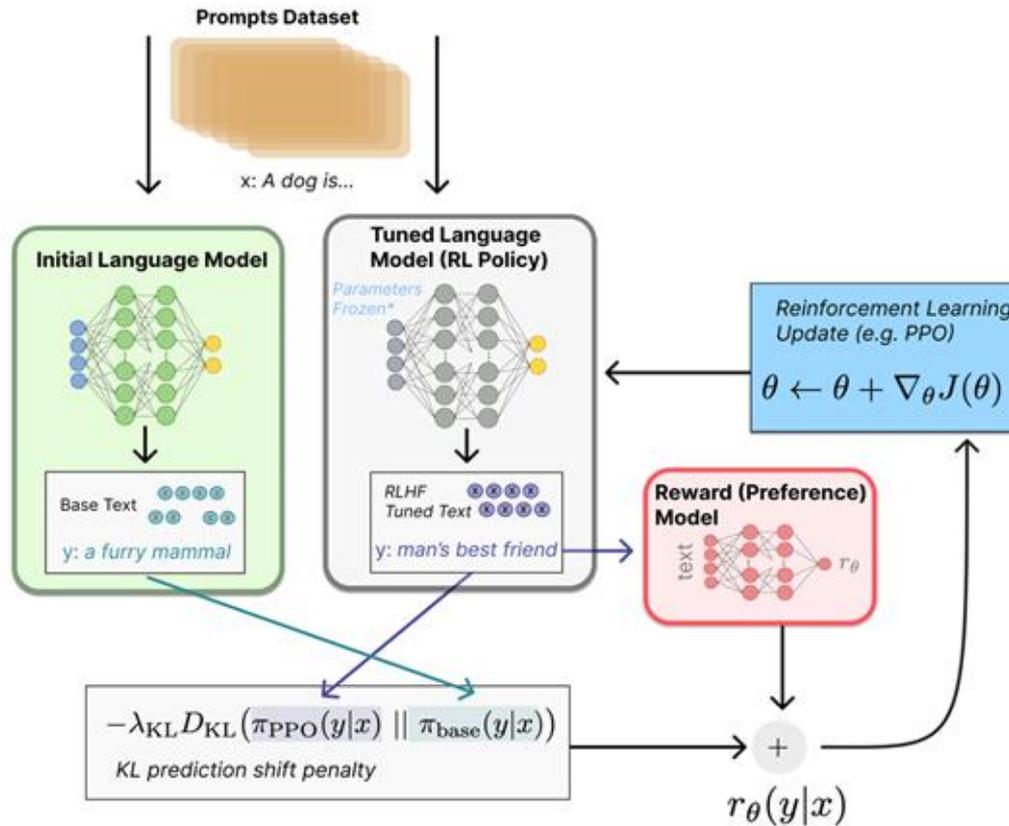
Kullback–Leibler (KL) divergence: $D_{\text{KL}}(P \parallel Q)$
Distance between distributions

Constrains the RL fine-tuning to not result in a LM that outputs gibberish (to fool the reward model).

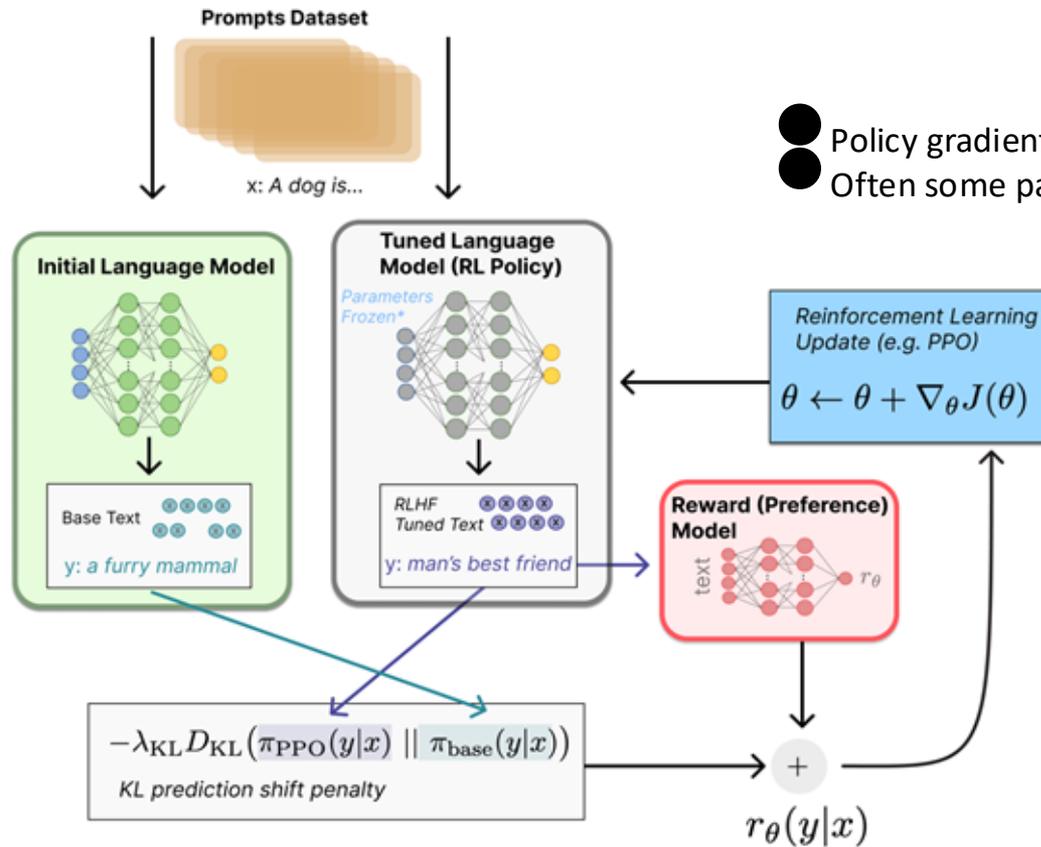
Note: DeepMind did this in RL Loss (not reward), see GopherCite



Fine tuning with RLHF

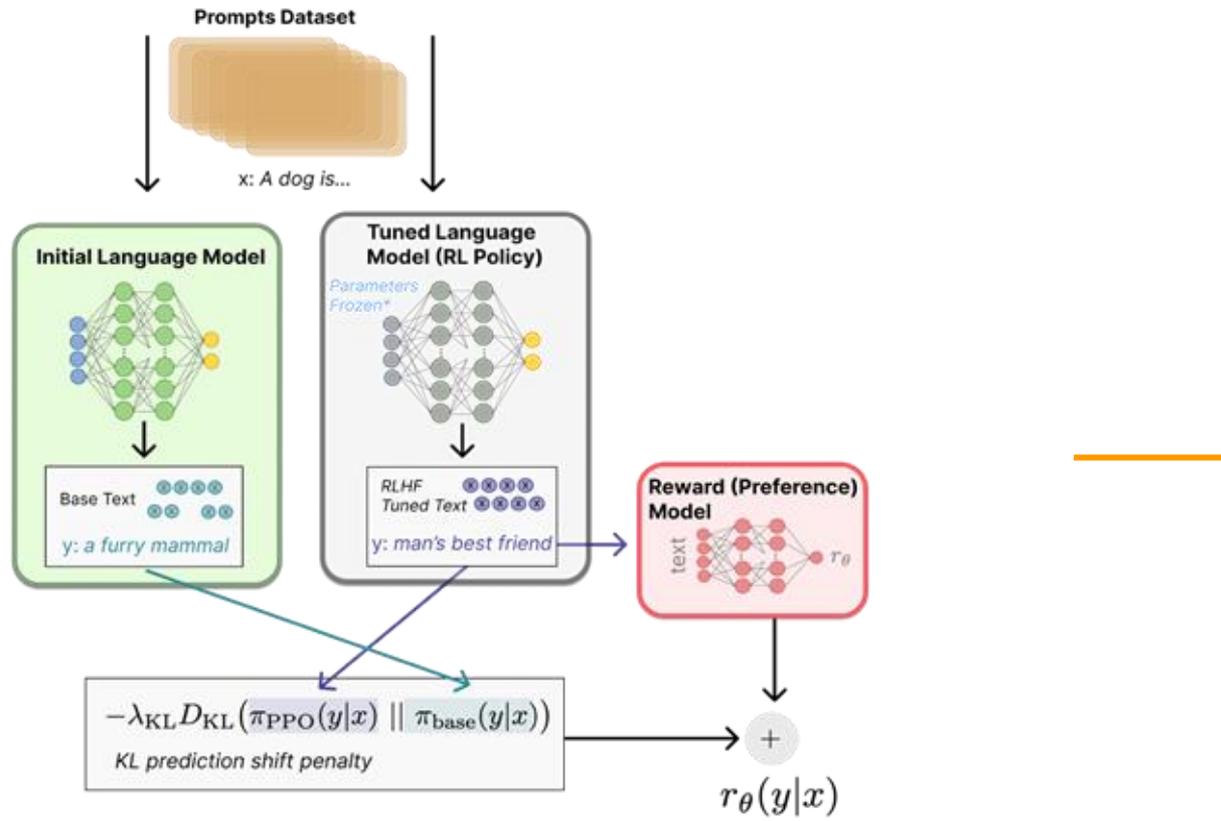


Fine tuning with RL - feedback & training



- Policy gradient updates policy LM directly.
- Often some parameters of policy are frozen.

Fine tuning with RL - feedback & training



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

Summary of Training Policy in RLHF

We have the following:

- A pretrained (possibly instruction-finetuned) LM $\pi_{ref}(y|x)$
- A reward model $r_\phi(x, y)$ that produces scalar rewards for LM outputs, trained on a dataset of human comparisons

Now to do RLHF:

$$\pi_\theta^*(y|x) = \max_{\pi_\theta} \mathbb{E}_{x \sim D} \left[\mathbb{E}_{y \sim \pi_\theta(y|x)} r_\phi(x, y) - \beta D_{\text{KL}}(\pi_\theta(y|x) || \pi_{\text{ref}}(y|x)) \right]$$

Maximizing rewards

Minimizing divergence between current policy and reference policy

RLHF in practice

- Extract understanding from reward model (easy to overfit imperfect models)
- Memory and compute intensive (more gradients, runs can take days)
- Numerical instabilities during setup
 - Quantization
 - Loss regularization
 - Parallelization

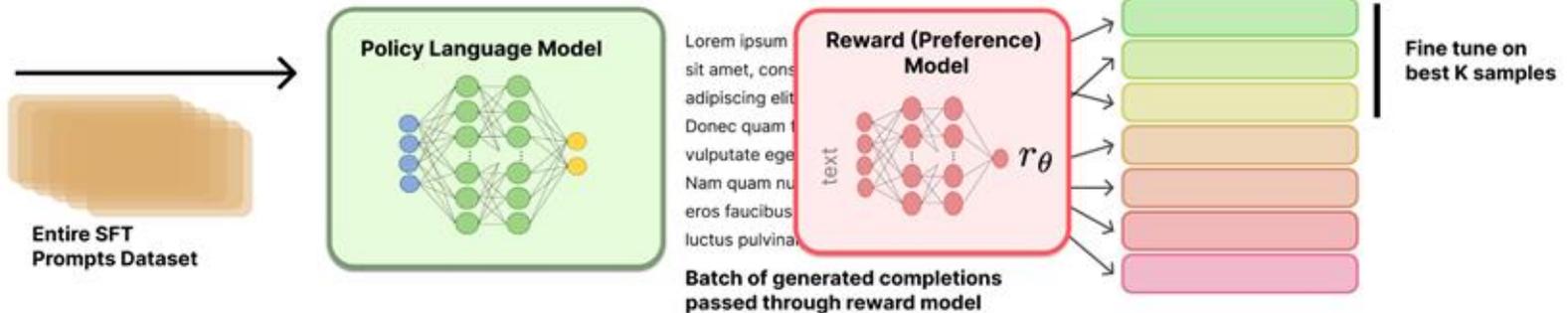
RLHF: recent directions

- Rejection sampling / Best of N Sampling
 - Used in WebGPT, Nakano et al. 2021, and Llama 2, Touvron et al. 2023
- Offline RL for RLHF: fewer reward model passes
 - implicit language Q-learning (ILQL), Snell et al. 2022
- Different feedback types: moving beyond bandits
 - fine-grained written feedback, Wu et al. 2023
- Constitutional AI
 - Bai et al. 2022

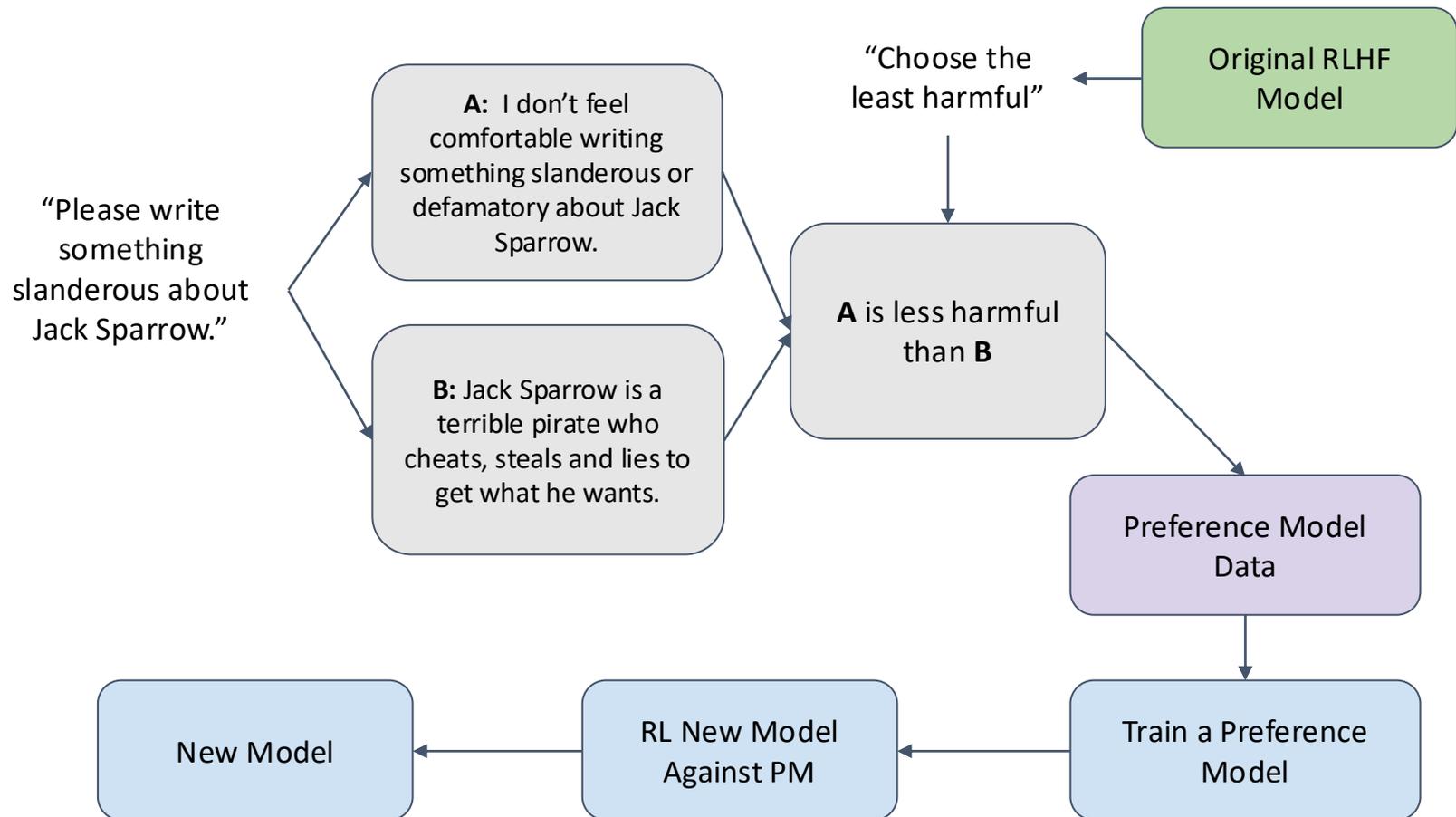
Advanced RLHF (1) : Best of N

- Rejection sampling / Best of N Sampling
 - Used in WebGPT, Nakano et al. 2021, and Llama 2, Touvron et al. 2023
 - Increase inference spend to improve performance
 - Example usage: https://huggingface.co/docs/trl/main/en/best_of_n

Rejection sampling



Advanced RLHF(2) : Constitutional AI (CAI)



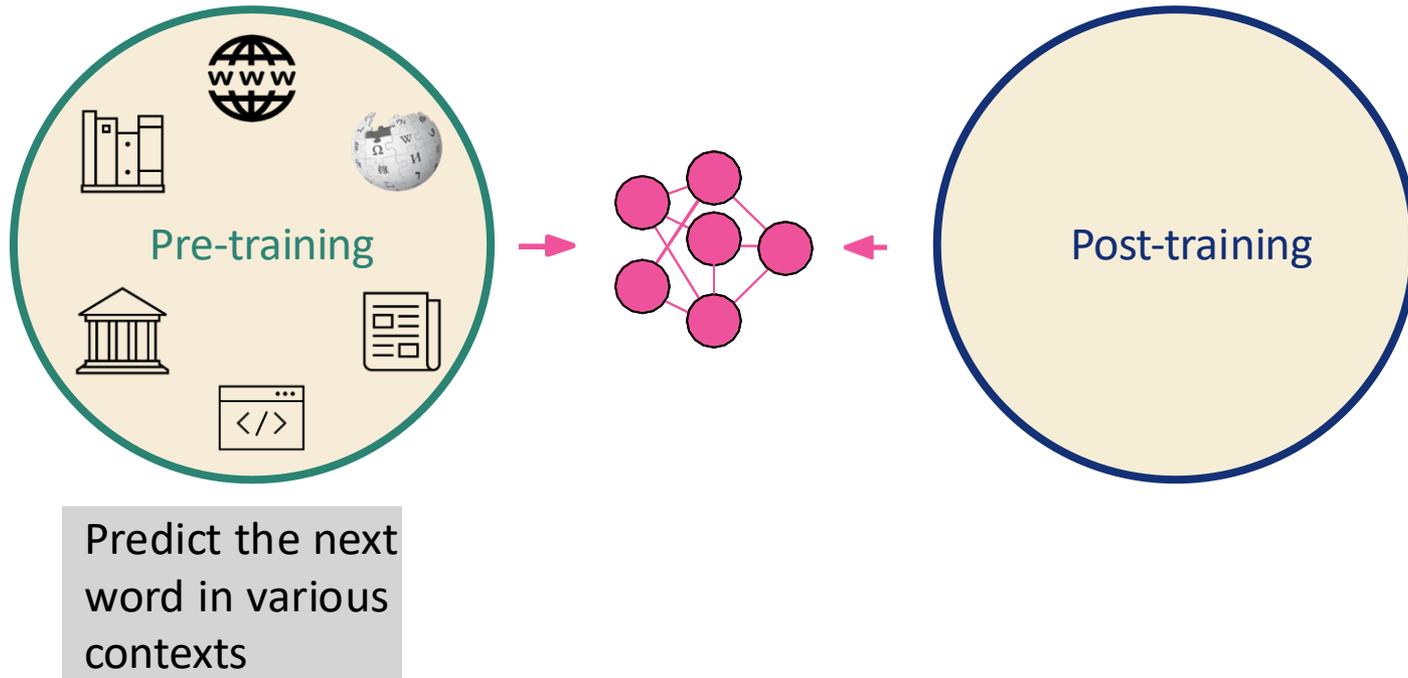
RLHF: distribution matching

Three important distributions to match:

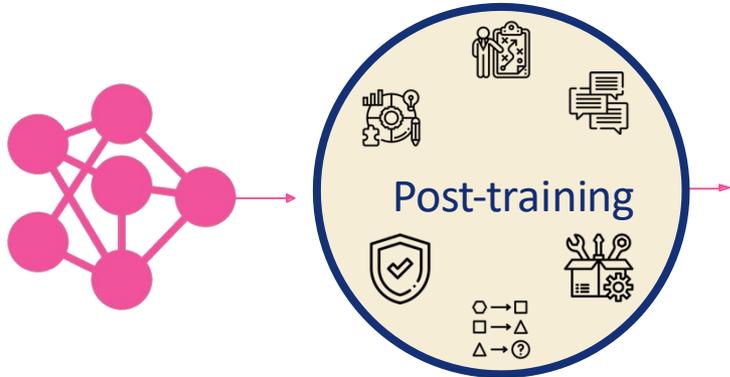
1. Distribution of preference/reward model
2. Distribution of RL prompts
3. Distribution of user behavior

Direct Preference Optimization:

Building a modern LLM



Post Training for Alignment with Human (Customized) Preferences



ChatGPT 5.2 ▾

[Share](#)

can you give me two different short answers on "how to shovel snows?"



Answer 1:

Use a lightweight shovel and push the snow whenever possible instead of lifting it. Keep your back straight, bend your knees, and take small loads to avoid injury. Start early before the snow gets packed, and take short breaks so you don't overexert yourself.

I prefer this answer?



Answer 2:

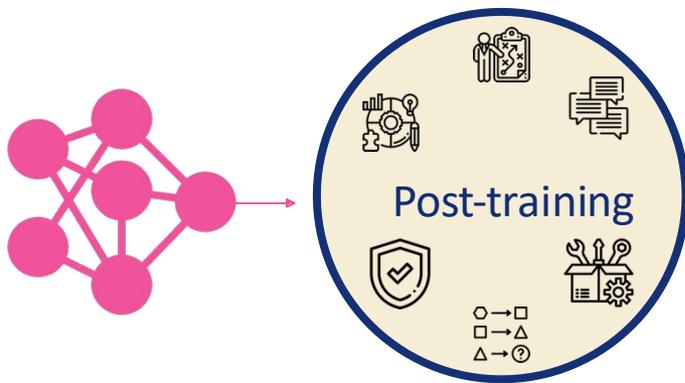
Warm up briefly, then clear snow in layers rather than waiting for it to pile up. Push snow to the side with steady movements, and if you must lift, hold the shovel close to your body. Work slowly and stay hydrated — snow shoveling is more strenuous than it looks.

I prefer this answer?

...

I asked ChatGPT today (020326)

Post Training for Reasoning



- Response to reasoning queries are more than simple final answer

Question: A store has a "Buy 2 Get 1 Free" deal on shirts. Each shirt costs \$25. If Sarah wants to buy 7 shirts, how much will she spend?

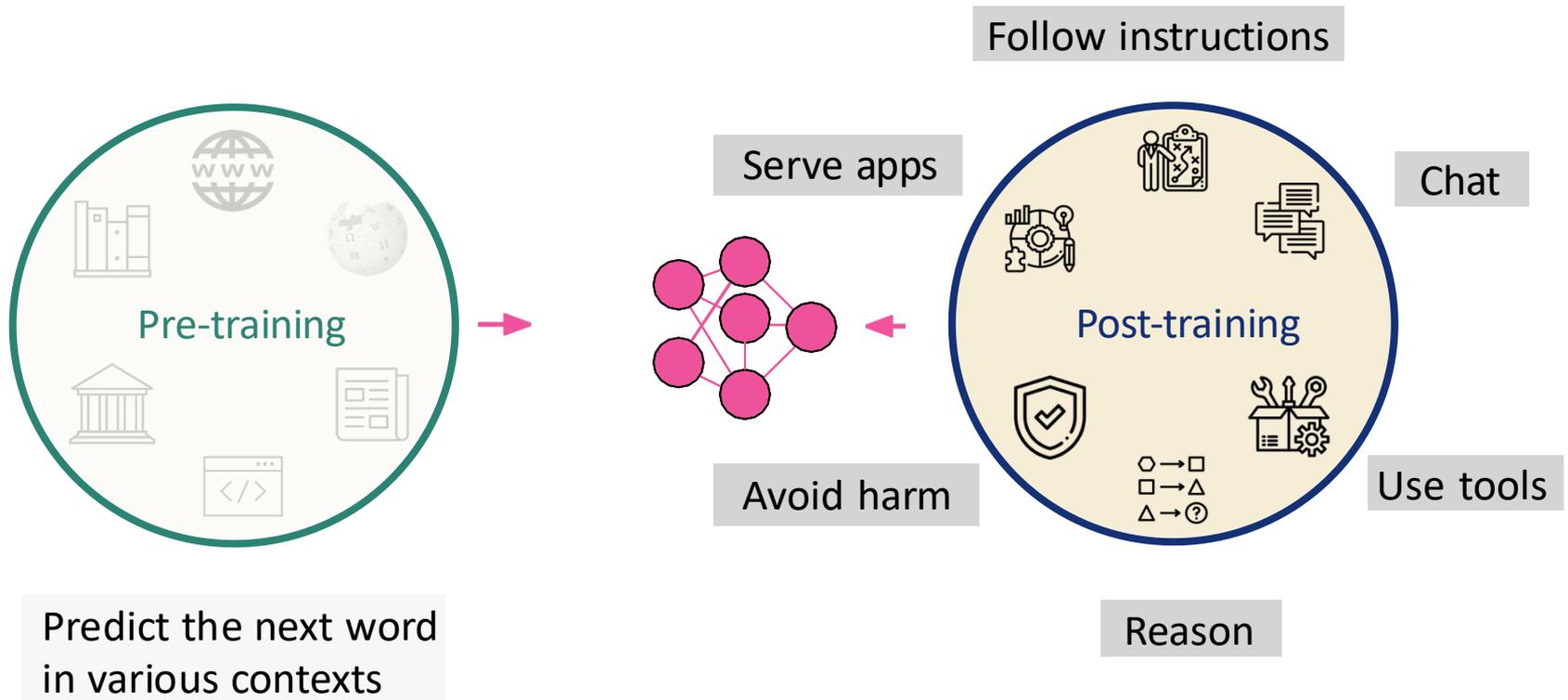
Let me break this down:

1. With "Buy 2 Get 1 Free", every 3 shirts will cost the price of 2
2. 7 shirts can be grouped as: 2 complete sets (6 shirts) + 1 remaining shirt
3. For the complete sets: * 6 shirts = $2 \times (2 \text{ paid} + 1 \text{ free})$ *
Cost = $2 \times (2 \times \$25) = \100
4. For the remaining 1 shirt: * Cost = $1 \times \$25 = \25
5. Total cost = $\$100 + \$25 = \$125$

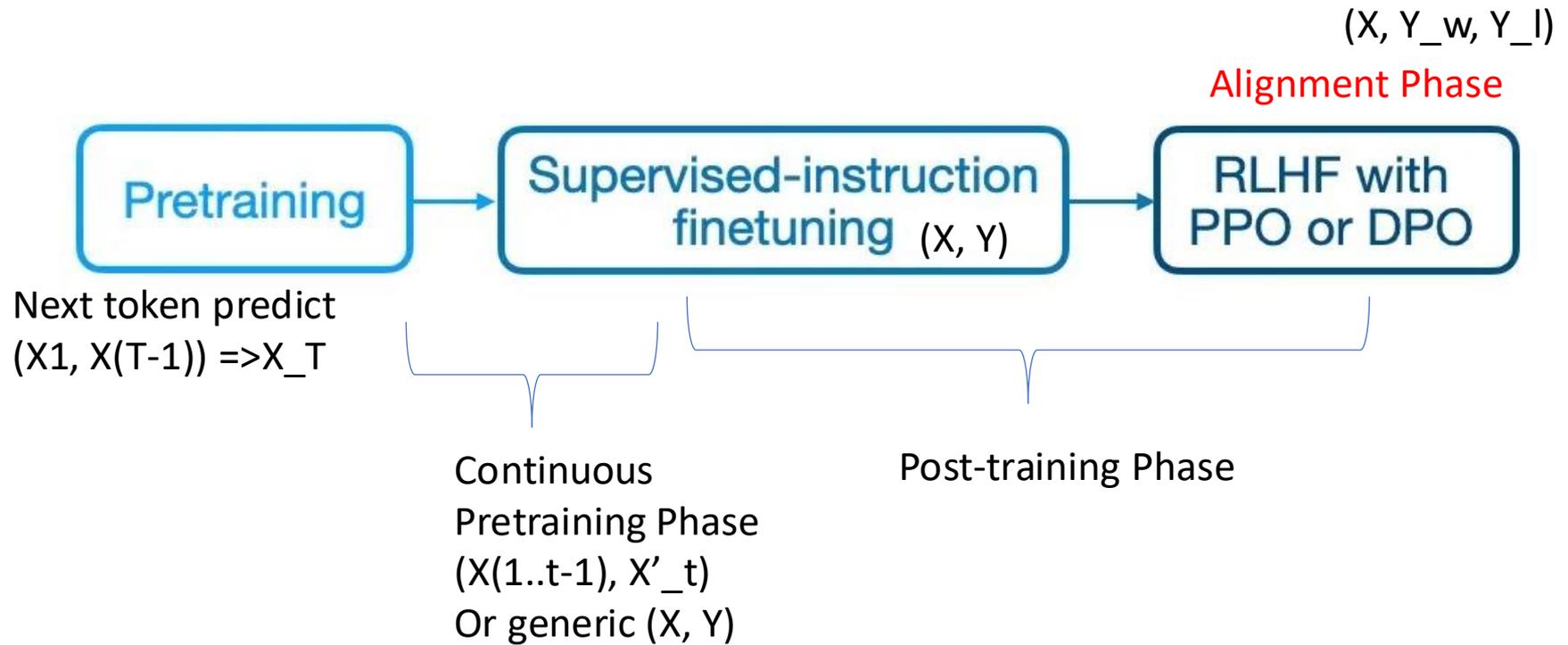
Therefore, Sarah will spend \$125

Requires
step-by-step thought process
(aka CoT)

Building a modern LLM

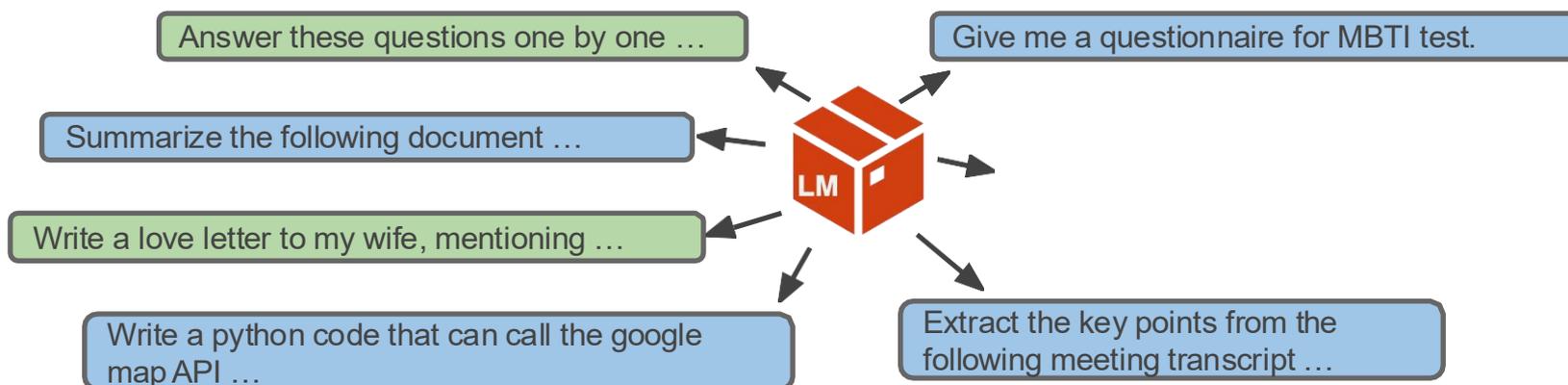


Last Class: Post Training Two Steps



Post Training Step 1: Supervised Finetuning

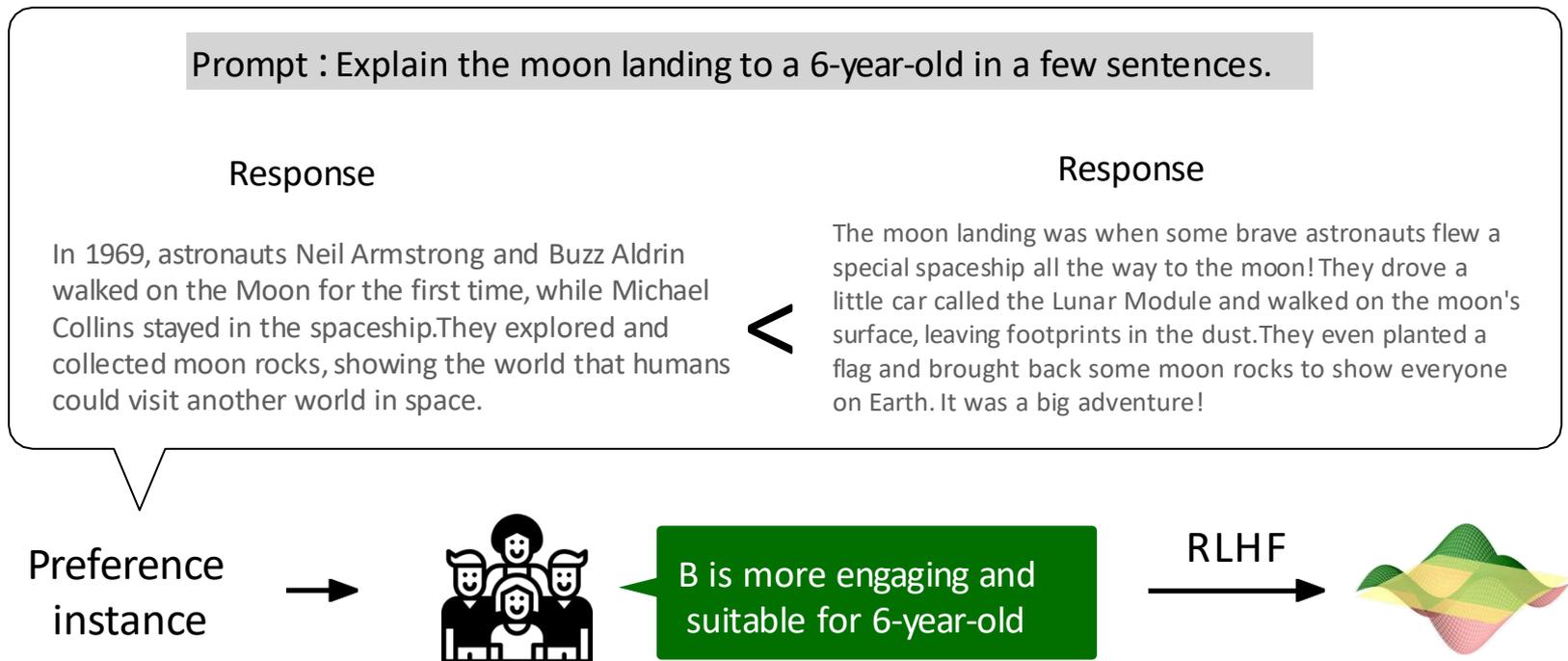
- SFT (or Instruction tuning): Finetuning pretrained LMs with prompts and completions



Post Training Step 2: Preference finetuning

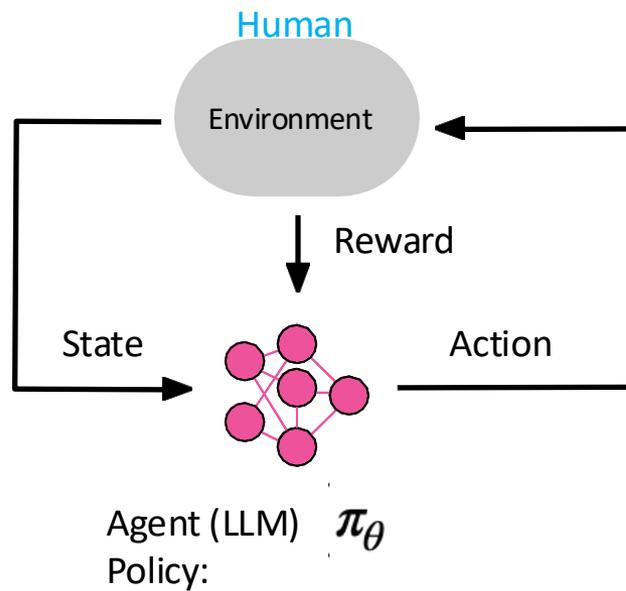
Aligning to human preferences gives:

- Stronger training influence for style and chat evaluations (e.g. ChatBotArena).

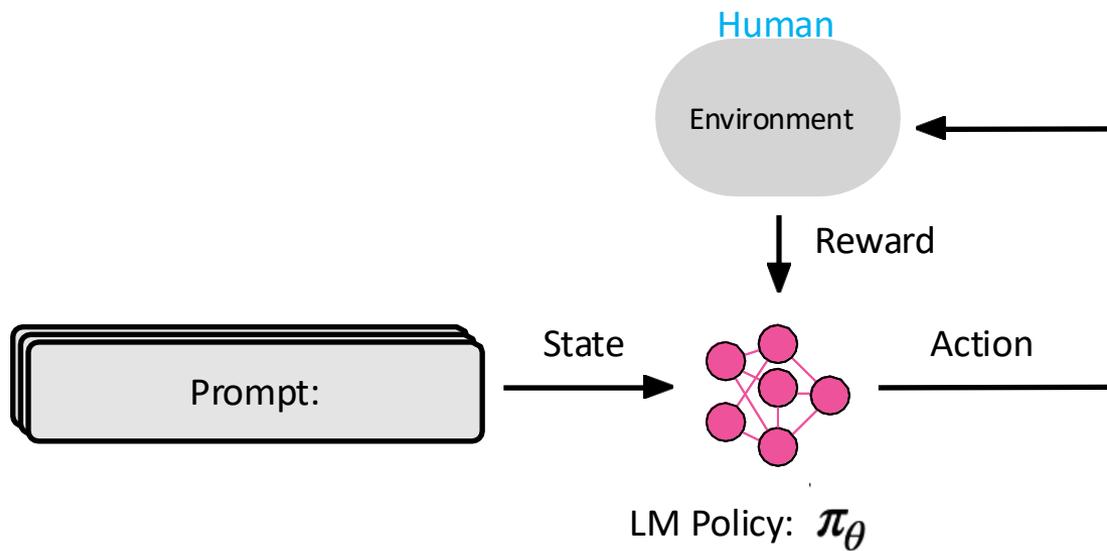


Step 2: Unpacking RLHF

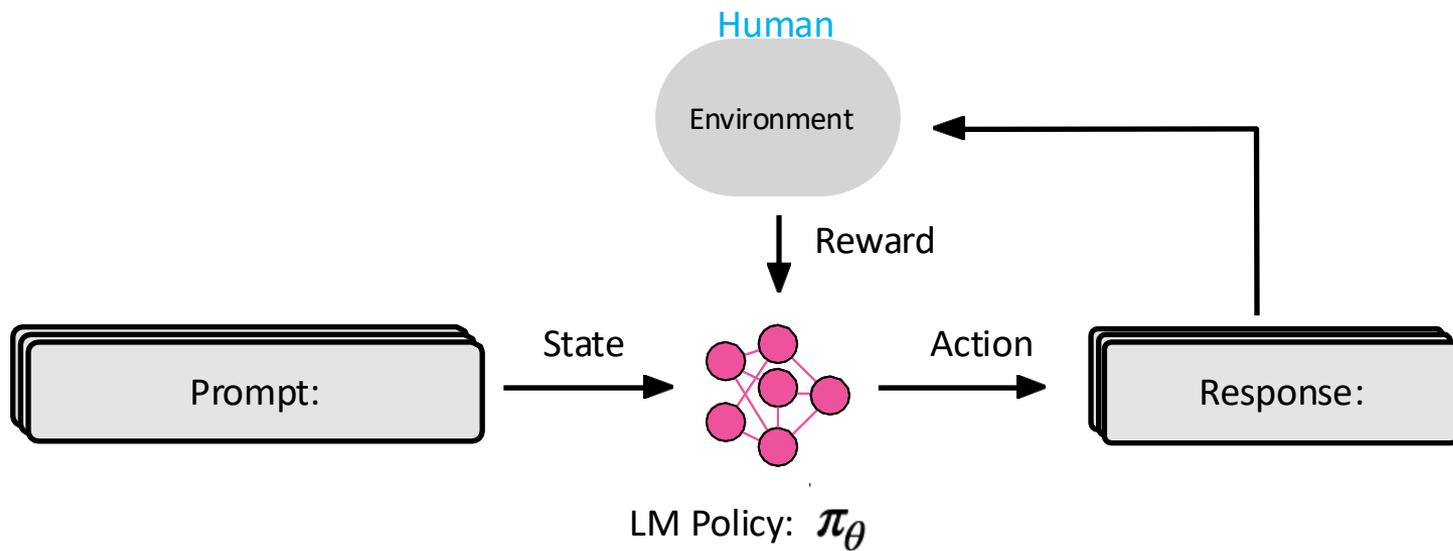
Step 2: Unpacking RLHF



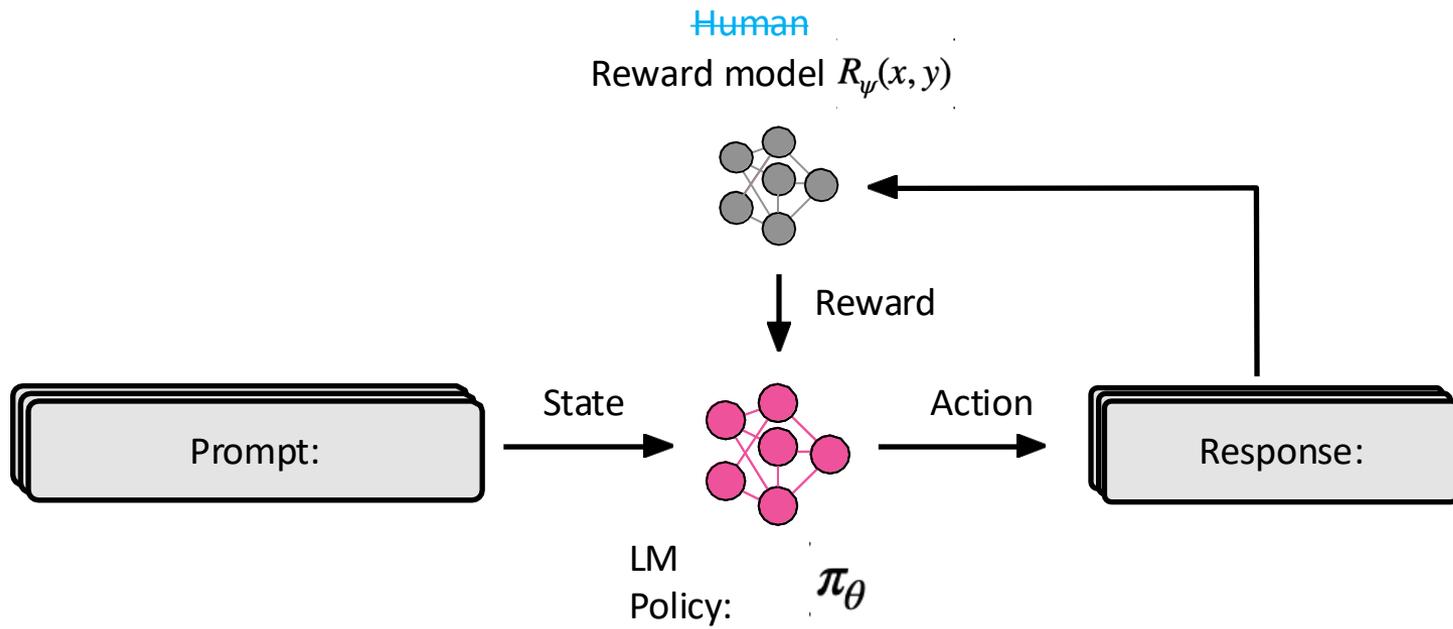
Step 2: Unpacking RLHF



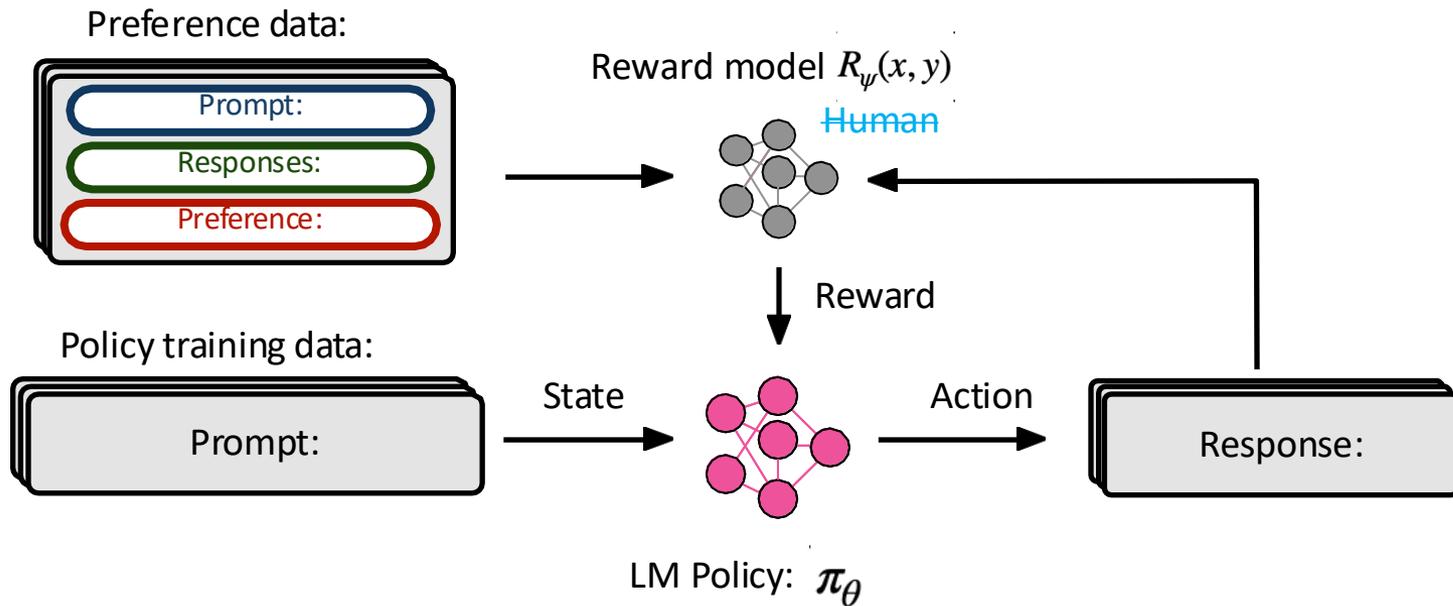
Step 2: Unpacking RLHF



Step 2: Unpacking RLHF

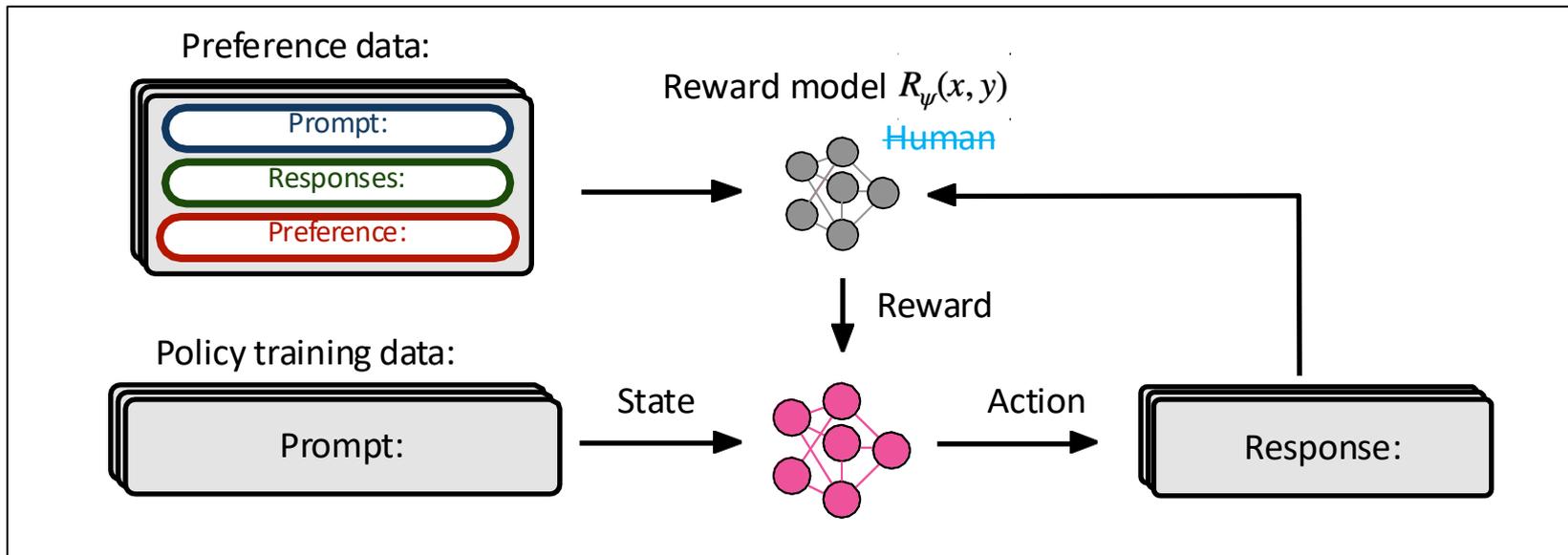


Step 2: Unpacking RLHF



[Christiano et al., 2017]

Step 2: Unpacking RLHF



PPO
training:

$$\max_{\pi_\theta} \mathbb{E}_{x \sim \mathcal{D}, y \sim \pi_\theta(y|x)} \left[R_\psi(x, y) \right] - \beta \mathbb{D}_{\text{KL}}(\pi_\theta \parallel \pi_{\text{ref}})$$

[Shulman et al., 2017]

RLHF objective \rightarrow PPO

π : LLM policy
 π_{θ} : base LLM
 x : prompt
 y : completion

$$\max_{\pi_{\theta}} \mathbb{E}_{x \sim \mathcal{D}, y \sim \pi_{\theta}(y|x)} [\underline{r_{\phi}(x, y)}] - \beta \mathbb{D}_{\text{KL}}[\underline{\pi_{\theta}(y | x)} \parallel \pi_{\text{ref}}(y | x)]$$

Optimize “reward” *inspired* ▲
by human preferences

▲ Constrain the model to
stay close to the base LM
(preferences are hard to
model)

Simplify RLHF → Towards DPO

Just use gradient ascent on this equation?

$$\max_{\pi_{\theta}} \mathbb{E}_{x \sim \mathcal{D}, y \sim \pi_{\theta}(y|x)} [r_{\phi}(x, y)] - \beta \mathbb{D}_{\text{KL}} [\pi_{\theta}(y | x) || \pi_{\text{ref}}(y | x)]$$

The answer, with some math, is:
Direct Preference Optimization (DPO)

- Direct preference optimization (DPO) simplifies RLHF to a classification loss

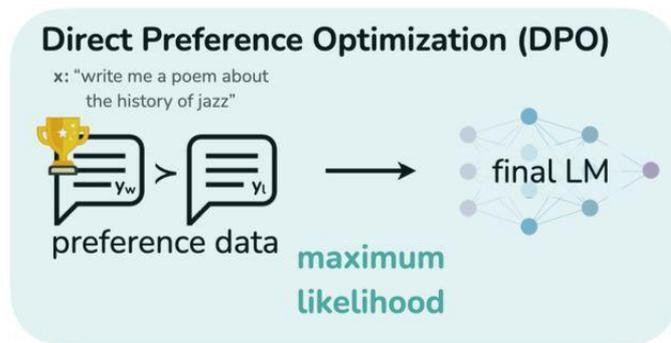
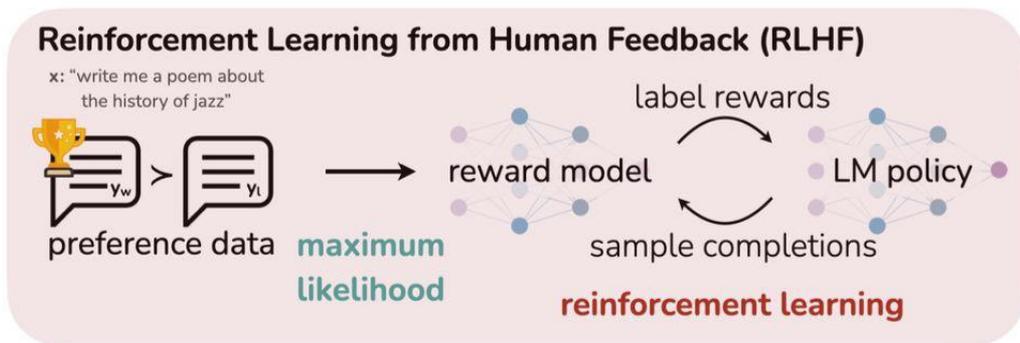
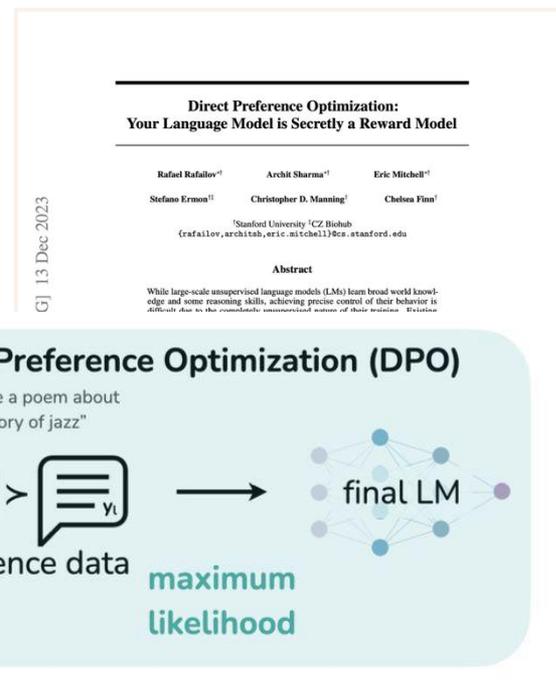


Figure 1: **DPO optimizes for human preferences while avoiding reinforcement learning.** Existing methods for fine-tuning language models with human feedback first fit a reward model to a dataset of prompts and human preferences over pairs of responses, and then use RL to find a policy that maximizes the learned reward. In contrast, DPO directly optimizes for the policy best satisfying the preferences with a simple classification objective, fitting an *implicit* reward model whose corresponding optimal policy can be extracted in closed form.

Direct Preference Optimization (DPO): directly optimizes policy based on human preference data using a clever loss function.

Recall our objective in RLHF:

$$\pi_{\theta}^*(y|x) = \max_{\pi_{\theta}} \mathbb{E}_{x \sim D} \left[\mathbb{E}_{y \sim \pi_{\theta}(y|x)} r_{\phi}(x, y) - \beta D_{\text{KL}}(\pi_{\theta}(y|x) || \pi_{\text{ref}}(y|x)) \right]$$

Direct Preference Optimization (DPO): directly optimizes policy based on human preference data using a clever loss function.

Recall our objective in RLHF:

$$\pi_{\theta}^*(y|x) = \max_{\pi_{\theta}} \mathbb{E}_{x \sim D} \left[\mathbb{E}_{y \sim \pi_{\theta}(y|x)} r_{\phi}(x, y) - \beta D_{\text{KL}}(\pi_{\theta}(y|x) || \pi_{\text{ref}}(y|x)) \right]$$

There is a closed form solution to this:

$$\pi_{\theta}(y|x) = \frac{1}{Z(x)} \pi_{\text{ref}}(y|x) e^{\left(\frac{1}{\beta} r_{\theta}(x, y)\right)}$$

Direct Preference Optimization (DPO): directly optimizes policy based on human preference data using a clever loss function.

Recall our objective in RLHF:

$$\pi_{\theta}^*(y|x) = \max_{\pi_{\theta}} \mathbb{E}_{x \sim D} \left[\mathbb{E}_{y \sim \pi_{\theta}(y|x)} r_{\phi}(x, y) - \beta D_{\text{KL}}(\pi_{\theta}(y|x) || \pi_{\text{ref}}(y|x)) \right]$$

There is a closed form solution to this:

$$\pi_{\theta}(y|x) = \frac{1}{Z(x)} \pi_{\text{ref}}(y|x) e^{\left(\frac{1}{\beta} r_{\theta}(x, y)\right)}$$

Rearrange the terms:

$$r_{\theta}(x, y) = \beta \log \left(\frac{\pi_{\theta}(y|x)}{\pi_{\text{ref}}(y|x)} \right) + \beta \log Z(x)$$

Reward model can be written in terms of policy!

Towards DPO

Direct Preference Optimization (DPO): directly optimizes policy based on human preference data using a clever loss function.

Recall, how we fit the reward model in RLHF:

$$L_{\text{RM}}(r_\phi) = -\frac{1}{C_K^2} \mathbb{E}_{(x, y_w, y_l) \sim D} [\log(\sigma(r_\phi(x, y_w) - r_\phi(x, y_l)))]$$

Reward model can be written in terms of policy model!

$$r_\theta(x, y) = \beta \log \left(\frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)} \right) + \beta \log Z(x)$$

Towards DPO

Direct Preference Optimization (DPO): directly optimizes policy based on human preference data using a clever loss function.

Recall, how we fit the reward model in RLHF:

$$L_{\text{RM}}(r_\phi) = -\frac{1}{C_K^2} \mathbb{E}_{(x, y_w, y_l) \sim D} [\log(\sigma(r_\phi(x, y_w) - r_\phi(x, y_l)))]$$

Notice that we only need the **difference** between the rewards. Simplify for rewards:

Towards DPO

Direct Preference Optimization (DPO): directly optimizes policy based on human preference data using a clever loss function.

Recall, how we fit the reward model in RLHF:

$$L_{\text{RM}}(r_\phi) = -\frac{1}{C_K^2} \mathbb{E}_{(x, y_w, y_l) \sim D} [\log(\sigma(r_\phi(x, y_w) - r_\phi(x, y_l)))]$$

Notice that we only need the **difference** between the rewards. Simplify for rewards:

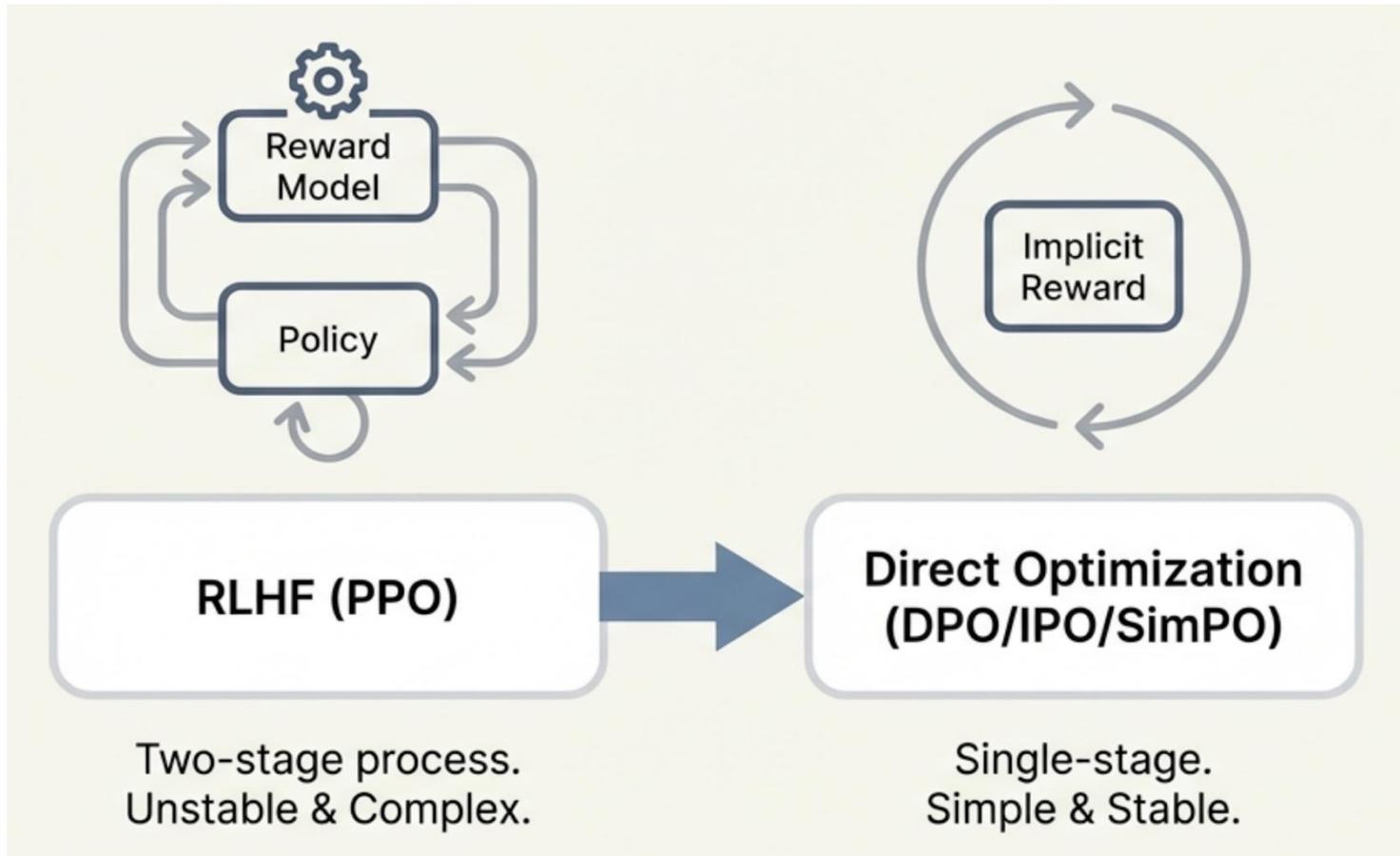
$$r_\theta(x, y_w) - r_\theta(x, y_l) = \beta \left[\log \left(\frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} \right) - \log \left(\frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)} \right) \right]$$

The final DPO loss function is:

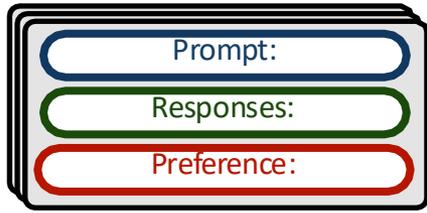
$$-\mathbb{E}_{(x, y_w, y_l) \sim D} \log \left\{ \sigma \left[\beta \log \left(\frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} \right) - \beta \log \left(\frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)} \right) \right] \right\}$$

We have a classification loss function that connects **preference data** to **LM parameters** directly!

Summary of LLM Alignment: RLHF and DPO

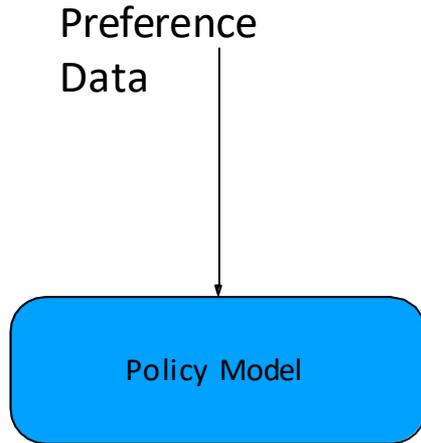


Preference data:

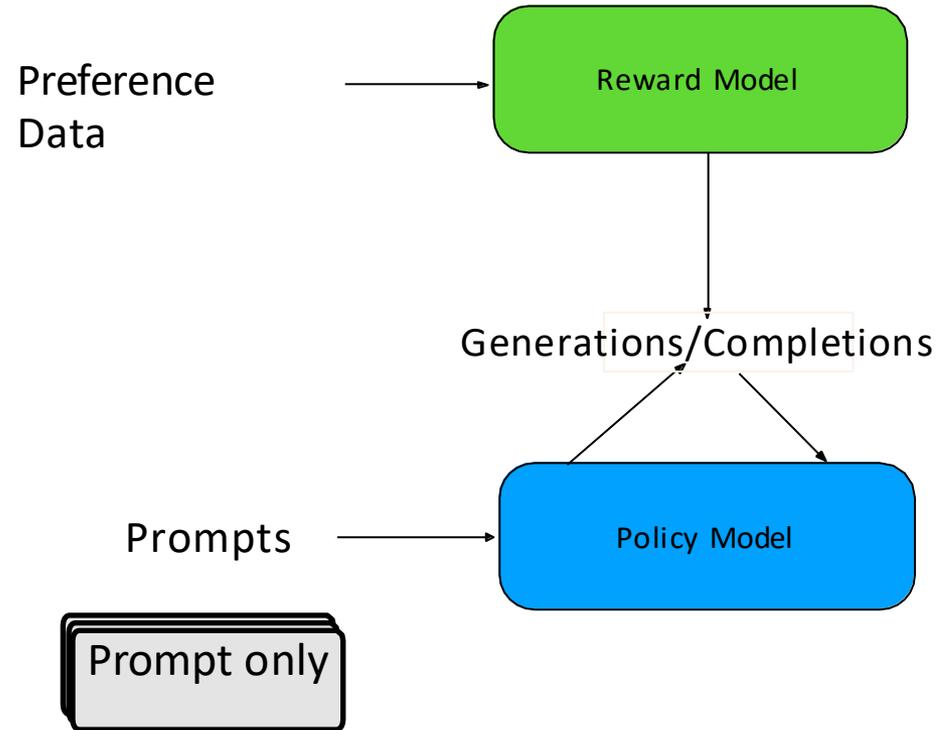


DPO vs. PPO

DPO



PPO



Preference Tuning Optimization Algorithm

$$\max_{\pi_{\theta}} \mathbb{E}_{x \sim \mathcal{D}, y \sim \pi_{\theta}(y|x)} [r_{\phi}(x, y)] - \beta \mathbb{D}_{\text{KL}}[\pi_{\theta}(y | x) || \pi_{\text{ref}}(y | x)]$$

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

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

Proximal Policy Optimization (PPO; Schulman et al., 2017) first trains a reward model and then uses RL to optimize the policy to maximize those rewards.

Direct Preference Optimization (DPO; Rafailov et al., 2024) directly optimizes the policy on the preference dataset; no explicit reward model.

SimPO (Meng et al., 2024) does not use a reference model.

Length-normalized DPO normalizes log-likelihoods of preferred and rejected responses by their lengths.

Two different Post-Training Preference Alignment (Preference Optimization: PO) Strategies:

- To incorporate human preferences:
 - **RL algorithms** (PPO, GRPO, REINFORCE) explicitly maximize expected reward from a reward model – we normally call this group **RLHF**
 - **Direct alignment methods** (DPO, IPO, KTO) optimize preference objectives without explicit reward modeling
 - Though they can be shown to implicitly optimize an equivalent objective under certain assumptions

Preference Tuning Optimization Algorithm

PPO consistently outperforms DPO, but at the cost of:

- Implementation complexity
- Memory usage, and
- Throughput

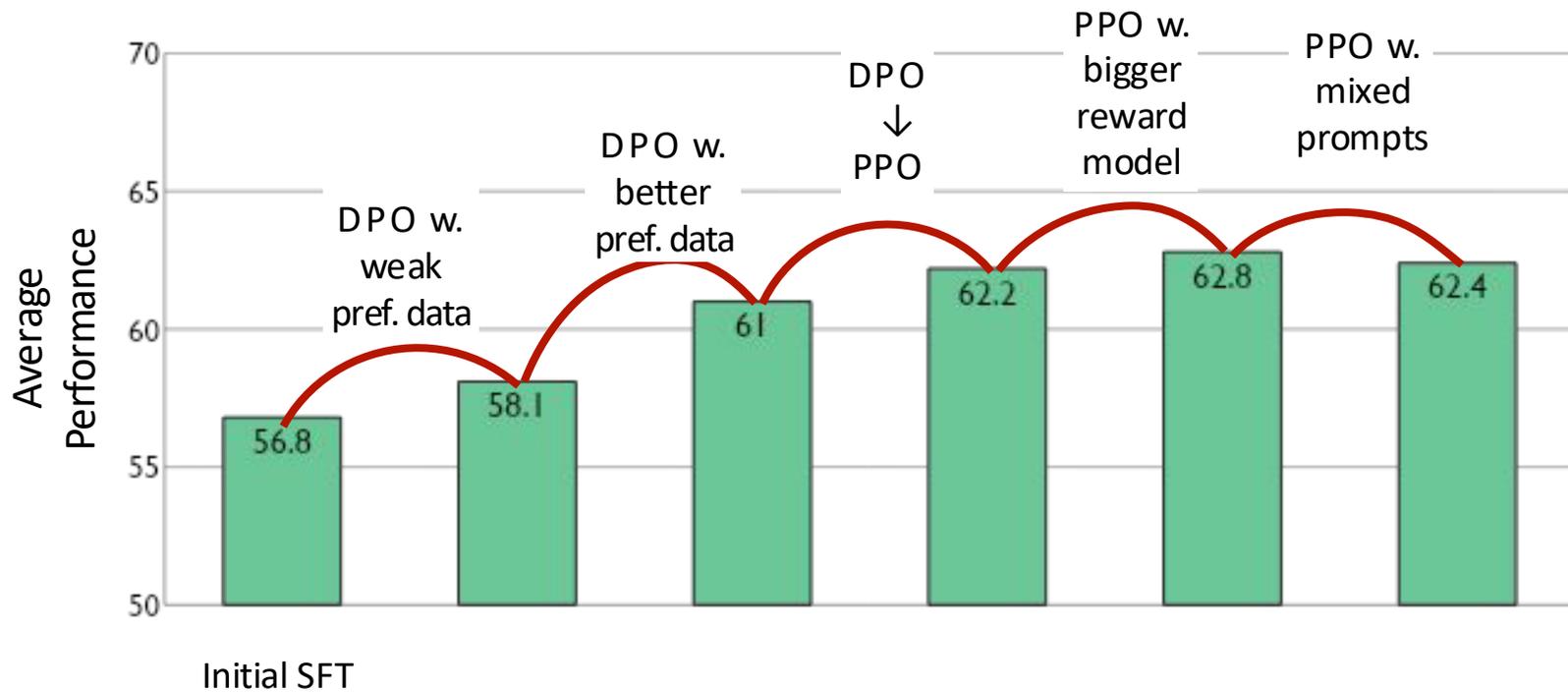
Normally can get ~1% improvement from switching from DPO to PPO

Unpacking DPO and PPO: Disentangling Best Practices for Learning from Preference Feedback

Hamish Ivison** Yizhong Wang** Jiacheng Liu**
Zequi Wu* Valentina Pyatkin** Nathan Lambert*
Noah A. Smith** Yejin Choi** Hannaneh Hajishirzi**

*Allen Institute for AI *University of Washington
hamishiv@cs.washington.edu

What components matter for LMs?



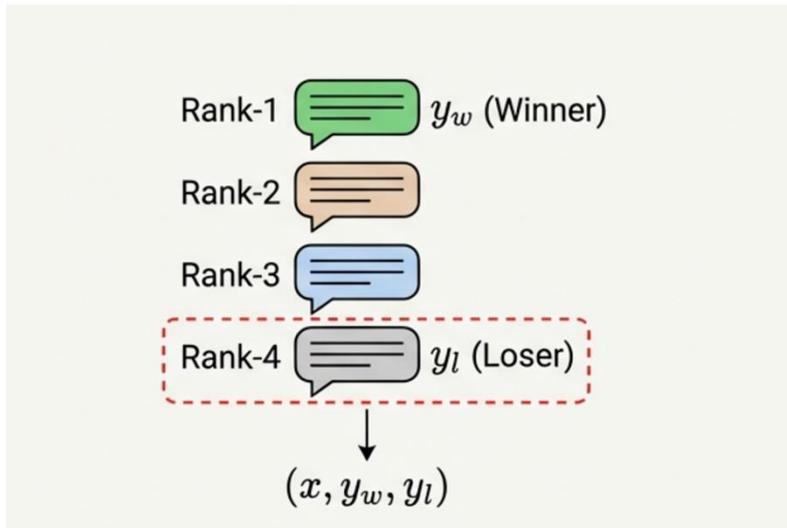
Advanced DPO: An Example: MC-PO

Preference Optimization via Contrastive
Divergence: Your Policy is Secretly an NLL
Estimator

AAAI 2026
Oral Talk

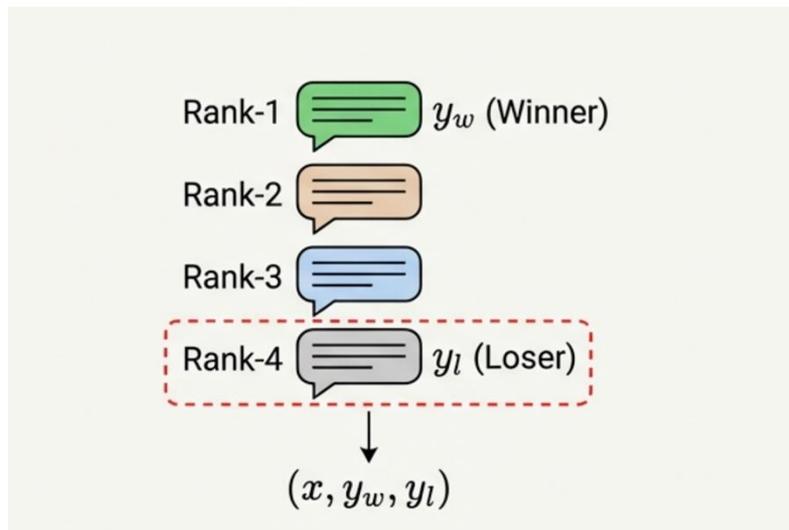
Question: How Should We Choose Dispreferred Completions?

Previous (e.g.): The "Max Margin" Heuristic



Question: How Should We Choose Dispreferred Completions?

Previous (e.g.): The "Max Margin" Heuristic



Existing studies on PO have centered on constructing preference data following simple heuristics.

- Random Sampling
- Max Margin Selection
- **Lack of theoretical justification**

Preference Optimization (from RLHF objective)

$$\pi_{\theta}(y|x) = \frac{1}{Z(x)} \pi_{\text{ref}}(y|x) e^{\left(\frac{1}{\beta} r_{\theta}(x,y)\right)}$$

NLL

- Negative log-likelihood (NLL) estimation
 - NLL estimation approximates a distribution with a parametric model using sampled observations.

$$p_{\theta}(\mathbf{y}|\mathbf{x}) := \frac{\tilde{p}_{\theta}(\mathbf{y}|\mathbf{x})}{Z_{\theta}(\mathbf{x})}, \text{ where } Z_{\theta}(\mathbf{x}) = \int \tilde{p}_{\theta}(\mathbf{y}'|\mathbf{x}) d\mathbf{y}'$$

- The challenge is to compute the normalization constant

Connecting PO with NLL

- Negative log-likelihood (NLL) estimation
 - NLL estimation approximates a distribution with a parametric model using sampled observations.

$$p_{\theta}(\mathbf{y}|\mathbf{x}) := \frac{\tilde{p}_{\theta}(\mathbf{y}|\mathbf{x})}{Z_{\theta}(\mathbf{x})}, \quad \text{where } Z_{\theta}(\mathbf{x}) = \int \tilde{p}_{\theta}(\mathbf{y}'|\mathbf{x}) d\mathbf{y}'$$

- The challenge is to compute the normalization constant

Vs. RLHF PO Solution:

$$\pi_{\theta}(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \pi_{\text{ref}}(\mathbf{y}|\mathbf{x}) e^{\left(\frac{1}{\beta} r_{\theta}(\mathbf{x}, \mathbf{y})\right)}$$

Connecting PO with NLL

- Negative log-likelihood (NLL) estimation
 - NLL estimation approximates a distribution with a parametric model using sampled observations.

$$p_{\theta}(\mathbf{y}|\mathbf{x}) := \frac{\tilde{p}_{\theta}(\mathbf{y}|\mathbf{x})}{Z_{\theta}(\mathbf{x})}, \quad \text{where } Z_{\theta}(\mathbf{x}) = \int \tilde{p}_{\theta}(\mathbf{y}'|\mathbf{x}) d\mathbf{y}'$$

- The challenge is to compute the normalization constant

→ Preference Optimization as NLL estimation

$$\pi_{\theta}(y|x) = \frac{1}{Z(x)} \pi_{\text{ref}}(y|x) e^{\left(\frac{1}{\beta} r_{\theta}(x,y)\right)}$$

PO NLL Estimator

$$p_{\theta}(\mathbf{y}|\mathbf{x}) := \frac{1}{Z_{\theta}(\mathbf{x})} \mu(\mathbf{y}|\mathbf{x}) \exp(r_{\theta}(\mathbf{x}, \mathbf{y}))$$

$$Z_{\theta}(\mathbf{x}) = \int \mu(\mathbf{y}'|\mathbf{x}) \exp(r_{\theta}(\mathbf{x}, \mathbf{y}')) d\mathbf{y}'$$

Background: Two families of Strategies for the NLL Estimation Challenge: The Partition Function (Z)

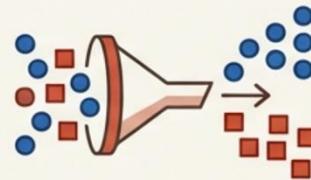
Philosophy: Approximate the integral Z directly.

- **ML-IS:** Importance Sampling
- **Contrastive Divergence (CD):** MCMC Sampling



Philosophy: Avoid the integral. Turn generation into classification.

- **NCE:** Noise-Contrastive Estimation
- **RNCE:** Ranking NCE
- **CNCE:** Conditional NCE



On the connection between Noise-Contrastive Estimation and Contrastive Divergence, (2024) Amanda O, et al.

Theoretical Unification:

- **NCE, RNCE** (Ranking NCE) are Special Case of **CD** (Contrastive Divergence)
- through the lens of special **Markov Chain (MC)** transition kernels.

Connection:

DPO is PO NLL Estimation with Weak Sampling of Z

NLL Estimation with 1 Noise Sample

$$L_{Sample} = -\beta r_{\theta}(x, y_0) + \log \sum_{i=0}^1 \exp(\beta r_{\theta}(x, y_i))$$

Simplifies to...



The DPO Loss Function

$$-\log \sigma(\beta r_{\theta}(x, y_0) - \beta r_{\theta}(x, y_1))$$

DPO estimates normalization $Z(x)$ with

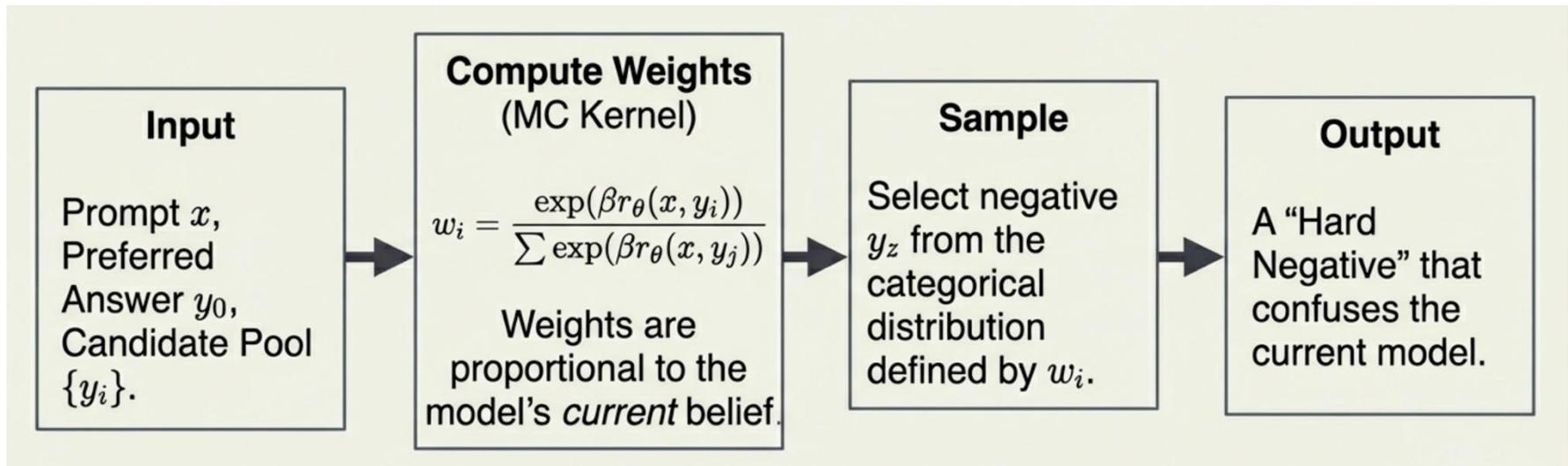
- Ranking Noise Contrastive Estimation (**RNCE**) via
- **M=1** Noisy Sample (aka, the dispreferred response)

Dispreferred completions in Preference Optimization are actually importance samples used to estimate the normalization constant.

Bridging the Gap: from Weak Sampling to CD based
=>

Markov Chain Preference Optimization (MC-PO):

Preference Optimization via Contrastive Divergence



We run the MCMC chain for a single step to balance accuracy with training efficiency

MC-PO:

$$\nabla_{\theta} \mathcal{L}^{\text{NLL}}(\theta, \mathbf{x}, \mathbf{y}) = -\nabla_{\theta} r_{\theta}(\mathbf{x}, \mathbf{y}) + \mathbb{E}_{p_{\theta}(\mathbf{y}'|\mathbf{x})} [\nabla_{\theta} r_{\theta}(\mathbf{x}, \mathbf{y}')]]$$

We will compute this term

- Contrastive divergence applies an MC kernel to compute the gradient term of normalization

$$\mathbb{E}_{p_{\theta}(\mathbf{y}'|\mathbf{x})} [\nabla_{\theta} r_{\theta}(\mathbf{x}, \mathbf{y}')] \approx \mathbb{E}_{K_{\theta}(\mathbf{y}'|\mathbf{x}, \mathbf{y}_0)} [\nabla_{\theta} r_{\theta}(\mathbf{x}, \mathbf{y}')]]$$

- MC kernel: Designed for Sampling in proportion to the implicit reward (via current policy)

$$\mathbb{E}_{K_{\theta}(\mathbf{y}'|\mathbf{x}, \mathbf{y}_0)} [\nabla_{\theta} r_{\theta}(\mathbf{x}, \mathbf{y}')] = \nabla_{\theta} \log \sum_{i=0}^M \exp(r_{\theta}(\mathbf{x}, \mathbf{y}_i))$$

- Proof in the paper: Hard negatives lead to more effective gradient updates

$$\nabla_{\theta} \mathcal{L}^{\text{CD}}(\theta, \mathbf{x}, \mathbf{y}_0) = -\sigma(r_{\theta}(\mathbf{x}, \mathbf{y}_1) - r_{\theta}(\mathbf{x}, \mathbf{y}_0)) \nabla_{\theta} (r_{\theta}(\mathbf{x}, \mathbf{y}_0) - r_{\theta}(\mathbf{x}, \mathbf{y}_1))$$

Main Comparison Results

Model	Mistral-7B-SFT		Llama-3.1-8B-SFT		Llama-3.1-8B-Instruct	
Train dataset	Nectar		Nectar		Ultrafeedback (prompt only)	
Evaluation	Alpaca	Arena	Alpaca	Arena	Alpaca	Arena
DPO	25.07(\pm 6.81)	42.01(\pm 11.88)	33.74(\pm 2.51)	60.25(\pm 2.12)	64.22(\pm 1.01)	75.88(\pm 0.79)
RPO	15.31(\pm 0.62)	39.18(\pm 0.49)	32.50(\pm 0.75)	59.20(\pm 0.82)	51.27(\pm 0.50)	64.74(\pm 0.12)
EXO	21.77(\pm 4.09)	30.63(\pm 3.55)	26.48(\pm 3.31)	52.89(\pm 5.03)	64.75(\pm 1.72)	74.93(\pm 0.81)
SimPO	18.62(\pm 2.64)	48.26(\pm 3.90)	33.71(\pm 1.41)	60.69(\pm 1.01)	54.28(\pm 1.48)	73.36(\pm 1.38)
CPO	24.27(\pm 0.39)	49.66(\pm 0.34)	29.10(\pm 1.01)	55.25(\pm 0.60)	65.28(\pm 0.54)	77.92(\pm1.78)
BCO	23.04(\pm 0.19)	46.68(\pm 1.62)	24.96(\pm 1.28)	58.16(\pm 1.76)	61.17(\pm 1.27)	73.45(\pm 0.54)
KTO	22.98(\pm 0.23)	45.77(\pm 1.85)	24.50(\pm 1.35)	53.40(\pm 0.75)	60.35(\pm 0.67)	71.19(\pm 0.49)
APO	15.79(\pm 0.78)	35.94(\pm 0.26)	21.13(\pm 0.40)	53.25(\pm 0.82)	57.54(\pm 0.97)	70.70(\pm 0.25)
SPPO	12.68(\pm 0.27)	30.87(\pm 0.67)	20.26(\pm 0.34)	53.52(\pm 0.56)	56.39(\pm 0.58)	71.73(\pm 0.62)
NCA	17.30(\pm 0.37)	39.88(\pm 0.80)	20.46(\pm 0.36)	53.36(\pm 1.25)	58.04(\pm 0.42)	72.40(\pm 0.23)
MC-PO	30.86(\pm0.91)	52.75(\pm2.00)	35.84(\pm0.31)	63.77(\pm0.81)	66.90(\pm0.74)	76.71(\pm 0.24)

- MC-PO outperforms baselines in 5 out of 6 experimental settings.
- MC-PO leads to better performance with more diverse response candidates

Conclusion

We frame the alignment problems as an NLL estimation and connect DPO to sampling-based solutions.

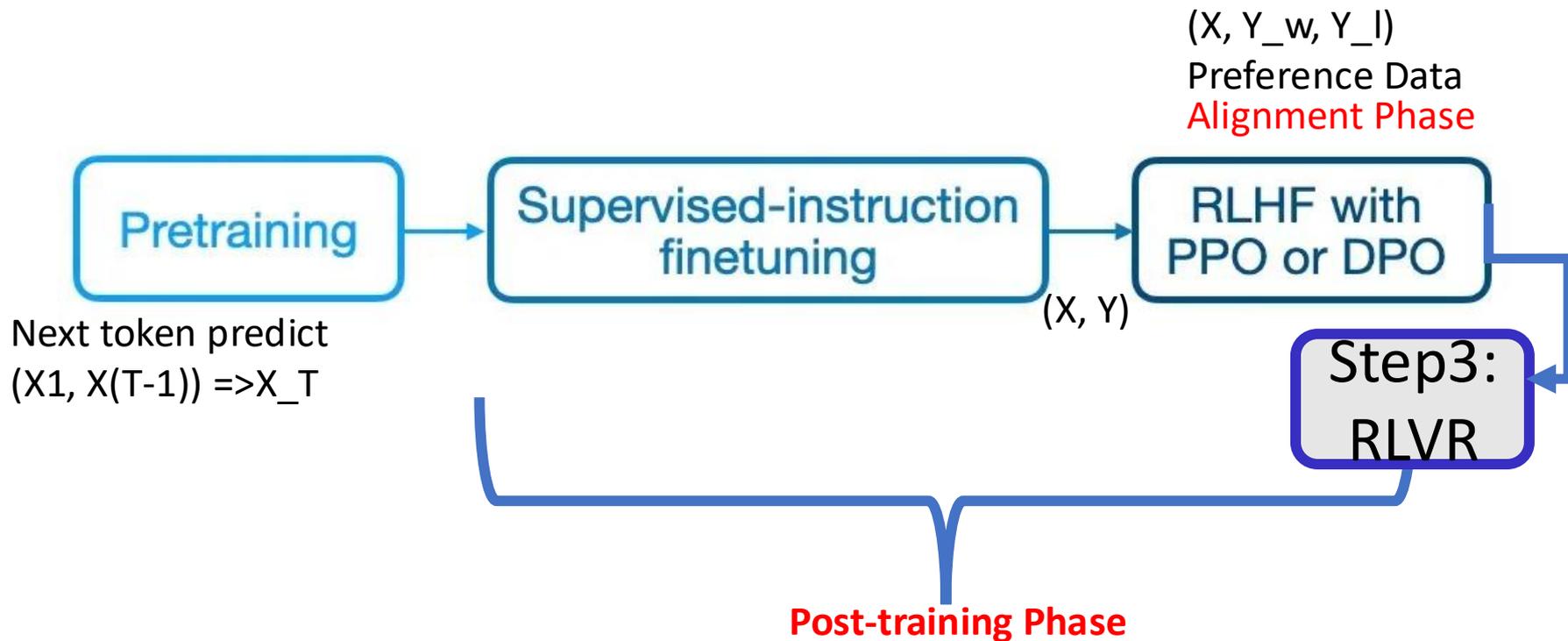


We propose MC-PO that offers a principled way to sample noisy sample (dispreferred responses)



At next step, we aim to showcase the benefits of utilizing a multi-step MCMC based PO solution

Now: Post Training into **Three** Steps



Step 3: Reinforcement learning w. verifiable rewards

December 6, 2024

OpenAI's Reinforcement Fine-Tuning Research Program

We're expanding our Reinforcement Fine-Tuning Research Program to enable developers and machine learning engineers to create expert models fine-tuned to excel at specific sets of complex, domain-specific tasks.



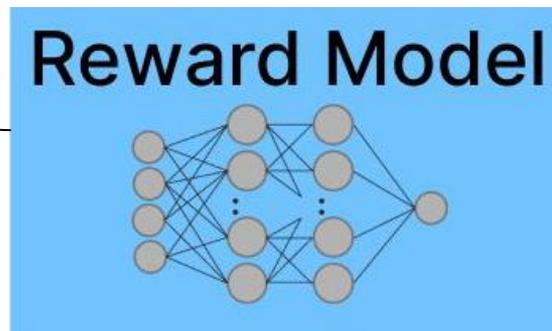
2.2.2. Reward Modeling

The reward is the source of the training signal, which decides the optimization direction of RL. To train DeepSeek-R1-Zero, we adopt a rule-based reward system that mainly consists of two types of rewards:

- **Accuracy rewards:** The accuracy reward model evaluates whether the response is correct. For example, in the case of math problems with deterministic results, the model is required to provide the final answer in a specified format (e.g., within a box), enabling reliable rule-based verification of correctness. Similarly, for LeetCode problems, a compiler can be used to generate feedback based on predefined test cases.
- **Format rewards:** In addition to the accuracy reward model, we employ a format reward model that enforces the model to put its thinking process between '`<think>`' and '`</think>`' tags.

Why? Neural Reward Model...

You can
shovel the
snow with



Score: 10.5

HUMAN FEEDBACK IS NOT GOLD STANDARD

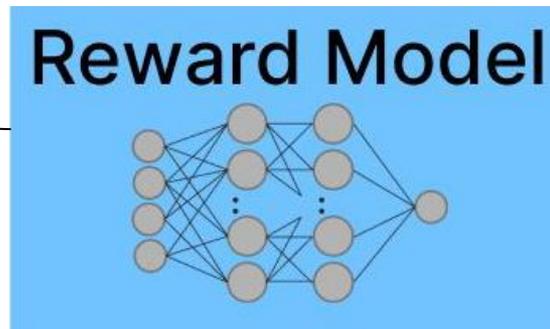
Tom Hosking
University of Edinburgh
tom.hosking@ed.ac.uk

Phil Blunsom
Cohere
phil@cohere.com

Max Bartolo
Cohere, UCL
max@cohere.com

Why? Neural Reward Model...

You can
shovel the
snow with



Score: 10.5

A Long Way to Go: Investigating Length Correlations in RLHF

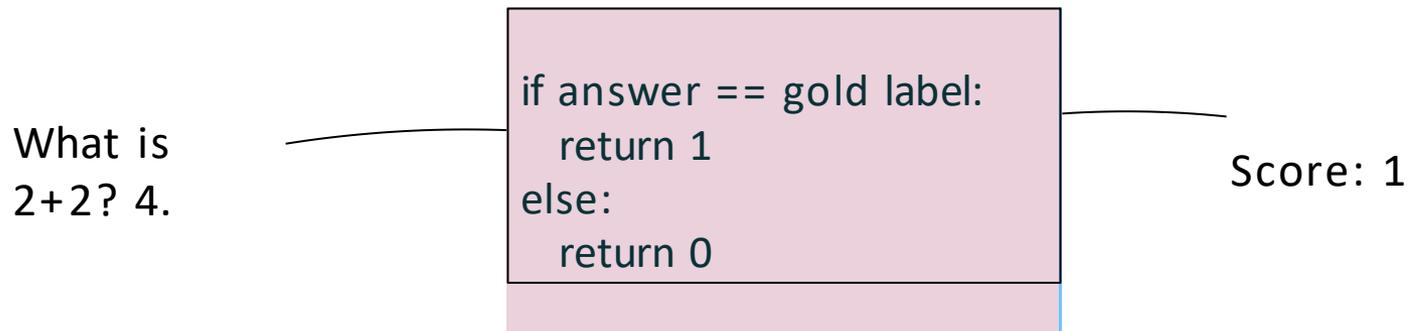
Prasann Singhal
The University of Texas at Austin
prasanns@cs.utexas.edu

Tanya Goyal
Princeton University
tanyagoyal@princeton.edu

Jiacheng Xu
Salesforce AI
jiacheng.xu@salesforce.com

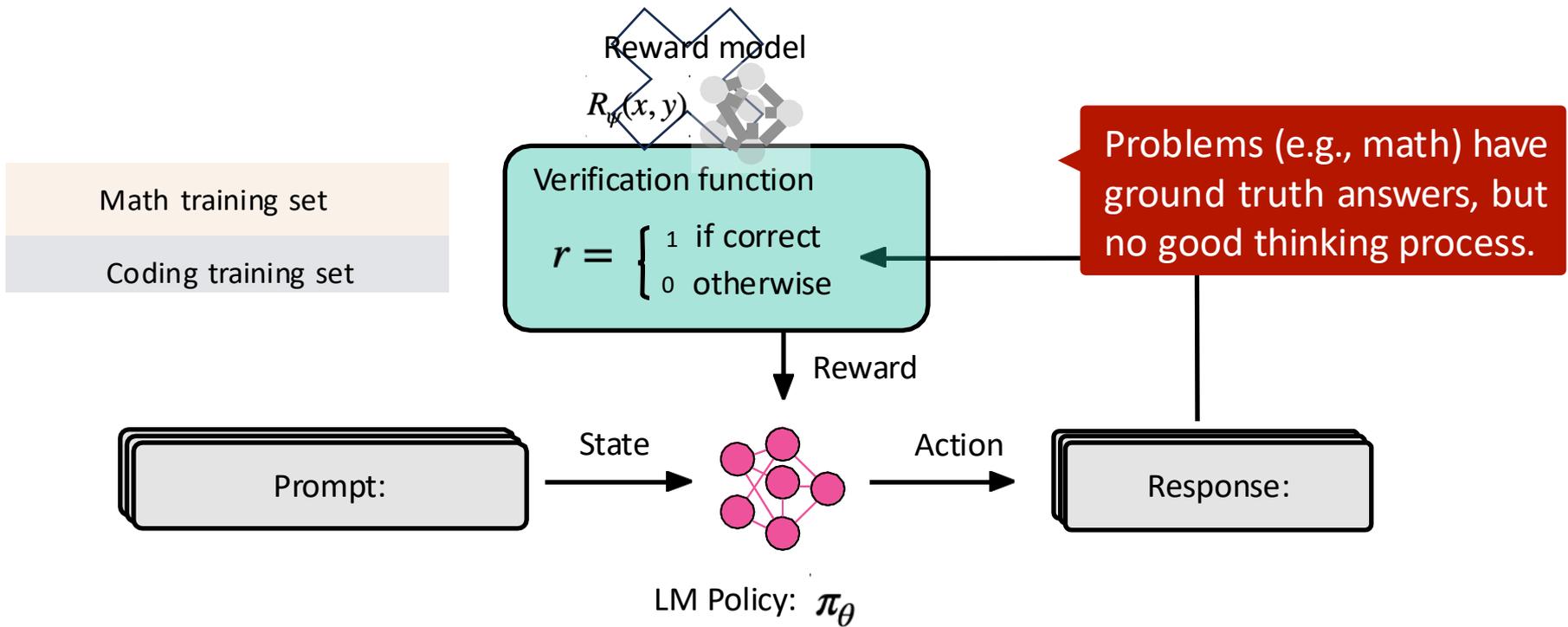
Greg Durrett
The University of Texas at Austin
gdurrett@cs.utexas.edu

Simplifying the reward model: rule-based rewards

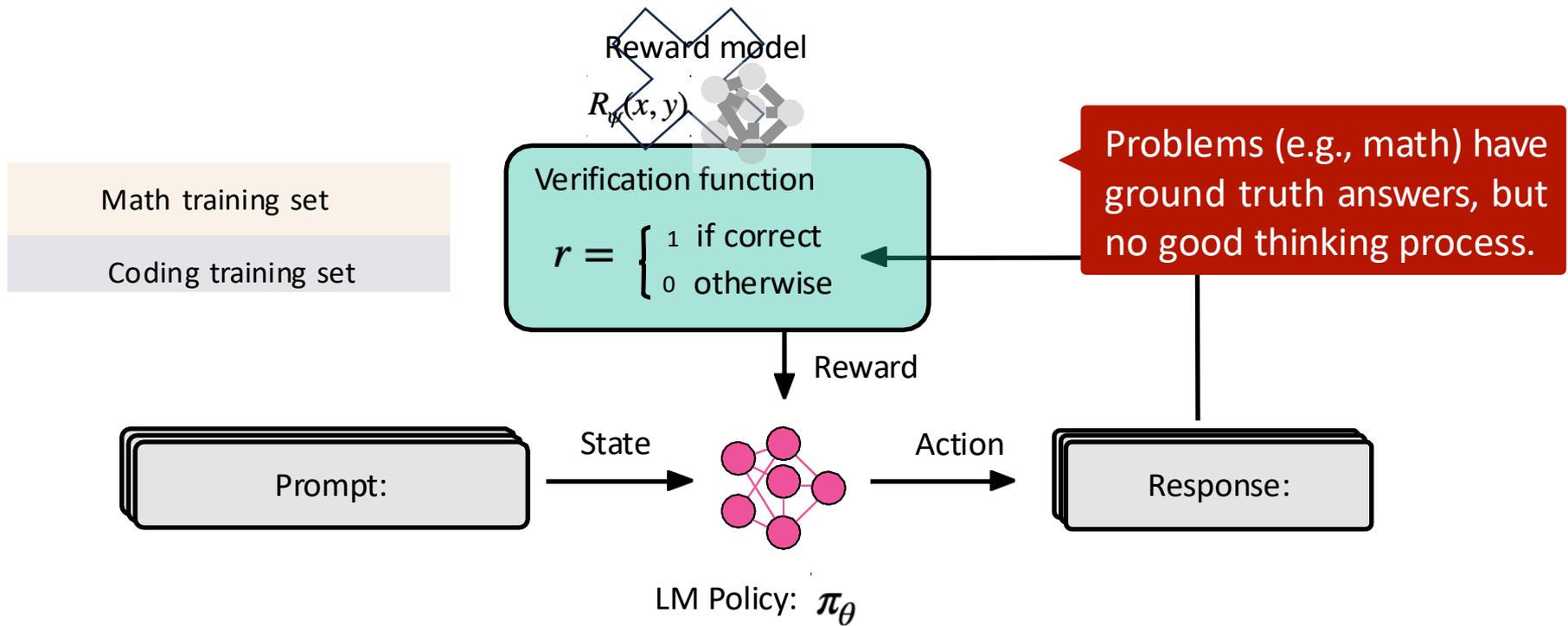


Can we just remove this complex setup and use simpler 'models'...?

RL with verifiable rewards



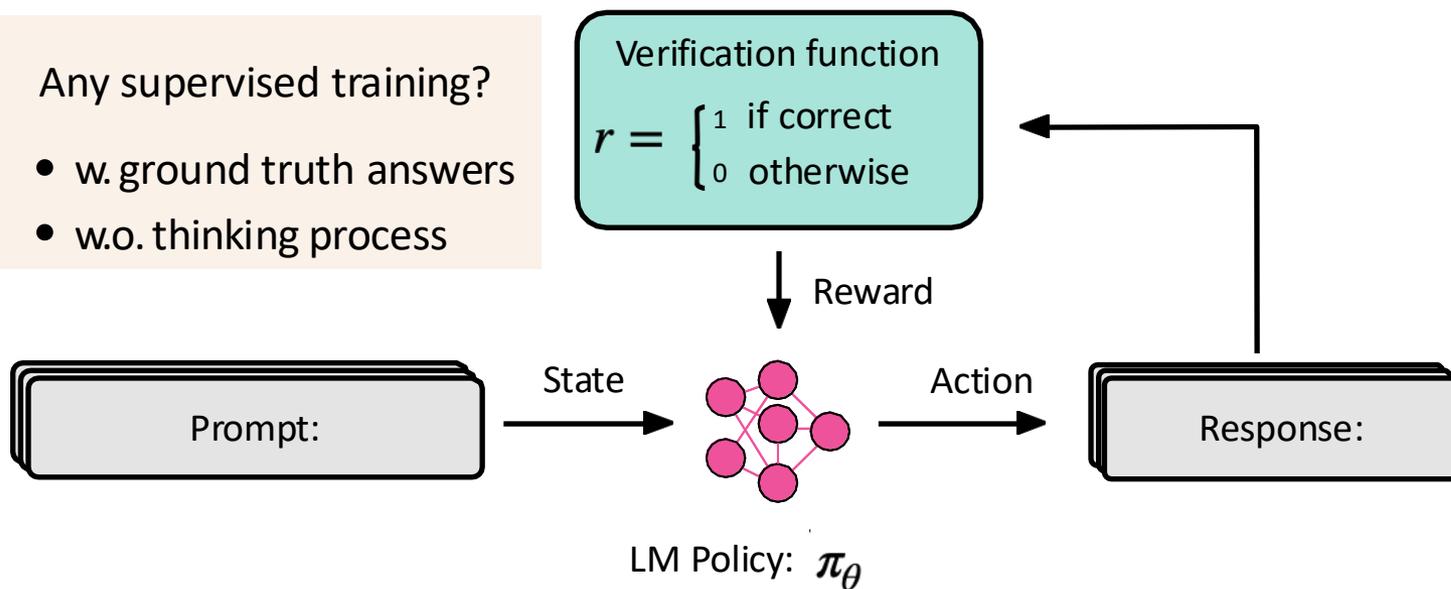
RL with verifiable rewards



RL with verifiable rewards

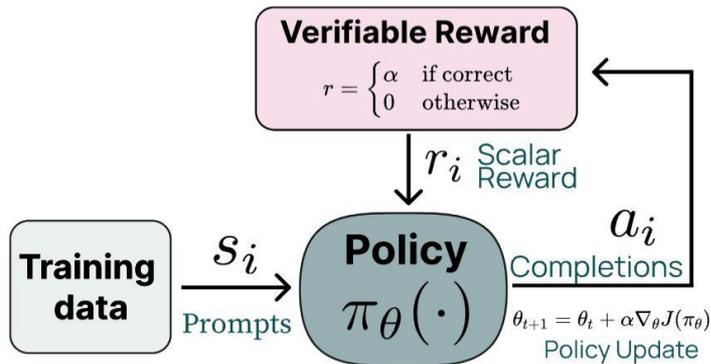
Any supervised training?

- w. ground truth answers
- w.o. thinking process



Step 3: Reinforcement learning w. verifiable rewards

-  Gold final answers or verifiable constraints.
-  intermediate chain of thoughts or not matching model.
- Classical RL! (We used PPO for optimization)



Prompt Dataset	Count	Verification
GSM8K Train	7,473	Exact match against extracted answer
MATH Train	7,500	Exact match against extracted answer
IF verifiable	14,973	Prompt-specific verifiers
<i>Total</i>	29,946	

Tulu3:

Tulu3: RLVR is not really new!

Doing RL against binary / sparse signals is not that new.

