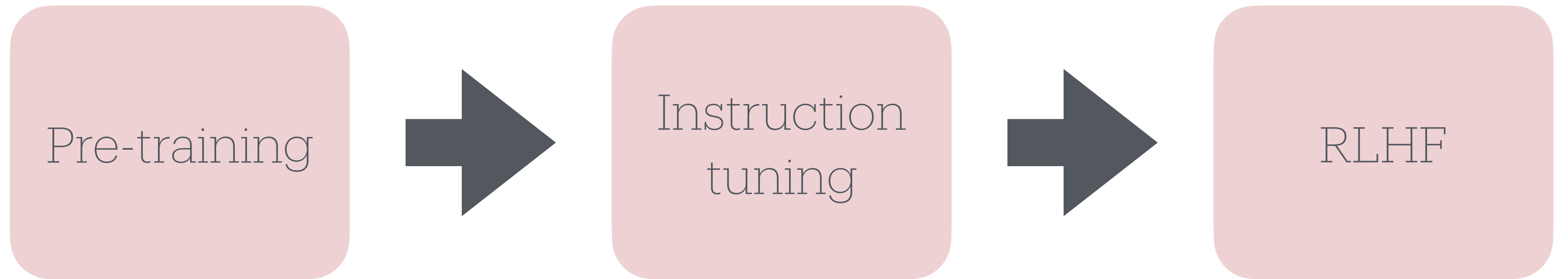


# DPO

Direct Preference Optimization

Philipp Krähenbühl, UT Austin

# RLHF

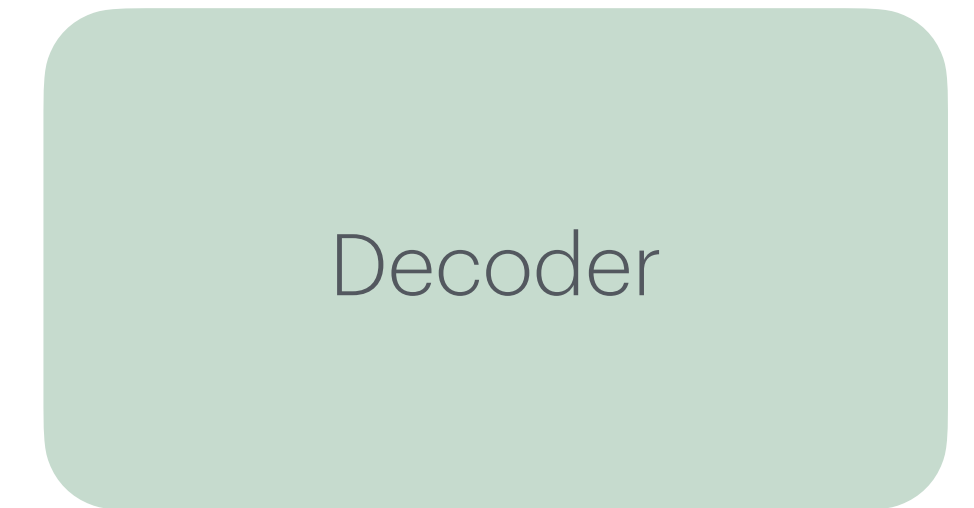




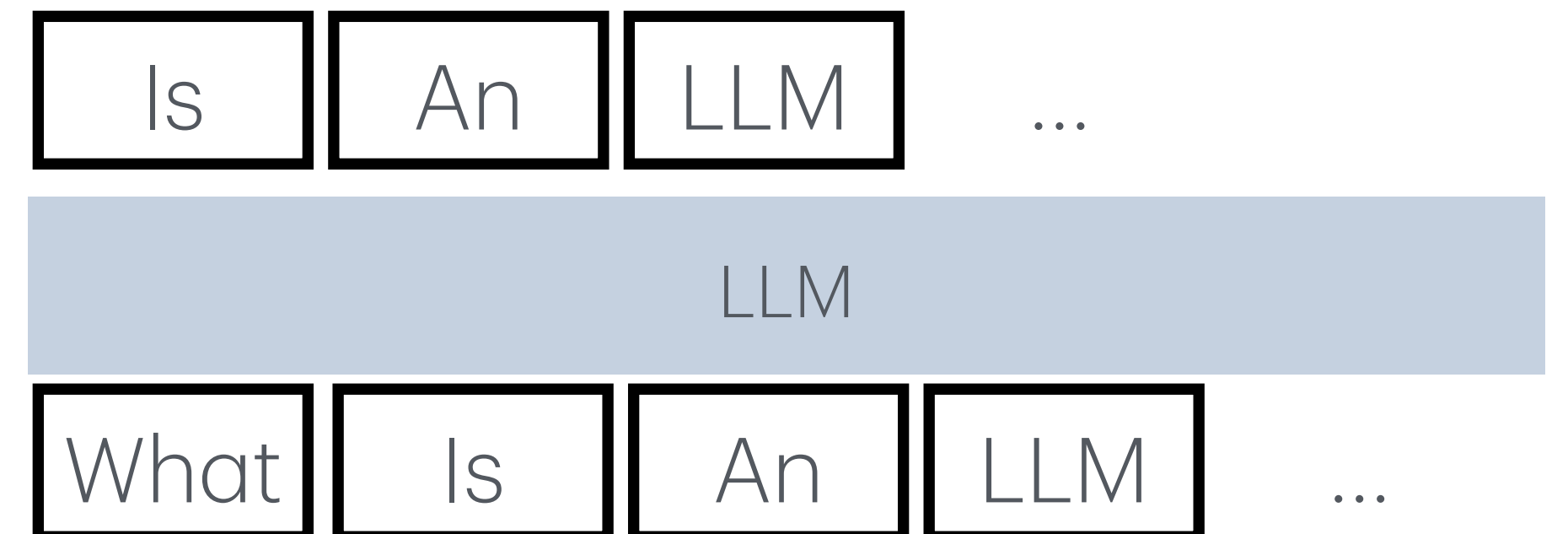
# Language Models

- Decoder-only LLMs
  - Modeling auto-regressive distribution over tokens
  - $P(\mathbf{t}) = P(t_1)P(t_2 | t_1)P(t_3 | t_1, t_2)P(t_4 | t_1 \dots t_3) \dots$
- Generation / Sampling:  $\mathbf{t} \sim P$

Distributions / logits



Embeddings  
Output



# Instruction tuning

- Teach a LLM to follow a certain template in answering
  - Tell LLM **what to do**
  - Does **not** tell LLM **what not to do**

> I want to refocus my diet on only eating Apples and supplements.

Let me provide information on how to make the most of eating apples as a primary food source.

**\*\*Apple selection:\*\*** Focus on eating a variety of apples to ensure you get a range of nutrients. Some apple varieties are higher in certain nutrients than others. For example:

\* Granny Smith and Braeburn apples have more malic acid, which can help with digestion.

\* Gala and Fuji apples contain more vitamins A and C.

...

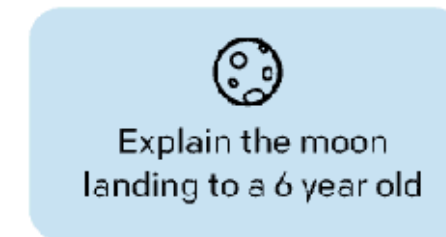
# RLHF

## Reinforcement Learning from Human Feedback

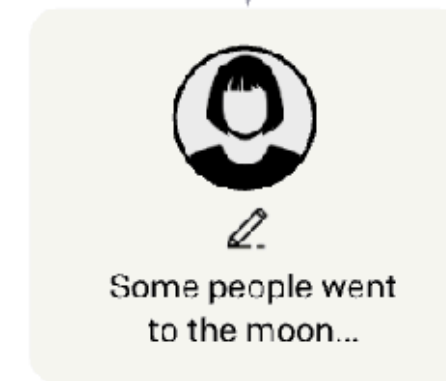
Step 1

**Collect demonstration data, and train a supervised policy.**

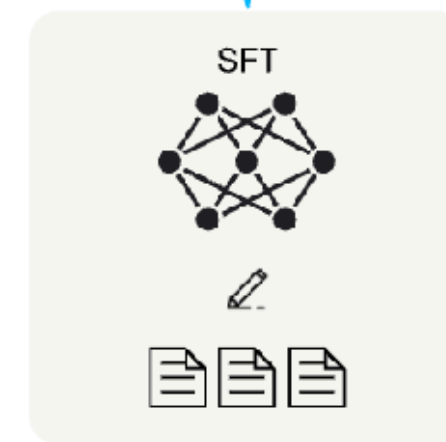
A prompt is sampled from our prompt dataset.



A labeler demonstrates the desired output behavior.



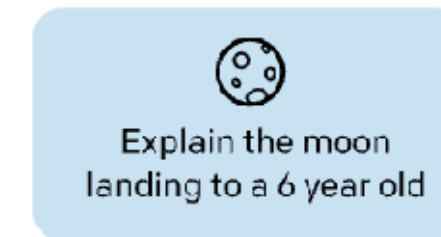
This data is used to fine-tune GPT-3 with supervised learning.



Step 2

**Collect comparison data, and train a reward model.**

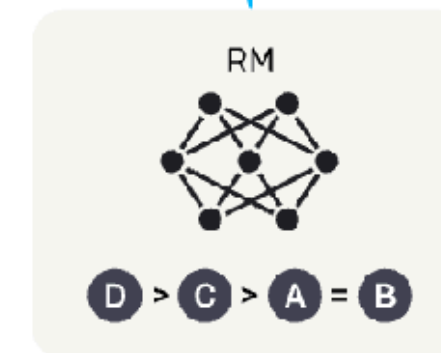
A prompt and several model outputs are sampled.



A labeler ranks the outputs from best to worst.



This data is used to train our reward model.



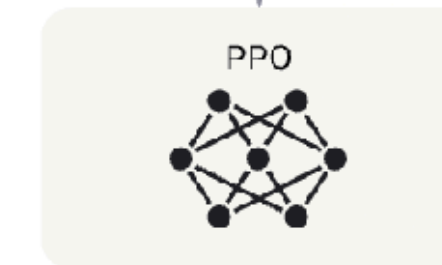
Step 3

**Optimize a policy against the reward model using reinforcement learning.**

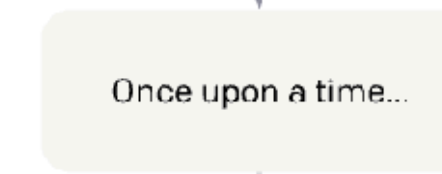
A new prompt is sampled from the dataset.



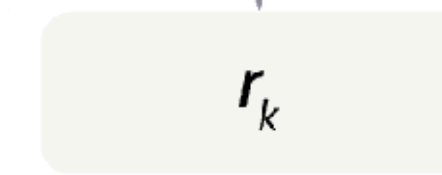
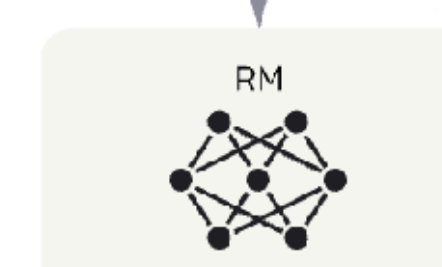
The policy generates an output.



The reward model calculates a reward for the output.



The reward is used to update the policy using PPO.



# RLHF - a recap

- Learn reward:  $\ell = E_{x, y_+, y_-} \left[ \log \sigma (r(x, y_+) - r(x, y_-)) \right]$
- Optimize:  $E_{y \sim P(\cdot | x)} \left[ (r(y, x)) \nabla \log P(y | x) \right] - \beta D_{KL} \left[ P(y | x) | P_{ref}(y | x) \right]$

# DPO

- Learn reward:  $\ell = E_{x, y_+, y_-} \left[ \log \sigma (r(x, y_+) - r(x, y_-)) \right]$
- Optimize:  $E_{y \sim P(\cdot | x)} \left[ (r(y, x)) \nabla \log P(y | x) \right] - \beta D_{KL} \left[ P(y | x) | P_{ref}(y | x) \right]$
- Closed form solution:  $P(y | x) = \frac{1}{Z(x)} P_{ref}(y | x) \exp \left( \frac{1}{\beta} r(x, y) \right)$



# DPO

- Learn reward:  $\ell = E_{x, y_+, y_-} [\log \sigma (r(x, y_+) - r(x, y_-))]$
- Optimize:  $E_{y \sim P(\cdot | x)} [(r(y, x)) \nabla \log P(y | x)] - \beta D_{KL} [P(y | x) | P_{ref}(y | x)]$
- Closed form solution:  $P(y | x) = \frac{1}{Z(x)} P_{ref}(y | x) \exp \left( \frac{1}{\beta} r(x, y) \right)$
- $r(x, y) = \beta \frac{P(y | x)}{P_{ref}(y | x)} + \beta \log Z(x)$

# DPO

- Learn reward:  $\ell = E_{x, y_+, y_-} \left[ \log \sigma \left( r(x, y_+) - r(x, y_-) \right) \right]$ 
  - Closed form  $\ell_{DPO} = E_{x, y_+, y_-} \left[ \log \sigma \left( \beta \frac{rP(x, y_+)}{P_{ref}(x, y_+)} - \beta \frac{rP(x, y_-)}{P_{ref}(x, y_-)} \right) \right]$
- Optimize:  $E_{y \sim P(\cdot | x)} \left[ (r(y, x)) \nabla \log P(y | x) \right] - \beta D_{KL} \left[ P(y | x) | P_{ref}(y | x) \right]$ 
  - Closed form solution:  $P(y | x) = \frac{1}{Z(x)} P_{ref}(y | x) \exp \left( \frac{1}{\beta} r(x, y) \right)$
  - $r(x, y) = \beta \frac{P(y | x)}{P_{ref}(y | x)} + \beta \log Z(x)$

# DPO

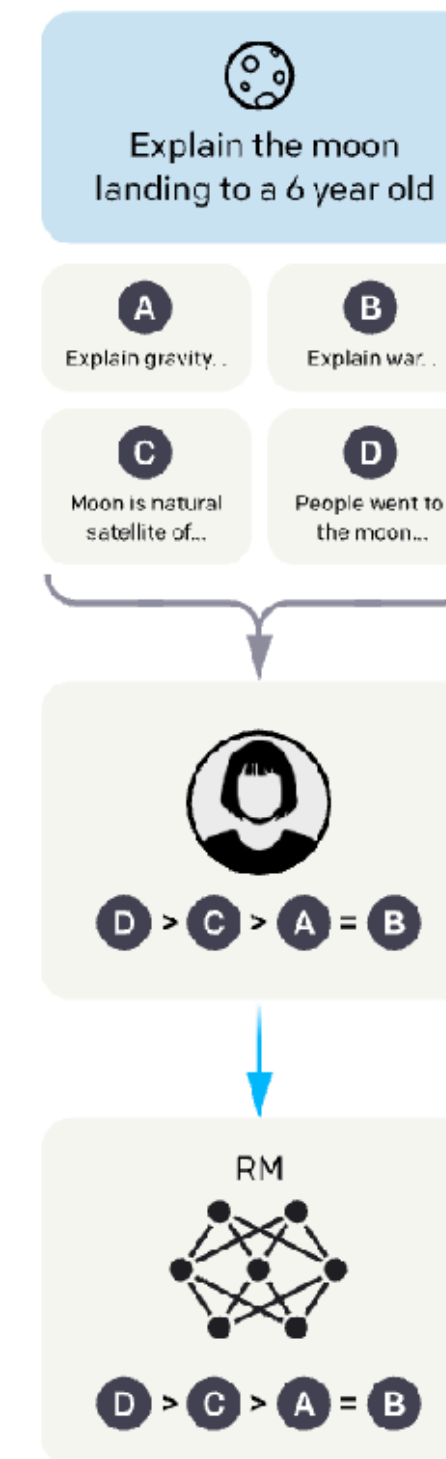
- Closed form solution to reward models + RL
- Supervised learning
- Easy to implement
- Efficient

$$\ell_{DPO} = E_{x,y_+,y_-} \left[ \log \sigma \left( \beta \frac{rP(x, y_+)}{P_{ref}(x, y_+)} - \beta \frac{rP(x, y_-)}{P_{ref}(x, y_-)} \right) \right]$$

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

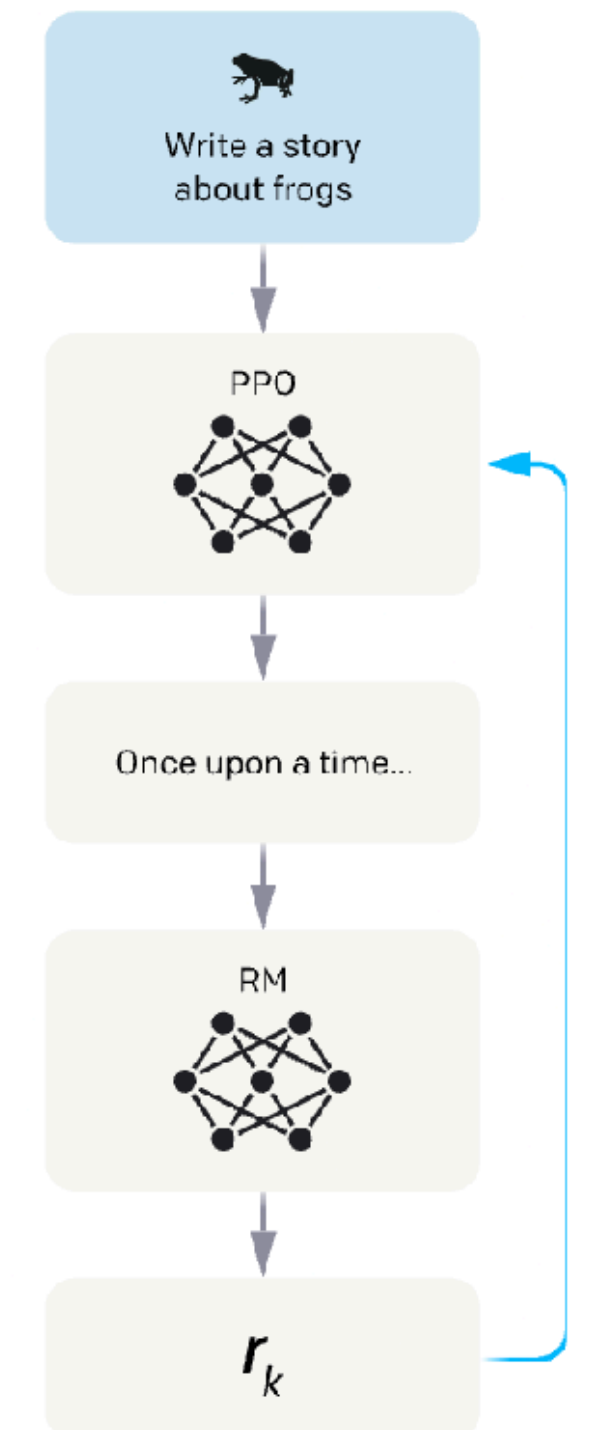
**Optimize a policy against the reward model using reinforcement learning.**

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.



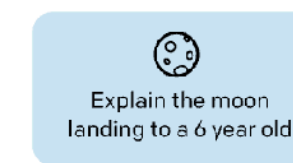
# DPO vs RLHF

$$\ell_{DPO} = E_{x, y_+, y_-} \left[ \log \sigma \left( \beta \frac{rP(x, y_+)}{P_{ref}(x, y_+)} - \beta \frac{rP(x, y_-)}{P_{ref}(x, y_-)} \right) \right]$$

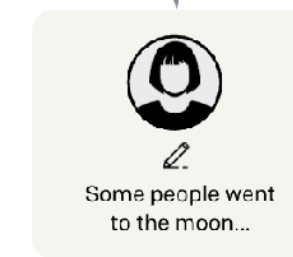
- DPO
  - Easier to make work
  - Can only learn on preference data
  - Generally produces long outputs
- RLHF
  - Requires quite a bit of RL knowledge
  - Higher ceiling (can use smaller preference data, larger fine-tuning data)

Step 1  
**Collect demonstration data, and train a supervised policy.**

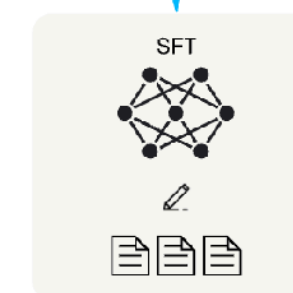
A prompt is sampled from our prompt dataset.



A labeler demonstrates the desired output behavior.

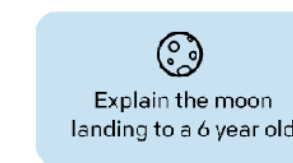


This data is used to fine-tune GPT-3 with supervised learning.

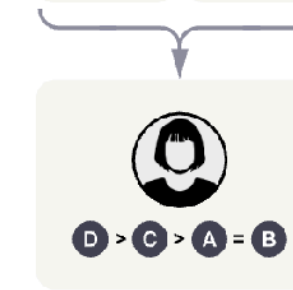


Step 2  
**Collect comparison data, and train a reward model.**

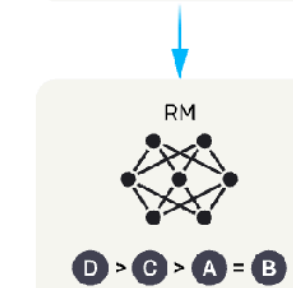
A prompt and several model outputs are sampled.



A labeler ranks the outputs from best to worst.



This data is used to train our reward model.

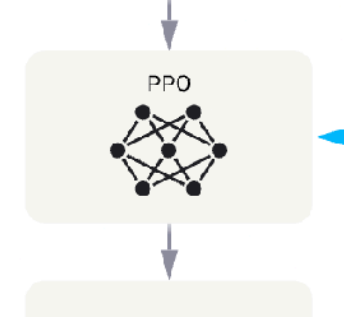


Step 3  
**Optimize a policy against the reward model using reinforcement learning.**

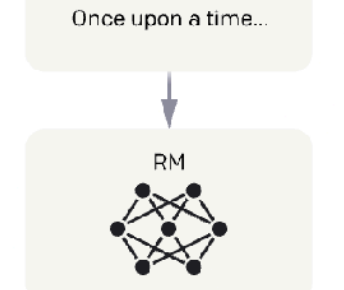
A new prompt is sampled from the dataset.



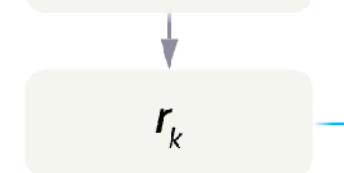
The policy generates an output.



The reward model calculates a reward for the output.



The reward is used to update the policy using PPO.



# Full Picture



# References

- [1] Training language models to follow instructions with human feedback. Ouyang etal 2022.
- [2] Direct Preference Optimization: Your Language Model is Secretly a Reward Model, Rafailov etal 2023.