Alignment of Language Models
(DPO)

Tanmoy Chakraborty
Associate Professor, IIT Delhi
https://tanmoychak.com/

Policy Gradient/PPO for LLM alignment

- Collect human preferences (x, y_+, y_-)
- Learn a reward model

$$\phi^* = \underset{\phi}{\operatorname{argmax}} \sum_{(x, y_+, y_-) \in D} \log \sigma(r_\phi(x, y_+) - r_\phi(x, y_-))$$

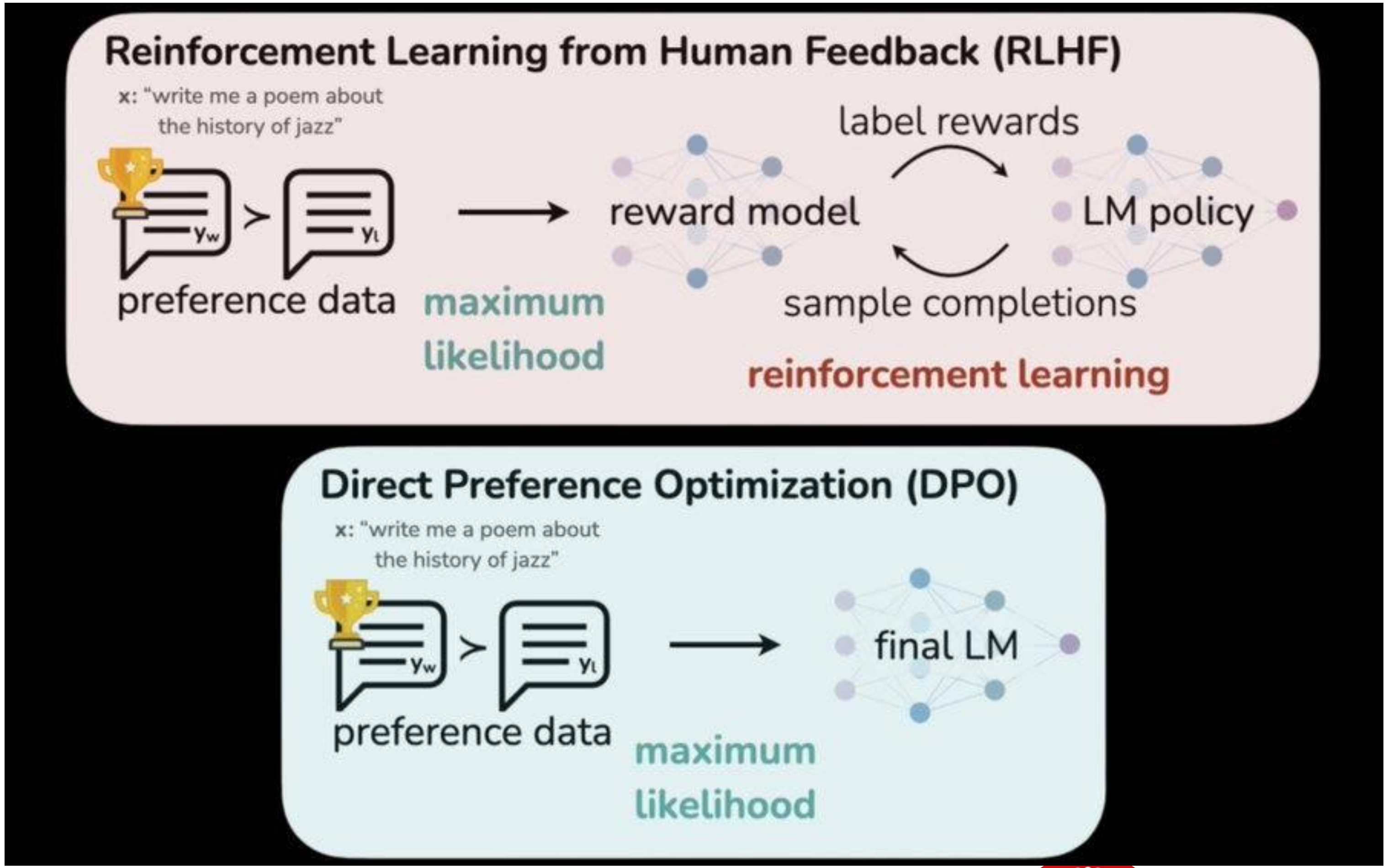
- Train the policy

$$\theta^* = \underset{\theta}{\operatorname{argmax}} E_{\pi_\theta(y|x)} r_{\phi^*}(x, y) - \beta . KL(\pi_\theta(y|x) || \pi_{ref}(y|x))$$

- Optionally
  - Also learn the value function

Question: Why do we need this intermediate step of learning reward model?

Direct Preference Optimization on preferences



Credit: https://arxiv.org/pdf/2305.18290

The non-parametric case

Assume that the policy & reward model can be arbitrary

- Learn a reward model

$$r^* = \underset{r}{\operatorname{argmax}} \sum_{(x,y_+,y_-)\in D} \log \sigma(r(x,y_+) - r(x,y_-))$$

→ optimize this exactly

- Train the policy

$$\pi^* = \underset{\pi}{\operatorname{argmax}}\, E_{\pi(y|x)} r^*(x,y) - \beta . KL(\pi(y|x) || \pi_{ref}(y|x))$$

→ optimize this also exactly

Primary idea of DPO: Cut out the middle-man r^*

The optimal policy & reward (π^*, r^*)

- Question: What does the optimal policy look like?

$$\pi^* = \underset{\pi}{\text{argmax}}\ \overbrace{E_{\pi(y|x)} r^*(x, y) - \beta . KL(\pi(y|x)||\pi_{ref}(y|x))}\ \rightarrow$$

Regularized reward-maximization objective

$$\text{subject to } \textstyle\sum_{y \in Y} \pi(y|x) = 1$$

$$\mathcal{L}(\pi, \lambda) = E_{\pi(y|x)} r^*(x, y) - \beta KL(\pi(y|x) || \pi_{ref}(y|x)) + \lambda \left(\sum_{y \in Y} \pi(y|x) - 1 \right)$$

$$\nabla_{\pi(y_0|x)} \mathcal{L}(\pi, \lambda) = 0$$

The optimal policy & reward (π^*, r^*)

$$\mathcal{L}(\pi,\lambda) = \sum_{y \in Y} \pi(y|x)\, r^*(x,y) - \underbrace{\sum_{y \in Y} \pi(y|x) \log \frac{\pi(y|x)}{\pi_{ref}(y|x)}} + \lambda\left(\sum_{y \in Y} \pi(y|x) - 1\right)$$

$$\nabla_{\pi(y_0|x)} \mathcal{L}(\pi, \lambda) = r^*(x, y_0) - \left[1 + \log \frac{\pi(y_0|x)}{\pi_{ref}(y_0|x)}\right] + \lambda$$

We know $\nabla_{\pi^*(y_0|x)} = 0$

$$\Rightarrow \quad r^*(x, y_0) - 1 - \log \frac{\pi^*(y_0|x)}{\pi_{ref}(y_0|x)} + \lambda = 0$$

The optimal policy & reward (π^*, r^*)

$$r^*(x, y_0) + \underbrace{\lambda - 1}_{\bar{\lambda}} = \log \frac{\pi^*(y_0|x)}{\pi_{ref}(y_0|x)}$$

$$e^{r^*(x, y_0) + \bar{\lambda}} = \frac{\pi^*(y_0|x)}{\pi_{ref}(y_0|x)}$$

$$\Rightarrow \quad \pi^*(y_0|x) = \pi_{ref}(y_0|x) \exp\left(r^*(x, y_0) + \bar{\lambda}\right)$$

Since $\sum_{y \in Y} \pi^*(y|x) = 1 \Rightarrow \sum_{y \in Y} \pi_{ref}(y|x) \exp\left(r^*(x,y) + \bar{\lambda}\right) = 1$

$$\Rightarrow \quad \exp(\bar{\lambda}) = \frac{1}{\sum_{y \in Y} \pi_{ref}(y|x) \exp\left(r^*(x,y)\right)}$$

The optimal policy & reward (π^*, r^*)

Write r in terms of optimal policy

$$\pi^*(y|x) = \frac{\pi_{ref}(y|x)\exp(r^*(x,y))}{Z}$$

$$r^*(x,y_0) + \bar{\lambda} = \log \frac{\pi^*(y_0|x)}{\pi_{ref}(y_0|x)}$$

$$\Rightarrow \quad r^*(x,y_0) = \log \frac{\pi^*(y_0|x)}{\pi_{ref}(y_0|x)} - \bar{\lambda}$$

$$\boxed{r^*(x,y_0) = \log \frac{\pi^*(y_0|x)}{\pi_{ref}(y_0|x)} - \log Z}$$

The parametric policy & reward (π_θ, r_θ)

- In reality, the policy will be parametrized as a language model π_θ
- Idea: Let's parameterize the reward function in terms of the policy parameters.

$$r_\theta(x, y) = \beta . \log \frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)} - \log Z_x(\theta)$$

- Next, train these parameterized reward function directly on human-preferences.

Training the reward function

Given a pair of human preferences (x, y_+, y_-)

- Reward of the positive output

$$r_\theta(x, y_+) = \beta.\log\frac{\pi_\theta(y_+|x)}{\pi_{ref}(y_+|x)} - \log Z_x(\theta)$$

- Reward of the negative output

$$r_\theta(x, y_-) = \beta.\log\frac{\pi_\theta(y_-|x)}{\pi_{ref}(y_-|x)} - \log Z_x(\theta)$$

- Training objective

$$\underset{\theta}{\operatorname{argmax}} \sum_{(x,y_+,y_-)\in D} \log\sigma(r_\theta(x, y_+) - r_\theta\,(x, y_-))$$

The training objective

(x, y_+, y_-)

$$\log \sigma \left(r_\theta(x, y_+) - r_\theta(x, y_-) \right)$$

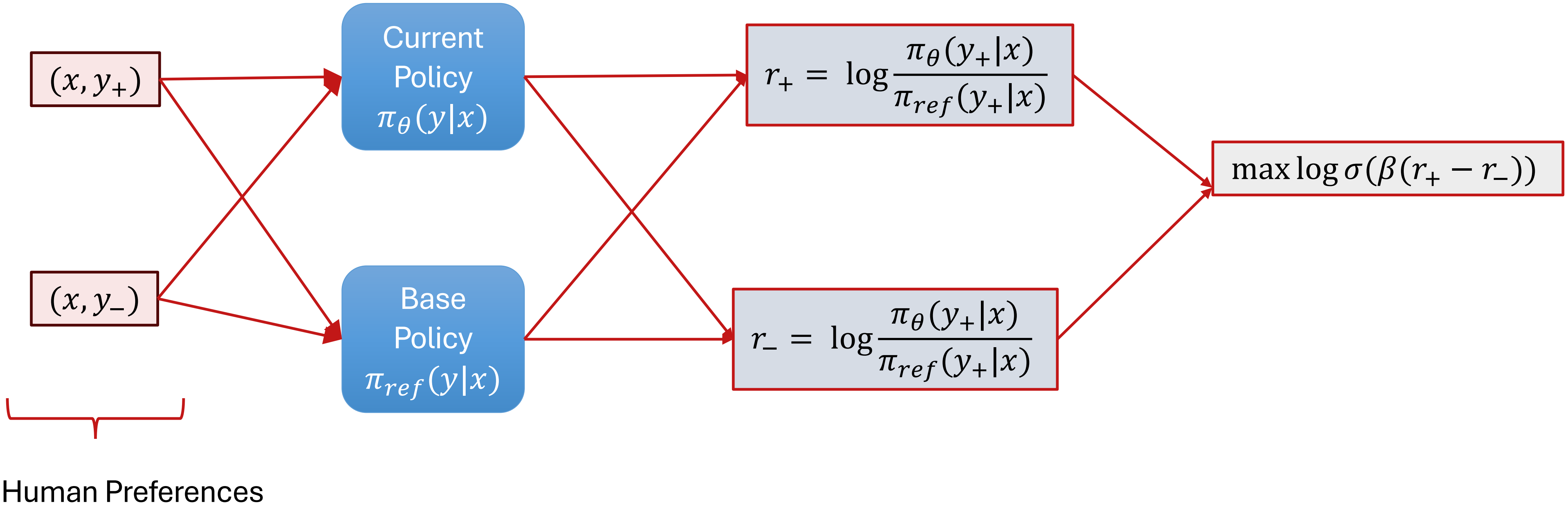
$$= \log \sigma \left(\left[\beta \log \frac{\pi_\theta(y_+|x)}{\pi_{ref}(y_+|x)} - \cancel{\log Z_x(\theta)} \right] - \left[\beta \log \frac{\pi_\theta(y_-|x)}{\pi_{ref}(y_-|x)} - \cancel{\log Z_x(\theta)} \right] \right)$$

$$= \log \sigma \left(\beta \left[\log \frac{\pi_\theta(y_+|x)}{\pi_{ref}(y_+|x)} - \log \frac{\pi_\theta(y_-|x)}{\pi_{ref}(y_-|x)} \right] \right)$$

$$= \log \frac{\exp\left(\beta \log \frac{\pi_\theta(y_+|x)}{\pi_{ref}(y_+|x)} \right)}{\exp\left(\beta \log \frac{\pi_\theta(y_+|x)}{\pi_{ref}(y_+|x)} \right) + \exp\left(\beta \log \frac{\pi_\theta(y_-|x)}{\pi_{ref}(y_-|x)} \right)}$$

$\longrightarrow$ logits of y_+

The DPO objective



Interpreting the objective

- For a positive output, $\left(\frac{\pi_\theta(y_+|x)}{\pi_{ref}(y_+|x)}\right)$ should be high
- If the reference model already assigned high probability to y_+ (say, 0.8) –
  - $\pi_\theta(y_+|x)$ will have to be relatively higher (say 0.9) →
- If the reference model assigned low probability to y_+ (say, 0.1)
  - $\pi_\theta(y_+|x)$ will be relatively higher that $\pi_{ref}(y_+|x)$ (say, 0.11)
  - In absolute terms, it might still be low

$$\frac{0.9}{0.8} \approx \frac{0.11}{0.1}$$

Adjust variable length output generated by the policy model

Interpreting β

$(0.3 - 0.003)$

$$\log\sigma\left(\beta\left[\log\frac{\pi_\theta(y_+|x)}{\pi_{ref}(y_+|x)} - \log\frac{\pi_\theta(y_-|x)}{\pi_{ref}(y_-|x)}\right]\right)$$

- Higher the value of β, more the model attempts to increase the gap between the reward of +ve and –ve outputs.

PPO vs DPO

- Ongoing debate about the efficacy of the two algorithms
- PPO is difficult to implement
- DPO is simpler – no reward function or value functions are required
- DPO is prone to generating a biased-policy that favors out-of-distribution responses.
- PPO can capture spurious correlations in the reward function.
  - Many reward functions have a length bias – Higher length outputs have higher rewards.
  - PPO training with these reward functions results in longer outputs from the policy.

Why is DPO biased?

$$\log \sigma\left(\beta\left[\log \frac{\pi_\theta(y_+|x)}{\pi_{ref}(y_+|x)} - \log \frac{\pi_\theta(y_-|x)}{\pi_{ref}(y_-|x)}\right]\right)$$

Why is DPO biased?

$$\log \sigma \left(\beta \left[\log \frac{\pi_\theta(y_+|x)}{\underbrace{\boxed{\pi_{ref}(y_+|x)}}_{0.5}} - \log \frac{\pi_\theta(y_-|x)}{\underbrace{\boxed{\pi_{ref}(y_-|x)}}_{0.5}} \right] \right)$$

$\pi_{ref}(y_o|x) = 0$

Say $y_0 = (the, the, the)$

Why is DPO biased?

At the beginning of training

$$\log \sigma \left(\beta \left[\log \frac{\overbrace{\boxed{\pi_\theta(y_+|x)}}^{0.5}}{\pi_{ref}(y_+|x)} - \log \frac{\overbrace{\boxed{\pi_\theta(y_-|x)}}^{0.5}}{\pi_{ref}(y_-|x)} \right] \right) \qquad \pi_\theta(y_o|x) = 0$$

After few steps of training, either $\pi_\theta(y_+|x)$ will increase or $\pi_\theta(y_-|x)$ will decrease

Why is DPO biased?

- If $\pi_\theta(y_+|x)$ increases, there is no issue
- If $\pi_\theta(y_-|x)$ decreases, where does the probability go?
  - Ideally, it should go to y_+
  - Most often it goes to y_+ & others (y_o)
- After training, you might end up with

y_- $\quad 0.5 \rightarrow 0.1$

$0.4 \swarrow y_+ \quad \searrow y_o$

$$\log \sigma \left(\beta \left[\log \frac{\overbrace{\boxed{\pi_\theta(y_+|x)}}^{0.5 \rightarrow 0.6}}{\pi_{ref}(y_+|x)} - \log \frac{\overbrace{\boxed{\pi_\theta(y_-|x)}}^{0.1}}{\pi_{ref}(y_-|x)} \right] \right)$$

$0 \downarrow$

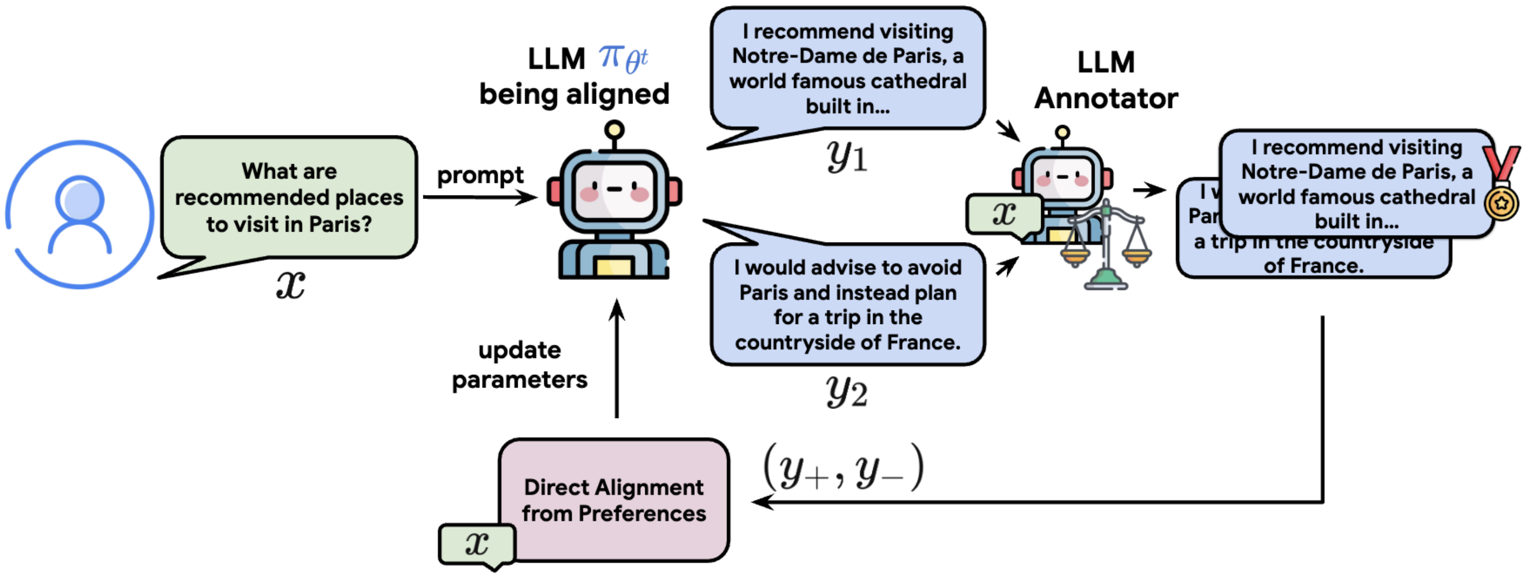
$\pi_\theta(y_o|x) = 0.3$

Say $y_0 = (the, the, the)$

- Unfortunately, this is quite common

How to deal with out-of-distribution bias in DPO?

- Possible Solution: Online DPO



- If the probability of a certain OOD output increases
  - It gets sampled in online DPO
  - Gets a low reward
  - Its probability decreases
- Resampling should be done frequently to prevent OOD bias

- Open Problem: How to deal with out-of-distribution bias in offline DPO?

Credit: Direct Language Model Alignment from Online AI Feedback

Performance Comparison: Offline vs Online DPO

Method	Win	Tie	Loss	Quality
TL;DR				
Online DPO	**63.74%**	28.57%	7.69%	**3.95**
Offline DPO	7.69%		63.74%	3.46
Helpfulness				
Online DPO	**58.60%**	21.20%	20.20%	**4.08**
Offline DPO	20.20%		58.60%	3.44
Harmlessness				
Online DPO	**60.26%**	35.90%	3.84%	**4.41**
Offline DPO	3.84%		60.26%	3.57

Table 2: Win/tie/loss rate of DPO with OAIF (online DPO) against vanilla DPO (offline DPO) on the `TL;DR`, `Helpfulness`, `Harmlessness` tasks, along with the quality score of their generations, judged by *human raters*.

Credit: Direct Language Model Alignment from Online AI Feedback

Main Takeaways

- DPO can learn the policy directly from human/AI preferences
  - No reward model or value function needed
- Can be biased towards OOD samples
- To prevent bias
  - A reward model can be trained
  - Outputs can be sampled frequently from the policy and ranked using the reward model