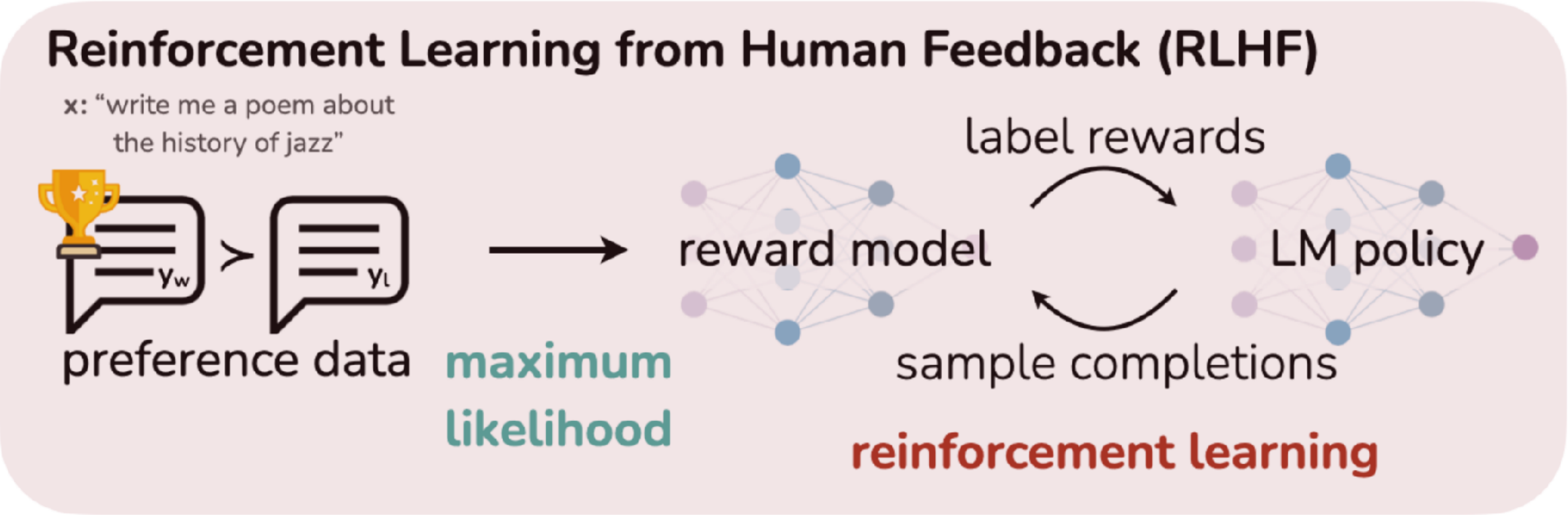
ReFoRM Reading Group

Feb 5, 2025

“Classical” RLHF



Two-stage pipeline: (1) Fitting a reward model through maximum likelihood (2) Learning the optimal policy implied by this reward through RL

Step 1: Maximum Likelihood

Goal: Train a reward model that predicts, given a prompt x, which of two responses (y_1, y_2) will be preferred by humans

- To make this tractable, we will assume there exists some **reward function** $r(x, y)$ such that the values of $r(y_1, x)$ and $r(y_2, x)$ determine the likelihood of a human preferring y_1 to y_2 in response to x
- Accomplishing our goal then reduces to learning this function

The Bradley-Terry Model

- Given i and j with "strengths" β_i and β_j, the probability of preferring i to j is:

$$\mathbb{P}(i \succ j) = \frac{1}{1 + \exp(\beta_j - \beta_i)}$$

- Given a dataset of pairwise comparisons $\mathcal{D}$, the resulting empirical log-likelihood is:

$$\frac{1}{|\mathcal{D}|} \sum_{(i,j) \in \mathcal{D}} \log \sigma(\beta_i - \beta_j)$$

- Maximum likelihood will then recover "optimal" strengths $\hat{\beta}$

Bradley-Terry in Context

- Instead of directly parameterizing the “strength” of prompts, we parameterize the reward function

- Given a dataset $\mathcal{D} = \{(x, y_1 \succ y_2)\}_{i=1}^{N}$ of prompts and preferences:

$$\text{Empirical Log-Likelihood} = \frac{1}{|\mathcal{D}|} \sum_{x, y_1, y_2} \log \sigma(r_\theta(x, y_1) - r_\theta(x, y_2))$$

- Learning r_θ via maximum-likelihood gives us the desired reward model

Step 2: Reinforcement Learning

- Relatively simple loop —
  1. Given a collection of prompts, sample completions
  2. Use the trained reward model $r_\phi(x, y)$ as the reward in the following objective:

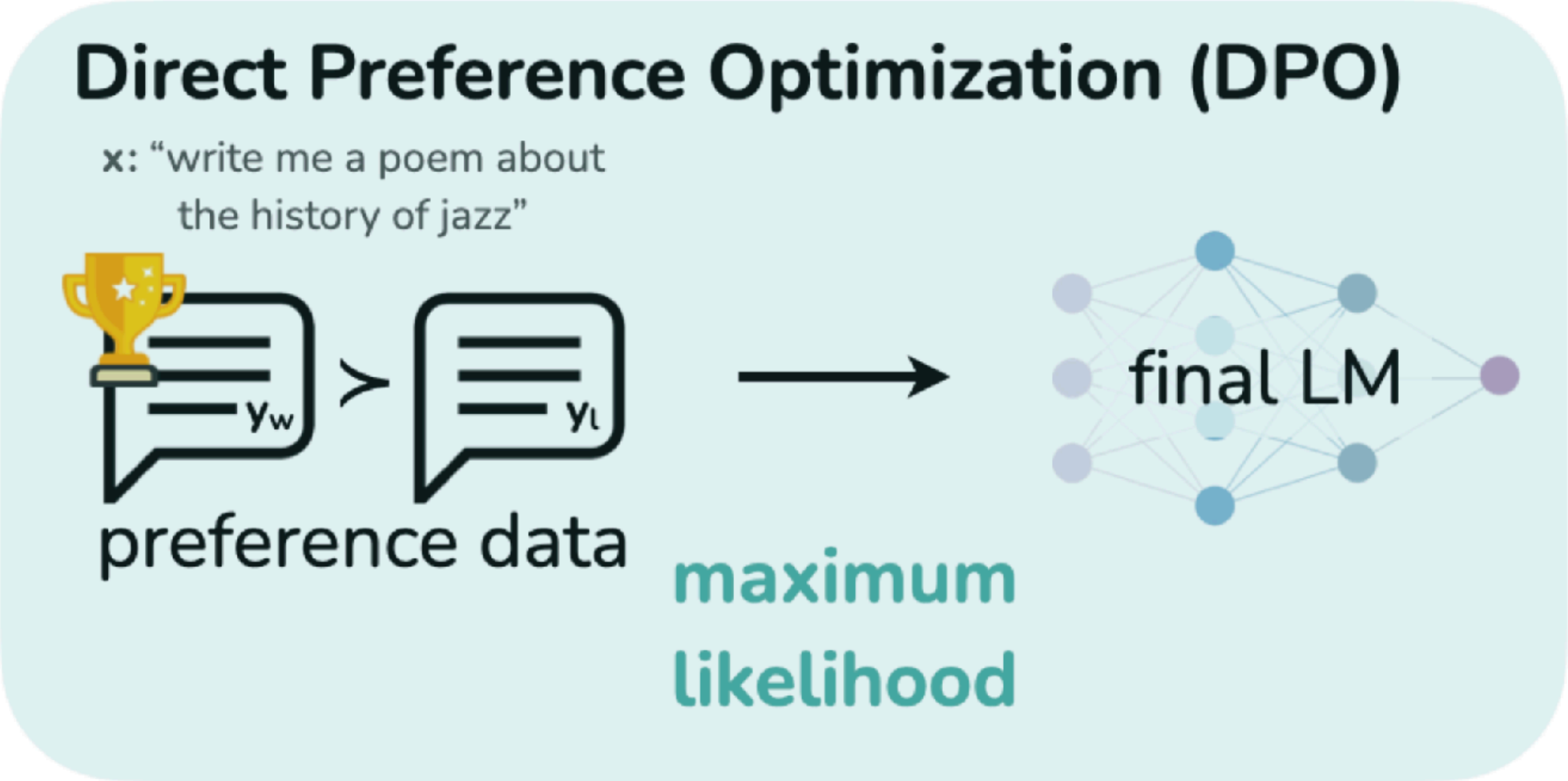
$$\max_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta(y|x)}\left[r_\phi(x, y)\right] - \beta\mathbb{D}_{\mathrm{KL}}\left[\pi_\theta(y \mid x) \mid\mid \pi_{\mathrm{ref}}(y \mid x)\right]$$

- In the above, $\pi_{\text{ref}}(y \mid x)$ is a reference policy that we do not wish to deviate too strongly from (typically the result of supervised fine-tuning)

Issues with RLHF?

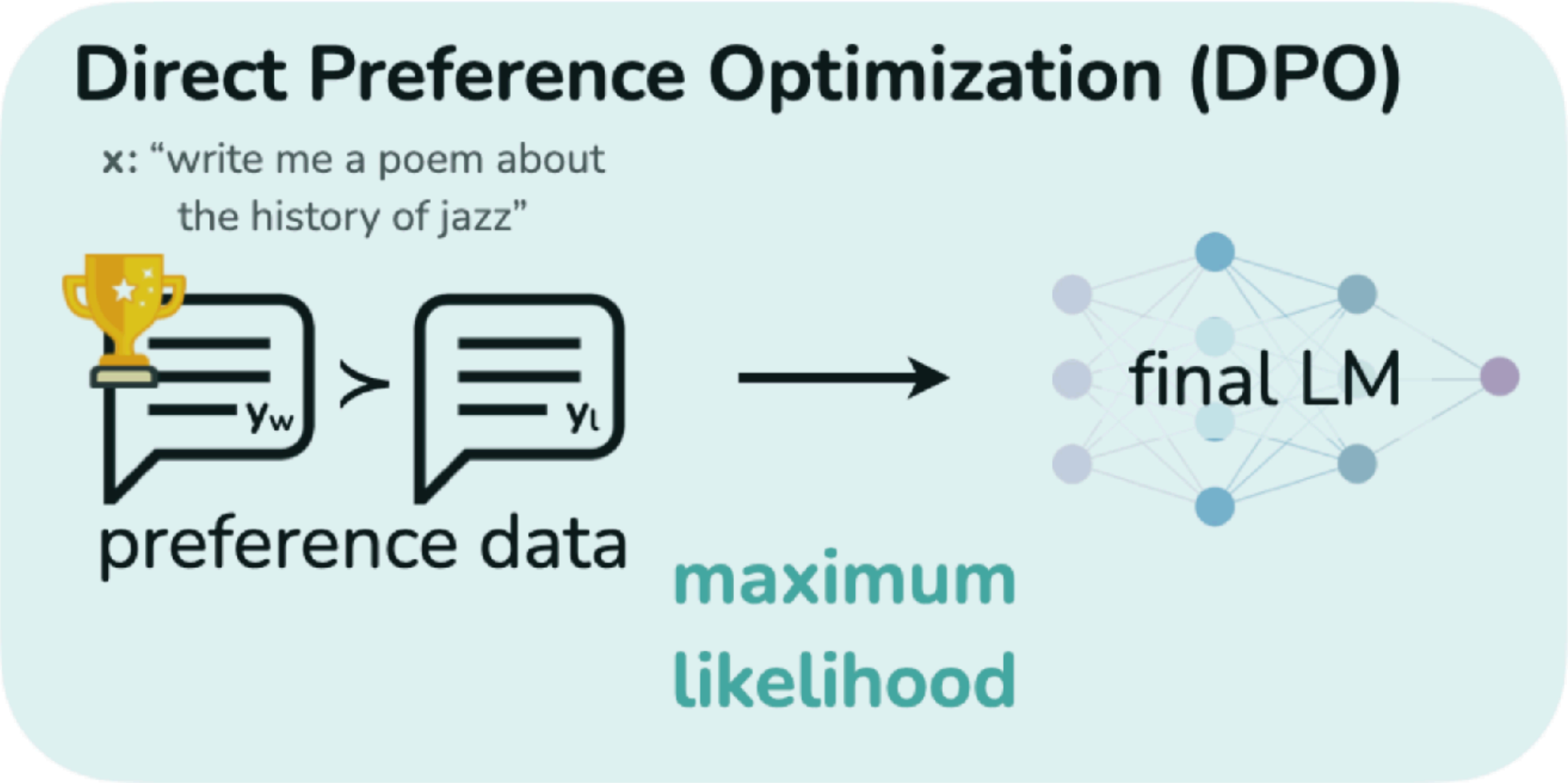
- Involves training a separate reward model
- Reinforcement learning step can be computationally expensive
- In general, the method is very **indirect** — one might wonder if preferences can be mapped to model changes directly

DPO



RLHF, without Reinforcement Learning

DPO



How is this possible?

Key Trick: Change of Variables

- Recall the RL objective is:

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

- The optimal policy π_r has a **closed-form** solution:

$$\pi_r(y \mid x) = \frac{1}{Z(x)} \pi_{\mathrm{ref}}(y \mid x) \exp\left(\frac{1}{\beta} r(x, y) \right)$$

- Upon rearrangement, the corresponding **reward function** is:

$$r(x, y) = \beta \log \frac{\pi_r(y \mid x)}{\pi_{\mathrm{ref}}(y \mid x)} + \beta \log Z(x)$$

Change of Variables cont.

- Let π^* denote the optimal policy corresponding to the **true** reward $r(x, y)$
- By the results of the previous slide, we can freely translate between the two
- In particular, we can write preference probabilities under the Bradley-Terry model as follows:

$$p^*(y_1 \succ y_2 \mid x) = \frac{1}{1 + \exp\left(\beta \log \frac{\pi^*(y_2|x)}{\pi_{\text{ref}}(y_2|x)} - \beta \log \frac{\pi^*(y_1|x)}{\pi_{\text{ref}}(y_1|x)}\right)}$$

- Note that the log-partition terms have cancelled!

Change of Variables cont.

- Something subtle has happened — **we can now directly recover π^* through maximum likelihood estimation**

- The empirical log-likelihood corresponding to our new Bradley-Terry is:

$$\frac{1}{|\mathcal{D}|}\sum_{x,y_1,y_2} \log \sigma\left(\beta \log \frac{\pi_\theta(y_1 \mid x)}{\pi_{\text{ref}}(y_1 \mid x)} - \beta \log \frac{\pi_\theta(y_2 \mid x)}{\pi_{\text{ref}}(y_2 \mid x)}\right)$$

- Maximum likelihood will give us $\hat{\pi}_\theta \approx \pi^*$ — no RL needed!

Is DPO a “Free Lunch”?

- In practice, DPO seems to underperform RLHF — e.g. Ivison, Wang et al (2024)

- Rather surprising since mathematically, DPO seems to do “as well as you can” given preference data

- **Key difference seems to be that DPO is (implicitly) an offline RL method, while RLHF is online**

Example: Catastrophic Likelihood Displacement

Razin, Malladi et al (2024)

Intuition: learning the correct differences between pairs does **not** imply good global control over behavior!