RLHF: aligning LLMs with human intents

→ start from a large pretrained LLM

→ Step 1: instruction tuning (SFT)

Limitations of instruction tuning:

→ don't learn from negative feedback

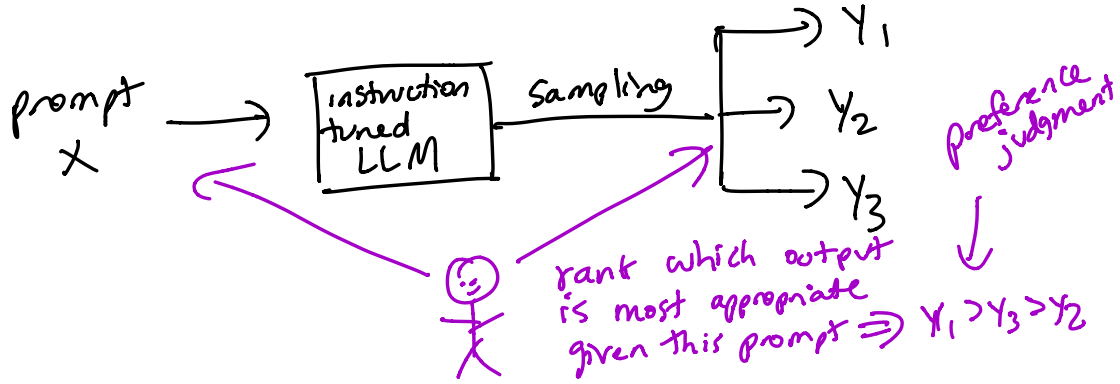
→ some prompts (e.g. creative) have many acceptable outputs, we only train on one of them

→ hard to encourage abstaining when the model doesn't know something

→ does not directly involve human prefs

---

how do we incorporate human prefs to address the above issues?



→ limitation: extremely expensive to collect

→ idea: can we train a model to predict human pref judgment

→ reward model

→ input: prompt x, output y_i

→ output: scalar score

→ Bradley-Terry pairwise preference model

→ y_w ⇒ preferred by humans

→ y_L ⇒ not preferred

→ $r(x, y)$ ⇒ reward for output y given x, scalar score

$$p(y_w > y_L \mid x) = \frac{\exp(r(x, y_w))}{\exp(r(x, y_w)) + \exp(r(x, y_L))}$$

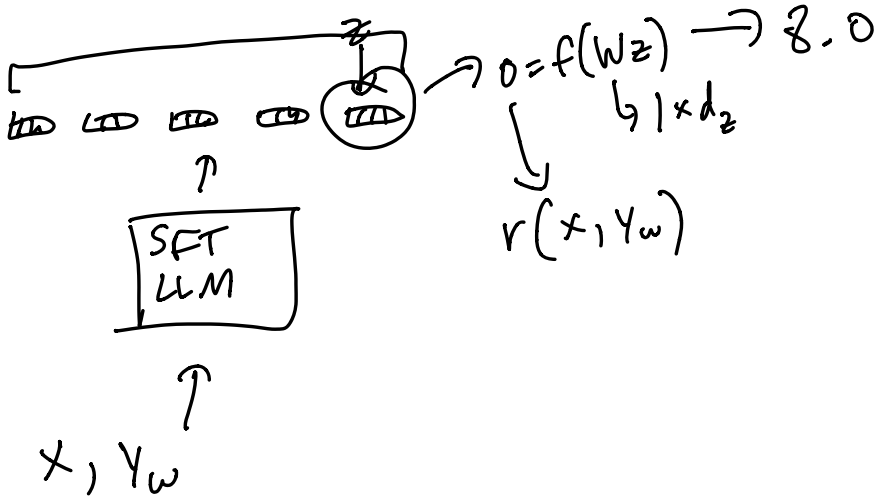
$$L = -\log\left(\quad\right)$$

↓ simplify

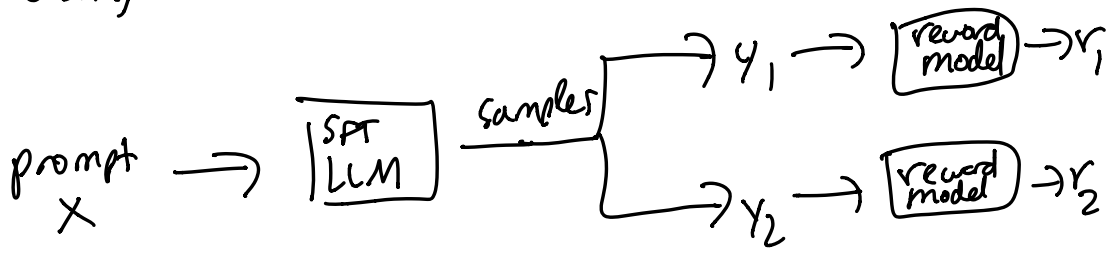
$$\sigma(x) = \frac{1}{1+e^{-x}}$$

$$L = -\log \sigma\left(r(x, y_w) - r(x, y_L)\right)$$

→ intuition: good sample y_w's reward should be greater than y_l's reward



Using the reward model



how do we use this ↗ to align LLMs to human prefs?

1. "best-of-n" sampling (rejection sampling)
   - ↳ generate n samples for a given prompt, score each w/ reward model, choose sample with highest reward
   - → very expensive

2. Just fine-tune the LM to maximize $p(y_w|x)$
   ↳ RAFT

3. use reinforcement learning to increase $p(y_w|x)$ by a small amount, decrease $p(y_L|x)$ by a small amount, where amounts are functions of $R(x, y_w)$, $R(x, y_L)$

RLHF step 3:

↳ we observe a reward only after generating a complete sequence

π_{ref} ⇒ SFT LLM checkpoint

π ⇒ current policy model
↑ ⇒ init. to π_{ref}
final aligned model

$$\max_{\pi} \; \mathop{E}_{x,y} \Big[\underbrace{r(x,y)}_{\text{reward}} - \underbrace{\beta D_{KL}\big(\pi(y|x) \,||\, \pi_{ref}(y|x)\big)}_{\text{KL penalty to prevent huge deviations from } \pi_{ref}} \Big]$$

$$D_{KL}(\pi(y|x) \,||\, \pi_{ref}(y|x)) =$$

$p_{FINAL\ RLHF\ MODEL}$ → $\log \frac{\pi(w_i | w_{1 \ldots i-1}, x)}{\pi_{ref}(w_i | w_{1 \ldots i-1}, x)}$

→ optimize using PPO algorithm
  → Schulman, 2016 → ChatGPT, GPT4
→ can also use REINFORCE
  ↳ Williams 1992
  ↳ Gemini

Final RLHF pipeline: