$$L(\theta) = -\frac{1}{N} \sum_{i=1}^{N} A_i \log \pi_\theta(y_i | x)$$

↳ advantage: $r(x, y_i) - b$

↳ baseline b:

↳ GRPO: group mean reward

↳ PPO: learned value fn

↳ PPO vs. REINFORCE

↳ PPO: we take multiple steps per batch, we use clipping on the policy ratio π_θ / π_{old} to prevent policy from shifting too far

↳ all of these methods need reward $r(x, y)$

↳ not always easily verifiable

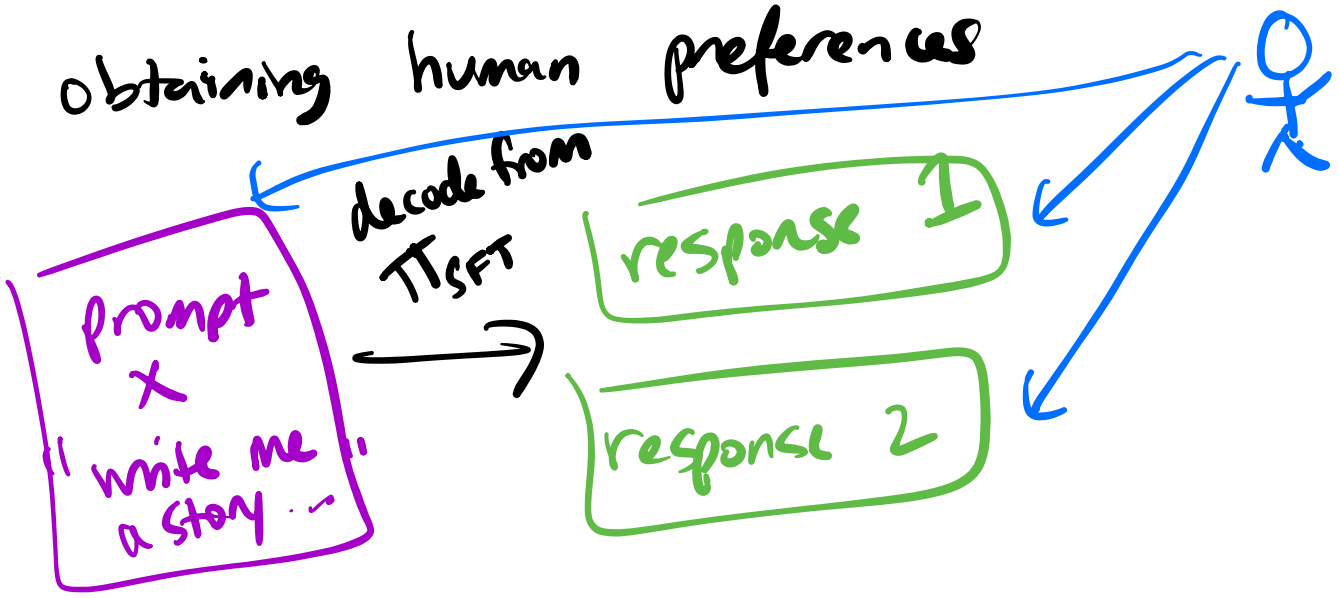
↳ today: learning a reward model from human preferences

↳ RLHF: PPO w/ learned reward model

+ KL penalty to prevent going too far from the π_{SFT}

---

obtaining human preferences



↳ why not ask ppl to rate each response from e.g 1-10

↳ exact numbers are subjective

↳ people are more reliable
(ie they agree more w/ each other)
when simply ranking two responses

---

preference data: ↳ approach to training LMs first introduced in OpenAI InstructGPT

prompt x, preferred response y_w
losing response y_L

reward model that we are training

reward model $RM(x, y)$ and outputs a score $r(x, y)$

$$r_w = RM(x, y_w), \quad r_L = RM(x, y_L)$$

↳ two forward passes thru RM

goal: train RM such that $r_w > r_L$

↳ we can treat these two scores r_w and r_L as logits, and just compute softmax over them

$$\text{softmax}([r_w, r_L]) = \left[\frac{e^{r_w}}{e^{r_w}+e^{r_L}}, \frac{e^{r_L}}{e^{r_w}+e^{r_L}}\right]$$

$$= \left[P(y_w \succ y_L \mid x), P(y_L \succ y_w \mid x)\right]$$

↳ Bradley-Terry preference model

training loss

target: $[1, 0]$

cross-entropy:

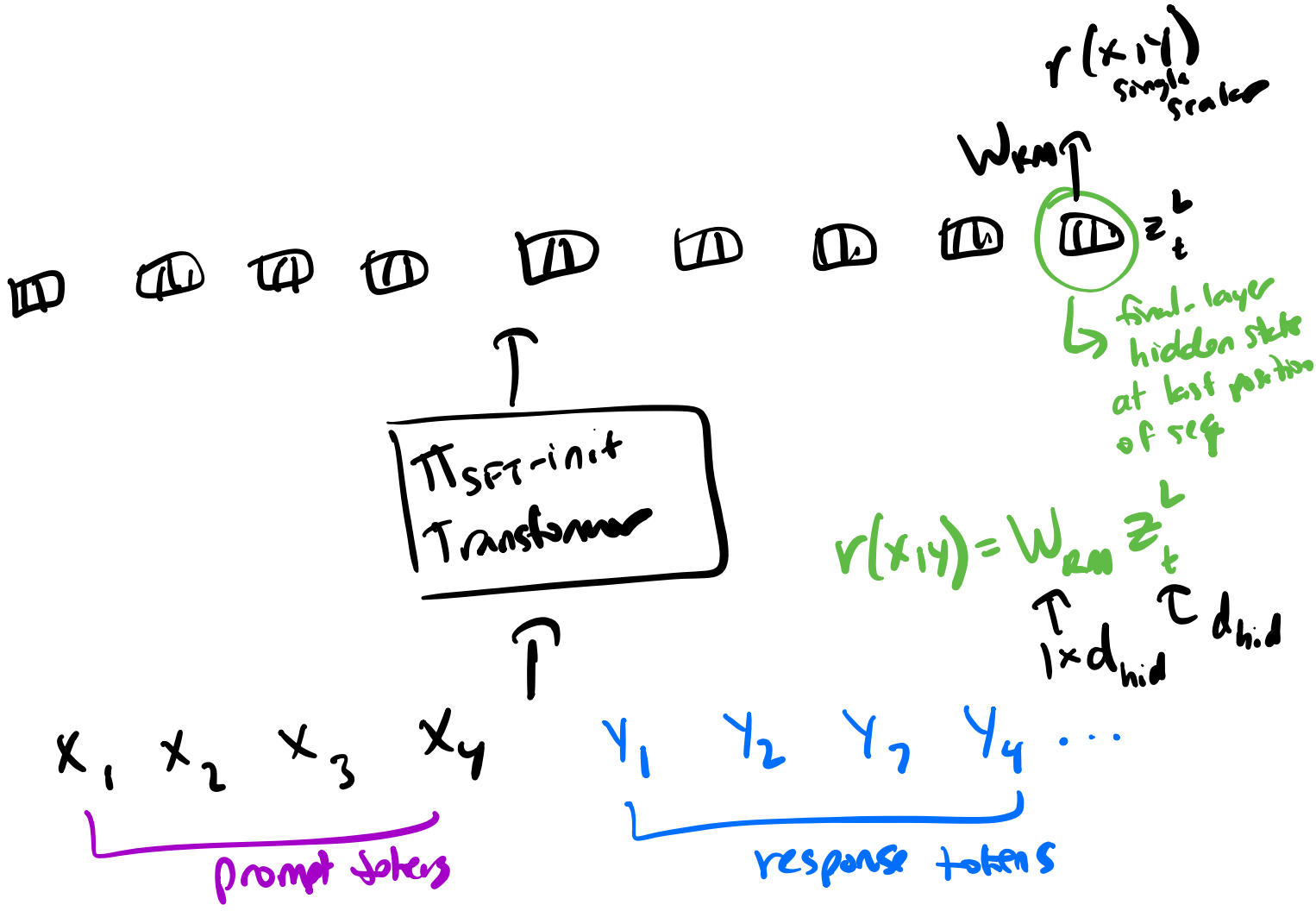
$$L(\theta_{RM}) = -\left[1 \cdot \log P(y_w \succ y_L \mid x) + \cancel{0 \cdot \log P(y_L \succ y_w \mid x)}\right]$$

$$= -\log P(y_w \succ y_L \mid x)$$

reward model architecture:

↳ Transformer init. with π_{SFT}

↳ further train w/ Bradley-Terry loss on preference pairs



W_{rm} $(1 \times d_{hid})$ is a new param, randomly init and trained thru backprop

↳ same setup as the value network in PPO

↳ we train this RM w/ Bradley-Terry loss on human preferences

↳ then, we can plug in $r_{RM}(x, y)$ into GRPO, PPO, REINFORCE, etc

↳ problem: r_{RM} is not as meaningful as verifable reward

↳ noisy, subjective data

↳ y_w, y_L come from one model (π_{SFT}) and so your RM may not generalize well to responses from diff models (π_θ)

↳ if we train our policy to maximize $E[r_{RM}(x, y)]$, it often finds

responses that people wouldn't like but are scored highly by RM

↳ reward hacking