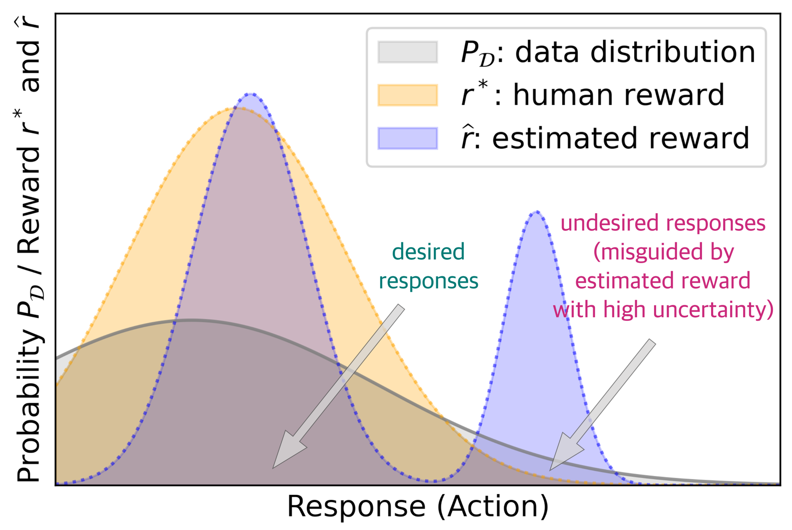
Provably Mitigating Overoptimization in RLHF: Your SFT Loss is Implicitly an Adversarial Regularizer

Zhihan Liu[1] Miao Lu[2] Shenao Zhang[1] Boyi Liu[3]
Hongyi Guo[1] Yingxiang Yang[3] Jose Blanchet[2] Zhaoran Wang[1]

[1]Northwestern University [2]Stanford University [3]ByteDance Inc.

Background on RLHF

❶ Aligning generative models with human preference via RLHF typically suffers from *overoptimization*, where an imperfectly learned reward model can misguide the generative model to output undesired responses.



action	a	b	c
$r^\star$	1	0.5	0
Dataset	$\mathcal{D} = \{(a, b, 1)\}$		
π^{ref}	0.45	0.45	0.1
π^{DPO}	0.5	0	**0.5**
π^{RPO}	**1.0**	0	0

❷ **Question:** How to *mitigate reward overoptimization* in RLHF in a principled and efficient manner for better alignment?

Our Contributions

❶ **A theoretical algorithm under general function approximation.**

❷ **An equivalent and easy-to-implement practical objective: Regularized Preference Optimization (RPO).**

❸ **Empirical evaluations on the LLM Alignment Tasks.**

Algorithm and Theory

Theoretical algorithm: Maximin objective.

❶ Output the policy **max**imizing an adversarially chosen reward model that **min**imizes the sum of: **(a)** the MLE loss for estimating the underlying reward; and **(b)** a reward expected value term as a penalty that prevents spuriously high reward estimation caused by data uncertainty and insufficient coverage.

❷ We prove the finite-sample suboptimality gap of (Maximin Objective) as $\widetilde{\mathcal{O}}(C^2_{\mathrm{coverage}}\sqrt{\mathcal{N}_{\mathcal{R}}/N})$ when competing with any LLM in terms of the underlying human reward. ($N = \#$ of preference data)

Practical algorithm: Minimax objective (RPO).

❶ Under mild assumptions: (Maximin Obj.) $\Leftrightarrow$ (Minimax Obj.)

❷ With reward-policy duality reparametrization, the minimax objective takes a quite simple form: Imitation (SFT) Loss + Preference Optimization Loss!

❸ We select the baseline policy π^{base} as the chosen policy of the preference dataset (does not induce extra computation overhead).

Model Name	GSM8K (%)	ARC Easy (%)	ARC Challenge (%)	MBPP (Pass @1) Normal (%)	MBPP (Pass @1) Plus (%)
RPO	**49.9**	**79.1**	49.8	54.2	**46.3**
DPO	45.3	75.7	**50.0**	54.2	43.9
Ref.	45.4	75.0	45.8	50.3	44.2
Released	47.3	77.6	48.6	**54.5**	44.7

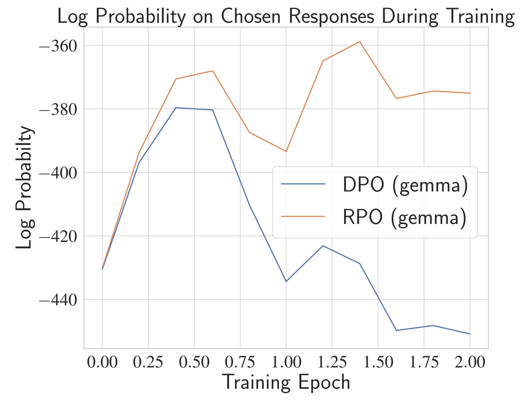
Experiments

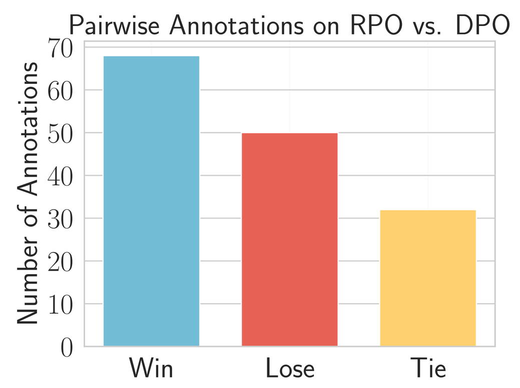
Conlusions based on Experiments:

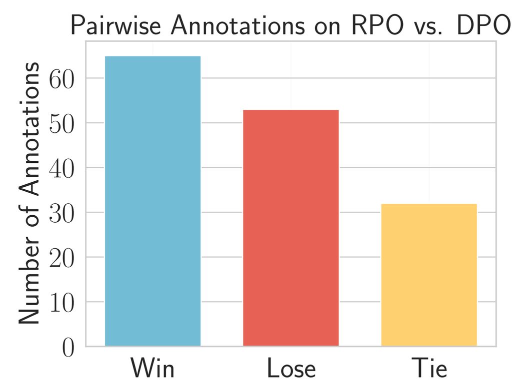
❶ **RPO alleviates overoptimization.**

❷ **RPO improves alignment for in-data distribution.**

❸ **RPO improves the alignment and reasoning benchmark performance.**







Model Name	MT-Bench Score	AlpacaEval 2.0 LC win rate (%)	AlpacaEval 2.0 win rate (%)
RPO (beta)	**7.381**	**23.28**	**21.01**
Ref. (beta)	5.088	7.19	4.69
DPO (beta)	7.278	21.15	17.27
`zephyr-beta-7b`	7.200	13.20	10.99

Model Name	MT-Bench Score	AlpacaEval 2.0 LC win rate (%)	AlpacaEval 2.0 win rate (%)
RPO (gemma)	**7.916**	**15.51**	**13.85**
Ref. (gemma)	7.266	8.35	4.61
DPO (gemma)	7.688	15.36	13.69
Released	7.719	14.78	12.14

Detailed Algorithm Design

$$\hat{\pi} \in \operatorname*{argmax}_{\pi \in \Pi} \min_{r \in \mathcal{R}} \left\{ \eta \cdot \mathbb{E}_{\substack{x \sim d_0, a^1 \sim \pi(\cdot|x), \\ a^0 \sim \pi^{\mathrm{base}}(\cdot|x)}} \Big[r(x, a^1) - r(x, a^0) - \beta \cdot \mathrm{KL}\big(\pi(\cdot|x) \| \pi^{\mathrm{ref}}(\cdot|x)\big) \Big] + \mathcal{L}_{\mathcal{D}}(r) \right\}. \quad \text{(Maximin Objective)}$$

$$\min_{\theta \in \Theta} \left\{ \mathcal{L}_{\mathrm{RPO}}(\theta) := \eta\beta \cdot \underbrace{\mathbb{E}_{x \sim d_0, a^0 \sim \pi^{\mathrm{base}}(\cdot|x)} \Big[-\log(\pi_\theta(a^0|x)) \Big]}_{\text{Imitation (SFT) loss}} + \underbrace{\mathcal{L}_{\mathcal{D}} \left(\beta \cdot \log \left(\frac{\pi_\theta(\cdot|\cdot)}{\pi^{\mathrm{ref}}(\cdot|\cdot)} \right) \right)}_{\text{Preference opt. loss}} \right\}. \quad \text{(Minimax Objective / RPO)}$$