How Reinforcement Learning Is Used to Fine-Tune Large Language Models

A Comparison of PPO-based RLHF and DPO

Ningyuan Xie

Master of Computer Science
Siebel School of Computing and Data Science
University of Illinois at Urbana-Champaign
nxie3@illinois.edu

CS440: Artificial Intelligence, Summer 2026

What This Talk Teaches

1. Why pretraining leaves a gap that post-training must fill
2. Why supervised imitation alone is a limited fix
3. How PPO-based RLHF turns preferences into a reward and an RL update
4. How DPO trains from preferences without a separate RM or PPO loop
5. What tradeoffs remain between the two approaches

Pretraining Makes a Text Predictor; Assistants Need More

What pretraining does: predict the next token on internet text.

What scaling alone does not guarantee:

- Better instruction following
- Answers that stay accurate and safe
- Helpful behavior for real users

Why that matters:

- Chat assistants need more than fluent continuation
- Models used in applications need helpful, safe instruction following

The Core Gap

A pretrained model asked *"Explain quantum entanglement in one sentence"* may continue the sentence fragment rather than answer the question.

Post-training fills this gap: further training on demonstrations, preferences, or other feedback.

Supervised Fine-Tuning (SFT): Useful, But Incomplete

SFT approach: train on written demonstration responses with a supervised next-token loss.

What SFT achieves:

- Trains on written demonstration responses
- Common initialization for later RL
- In studied settings, helps lock in response format

The fundamental limitation:

- Trains only on a *fixed* demonstration set
- No direct notion of which response is "better"
- Classical imitation learning warns about compounding errors under distribution shift

Reinforcement Learning from Human Feedback (RLHF)

Key idea: replace single demonstrations with *pairwise comparisons*; train a reward model; optimize the policy with RL.

Stage 1: SFT initialization

- Train on demonstrations to establish format

Stage 2: Reward modeling

- Annotators compare or rank multiple responses as pairs $(y_w \succ y_l)$
- Fit reward model $r_\phi(x, y)$ via Bradley–Terry loss:

$$\mathcal{L}_{\text{RM}} = -\mathbb{E}\left[\log\left(\sigma(r_\phi(x, y_w) - r_\phi(x, y_l))\right)\right]$$

Stage 3: Policy optimization (PPO + KL)

- Maximize expected reward while staying near a reference policy:

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

Why KL + PPO?

Without a KL penalty, the policy can exploit reward-model errors or collapse to degenerate high-reward text. RLHF is the broader paradigm; PPO is a popular Stage 3 optimizer.

InstructGPT result

In pairwise human evaluations, 175B InstructGPT was preferred to 175B GPT-3 $85 \pm 3\%$ of the time (Ouyang et al., 2022).

Direct Preference Optimization (DPO)

Key insight: for KL-regularized preference optimization with a fixed comparison set, the best policy can be written directly in terms of the reward.
Substituting that into Bradley–Terry yields a loss on preference pairs with *no separate reward network*.

DPO loss (Rafailov et al., 2023):

$$\mathcal{L}_{\text{DPO}} = -\mathbb{E}\left[\log \sigma\left(\beta \log \frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta \log \frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)}\right)\right]$$

What this means in plain terms:

- Raise likelihood of preferred responses relative to dispreferred ones
- The policy encodes a reward-like ranking over responses
- The reference enters through log-probability ratios
- No separate reward fitting and no PPO sampling loop

DPO vs. RLHF at a glance

PPO-based RLHF	DPO
Reward model + PPO	Preferences used directly
On-policy samples	Fixed preference pairs
Separate RM and policy stages	Single offline preference loss

TL;DR result

DPO ≈61% vs. PPO ≈57% GPT-4 win rate against reference summaries (Rafailov et al., 2023).

The Lesson in One Pass

1. **Pretraining** yields a strong text predictor; post-training is what makes an assistant.
2. **SFT** trains on fixed demonstrations; useful start, limited notion of "better."
3. **PPO-based RLHF** scores fresh policy samples with a learned reward; separate RM and policy stages.
4. **DPO** trains an offline preference loss on a fixed comparison set, without a separate RM or PPO loop.

Still debated

How much new capability RL adds versus reshaping existing ones, and when an explicit RL loop is required.