# RLHF for Open-Ended Instruction Following

Chetan Goenka

**Abstract**

We study reinforcement learning from human feedback (RLHF) for a small open-weight instruction-following model, fine-tuning Qwen2.5-1.5B-Instruct on roughly 5,000 preference pairs from the WildChat benchmark. Every trained policy is evaluated by head-to-head win rate against the frozen base model, judged by an LLM (GPT-5.4). We implement and compare three offline preference-optimization methods (DPO, IPO, AOT) and three online policy-gradient methods (GRPO, DrGRPO, GSPO), then extend the online setup with three ideas: a pessimistic reward-model ensemble, a leave-one-out advantage baseline, and replay-based online IPO. Among baselines, IPO is strongest offline (78.8% win rate) and DrGRPO strongest online (70.0%). Our best extension, a pessimistic four-checkpoint reward-model ensemble with tuned optimization, reaches a **75.0%** online win rate, beating the strongest online baseline.

## 1 Setup

The benchmark is roughly 5,000 human preference pairs from WildChat, split into preference pairs for offline training and reward-model evaluation and held-out prompts for online rollouts. The base policy is Qwen2.5-1.5B-Instruct; the frozen base model serves as both the reference policy $\pi_{\text{ref}}$ and the evaluation opponent. The primary metric throughout is win rate against that frozen base model on a held-out prompt set, judged by GPT-5.4. We ask: (i) how do offline methods compare? (ii) how do online methods compare with one another and with offline? and (iii) can changes to the reward signal, the advantage estimator, or the training objective improve the online policies?

## 2 Methods

**Offline preference optimization.** All offline methods operate on preference triples $(x, y^+, y^-)$ through a reference-corrected margin that measures how much more the current policy prefers the chosen response than the rejected one, relative to the frozen reference:

$$\Delta_\theta(x, y^+, y^-) = \log \pi_\theta(y^+ \mid x) - \log \pi_\theta(y^- \mid x) - \log \pi_{\text{ref}}(y^+ \mid x) + \log \pi_{\text{ref}}(y^- \mid x). \tag{1}$$

**DPO** applies a logistic loss to this margin. **IPO** instead regresses the margin toward a fixed target $\frac{1}{2\beta}$, which is more conservative and prevents the model from satisfying the objective by simply pushing both probabilities down. **AOT** compares the full minibatch distributions of chosen vs. rejected scores (sorting each and penalizing quantiles where rejected exceeds chosen), a weaker but more global condition than pairwise ranking.

**Reward model.** For the online methods we train a reward model $r_\phi(x, y)$ with the Bradley-Terry objective, $\mathcal{L}_{\text{RM}} = -\log \sigma\big(r_\phi(x, y^+) - r_\phi(x, y^-)\big)$, reaching 83.6% held-out pair accuracy.

**Online policy gradients.** Online methods sample $G$ responses per prompt, score them with $r_\phi$, compute group-relative advantages, and update with a clipped surrogate:

$$A_{i,j} = \frac{r_{i,j} - \mu_i}{\sigma_i + \varepsilon}, \qquad \mu_i = \tfrac{1}{G}\sum_j r_{i,j}, \qquad \sigma_i = \sqrt{\tfrac{1}{G}\sum_j (r_{i,j} - \mu_i)^2}. \tag{2}$$

**GRPO** clips per-token likelihood ratios; **DrGRPO** drops the standard-deviation and per-sequence length normalization, changing the update's length dependence; **GSPO** clips at the sequence level, treating the whole response as one action.

**Extensions.** We targeted three different components of the online pipeline:

- **Pessimistic reward-model ensemble (reward signal).** We aggregate several reward-model checkpoints by an elementwise minimum, $r_{\text{ens}}(x, y) = \min_k r_{\phi_k}(x, y)$, so a response is rewarded only when *every* member agrees it is good, making the signal harder to game.
- **Leave-one-out advantage baseline (advantage estimator).** We replace the group mean with a per-sample baseline that excludes the response being scored, $A_{i,j} = r_{i,j} - \frac{1}{G-1} \sum_{k \neq j} r_{i,k}$, a lower-variance estimator of the expected reward.
- **Replay-based online IPO (training objective).** Rather than discarding rollouts after each update, we store them as max-vs-rest preference pairs in a FIFO replay buffer and train an IPO objective on minibatches sampled from it, reusing preference signal across steps.

## 3 Results

**Baselines.** Table 1 reports win rates against the frozen base model. Offline methods outperform online ones on this benchmark, with IPO strongest overall (78.8%) and DrGRPO the best online method (70.0%). Notably, IPO wins most often under the judge despite the *lowest* reference-corrected preference accuracy, and GSPO ranks second online despite the lowest fraction of scores above reference; the internal proxy metrics correlate only loosely with the downstream LLM-judge win rate.

Table 1: Win rate against the frozen base model (GPT-5.4 judge).

| Category | Method | Win rate |
|---|---|---|
| Offline | DPO | 0.766 |
| | IPO | **0.788** |
| | AOT | 0.726 |
| Online | GRPO | 0.659 |
| | DrGRPO | **0.700** |
| | GSPO | 0.684 |

**Reward-model ensemble.** The ensemble was our most effective extension. The best run aggregates four reward-model checkpoints and, combined with tuned optimization (3 PPO epochs, learning rate $\eta = 2 \times 10^{-5}$, and four-path sampling on GRPO), reaches a **75.0%** online win rate, a 5-point gain over the best online baseline (DrGRPO, 70.0%). As Figure 2 shows, the gain comes from the interaction of the pessimistic signal and the optimization changes: the extra learning rate and PPO epoch helped only in combination with the ensemble reward.

**Leave-one-out advantage.** The LOO baseline (Table 2) improves GRPO and GSPO with no other changes, but slightly hurts DrGRPO, consistent with the fact that DrGRPO already removes standard-deviation normalization, making it more sensitive to changes in the advantage estimator.

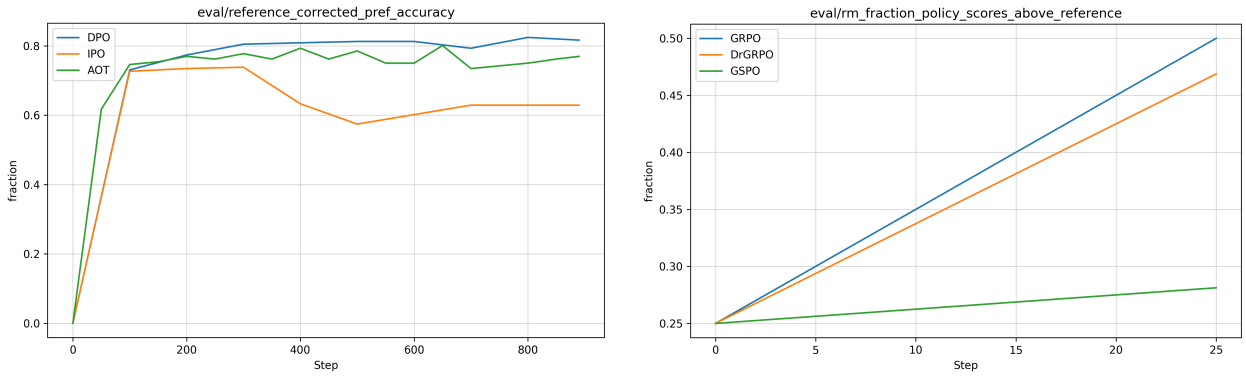


Figure 1: Training diagnostics for the baseline methods. Left: reference-corrected preference accuracy for the offline methods. Right: fraction of on-policy generations scoring above the reference for the online methods.

Table 2: Leave-one-out advantage baseline vs. the corresponding online baseline (GPT-5.4 judge).

| Method | LOO win rate | Baseline win rate |
|---|---|---|
| GRPO | **0.708** | 0.659 |
| GSPO | **0.700** | 0.684 |
| DrGRPO | 0.653 | **0.700** |

**Replay-based online IPO.** This extension did not pan out. The best variant (buffer size 200, $\beta = 0.1$, with reward weighting) reached a reward-model win rate of only 0.36 against the reference, below GRPO (0.50) and DrGRPO (0.46), and ablations over buffer size and $\beta$ moved the needle little. The likely cause: the replay buffer feeds stale off-policy data into an IPO objective that lacks the clipped surrogate and KL regularization that keep the policy-gradient methods stable.

## 4 Discussion

Two takeaways stand out. First, **the pessimistic ensemble works because it is harder to hack**: min-aggregation means the policy cannot earn reward by exploiting a weakness in any single checkpoint, which pushes it toward responses that are good on the whole rather than good according to one idiosyncratic reward model. Second, **internal metrics are imperfect proxies for the real objective**: DPO's reference-corrected accuracy looked strongest throughout training, yet its generations were fluent but shallow (numbered lists and safe filler) and lost to IPO under the actual judge. Optimizing the proxy too hard is precisely how reward hacking begins.

With more budget, the most promising direction is the ensemble: scaling to more members (8, 12, and beyond) to see whether the pessimism keeps paying off, and ablating mean vs. min aggregation to isolate whether it is the pessimism specifically, rather than simply averaging more signals, that drives the gain.

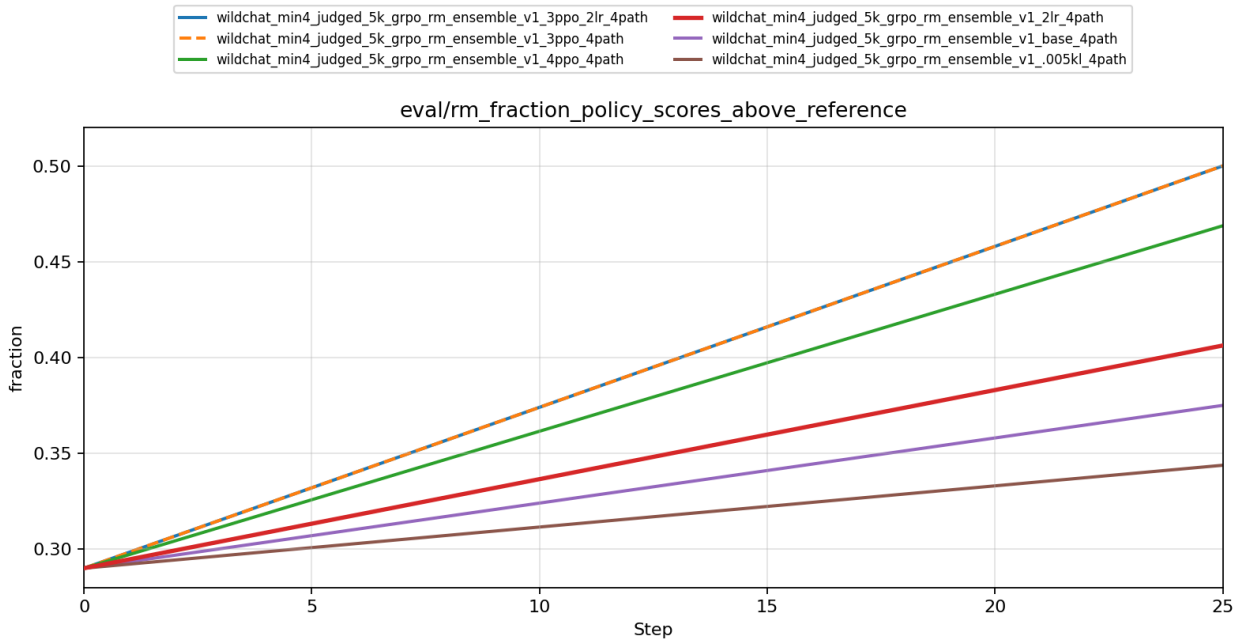


Figure 2: Online training under the reward-model ensemble across six configurations varying PPO depth, learning rate, KL coefficient, and sampling width. The strongest configuration (3 PPO epochs, doubled learning rate, four-path sampling) reaches the top of this metric and is the one that achieves the 75.0% judged win rate.