REINFORCED DPO: SINGLE-STAGE PREFERENCE OPTIMIZATION PROVABLY EQUIVALENT TO RLHF

Anonymous authors
Paper under double-blind review

ABSTRACT

Direct preference optimization (DPO) aligns language models in a single training stage, offering a simpler and more stable alternative to the two-stage pipeline of reinforcement learning from human feedback (RLHF). Yet despite being derived from the RLHF objective, DPO fails to reproduce the desirable properties of RLHF: it learns only from offline preference data without on-policy learning, and it does not effectively regularize the policy toward the reference model. In this paper, we first prove that DPO provably deviates from RLHF in both respects. We then trace this inequivalence to a gap in DPO's derivation: the change of variables from rewards to Q-functions silently drops the reward-boundedness constraint that makes RLHF's KL-regularized RL stage well posed. Reinstating this constraint, which takes the form of a Bellman constraint on Q-functions, yields Reinforced Direct Preference Optimization (ReDPO). We prove that ReDPO is equivalent to RLHF and thus inherits its bound on the divergence from the reference policy. Crucially, the Bellman constraint induces a value flow that propagates both the successor Q-values and the reference log-probability into the current Q-value, carrying out KL-regularized RL through dynamic programming. Experiments show that ReDPO clearly outperforms DPO and its variants, while keeping the policy closer to the reference model and mitigating the forgetting of general capabilities. To our knowledge, ReDPO is the first single-stage preference optimization method that is provably equivalent to RLHF.

1 INTRODUCTION

Reinforcement learning from human feedback (RLHF) (Christiano et al., 2017) is a central paradigm for aligning large language models (LLMs) with human preferences. It proceeds in two stages: a reward model is first learned from human preference data, and the policy is then optimized against this reward using KL-regularized reinforcement learning. RLHF has been instrumental in the development of leading LLMs such as ChatGPT (Ouyang et al., 2022) and Claude (Bai et al., 2022a).

However, this two-stage pipeline is complex and often unstable to train. Direct preference optimization (DPO) (Rafailov et al., 2023; 2024) addresses these issues by collapsing the two stages into one. Specifically, DPO exploits the closed-form solution to the KL-regularized RL problem: a change of variables from rewards to policies turns the reward-learning objective into an objective defined directly over policies. The resulting supervised objective is simple and stable to optimize, and has inspired numerous variants (Gheshlaghi Azar et al., 2024; Hong et al., 2024; Meng et al., 2024).

Although DPO is derived from the RLHF objective, several studies find that it fails to reproduce the desirable properties of RLHF. First, DPO learns only from a fixed, offline preference dataset, whereas RLHF can draw on additional on-policy rollouts to reduce estimation error and thereby achieve better performance (Li et al., 2023). This offline training has also been identified as a partial cause of the squeezing effect, a failure mode unique to DPO in which the probabilities of both the preferred and the dispreferred responses decrease during training (Ren & Sutherland, 2025; Razin et al., 2025). Second, DPO tends to overfit the preference data and does not effectively constrain the policy's divergence from the reference model, so it may forget knowledge encoded in the reference model (Gheshlaghi Azar et al., 2024; Xu et al., 2024; Fisch et al., 2025).

Despite these empirical findings, a theoretical understanding of the gap between DPO and RLHF remains limited. Yang et al. (2026) prove that when the reference policy conflicts with the preference

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

data, RLHF, owing to its regularization toward the reference policy, may learn a policy that contradicts the preferences, whereas DPO follows the preference data. This result is consistent with the observation that DPO does not effectively constrain divergence from the reference model, but it does not identify the source of this discrepancy. Shi et al. (2026); Swamy et al. (2026) show that DPO and RLHF are equivalent under the idealized condition that the reward class of RLHF and the policy class of DPO are isomorphic. However, neither work examines whether this condition holds in general or how it can be enforced. These gaps motivate the following question:

Question

What is the theoretical source of the discrepancy between DPO and RLHF, and can it be eliminated to obtain a single-stage method that is provably equivalent to RLHF?

Contributions. This paper answers this question affirmatively through a systematic theoretical analysis of DPO. Our contributions are threefold.

First, we put the two empirical discrepancies between DPO and RLHF discussed above on a rigorous theoretical footing. Regarding on-policy learning, we prove that RLHF can identify the optimal action at states absent from the preference dataset, since its on-policy rollouts carry the training signal to the states the policy visits; DPO, whose training signal is confined to the preference data, leaves the policy at such states arbitrary (Proposition 1). Regarding reference regularization, we prove that the DPO policy can drift arbitrarily far from the reference policy no matter how the regularization coefficient is tuned, whereas the RLHF policy always stays within a bound that tightens as the coefficient grows (Proposition 2). These results turn previously observed empirical gaps into provable differences, establishing that DPO is not equivalent to RLHF.

Second, we pinpoint the root cause of this inequivalence in DPO's derivation and eliminate it, restoring equivalence with RLHF. Specifically, we find that the key step of the derivation, a change of variables from rewards to Q-functions (which in turn determine policies), silently drops the reward-boundedness constraint that makes RLHF's KL-regularized RL stage well posed. In the space of Q-functions, this constraint takes the form of a Bellman constraint, and reinstating it yields *Reinforced Direct Preference Optimization* (ReDPO). We prove that ReDPO is equivalent to RLHF and therefore inherits its bound on the divergence from the reference policy (Theorem 1). In practice, ReDPO enforces the Bellman constraint through a temporal-difference (TD) penalty (Sutton, 1988) evaluated on on-policy rollouts. This penalty induces a value flow that propagates both the successor Q-values and the reference log-probability into the current Q-value, thereby carrying out KL-regularized RL through dynamic programming. To our knowledge, ReDPO is the first single-stage preference optimization method that is provably equivalent to RLHF.

Third, we validate these theoretical claims through experiments on LLM alignment, with three main findings. (i) *Improved alignment performance:* on standard preference benchmarks, ReDPO consistently outperforms DPO and its variants on almost all metrics. This shows that the value flow turns on-policy rollouts into concrete performance gains. (ii) *Effective reference regularization and reduced forgetting:* ReDPO keeps the policy markedly closer to the reference model than DPO throughout training, as predicted by Theorem 1. This regularization in turn mitigates the forgetting of general capabilities, eliminating the alignment tax: whereas DPO falls 3.76 points below the reference model on average, ReDPO exceeds the reference model by 0.27 points. (iii) *Mitigated squeezing effect:* ReDPO substantially slows the decline in the log-probabilities of both preferred and dispreferred responses, confirming that its on-policy learning counteracts this failure mode of DPO's offline training.

2 PRELIMINARIES

Markov Decision Process. We adopt a reinforcement learning (RL) perspective and consider a finite-horizon Markov decision process (MDP) $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, r, P, \rho, H \rangle$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, H is the horizon, and ρ is the initial state distribution. Each trajectory begins with an initial state $s_1 \sim \rho$. At each step $h \in [H]$, the agent selects an action $a_h \in \mathcal{A}$ according to a policy $\pi(\cdot|s_h)$ and receives a reward $r(s_h, a_h) \in [R_{\min}, R_{\max}]$. The environment then transitions to the next state $s_{h+1} \sim P(\cdot|s_h, a_h)$, and the trajectory terminates after H steps. Language generation can be formalized as an MDP: the action space is the vocabulary, each state is the prompt followed by the tokens generated so far, and the transition appends the generated token to the current state, i.e.,

$s_{h+1} = s_h \oplus a_h$, where $\oplus$ denotes concatenation. Following RL convention, we refer to the LLM as a policy π and use the two terms interchangeably.

Reinforcement Learning from Human Feedback. RLHF aligns a language model in two stages (Christiano et al., 2017). In the first stage, a reward model $\widehat{r}$ is trained to predict human preferences under the Bradley–Terry model (Bradley & Terry, 1952) by maximizing the log-likelihood

$$\widehat{r} \in \operatorname*{argmax}_{r \in \mathcal{R}} \mathbb{E}_{(s_1, \tau^w, \tau^l) \sim \mathcal{D}^{\mathrm{pref}}} \left[\log \sigma \left(\sum_{h=1}^{H} r(s_h^w, a_h^w) - \sum_{h=1}^{H} r(s_h^l, a_h^l) \right) \right], \tag{1}$$

where $\mathcal{D}^{\mathrm{pref}} = \{(s_1, \tau^w, \tau^l)\}$ is a preference dataset in which τ^w and τ^l denote the preferred and dispreferred responses, respectively; $\mathcal{R} = \{r : r(s,a) \in [R_{\min}, R_{\max}], \forall (s,a) \in \mathcal{S} \times \mathcal{A}\}$ is the class of feasible bounded reward functions; and $\sigma(x) = 1/(1+\exp(-x))$ is the sigmoid function. In the second stage, a policy is learned by solving the KL-regularized RL problem against the learned reward:

$$\max_{\pi \in \Pi} \mathbb{E}_{\tau \sim \pi} \left[\sum_{h=1}^{H} \widehat{r}(s_h, a_h) \right] - \beta \, D_{\mathrm{KL}}(\pi, \pi^{\mathrm{ref}}), \tag{2}$$

where π^{ref} is a fixed reference policy (typically the supervised fine-tuned model from which π is initialized), $D_{\mathrm{KL}}(\pi, \pi^{\mathrm{ref}}) := \mathbb{E}_{\tau \sim \pi} \big[\sum_{h=1}^{H} D_{\mathrm{KL}}\big(\pi(\cdot|s_h), \pi^{\mathrm{ref}}(\cdot|s_h)\big) \big]$ is the trajectory-level KL divergence, and $\beta > 0$ is a regularization coefficient controlling how far π may deviate from π^{ref}.

Direct Preference Optimization. As a singe-stage method, DPO (Rafailov et al., 2023; 2024) bypasses explicit reward modeling and trains the policy directly on preference data by solving

$$\max_{\pi \in \Pi} \mathbb{E}_{(s_1, \tau^w, \tau^l) \sim \mathcal{D}^{\mathrm{pref}}} \left[\log \sigma \left(\beta \sum_{h=1}^{H} \log \frac{\pi(a_h^w \mid s_h^w)}{\pi^{\mathrm{ref}}(a_h^w \mid s_h^w)} - \beta \sum_{h=1}^{H} \log \frac{\pi(a_h^l \mid s_h^l)}{\pi^{\mathrm{ref}}(a_h^l \mid s_h^l)} \right) \right]. \tag{3}$$

Intuitively, this objective increases the likelihood of the preferred response relative to the dispreferred one, where each likelihood is measured relative to the reference policy π^{ref}.

3 DPO Could Deviate from RLHF

Although DPO is derived from the RLHF objective, several studies (Li et al., 2023; Gheshlaghi Azar et al., 2024; Xu et al., 2024) find that the policy it learns does not retain the desirable properties of the policy learned by RLHF. In this section, we first summarize the two empirical discrepancies between DPO and RLHF, and then put each on a rigorous theoretical footing, establishing that DPO is not equivalent to RLHF. The first discrepancy concerns the data used to train the policy.

Difference 1: No On-Policy Learning

RLHF optimizes the policy on rollouts sampled from the policy itself. DPO, in contrast, learns only from the fixed responses in $\mathcal{D}^{\mathrm{pref}}$ and never uses on-policy samples.

Under this offline paradigm, DPO's learning signal comes entirely from the preference data, which makes it difficult for DPO to identify the good action at states beyond that data. We formalize this difference below by showing that RLHF can identify the optimal action at states not covered by the preference data, whereas DPO cannot.

Proposition 1. *Suppose that π^{RLHF} is the RLHF policy learned by solving Eq.* (2) *and π^{DPO} is the DPO policy learned by solving Eq.* (3)*. There exists an MDP $\mathcal{M}$ with a deterministic optimal policy $\pi^\star$, preference dataset $\mathcal{D}^{\mathrm{pref}}$ and reference policy π^{ref} such that for a state s uncovered by $\mathcal{D}^{\mathrm{pref}}$,*

$$\pi^{\mathrm{RLHF}}(\pi^\star(s)|s) > \pi^{\mathrm{RLHF}}(a|s), \forall a \neq \pi^\star(s).$$

In contrast, π^{DPO} exhibits an arbitrary distribution on such a state.

The proof of Proposition 1 is presented in Appendix B.1. Proposition 1 shows that DPO is not equivalent to RLHF: RLHF can identify the optimal action at states beyond the preference data, whereas DPO cannot. The reason is structural. Because RLHF optimizes the policy on on-policy rollouts, its training signal reaches every state the policy can visit, including those absent from $\mathcal{D}^{\mathrm{pref}}$,

where it can learn which actions are good based on the future value. DPO, by contrast, trains directly on $\mathcal{D}^{\text{pref}}$, so its training signal never extends beyond the support of that dataset. The lack of on-policy learning also has a side effect: Ren & Sutherland (2025); Razin et al. (2025) show that it partly accounts for the *squeezing effect*, a failure mode unique to DPO in which the probabilities of both the preferred and the dispreferred responses decrease during training.

The second difference concerns whether the training objective effectively controls how far the policy diverges from the reference policy.

Difference 2: Loss of Reference Regularization

DPO's objective drives the policy toward a sharp policy that assigns probability one to the preferred response and zero to the dispreferred one, leaving its divergence from π^{ref} uncontrolled. RLHF, in contrast, keeps the policy close to π^{ref} effectively.

Without this control, DPO tends to overfit the limited preference dataset and forget the broader knowledge encoded in the reference policy (Gheshlaghi Azar et al., 2024), which can result in an alignment tax. Proposition 2 formalizes this difference.

Proposition 2. *Suppose that π^{RLHF} is the RLHF policy learned by solving Eq.* (2) *and π^{DPO} is the DPO policy learned by solving Eq.* (3)*. For any $C, \beta > 0$, there exists an MDP $\mathcal{M}$, preference dataset $\mathcal{D}^{\text{pref}}$ and reference policy π^{ref} such that $D_{\text{KL}}(\pi^{\text{DPO}}, \pi^{\text{ref}}) > C$. In contrast, for any MDP $\mathcal{M}$, preference dataset $\mathcal{D}^{\text{pref}}$ and reference policy π^{ref}, the RLHF policy satisfies that $D_{\text{KL}}(\pi^{\text{RLHF}}, \pi^{\text{ref}}) \leq (H(R_{\max} - R_{\min}))/\beta$.*

The proof of Proposition 2 is presented in Section B.2. Proposition 2 shows that the DPO policy π^{DPO} can drift arbitrarily far from π^{ref} in KL divergence no matter how β is tuned, whereas π^{RLHF} always stays within a bound that tightens as β grows.

Together, Propositions 1 and 2 establish that DPO is not equivalent to RLHF with respect to both on-policy learning and reference regularization. In the next section, we identify the root cause of this inequivalence and eliminate it, yielding a single-stage method that is provably equivalent to RLHF.

4 Reinforced Direct Preference Optimization

Section 3 established that DPO is not equivalent to RLHF. In this section, we first trace this inequivalence to a gap in DPO's derivation. Closing this gap yields Reinforced Direct Preference Optimization (ReDPO), which we prove is equivalent to RLHF. Finally, we present a practical implementation of ReDPO and explain how it eliminates both differences between DPO and RLHF.

4.1 Pinpointing the Gap in DPO's Derivation

Although DPO is derived from the RLHF objective, it is not equivalent to RLHF. We now trace where the two diverge by revisiting the derivation of DPO in its MDP formulation (Rafailov et al., 2024), which matches the RLHF setup. The derivation proceeds in two steps.

Step I: Reward-to-Q correspondence. Rafailov et al. (2024) observe that the optimal soft Q-function of the KL-regularized RL problem in Eq. (2) satisfies the soft Bellman optimality equation[1]

$$Q^{\star}_{r_\beta}(s_h, a_h) = \underbrace{\widehat{r}(s_h, a_h) + \beta \log \pi^{\text{ref}}(a_h|s_h)}_{:= r_\beta(s_h, a_h)} + V_{Q^{\star}_{r_\beta}}(s_{h+1}), \tag{4}$$

where r_β is the augmented reward, $Q^{\star}_{r_\beta}$ is the optimal soft Q-function with respect to r_β, and $V_{Q^{\star}_{r_\beta}}(s) := \beta \log \sum_{a \in \mathcal{A}} \exp\left(Q^{\star}_{r_\beta}(s, a)/\beta\right)$ is the corresponding optimal soft V-function. Eq. (4) thus establishes a one-to-one correspondence between reward functions and optimal soft Q-functions.

Step II: Change of variables. Building on Eq. (4), Rafailov et al. (2024) substitute $r(s_h, a_h) = Q(s_h, a_h) - \beta \log \pi^{\text{ref}}(a_h|s_h) - V_Q(s_{h+1})$ into Eq. (1), which reparameterizes the reward-learning objective in terms of Q-functions:

$$\max_Q \mathbb{E}_{(s_1, \tau^w, \tau^l) \sim \mathcal{D}^{\text{pref}}} \left[\log \sigma \left(\sum_{h=1}^{H} \left(Q(s_h^w, a_h^w) - \beta \log \pi^{\text{ref}}(a_h^w|s_h^w) - V_Q(s_{h+1}^w) \right) \right. \right.$$

[1] See Appendix A for details.

$$-\sum_{h=1}^{H}\left(Q(s_h^l,a_h^l)-\beta\log\pi^{\text{ref}}(a_h^l|s_h^l)-V_Q(s_{h+1}^l)\right)\Bigg)\Bigg]. \tag{5}$$

The optimal policy of the KL-regularized RL problem is in turn the softmax of the optimal soft Q-function, i.e., $\pi_{Q^\star_{r_\beta}}(a_h|s_h)\propto\exp\left(Q^\star_{r_\beta}(s_h,a_h)/\beta\right)$, which is precisely how a LLM converts its logits into an output distribution. This correspondence motivates identifying the LLM's logits with the Q-function and its output distribution with the policy (Rafailov et al., 2024; Yuan et al., 2025). Under this identification, the objective therefore reduces exactly to the DPO objective in Eq. (3)[2]:

$$\max_Q \mathbb{E}_{(s_1,\tau^w,\tau^l)\sim\mathcal{D}^{\text{pref}}}\left[\log\sigma\left(\beta\sum_{h=1}^{H}\log\frac{\pi_Q(a_h^w|s_h^w)}{\pi^{\text{ref}}(a_h^w|s_h^w)}-\beta\sum_{h=1}^{H}\log\frac{\pi_Q(a_h^l|s_h^l)}{\pi^{\text{ref}}(a_h^l|s_h^l)}\right)\right]. \tag{6}$$

We find that the issue lies precisely in the change of variables from r to Q in Step II. The original reward-learning objective in Eq. (1) is optimized over *bounded* reward functions, which keeps the subsequent KL-regularized RL problem well posed. Once r is replaced by Q, however, the Q-function in Eq. (6) ranges freely over all functions, and nothing constrains the induced reward $r_Q(s_h,a_h):=Q(s_h,a_h)-\beta\log(\pi^{\text{ref}}(a_h|s_h))-V_Q(s_{h+1})$ (Rafailov et al., 2024) to remain bounded. It is exactly this loss of reward boundedness that breaks the equivalence between DPO and RLHF and gives rise to the two differences identified in Section 3, as explained below.

Remark 1. *Reward boundedness is not a mere technicality. Without it, fitting the preference data in DPO drives the implicit reward toward $+\infty$ on preferred responses and $-\infty$ on dispreferred ones. This signal collapses the KL-regularized RL process into learning on the preference data alone: the policy neither maximizes the long-term reward globally through on-policy rollouts nor stays regularized toward the reference model, giving rise to Differences 1 and 2, respectively.*

4.2 Reinforced DPO Based on Bellman Constraints

The diagnosis in Section 4.1 attributes the failure of DPO's derivation to a single omission: the change of variables drops the reward-boundedness constraint. Repairing the derivation therefore amounts to reinstating exactly this constraint, that is, restricting the optimization in Eq. (6) to Q-functions whose induced reward r_Q remains bounded. This yields the following method, which we call *Reinforced Direct Preference Optimization* (ReDPO):

$$\begin{aligned}&\max_Q \mathbb{E}_{(s_1,\tau^w,\tau^l)\sim\mathcal{D}^{\text{pref}}}\left[\log\sigma\left(\beta\sum_{h=1}^{H}\log\frac{\pi_Q(a_h^w|s_h^w)}{\pi^{\text{ref}}(a_h^w|s_h^w)}-\beta\sum_{h=1}^{H}\log\frac{\pi_Q(a_h^l|s_h^l)}{\pi^{\text{ref}}(a_h^l|s_h^l)}\right)\right]\\ &\text{s.t. } Q(s_h,a_h)-\beta\log\pi^{\text{ref}}(a_h|s_h)-V_Q(s_{h+1})\in[R_{\min},R_{\max}],\quad\forall(s_h,a_h)\in\mathcal{S}\times\mathcal{A}.\end{aligned} \tag{7}$$

Since Q corresponds to the logits of a language model, this constraint imposes reward boundedness directly on the model's logits. We call it the *Bellman constraint*, as it is an inequality analogue of the Bellman equation. The following remarks elaborate on its role.

Remark 2. *The Bellman constraint is what allows ReDPO to carry out KL-regularized RL through dynamic programming (Bertsekas & Tsitsiklis, 1996). This becomes explicit after rearranging it as*

$$Q(s_h,a_h)\in\Big[\beta\log\pi^{\text{ref}}(a_h|s_h)+V_Q(s_{h+1})+R_{\min},\ \beta\log\pi^{\text{ref}}(a_h|s_h)+V_Q(s_{h+1})+R_{\max}\Big].$$

The constraint thus propagates two quantities into the current Q-value: the Q-value of the successor state and the reference log-probability. Propagating the successor value realizes the dynamic-programming mechanism that globally maximizes the long-term reward, eliminating Difference 1. Propagating the reference log-probability anchors the Q-function to π^{ref}, so that the induced softmax policy stays close to it, eliminating Difference 2.

Remark 3. *ReDPO admits a natural division of labor: the objective learns the implicit reward, while the Bellman constraint carries out the resulting KL-regularized RL. Unlike RLHF, which performs these two roles in separate stages, ReDPO fulfills both within a single training formulation. This division also explains why the Bellman constraint is an inequality rather than an equality like the Bellman equation: since the implicit reward is unknown and learned during training, the inequality leaves the Q-function room to be shaped by the objective within the admissible range.*

[2]Optimizing Eq. (6) over Q-functions is equivalent to optimizing Eq. (3) over policies, since a language model's output distribution is fully determined by its logits.

We prove that ReDPO is equivalent to RLHF and consequently inherits its KL-divergence bound.

Theorem 1. *Suppose that $\widehat{Q}$ is the Q-function learned by ReDPO by solving Eq.* (7) *and $\pi_{\widehat{Q}}$ is the corresponding softmax policy. Then the following hold:*

(a) Equivalence to RLHF: $\pi_{\widehat{Q}}$ is the optimal solution to the RLHF problem in Eq. (2)*.*

(b) Bounded KL divergence: $\pi_{\widehat{Q}}$ satisfies $D_{\mathrm{KL}}(\pi_{\widehat{Q}}, \pi^{\mathrm{ref}}) \leq H(R_{\max} - R_{\min})/\beta$.

The proof of Theorem 1 is presented in Section C.1. In sharp contrast to Propositions 1 and 2, which establish that DPO is not equivalent to RLHF, ReDPO, obtained by enforcing the Bellman constraint, is equivalent to RLHF. As a consequence, ReDPO inherits exactly the same KL-divergence bound established for RLHF in Proposition 2. Unlike DPO, ReDPO thus keeps the learned policy provably close to π^{ref}, which alleviates the forgetting of knowledge encoded in the reference model. To our knowledge, ReDPO is the first single-stage preference optimization method that is provably equivalent to RLHF.

Remark 4. *Shi et al. (2026); Swamy et al. (2026) prove that DPO and RLHF are equivalent under the idealized condition that the reward class of RLHF and the policy class of DPO are isomorphic. However, we find that this condition does not hold in general: RLHF optimizes over feasible bounded rewards, whereas the policy class of DPO, represented by unconstrained Q-functions, induces implicit rewards that may be arbitrarily large. Our analysis further reveals that the Bellman constraint is precisely what restores this isomorphism, establishing a one-to-one correspondence between bounded rewards and the Q-functions admissible in ReDPO.*

4.3 Practical Implementation

Having established the theoretical formulation of ReDPO and its equivalence to RLHF, we now turn to a practical implementation for LLM training. We enforce the Bellman constraint in Eq. (7) through a squared-error penalty that carries out backward dynamic programming: it pulls $Q(s_h, a_h)$ toward the Bellman target $\beta \log \pi^{\mathrm{ref}}(a_h|s_h) + V_Q(s_{h+1})$, thereby keeping the induced reward $r_Q(s_h, a_h)$ bounded. Crucially, this penalty is evaluated on *on-policy rollouts* freshly sampled from the current policy π_{Q_θ}, so that the training signal extends to states beyond the preference dataset. Combining this penalty with the preference objective in Eq. (7) yields the following training objective:

$$\mathcal{L}_{\text{ReDPO}}(\theta) := \mathbb{E}_{(s_1,\tau^w,\tau^l)\sim\mathcal{D}^{\mathrm{pref}}}\left[\log\sigma\left(\beta\sum_{h=1}^{H}\log\frac{\pi_{Q_\theta}(a_h^w|s_h^w)}{\pi^{\mathrm{ref}}(a_h^w|s_h^w)} - \beta\sum_{h=1}^{H}\log\frac{\pi_{Q_\theta}(a_h^l|s_h^l)}{\pi^{\mathrm{ref}}(a_h^l|s_h^l)}\right)\right]$$
$$- \alpha\,\mathbb{E}_{\tau\sim\mathcal{D}^{\mathrm{on}}}\left[\sum_{h=1}^{H}\Big(Q_\theta(s_h,a_h) - \beta\log\pi^{\mathrm{ref}}(a_h|s_h) - \mathrm{SG}\big(V_{Q_\theta}(s_{h+1})\big)\Big)^2\right]. \quad (8)$$

Here θ denotes the parameters of the LLM, Q_θ denotes its logits, $\mathcal{D}^{\mathrm{on}}$ denotes the on-policy rollouts generated by π_{Q_θ}, and $\alpha > 0$ controls the strength of the penalty. The penalty takes the form of a temporal-difference (TD) objective (Sutton, 1988) with TD target $\beta \log \pi^{\mathrm{ref}}(a_h|s_h) + \mathrm{SG}\big(V_{Q_\theta}(s_{h+1})\big)$, where SG denotes the stop-gradient operator. We train the LLM by maximizing $\mathcal{L}_{\text{ReDPO}}(\theta)$ with standard gradient-based optimization, as outlined in Algorithm 1.

Algorithm 1 Reinforced Direct Preference Optimization (ReDPO)

Input: Preference dataset $\mathcal{D}^{\mathrm{pref}} = \{(s_1, \tau^w, \tau^l)\}$, reference policy π^{ref}, regularization coefficient β, penalty coefficient α, learning rate η, number of iterations K

1: **for** $k = 1, \ldots, K$ **do**
2: Sample a mini-batch of preference data $\{(s_1, \tau^w, \tau^l)\}$ from $\mathcal{D}^{\mathrm{pref}}$
3: Generate on-policy rollouts $\mathcal{D}^{\mathrm{on}}$ by running π_{Q_θ} on the prompts in the mini-batch
4: Compute $\mathcal{L}_{\text{ReDPO}}(\theta)$ in Eq. (8) using the mini-batch and $\mathcal{D}^{\mathrm{on}}$
5: Update $\theta \leftarrow \theta + \eta \nabla_\theta \mathcal{L}_{\text{ReDPO}}(\theta)$
6: **end for**

The value flow mechanism. We close this subsection by explaining how ReDPO eliminates the two differences between DPO and RLHF identified in Section 3. The key is that ReDPO drives a *value*

flow, in which both the value of the successor state and the reference log-probability propagate into the current Q-value. The preference term in Eq. (8) serves as the source of this flow: it raises the Q-values along preferred responses and lowers those along dispreferred ones. The penalty term then propagates these values along on-policy rollouts, serving two purposes. First, if generating a token a_h leads to a successor state s_{h+1} with a high value, the penalty raises the Q-value of a_h accordingly. Through this dynamic-programming propagation, the preference signal reaches states beyond the preference data, enabling on-policy learning and eliminating Difference 1. Second, if π^{ref} assigns a high probability to a_h, the penalty likewise raises the Q-value of a_h, anchoring Q_θ to π^{ref} throughout training. The induced softmax policy therefore stays close to π^{ref}, eliminating Difference 2.

5 Related Work

Reinforcement learning from human feedback. RLHF (Christiano et al., 2017) aligns a language model through a two-stage pipeline. Early RLHF systems (Ouyang et al., 2022; Bai et al., 2022a;b) rely on Proximal Policy Optimization (PPO) (Schulman et al., 2017), which requires training an additional critic model and thus adds substantial complexity. To avoid this overhead, Li et al. (2024) propose ReMax, a critic-free method that applies REINFORCE (Williams, 1992) with a greedy-decoding reward baseline. Other critic-free methods include RLOO (Ahmadian et al., 2024), GRPO (Shao et al., 2024), and REINFORCE++ (Hu et al., 2025). These methods simplify the RL stage but retain RLHF's two-stage pipeline. In contrast, ReDPO is a single-stage method that is provably equivalent to RLHF: through its value flow mechanism, it learns from on-policy rollouts while remaining close to the reference policy, without training a separate reward model.

Direct preference optimization. DPO (Rafailov et al., 2023; 2024) merges the two stages of RLHF into one. Although DPO is derived from the RLHF objective, a theoretical understanding of how the two methods relate remains limited, and only recent works have begun to examine it. Shi et al. (2026); Swamy et al. (2026) prove that DPO and RLHF are equivalent under the idealized condition that the reward class of RLHF and the policy class of DPO are isomorphic. Swamy et al. (2026) further attribute the practical value of RLHF to other factors, such as the gap between generation and verification. Both works, however, take the isomorphism condition as given. In contrast, we show that this condition does not hold in general for DPO, since its implicit rewards may be unbounded, and identify the Bellman constraint as the key to enforcing it. This insight leads to ReDPO, a single-stage method that is provably equivalent to RLHF.

The change-of-variables technique. The change-of-variables technique studied in this paper originated in imitation learning (IL) (Osa et al., 2018), which learns from expert demonstrations. In that setting, it converts an adversarial method, which trains a reward and a policy against each other, into a non-adversarial one (Kostrikov et al., 2020; Garg et al., 2021; Xu et al., 2026). Notably, Xu et al. (2026) show that preserving the original constraint under the change of variables is crucial for resolving the well-known compounding-error issue in IL We instead study this technique in the setting of learning from preferences, and find that preserving the original reward-boundedness constraint under the change of variables is crucial for maintaining equivalence with RLHF.

6 Experiments

In this section, we empirically validate our theoretical claims by examining three aspects of ReDPO: (i) its alignment performance compared with DPO and its variants as well as RLHF methods; (ii) its ability to control the KL divergence from the reference model, as predicted by Theorem 1, and thereby mitigate forgetting; and (iii) the role of on-policy learning in mitigating the squeezing effect. We briefly describe the experimental setup below and defer details to Appendix D.

Setup. We fine-tune two instruction-tuned models, `meta-llama/Llama-3.2-3B-Instruct` and `Qwen/Qwen2.5-7B-Instruct`. Each checkpoint serves both as the SFT model from which the policy is initialized and as the reference policy for methods that require one. Following SimPO (Meng et al., 2024), we construct a preference dataset for each model from the prompts in the binarized UltraFeedback dataset. This dataset remains fixed during training; ReDPO additionally generates on-policy rollouts to evaluate the TD penalty that enforces the Bellman constraint.

Baselines. We compare ReDPO with two categories of baselines: DPO and its variants, including IPO (Gheshlaghi Azar et al., 2024), SimPO (Meng et al., 2024), and ORPO (Hong et al., 2024); and RLHF methods, implemented with PPO (Schulman et al., 2017) and GRPO (Shao et al., 2024).

6.1 Main Results

Table 1 reports the results on AlpacaEval 2 and Arena-Hard. On both models, ReDPO outperforms DPO and its variants on all metrics except one, where it slightly trails ORPO. The gains over DPO are consistent and grow on the stronger model: on Llama3.2-3B-Instruct, ReDPO improves the AlpacaEval 2 LC win rate by 3.72 points (a 15.1% relative improvement); on Qwen2.5-7B-Instruct, it improves the LC win rate by 7.13 points (17.3%) and the Arena-Hard SC win rate by 6.3 points (13.1%). These gains support the empirical effectiveness of ReDPO, which combines preference optimization with a TD penalty evaluated on on-policy rollouts.

Moreover, ReDPO outperforms the RLHF methods PPO and GRPO on six of the eight metrics and remains close on the other two. This suggests that ReDPO retains the benefits of RLHF while avoiding the instability of a separate reward-modeling stage.

Method	Llama3.2-3B-Instruct					Qwen2.5-7B-Instruct				
	AlpacaEval 2			Arena-Hard		AlpacaEval 2			Arena-Hard	
	WR(%)	LC(%)	Len.	WR(%)	SC(%)	WR(%)	LC(%)	Len.	WR(%)	SC(%)
SFT	21.80	21.61	2002	9.9	11.4	35.57	38.55	1933	46.8	40.8
IPO	25.03	24.87	2067	11.7	13.3	42.36	42.72	2036	52.1	47.0
SimPO	24.78	24.94	1966	12.7	14.3	36.77	39.36	1950	46.9	41.4
ORPO	22.24	22.29	1970	13.4	**15.0**	37.02	40.25	1926	47.8	43.5
DPO	25.59	24.63	3361	12.6	13.9	46.52	41.31	2477	55.3	48.2
ReDPO	**28.76**	**28.35**	2251	**13.8**	14.4	50.62	**48.44**	2188	**60.2**	**54.5**
GRPO	23.42	27.20	1803	13.7	14.7	48.82	43.74	2271	52.9	42.2
PPO	23.17	27.93	1775	11.5	14.3	**51.18**	47.67	2205	50.5	38.4

Table 1: Preference benchmark results. On AlpacaEval 2, we report the win rate (WR), length-controlled win rate (LC), and average response length (Len.). On Arena-Hard, we report the win rate (WR) and style-controlled win rate (SC). The best result in each column (excluding Len.) is in bold. ReDPO achieves the best result in six of the eight win-rate metrics.

6.2 Controlling KL Divergence and Mitigating Forgetting

We next examine whether ReDPO effectively controls the KL divergence from the reference model, as predicted by Theorem 1(b). As shown in Figure 1, the KL divergence of DPO keeps growing and fluctuates sharply, peaking above 300, whereas that of ReDPO quickly stabilizes at around 125 and remains there for the rest of training. This contrast mirrors our theory. Consistent with Proposition 2, the DPO policy drifts far from the reference policy. In ReDPO, by contrast, the Bellman constraint continually propagates the reference log-probability into the Q-values, anchoring the policy to the reference model and keeping its KL divergence under control.

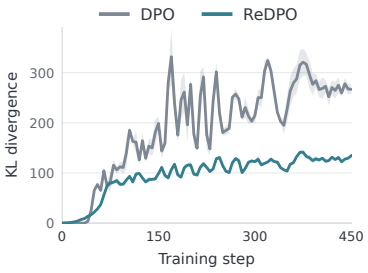

Figure 1: KL divergence between the policy and the reference model during training. The KL divergence of DPO keeps growing, whereas that of ReDPO stabilizes at a much lower level.

The lower KL divergence observed for ReDPO translates directly into less forgetting and a lower alignment tax. We evaluate general capabilities on seven benchmarks: GPQA Diamond, MMLU, ARC-C, HellaSwag, WinoGrande, GSM8K, and MATH. As shown in Figure 2, DPO underperforms the reference model on all seven benchmarks. ReDPO outperforms DPO on every benchmark and stays close to the reference model: it preserves performance on ARC-C and WinoGrande, and even improves over the reference model on GPQA Diamond and GSM8K. Averaged over the seven benchmarks, **ReDPO**

effectively eliminates the alignment tax, with a 0.27-point increase for ReDPO compared with a 3.76-point drop for DPO.

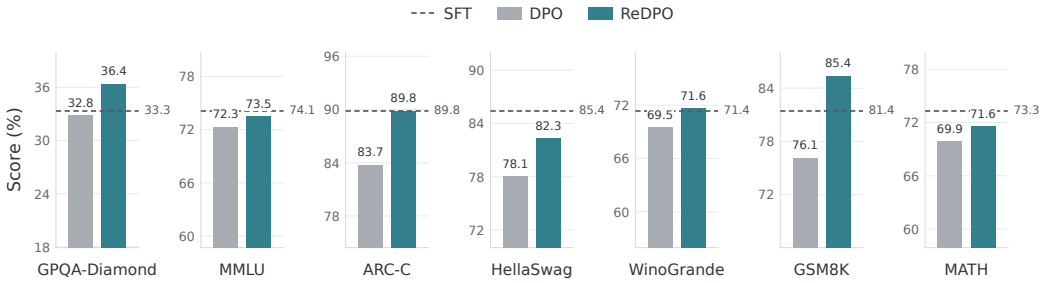

Figure 2: Performance on general-capability benchmarks after preference optimization. DPO falls 3.76 points below the reference model on average, whereas ReDPO exceeds it by 0.27 points, eliminating the alignment tax.

6.3 Mitigating the Squeezing Effect with On-Policy Rollouts

Finally, Figure 3 tracks the summed log-probabilities of the chosen and rejected responses during training. DPO exhibits a pronounced squeezing effect: the log-probabilities of both chosen and rejected responses drop sharply early in training and continue to decline thereafter. ReDPO substantially slows this decline, keeping the log-probability of the chosen responses markedly higher than that of DPO throughout training. This confirms that, through the value flow, the on-policy rollouts in ReDPO provide training signal beyond the fixed preference dataset, thereby counteracting the squeezing effect caused by DPO's offline training.

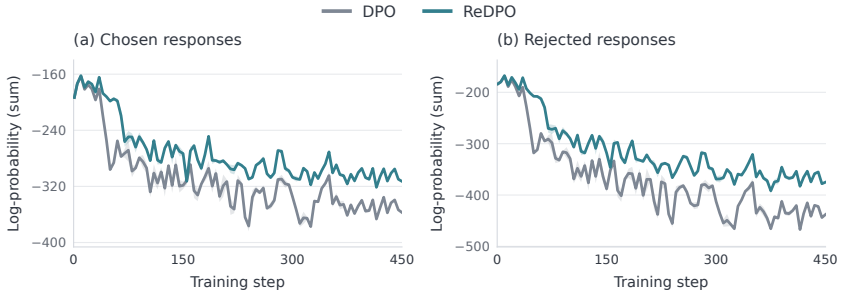

Figure 3: Summed log-probabilities of (a) chosen and (b) rejected responses in the preference dataset during training. ReDPO substantially slows the decline in both, mitigating the squeezing effect observed in DPO.

7 Conclusion

In this paper, we revisit the relationship between DPO and RLHF from a theoretical perspective. We prove that DPO is not equivalent to RLHF: it cannot learn beyond the preference data through on-policy learning, and it leaves its KL divergence from the reference policy uncontrolled. We trace both discrepancies to a single gap in DPO's derivation, namely the Bellman constraint dropped by the change of variables from rewards to Q-functions. Reinstating this constraint yields ReDPO, a single-stage method that is provably equivalent to RLHF and thus inherit its KL-divergence guarantee. Through a value flow mechanism, ReDPO learns from on-policy rollouts while remaining anchored to the reference model. Experiments on LLM alignment confirm that ReDPO outperforms DPO and its variants, remains competitive with RLHF methods, better controls KL divergence, mitigates forgetting, and alleviates the squeezing effect. We hope this work offers a principled foundation for designing single-stage preference optimization methods that retain the guarantees of RLHF.

AI USE STATEMENT

In this work, we used generative AI tools to edit the manuscript for readability, including polishing the language and improving the clarity of the text. We did not use generative AI tools to develop the theoretical framework, formulate or prove mathematical claims, design or implement the method, design experiments, or analyze and interpret results; the remaining required-disclosure tasks are not applicable to this work. All AI-assisted edits were reviewed and revised by the authors. We take full responsibility for the final content of this work, including all text and claims.

ETHICS STATEMENT

This paper investigates the theoretical underpinnings of direct preference optimization and conforms with the ICLR Code of Ethics in every respect.

REPRODUCIBILITY STATEMENT

This paper provides all the information needed to reproduce the main results. For all theoretical results, the complete proof is provided in Appendix B and Appendix C. For experimental results, we present all implementation details in Appendix D.

REFERENCES

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting REINFORCE-style optimization for learning from human feedback in LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12248–12267, 2024.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022a.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022b.

Dimitri P. Bertsekas and John N. Tsitsiklis. *Neuro-Dynamic Programming*, volume 3 of *Optimization and Neural Computation*. Athena Scientific, 1996.

Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Advances in neural information processing systems*, volume 30, 2017.

Adam Fisch, Jacob Eisenstein, Vicky Zayats, Alekh Agarwal, Ahmad Beirami, Chirag Nagpal, Peter Shaw, and Jonathan Berant. Robust preference optimization through reward model distillation. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856.

Divyansh Garg, Shuvam Chakraborty, Chris Cundy, Jiaming Song, and Stefano Ermon. IQ-Learn: Inverse soft-q learning for imitation. In *Advances in Neural Information Processing Systems 34*, pp. 4028–4039, 2021.

Mohammad Gheshlaghi Azar, Zhaohan Daniel Guo, Bilal Piot, Remi Munos, Mark Rowland, Michal Valko, and Daniele Calandriello. A general theoretical paradigm to understand learning from human preferences. In *International Conference on Artificial Intelligence and Statistics*, pp. 4447–4455. PMLR, 2024.

Jiwoo Hong, Noah Lee, and James Thorne. ORPO: Monolithic preference optimization without reference model. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 11170–11189, 2024.

Jian Hu, Jason Klein Liu, Haotian Xu, and Wei Shen. REINFORCE++: Stabilizing critic-free policy optimization with global advantage normalization. *arXiv preprint arXiv:2501.03262*, 2025.

Ilya Kostrikov, Ofir Nachum, and Jonathan Tompson. Imitation learning via off-policy distribution matching. In *Proceedings of the 8th International Conference on Learning Representations*, 2020.

Ziniu Li, Tian Xu, and Yang Yu. Policy optimization in rlhf: The impact of out-of-preference data. *arXiv preprint arXiv:2312.10584*, 2023.

Ziniu Li, Tian Xu, Yushun Zhang, Zhihang Lin, Yang Yu, Ruoyu Sun, and Zhi-Quan Luo. Remax: A simple, effective, and efficient reinforcement learning method for aligning large language models. In *International Conference on Machine Learning*, 2024.

Yu Meng, Mengzhou Xia, and Danqi Chen. SimPO: Simple preference optimization with a reference-free reward. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Andrew Y Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the 16th International Conference on Machine Learning*, 1999.

Takayuki Osa, Joni Pajarinen, Gerhard Neumann, J. Andrew Bagnell, Pieter Abbeel, and Jan Peters. An algorithmic perspective on imitation learning. *Foundations and Trends in Robotics*, 7(1-2): 1–179, 2018.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744, 2022.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in neural information processing systems 36*, pp. 53728–53741, 2023.

Rafael Rafailov, Joey Hejna, Ryan Park, and Chelsea Finn. From r to Q^*: Your language model is secretly a Q-function. *arXiv preprint arXiv:2404.12358*, 2024.

Noam Razin, Sadhika Malladi, Adithya Bhaskar, Danqi Chen, Sanjeev Arora, and Boris Hanin. Unintentional unalignment: Likelihood displacement in direct preference optimization. In *Proceedings of the 13th International Conference on Learning Representations*, pp. 24791–24834, 2025.

Yi Ren and Danica Sutherland. Learning dynamics of llm finetuning. In *Proceedings of the 13th International Conference on Learning Representations*, pp. 70523–70563, 2025.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Ruizhe Shi, Minhak Song, Runlong Zhou, Zihan Zhang, Maryam Fazel, and Simon Shaolei Du. Understanding the performance gap in preference learning: A dichotomy of RLHF and DPO. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Richard Sutton. Learning to predict by the methods of temporal differences. *Machine learning*, 3: 9–44, 1988.

Gokul Swamy, Sanjiban Choudhury, Wen Sun, Steven Wu, and Drew Bagnell. All roads lead to likelihood: The value of reinforcement learning in fine-tuning. In *Proceedings of the 14th International Conference on Learning Representations*, pp. 143056–143082, 2026.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256, 1992.

Shusheng Xu, Wei Fu, Jiaxuan Gao, Wenjie Ye, Weilin Liu, Zhiyu Mei, Guangju Wang, Chao Yu, and Yi Wu. Is DPO superior to PPO for LLM alignment? a comprehensive study. In *International Conference on Machine Learning*, pp. 54983–54998. PMLR, 2024.

Tian Xu, Chenyang Wang, Xiaochen Zhai, Ziniu Li, Yi-Chen Li, and Yang Yu. Non-adversarial imitation learning provably free of compounding errors: The value flow mechanism. In *Proceedings of the Forty-third International Conference on Machine Learning*, 2026.

Zhiqin Yang, Yonggang Zhang, Wei Xue, Dong Fang, Bo Han, and Yike Guo. Conditional equivalence of DPO and RLHF: Implicit assumption, failure modes, and provable alignment. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Yurun Yuan, Fan Chen, Zeyu Jia, Alexander Rakhlin, and Tengyang Xie. Trajectory bellman residual minimization: A simple value-based method for llm reasoning. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

A KL-REGULARIZED REINFORCEMENT LEARNING

This section reviews the KL-regularized RL problem that underlies both RLHF and the MDP derivation of DPO (Rafailov et al., 2024). We show that this problem is equivalent to maximum-entropy RL with an augmented reward, define the soft value functions and the soft Bellman optimality equation, and characterize the resulting optimal policy.

Throughout, we fix the MDP $\mathcal{M}$ introduced in Section 2, a coefficient $\beta > 0$, and a reference policy π^{ref}. We denote by $\mathcal{S}_h$ the set of states at step h and adopt the convention that any function evaluated at a terminal state s_{H+1} equals zero. Given a prompt s_1, we write $\mathbb{E}_{\tau\sim\pi}[\,\cdot \mid s_1]$ for the expectation over the response $\tau = (a_1, \ldots, a_H)$ generated by π.

KL-regularized RL as maximum-entropy RL. KL-regularized RL solves

$$\max_{\pi\in\Pi} \; J_\beta(\pi) := \mathbb{E}_{\tau\sim\pi}\left[\sum_{h=1}^{H} \widehat{r}(s_h, a_h)\right] - \beta D_{\mathrm{KL}}(\pi, \pi^{\mathrm{ref}}). \tag{9}$$

Since $D_{\mathrm{KL}}(\pi(\cdot|s), \pi^{\mathrm{ref}}(\cdot|s)) = -\sum_{a\in\mathcal{A}} \pi(a|s) \log \pi^{\mathrm{ref}}(a|s) - \mathcal{H}(\pi(\cdot|s))$, where $\mathcal{H}(p) := -\sum_{a\in\mathcal{A}} p(a) \log p(a)$ denotes the Shannon entropy, the objective can be rewritten as

$$J_\beta(\pi) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{h=1}^{H} \Big(r_\beta(s_h, a_h) + \beta\mathcal{H}\big(\pi(\cdot|s_h)\big)\Big)\right], \tag{10}$$

where $r_\beta(s, a) := \widehat{r}(s, a) + \beta \log \pi^{\mathrm{ref}}(a|s)$. Hence, KL-regularized RL with reward $\widehat{r}$ is exactly maximum-entropy RL with the *augmented reward* r_β and temperature β.

Soft value functions and the soft Bellman optimality equation. Building on this connection, we now introduce the soft value functions and the soft Bellman optimality equation.

Definition 1 (Soft V-function and softmax policy)**.** *For any Q-function $Q : \bigcup_{h=1}^{H} \mathcal{S}_h \times \mathcal{A} \to \mathbb{R}$, its soft V-function and softmax policy are defined as*

$$V_Q(s) := \beta \log\Big(\sum_{a\in\mathcal{A}} \exp\big(Q(s, a)/\beta\big)\Big) = \beta\,\mathrm{LSE}(Q/\beta)(s),$$

$$\pi_Q(a|s) := \frac{\exp\big(Q(s, a)/\beta\big)}{\sum_{a'\in\mathcal{A}} \exp\big(Q(s, a')/\beta\big)},$$

with $V_Q(s_{H+1}) := 0$.

Definition 2 (Optimal soft Q- and V-functions)**.** *Given a reward $\widehat{r}$, the optimal soft Q-function $Q^\star_{r_\beta}$ is defined by backward induction over $h = H, H-1, \ldots, 1$ through the* soft Bellman optimality equation

$$Q^\star_{r_\beta}(s_h, a_h) = r_\beta(s_h, a_h) + V_{Q^\star_{r_\beta}}(s_{h+1}) = \widehat{r}(s_h, a_h) + \beta \log \pi^{\mathrm{ref}}(a_h|s_h) + V_{Q^\star_{r_\beta}}(s_{h+1}), \tag{11}$$

where $s_{h+1} = s_h \oplus a_h$ and $V_{Q^\star_{r_\beta}}(s_{H+1}) = 0$. We call $V_{Q^\star_{r_\beta}}$ the optimal soft V-function.

A standard result in maximum-entropy RL is that the softmax policy induced by $Q^\star_{r_\beta}$ is the unique optimal solution to Eq. (10). By the equivalence established above, it is therefore also the unique optimal solution to the KL-regularized RL problem in Eq. (9):

$$\begin{aligned}\pi_{Q^\star_{r_\beta}} &= \operatorname*{argmax}_{\pi\in\Pi} \mathbb{E}_{\tau\sim\pi}\left[\sum_{h=1}^{H} \Big(r_\beta(s_h, a_h) + \beta\mathcal{H}\big(\pi(\cdot|s_h)\big)\Big)\right] \\ &= \operatorname*{argmax}_{\pi\in\Pi} \mathbb{E}_{\tau\sim\pi}\left[\sum_{h=1}^{H} \widehat{r}(s_h, a_h)\right] - \beta D_{\mathrm{KL}}(\pi, \pi^{\mathrm{ref}}).\end{aligned}$$

Implicit reward as a shaping reward. In the token-level MDP, the implicit reward induced by the Q-function is formulated as

$$r_Q(s_h, a_h) := Q(s_h, a_h) - \beta \log(\pi^{\mathrm{ref}}(a_h|s_h)) - V_Q(s_{h+1}).$$

This reward is closely related to the well-known log-probability reward $\beta \log(\pi_Q(a|s)/\pi^{\text{ref}}(a|s))$ in (Rafailov et al., 2023). Specifically, since $\pi_Q(a|s) = \exp\big(Q(s,a)/\beta\big)/(\sum_{a' \in \mathcal{A}} \exp\big(Q(s,a')/\beta\big))$, we have that

$$\begin{aligned} r_Q(s_h, a_h) &= Q(s_h, a_h) - \beta \log \pi^{\text{ref}}(a_h|s_h) - V_Q(s_{h+1}) \\ &= \beta \log \frac{\pi_Q(a_h|s_h)}{\pi^{\text{ref}}(a_h|s_h)} + V_Q(s_h) - V_Q(s_{h+1}). \end{aligned}$$

That is, r_Q is a potential-based shaping (Ng et al., 1999) of the log-probability reward with the potential function $-V_Q(\cdot)$. Summing over a response, the potential terms telescope to $V_Q(s_1)$, which cancels between the two responses sharing the same prompt. This is also why the change of variables in Section 4.1 recovers exactly the DPO objective.

B Proof of Results in Section 3

B.1 Proof of Proposition 1

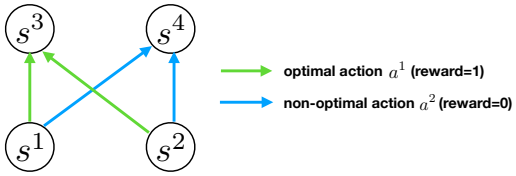

Figure 4: An MDP example corresponding to Proposition 1. The arrow denote the corresponding transition.

Proof. Here we leverage a minimal instance to demonstrate that RLHF can identify the optimal action at states beyond the preference dataset while DPO cannot. The MDP instance is shown in Figure 4. In this MDP, there are four states $\mathcal{S} = \{s^1, s^2, s^3, s^4\}$ and two actions $\mathcal{A} = \{a^1, a^2\}$. The horizon $H = 2$. Taking the optimal action a^1 from either s^1 or s^2 transitions to s^3 and yields reward 1, whereas taking the non-optimal action a^2 from either state transitions to s^4 and yields reward 0. The preference dataset $\mathcal{D}^{\text{pref}}$ consists of the single comparison $(s^1, a^1, s^3, a^1) \succ (s^1, a^2, s^4, a^1)$ generated at s^1, so the state s^2 is uncovered by $\mathcal{D}^{\text{pref}}$. We consider that the reference policy plays a uniform distribution at each state, i.e., $\pi^{\text{ref}}(a^1|s) = \pi^{\text{ref}}(a^2|s) = 1/2, \forall s \in \mathcal{S}$. In the following part, we characterize the RLHF policy π^{RLHF} and DPO policy π^{DPO} at the uncovered state s^2.

To analyze the RLHF policy, we first analyze the reward function learned by Eq. (1). In particular, the reward objective can be formulated as follows.

$$\widehat{r} \in \operatorname*{argmax}_{r \in \mathcal{R}} \log\left(\sigma\left(r(s^1, a^1) + r(s^3, a^1) - r(s^1, a^2) - r(s^4, a^1)\right)\right).$$

It is direct to derive the closed-form solution of the above optimization problem.

$$\widehat{r}(s^1, a^1) = R_{\max}, \widehat{r}(s^3, a^1) = R_{\max}, \widehat{r}(s^1, a^2) = R_{\min}, \widehat{r}(s^4, a^1) = R_{\min}.$$

The objective in Eq. (1) only involves the state-action pairs visited by the two compared trajectories. Therefore, it is invariant to the reward values at the uncovered pairs $(s^2, a^1), (s^2, a^2), (s^3, a^2)$ and (s^4, a^2), whose reward values are specified as the initial value. For concreteness, we assume that they share a common initial value $C \in [R_{\min}, R_{\max}]$.

$$\widehat{r}(s^2, a^1) = \widehat{r}(s^2, a^2) = \widehat{r}(s^3, a^2) = \widehat{r}(s^4, a^2) = C.$$

Then we analyze the RLHF policy learned by $\widehat{r}$.

$$\pi^{\text{RLHF}} \in \operatorname*{argmax}_{\pi \in \Pi} \mathbb{E}_{\tau \sim \pi}\left[\sum_{h=1}^{H} \left(\widehat{r}(s_h, a_h) - \beta D_{\text{KL}}\left(\pi(\cdot|s_h), \pi^{\text{ref}}(\cdot|s_h)\right)\right)\right].$$

The above KL-regularized RL objective can be re-formulated as the following maximum entropy RL objective with reward $r_\beta(s,a) := \widehat{r}(s,a) + \beta \log(\pi^{\text{ref}}(a|s))$.

$$\mathbb{E}_{\tau \sim \pi}\left[\sum_{h=1}^{H} \left(\widehat{r}(s_h, a_h) - \beta D_{\text{KL}}\left(\pi(\cdot|s_h), \pi^{\text{ref}}(\cdot|s_h)\right)\right)\right]$$

$$= \mathbb{E}_{\tau\sim\pi}\left[\sum_{h=1}^{H}\left(\underbrace{\widehat{r}(s_h,a_h)+\beta\log(\pi^{\mathrm{ref}}(a_h|s_h))}_{:=r_\beta(s_h,a_h)}+\beta\mathcal{H}(\pi(\cdot|s_h))\right)\right].$$

Thus we turn to analyze the optimal policy of this equivalent maximum entropy RL objective. Its soft optimal Q-function satisfies the soft Bellman optimality equation

$$Q^\star_{r_\beta}(s_h,a_h) = r_\beta(s_h,a_h) + V_{Q^\star_{r_\beta}}(s_{h+1}) = \widehat{r}(s_h,a_h) + \beta\log\left(\pi^{\mathrm{ref}}(a_h|s_h)\right) + V_{Q^\star_{r_\beta}}(s_{h+1}), \tag{12}$$

where $V_Q(s_h) = \beta\,\mathrm{LSE}(Q/\beta)(s_h) = \beta\log\left(\sum_{a'\in\mathcal{A}}\exp\left(Q(s_h,a')/\beta\right)\right)$ and $V_Q(s_{H+1}) = 0$. The induced optimal policy is the softmax policy with respect to $Q^\star_{r_\beta}/\beta$.

$$\pi^{\mathrm{RLHF}}(a_h|s_h) = \frac{\exp\left(Q^\star_{r_\beta}(s_h,a_h)/\beta\right)}{\sum_{a'\in\mathcal{A}}\exp\left(Q^\star_{r_\beta}(s_h,a')/\beta\right)}. \tag{13}$$

First, we can calculate the soft optimal Q-function at states s^3 and s^4. Since the trajectory ends after step $H = 2$, the future value term in Eq. (12) vanishes.

$$\begin{aligned}
Q^\star_{r_\beta}(s^3,a^1) &= \widehat{r}(s^3,a^1) + \beta\log\left(\pi^{\mathrm{ref}}(a^1|s^3)\right) = R_{\max} - \beta\log(2),\\
Q^\star_{r_\beta}(s^3,a^2) &= \widehat{r}(s^3,a^2) + \beta\log\left(\pi^{\mathrm{ref}}(a^2|s^3)\right) = C - \beta\log(2),\\
Q^\star_{r_\beta}(s^4,a^1) &= \widehat{r}(s^4,a^1) + \beta\log\left(\pi^{\mathrm{ref}}(a^1|s^4)\right) = R_{\min} - \beta\log(2),\\
Q^\star_{r_\beta}(s^4,a^2) &= \widehat{r}(s^4,a^2) + \beta\log\left(\pi^{\mathrm{ref}}(a^2|s^4)\right) = C - \beta\log(2).
\end{aligned}$$

Taking the log-sum-exp over actions yields the corresponding soft V-functions.

$$\begin{aligned}
V_{Q^\star_{r_\beta}}(s^3) &= \beta\log\left(\exp\left(\frac{R_{\max}}{\beta} - \log(2)\right) + \exp\left(\frac{C}{\beta} - \log(2)\right)\right),\\
V_{Q^\star_{r_\beta}}(s^4) &= \beta\log\left(\exp\left(\frac{R_{\min}}{\beta} - \log(2)\right) + \exp\left(\frac{C}{\beta} - \log(2)\right)\right).
\end{aligned}$$

Since $R_{\max} > R_{\min}$ and both $\exp(\cdot)$ and $\log(\cdot)$ are strictly increasing, we obtain the strict separation between the two terminal states.

$$V_{Q^\star_{r_\beta}}(s^3) > V_{Q^\star_{r_\beta}}(s^4). \tag{14}$$

Notice that this separation is inherited from the single preference comparison collected at s^1. The RL procedure transfers the learned preference to the terminal states s^3 and s^4, which are shared by s^1 and the uncovered state s^2.

Now we are ready to analyze the RLHF policy at the uncovered state s^2. Taking action a^1 at s^2 transitions to s^3 and taking action a^2 transitions to s^4. Then Eq. (12) implies that

$$\begin{aligned}
Q^\star_{r_\beta}(s^2,a^1) &= \widehat{r}(s^2,a^1) + \beta\log\left(\pi^{\mathrm{ref}}(a^1|s^2)\right) + V_{Q^\star_{r_\beta}}(s^3) = C - \beta\log(2) + V_{Q^\star_{r_\beta}}(s^3),\\
Q^\star_{r_\beta}(s^2,a^2) &= \widehat{r}(s^2,a^2) + \beta\log\left(\pi^{\mathrm{ref}}(a^2|s^2)\right) + V_{Q^\star_{r_\beta}}(s^4) = C - \beta\log(2) + V_{Q^\star_{r_\beta}}(s^4).
\end{aligned}$$

The two Q-values share the same reward and reference terms since $\widehat{r}(s^2,a^1) = \widehat{r}(s^2,a^2) = C$ and π^{ref} is uniform. Hence they differ only through the soft V-functions of the successor states. Combining with Eq. (14) yields that

$$Q^\star_{r_\beta}(s^2,a^1) > Q^\star_{r_\beta}(s^2,a^2).$$

Since the softmax mapping in Eq. (13) is strictly increasing in the Q-value, we conclude that RLHF identifies the optimal action $\pi^\star(s^2) = a^1$ at the uncovered state s^2.

$$\pi^{\mathrm{RLHF}}(a^1|s^2) > \pi^{\mathrm{RLHF}}(a^2|s^2).$$

Finally, we analyze the DPO policy at s^2. Instantiating Eq. (3) with the single comparison in $\mathcal{D}^{\text{pref}}$, the DPO objective is formulated as

$$\pi^{\text{DPO}} \in \operatorname*{argmax}_{\pi} \log \sigma \Bigg(\beta \log \frac{\pi(a^1|s^1)}{\pi^{\text{ref}}(a^1|s^1)} + \beta \log \frac{\pi(a^1|s^3)}{\pi^{\text{ref}}(a^1|s^3)} - \beta \log \frac{\pi(a^2|s^1)}{\pi^{\text{ref}}(a^2|s^1)} - \beta \log \frac{\pi(a^1|s^4)}{\pi^{\text{ref}}(a^1|s^4)} \Bigg).$$

This objective involves the policy only at s^1, s^3 and s^4, and contains no term of $\pi(\cdot|s^2)$. Hence $\pi^{\text{DPO}}(\cdot|s^2)$ can be an arbitrary distribution. Therefore, in contrast to RLHF, DPO fails to identify the optimal action at the state uncovered by the preference dataset, which completes the proof. □

Relevance of the example to LLMs. In the MDP used in the above proof, two different states lead to shared successor states. In LLMs, this structure arises naturally from how a transformer represents states. The transformer maps each prefix to a hidden representation and computes both the logits and the reward from it, so prefixes with similar representations are effectively treated as the same state. For example, consider two paraphrases of the same question, only one of which appears in the preference dataset. After the same partial answer is written to both, how the response should continue depends on their shared meaning rather than on the exact wording, and the transformer maps the two prefixes to similar representations. The two questions thus effectively share successor states, just as in our example. RLHF can then carry the value learned from the covered question back to the uncovered one through on-policy rollouts. DPO has no training signal on the uncovered question, so its behavior there is not guided by the values of the successor states. Function approximation may still let DPO generalize to the uncovered question, but only incidentally, when that question happens to resemble covered prompts in representation. What the policy does there is therefore determined by input similarity, not by which action leads to higher value, and no mechanism corrects it when the two diverge.

B.2 Proof of Proposition 2

Proof. We prove the two parts of Proposition 2 in turn.

We first prove the claim that DPO does not effectively control the KL divergence from the reference model. Fix $C > 0$. We consider a bandit problem (horizon $H = 1$) with two actions a^1, a^2, where the preference dataset consists of the single comparison $a^1 \succ a^2$, and the reference policy is $\pi^{\text{ref}}(a^1) = \exp(-(C+1))$, $\pi^{\text{ref}}(a^2) = 1 - \exp(-(C+1))$, which has full support. The DPO objective is

$$J_{\text{DPO}}(\pi(a^1)) = \log \sigma \left(\beta \log \big(\pi(a^1) \cdot \exp(C+1)\big) - \beta \log \left(\frac{1-\pi(a^1)}{1-\exp(-(C+1))} \right) \right) \tag{15}$$

for $\pi(a^1) \in (0,1)$. By examining the one-sided asymptotic behavior as $\pi(a^1) \to 0^+$ and $\pi(a^1) \to 1^-$, we obtain the boundary values $J_{\text{DPO}}(0) = -\infty$ and $J_{\text{DPO}}(1) = 0$. Let $z(\pi(a^1)) = \beta \log \big(\pi(a^1) \cdot \exp(C+1)\big) - \beta \log \left(\frac{1-\pi(a^1)}{1-\exp(-(C+1))} \right)$. For every $\pi(a^1) \in (0,1)$,

$$\frac{\mathrm{d}z}{\mathrm{d}\pi(a^1)} = \frac{\beta}{\pi(a^1)(1-\pi(a^1))} > 0, \qquad \frac{\mathrm{d}}{\mathrm{d}z} \log \sigma(z) = \sigma(-z) > 0,$$

so J_{DPO} is strictly increasing on $(0,1)$, and together with its values at the endpoints, J_{DPO} is strictly increasing on the whole closed interval $[0,1]$. Since Π is compact and J_{DPO} is upper semicontinuous on $[0,1]$, the maximum is attained, and by strict monotonicity it is attained uniquely at $\pi(a^1) = 1$: $\pi^{\text{DPO}}(a^1) = 1$ and $\pi^{\text{DPO}}(a^2) = 0$.

The KL divergence at this policy is,

$$D_{\text{KL}}\left(\pi^{\text{DPO}}, \pi^{\text{ref}}\right) = 1 \cdot \log \frac{1}{\exp(-(C+1))} + 0 \cdot \log \frac{0}{1-\exp(-(C+1))} = C + 1 > C,$$

proving the first part.

Then we prove the second claim that RLHF effectively controls the KL divergence. Fix an arbitrary MDP $\mathcal{M}$ with horizon H, preference dataset $\mathcal{D}^{\text{pref}}$, and reference policy π^{ref}. Since $\pi^{\text{ref}} \in \Pi$ is a feasible choice in Eq. (2) and $D_{\text{KL}}(\pi^{\text{ref}}, \pi^{\text{ref}}) = 0$, optimality of π^{RLHF} gives

$$\mathbb{E}_{\tau\sim\pi^{\text{RLHF}}}\left[\sum_{h=1}^{H}\widehat{r}(s_h,a_h)\right] - \beta D_{\text{KL}}(\pi^{\text{RLHF}},\pi^{\text{ref}}) \;\geq\; \mathbb{E}_{\tau\sim\pi^{\text{ref}}}\left[\sum_{h=1}^{H}\widehat{r}(s_h,a_h)\right].$$

Re-arranging the above inequality yields that

$$D_{\text{KL}}(\pi^{\text{RLHF}},\pi^{\text{ref}}) \leq \mathbb{E}_{\tau\sim\pi^{\text{RLHF}}}\left[\sum_{h=1}^{H}\widehat{r}(s_h,a_h)\right] - \mathbb{E}_{\tau\sim\pi^{\text{ref}}}\left[\sum_{h=1}^{H}\widehat{r}(s_h,a_h)\right] \leq \frac{H(R_{\max}-R_{\min})}{\beta}.$$

The last inequality follows that $\widehat{r}(s,a) \in [R_{\min}, R_{\max}]$. Since $\mathcal{M}$, $\mathcal{D}^{\text{pref}}$ and π^{ref} were arbitrary, this bound holds uniformly, proving the second part and completing the proof. □

C Proof of Results in Section 4

C.1 Proof of Theorem 1

Proof. We begin by proving the first claim. We first show that $r_{\widehat{Q}}$ is a feasible and optimal solution to the RLHF reward learning problem Eq. (1). By the constraint in Eq. (7), the implicit reward $r_{\widehat{Q}}(s_h,a_h) \in [R_{\min}, R_{\max}]$ for all (s_h,a_h), so $r_{\widehat{Q}} \in \mathcal{R}$. Moreover, substituting r_Q for r in Eq. (1), the reward learning objective becomes

$$\begin{aligned}
&\mathbb{E}_{(s_1,\tau^w,\tau^l)\sim\mathcal{D}^{\text{pref}}}\left[\log\sigma\big(r_Q(\tau^w) - r_Q(\tau^l)\big)\right] \\
&= \mathbb{E}_{(s_1,\tau^w,\tau^l)\sim\mathcal{D}^{\text{pref}}}\left[\log\sigma\left(\sum_{h=1}^{H}\big(r_Q(s_h^w,a_h^w) - r_Q(s_h^l,a_h^l)\big)\right)\right],
\end{aligned}$$

which is exactly the objective of Eq. (7). Since $\widehat{Q}$ maximizes this objective over all $Q \in \mathcal{Q}$ satisfying $r_Q \in \mathcal{R}$, it follows that $r_{\widehat{Q}}$ maximizes the RLHF reward learning objective over $\mathcal{R}$,

$$r_{\widehat{Q}} \in \operatorname*{argmax}_{r\in\mathcal{R}}\; \mathbb{E}_{(s_1,\tau^w,\tau^l)\sim\mathcal{D}^{\text{pref}}}\left[\log\sigma\left(\sum_{h=1}^{H} r(s_h^w,a_h^w) - \sum_{h=1}^{H} r(s_h^l,a_h^l)\right)\right].$$

Then we show that $\widehat{Q}$ satisfies the Bellman optimality equation for the KL-regularized RL problem with reward $r_{\widehat{Q}}$. According to the definition of the implicit reward, we have that

$$r_{\widehat{Q}}(s_h,a_h) = \widehat{Q}(s_h,a_h) - \beta\log\pi^{\text{ref}}(a_h|s_h) - \beta\,\text{LSE}(\widehat{Q}/\beta)(s_{h+1}),$$

for all (s_h,a_h,h). Rearranging the above equation yields that

$$\widehat{Q}(s_h,a_h) = r_{\widehat{Q}}(s_h,a_h) + \beta\log\pi^{\text{ref}}(a_h|s_h) + \beta\,\text{LSE}(\widehat{Q}/\beta)(s_{h+1}).$$

This is precisely the soft Bellman optimality equation for the KL-regularized RL problem Eq. (2), confirming that $\widehat{Q} = Q^\star_{r_\beta}$.

Finally, by the standard result of maximum-entropy RL, the unique optimal policy for the KL-regularized problem Eq. (2) is the softmax policy with respect to $Q^\star_{r_\beta}$:

$$\pi^{\text{RLHF}}(a_h|s_h) = \frac{\exp(\widehat{Q}(s_h,a_h)/\beta)}{\sum_{a'}\exp(\widehat{Q}(s_h,a')/\beta)} = \pi_{\widehat{Q}}(a_h|s_h).$$

Hence $\pi_{\widehat{Q}} = \pi^{\text{RLHF}}$, which completes the proof of the first claim.

Then we prove the second claim. Since $\pi_{\widehat{Q}}(\cdot|s_h) = \text{softmax}(\widehat{Q}(s_h,\cdot)/\beta)$, we have $\log\pi_{\widehat{Q}}(a_h|s_h) = \widehat{Q}(s_h,a_h)/\beta - \text{LSE}(\widehat{Q}/\beta)(s_h)$, where $\text{LSE}(\widehat{Q}/\beta)(s_h) = \log(\sum_{a\in\mathcal{A}}\exp(\widehat{Q}(s_h,a)/\beta))$. Then we have that

$$\log\frac{\pi_{\widehat{Q}}(a_h|s_h)}{\pi^{\text{ref}}(a_h|s_h)} = \frac{\widehat{Q}(s_h,a_h)}{\beta} - \text{LSE}(\widehat{Q}/\beta)(s_h) - \log\pi^{\text{ref}}(a_h|s_h)$$

$$= \frac{r_{\widehat{Q}}(s_h, a_h)}{\beta} + \mathrm{LSE}(\widehat{Q}/\beta)(s_{h+1}) - \mathrm{LSE}(\widehat{Q}/\beta)(s_h),$$

where the second equality uses the definition $r_{\widehat{Q}}(s_h, a_h) = \widehat{Q}(s_h, a_h) - \beta \log \pi^{\mathrm{ref}}(a_h|s_h) - \beta\, \mathrm{LSE}(\widehat{Q}/\beta)(s_{h+1})$. Summing over $h = 1, \ldots, H$ and applying telescoping cancellation with the terminal condition $\mathrm{LSE}(\widehat{Q}/\beta)(s_{H+1}) = 0$,

$$\sum_{h=1}^{H} \log \frac{\pi_{\widehat{Q}}(a_h|s_h)}{\pi^{\mathrm{ref}}(a_h|s_h)} = \frac{1}{\beta} \sum_{h=1}^{H} r_{\widehat{Q}}(s_h, a_h) - \mathrm{LSE}(\widehat{Q}/\beta)(s_1).$$

Taking expectations over $\tau \sim \pi_{\widehat{Q}}$,

$$\mathbb{E}_{\tau\sim\pi_{\widehat{Q}}}\left[\sum_{h=1}^{H} D_{\mathrm{KL}}\left(\pi_{\widehat{Q}}(\cdot|s_h), \pi^{\mathrm{ref}}(\cdot|s_h)\right)\right] = \frac{1}{\beta}\mathbb{E}_{\tau\sim\pi_{\widehat{Q}}}\left[\sum_{h=1}^{H} r_{\widehat{Q}}(s_h, a_h)\right] - \mathrm{LSE}(\widehat{Q}/\beta)(s_1).$$

The Bellman constraint in Eq. (7) gives $r_{\widehat{Q}}(s_h, a_h) \in [R_{\min}, R_{\max}]$ for all (s_h, a_h). For the first term, $r_{\widehat{Q}} \le R_{\max}$ implies

$$\frac{1}{\beta} \sum_{h=1}^{H} r_{\widehat{Q}}(s_h, a_h) \le \frac{H R_{\max}}{\beta}.$$

For the second term, the soft value $\beta\, \mathrm{LSE}(\widehat{Q}/\beta)(s_1)$ admits the variational form of

$$\begin{aligned}
\beta\, \mathrm{LSE}(\widehat{Q}/\beta)(s_1) &= \max_{\mu} \mathbb{E}_{\tau\sim\mu}\left[\sum_{h=1}^{H}\left(r_{\widehat{Q}}(s_h, a_h) - \beta D_{\mathrm{KL}}\left(\mu(\cdot|s_h), \pi^{\mathrm{ref}}(\cdot|s_h)\right)\right)\right] \\
&\ge \mathbb{E}_{\tau\sim\pi^{\mathrm{ref}}}\left[\sum_{h=1}^{H} r_{\widehat{Q}}(s_h, a_h)\right] \\
&\ge H R_{\min},
\end{aligned}$$

where the first inequality takes $\mu = \pi^{\mathrm{ref}}$ (so the KL term vanishes) and the second uses $r_{\widehat{Q}} \ge R_{\min}$. Hence $\mathrm{LSE}(\widehat{Q}/\beta)(s_1) \ge \frac{H R_{\min}}{\beta}$, so $-\mathrm{LSE}(\widehat{Q}/\beta)(s_1) \le -\frac{H R_{\min}}{\beta}$. Together, we can obtain

$$\mathbb{E}_{\tau\sim\pi_{\widehat{Q}}}\left[\sum_{h=1}^{H} D_{\mathrm{KL}}\left(\pi_{\widehat{Q}}(\cdot|s_h), \pi^{\mathrm{ref}}(\cdot|s_h)\right)\right] \le \frac{H R_{\max}}{\beta} - \frac{H R_{\min}}{\beta} = \frac{H(R_{\max} - R_{\min})}{\beta},$$

which finishes the proof of the second claim. □

D Experimental Details

D.1 Preference Data

Following the Instruct setting of SimPO (Meng et al., 2024), we construct a preference dataset for each model using prompts from the binarized UltraFeedback dataset. Specifically, we sample multiple responses to each prompt from the corresponding SFT model, score them with `ArmoRM-Llama3-8B-v0.1`, and select the highest-scoring and lowest-scoring responses as the preferred and dispreferred responses, respectively. This procedure produces preference pairs that better match the initial policy's response distribution. The resulting preference dataset is fixed during training; ReDPO additionally generates fresh on-policy rollouts to evaluate its Bellman penalty.

D.2 Implementation and Training Configuration

Implementations. We use the implementations of IPO, SimPO, ORPO and DPO in ms-swift 4.0.4, and implement ReDPO within the same framework. For SimPO, we set `cpo_alpha=0` to use the pure preference objective without an auxiliary SFT loss. PPO and GRPO are implemented using

Hyperparameter	Value
Fine-tuning	Full parameters
Precision	BF16
Training epochs	1
Per-device batch size	4 preference pairs
Gradient accumulation steps	4
Effective batch size	$16N_{\mathrm{dp}}$ preference pairs
Learning rate	10^{-6}
Learning rate schedule	Cosine decay
Warmup ratio / steps	0 / 0
Optimizer	Fused AdamW
Adam momentum coefficients	$(0.9, 0.95)$
Adam numerical stability constant	10^{-8}
Weight decay	0.1
Maximum gradient norm	1.0
Maximum sequence length	4096 tokens
ReDPO rollouts per prompt	2
Maximum ReDPO rollout length	2048 tokens

Table 2: Base training configuration for the ms-swift experiments. N_{dp} denotes the number of data-parallel workers. The final two rows apply to ReDPO's on-policy rollouts.

verl 0.7.1. For each model, the direct preference optimization methods use the same initial checkpoint and the same preference dataset described in Section 6. ReDPO combines this preference data with rollouts from the policy being optimized, as specified in Algorithm 1.

Training configuration. We use full-parameter fine-tuning with BF16 precision. The base training configuration for the ms-swift experiments is summarized in Table 2. We train for one epoch with a learning rate of 10^{-6} and cosine decay, without warmup. The optimizer is fused AdamW with weight decay 0.1, momentum coefficients $(0.9, 0.95)$ and numerical stability constant 10^{-8}. We clip the gradient norm at 1.0. Each device processes a mini-batch of four preference pairs, and gradients are accumulated over four steps. Thus, with N_{dp} data-parallel workers, the effective batch size is $16N_{\mathrm{dp}}$ preference pairs per optimizer update. The maximum input sequence length is 4096 tokens.

ReDPO rollouts and penalty. For each prompt in a preference mini-batch, ReDPO generates two responses from the current policy, with at most 2048 generated tokens per response. PPO and GRPO also generate two responses per prompt (`rollout_n=2`) to match ReDPO's rollout count; this corresponds to a group size of $G = 2$ for GRPO. We compute the Bellman penalty on ReDPO's freshly generated responses and stop the gradient through the successor value, following Eq. (8). In the implementation, the squared residual is averaged over valid generated response tokens. This token-mean reduction sets the scale of the penalty coefficient α. The selected ReDPO configuration is $\beta = 0.02$ and $\alpha = 0.04$ for both models.

D.3 Chat Templates

We use the following model-specific chat templates in all experiments. For reproducibility, we fix the date in the Llama system prompt to `Today Date: 01 Jun 2026` for every run.

Llama 3.2 Instruct

```
<|begin_of_text|><|start_header_id|>system<|end_header_id|>

Cutting Knowledge Date: December 2023
Today Date: 01 Jun 2026

<|eot_id|><|start_header_id|>user<|end_header_id|>

{user_message}<|eot_id|><|start_header_id|>assistant<|end_header_id|>

{assistant_response}<|eot_id|>
```

Method	Objective to minimize	Grid-search range
IPO	$\mathbb{E}_{\mathcal{D}^{\text{pref}}}\left[(\overline{d}_\theta - \frac{1}{2\beta})^2\right]$	β: $\{0.01, 0.05, 0.1, 0.2\}$
SimPO	$-\mathbb{E}_{\mathcal{D}^{\text{pref}}} \log \sigma\big(\beta(\overline{z}^w_\theta - \overline{z}^l_\theta) - \gamma\big)$	β: $\{1.0, 2.0, 3.0\}$ γ: $\{0.5, 1.0, 1.2\}$
ORPO	$\mathbb{E}_{\mathcal{D}^{\text{pref}}}\left[-\overline{z}^w_\theta - \lambda \log \sigma(\omega^w_\theta - \omega^l_\theta)\right]$	λ: $\{0.1, 0.2, 0.5, 1.0\}$
DPO	$-\mathbb{E}_{\mathcal{D}^{\text{pref}}} \log \sigma(\beta d_\theta)$	β: $\{0.01, 0.02, 0.05, 0.1\}$
PPO	$-\mathbb{E}_{\tau\sim\pi_\theta}\widehat{r}(\tau) + \beta_{\text{KL}} D_{\text{KL}}(\pi_\theta, \pi^{\text{ref}})$	β_{KL}: $\{0.0001, 0.0005, 0.001, 0.005\}$
GRPO	$-\mathbb{E}_{\tau\sim\pi_\theta}\widehat{r}(\tau) + \beta_{\text{KL}} D_{\text{KL}}(\pi_\theta, \pi^{\text{ref}})$	β_{KL}: $\{0.001, 0.005, 0.01, 0.02\}$
ReDPO	$-\mathbb{E}_{\mathcal{D}^{\text{pref}}} \log \sigma(\beta d_\theta) + \alpha \mathcal{B}(\theta)$	β: $\{0.01, 0.02, 0.05, 0.1\}$ α: $\{0.02, 0.04, 0.06, 0.08\}$

Table 3: Preference optimization objectives and hyperparameter search ranges, shared by both models. β_{KL} is the RLHF KL coefficient. PPO and GRPO share the displayed RL objective but use different advantage estimators; only their KL coefficients are searched.

Method	Parameter	Qwen2.5-7B-Instruct	Llama-3.2-3B-Instruct
IPO	β	0.05	0.1
SimPO	β	2.0	1.0
	γ	1.2	0.5
ORPO	λ	1.0	0.5
DPO	β	0.02	0.02
PPO	β_{KL}	0.001	0.001
GRPO	β_{KL}	0.01	0.01
ReDPO	β	0.02	0.02
	α	0.04	0.04

Table 4: Selected hyperparameters for the two instruction-tuned models. ReDPO, PPO and GRPO each use two on-policy rollouts per prompt for both models.

Qwen 2.5 Instruct

```
<|im_start|>system
You are Qwen, created by Alibaba Cloud. You are a helpful assistant.
<|im_end|>
<|im_start|>user
{user_message}<|im_end|>
<|im_start|>assistant
{assistant_response}<|im_end|>
```

D.4 Objectives and Hyperparameter Search

Table 3 summarizes the optimization objectives and hyperparameter search ranges, which are shared by both models. Each searched parameter has four candidate values, except for SimPO, whose β and γ each have three candidates. For PPO and GRPO, we search only the KL coefficient β_{KL}, keeping the rollout count fixed at two responses per prompt. The selected hyperparameters for each model are reported in Table 4.

D.5 Evaluation Protocol

Preference benchmarks. We evaluate instruction-following and conversational quality using the official implementations of AlpacaEval 2 and Arena-Hard v0.1. For AlpacaEval 2, we report the raw win rate (WR) and length-controlled win rate (LC); for Arena-Hard v0.1, we report WR and the style-controlled win rate (SC). These controlled metrics help account for response-length and formatting biases in pairwise judgments. All methods follow the same evaluation and judging protocol

Benchmark	Implementation	Judge	Metrics
AlpacaEval 2	Official	DeepSeek v3.2	WR, LC
Arena-Hard v0.1	Official	GPT-4.1-2025-04-14	WR, SC

Table 5: Evaluation frameworks, judge models and reported metrics for preference evaluation. WR denotes raw win rate, LC denotes length-controlled win rate, and SC denotes style-controlled win rate.

Evaluator	Human agreement (%)	Spearman corr.	Pearson corr.	Bias	Variance	Prefer longer	Prefer lists	Prefer first
GPT-4	69.17	0.97	0.93	28.40	14.62	0.68	0.73	0.50
DeepSeek v3.2	68.26	0.92	0.82	29.94	11.12	0.67	0.70	0.52
GPT-4 Turbo	65.73	0.78	0.77	33.90	23.65	0.61	0.57	0.53
Humans	65.66	1.00	1.00	0.00	34.34	0.64	0.60	0.52
Claude	65.32	0.93	0.90	32.41	18.48	0.66	0.67	0.49
text-davinci-003	64.08	0.85	0.83	33.80	22.72	0.70	0.66	0.47

Table 6: Judge quality measured using the official AlpacaEval evaluation procedure. The final three columns give the probabilities of preferring a longer answer, an answer with a list, and the first answer, respectively. DeepSeek v3.2 has human agreement and ranking correlations comparable to the GPT-4 evaluators in this comparison.

within each benchmark. Table 5 summarizes the evaluation frameworks, judge models, and reported metrics.

General capabilities. We also use OpenCompass to evaluate GPQA Diamond, MMLU, ARC-C, HellaSwag, WinoGrande, GSM8K and MATH. The SFT checkpoint and its preference-tuned policies are evaluated under the same task configuration, so that changes in performance measure the effect of preference optimization. We report the task-specific scores to retain the distinction between knowledge, commonsense reasoning and mathematical abilities.

D.6 Validating the AlpacaEval Judge

The retirement schedule of the GPT-4o model recommended by AlpacaEval raises a reproducibility concern for evaluations that depend on its continued availability. We therefore use DeepSeek v3.2 as the AlpacaEval 2 judge. To assess this choice, we apply the judge-quality evaluation procedure provided by the official AlpacaEval repository, comparing agreement with human judgments, correlations with human-based model rankings, bias, variance and preference tendencies.

Table 6 reports the results. DeepSeek v3.2 achieves a human agreement rate of 68.26%, compared with 69.17% for GPT-4 and 65.73% for GPT-4 Turbo. Its Spearman and Pearson correlations are 0.92 and 0.82, respectively, between those of GPT-4 and GPT-4 Turbo. These results indicate that DeepSeek v3.2 provides judgments of comparable quality to the GPT-4 evaluators considered here. Its measured preferences for longer answers, lists and the first answer are also similar in magnitude to those of these evaluators. We use the same DeepSeek judge for every method in the AlpacaEval 2 comparison.