ReAlign: Structured Revision for Small Language Model Alignment

Anonymous ACL submission

Abstract

Aligning small language models with human preferences remains a challenging problem: weak policies often struggle to produce informative on-policy samples and exhibit unstable gradients when trained on off-policy signals from stronger models or human annotators. In this work, we introduce *ReAlign*, a training framework that combines the stability of on-policy learning with the guidance of reviser-assisted supervision. In ReAlign, a lightweight external reviser is first trained to improve policy-generated responses using preference-based feedback, conditioned on both the prompt and the initial output. The policy is then optimized using a hybrid approach that leverages standard on-policy preference pairs alongside reviser-enhanced pairs framed as a structured revision task. These enhanced pairs provide richer, more informative supervision, and facilitate more effective optimization. Extensive experiments on AlpacaEval-2 and Arena-Hard demonstrate that ReAlign consistently improves alignment performance for small language models and outperforms strong preference optimization baselines.

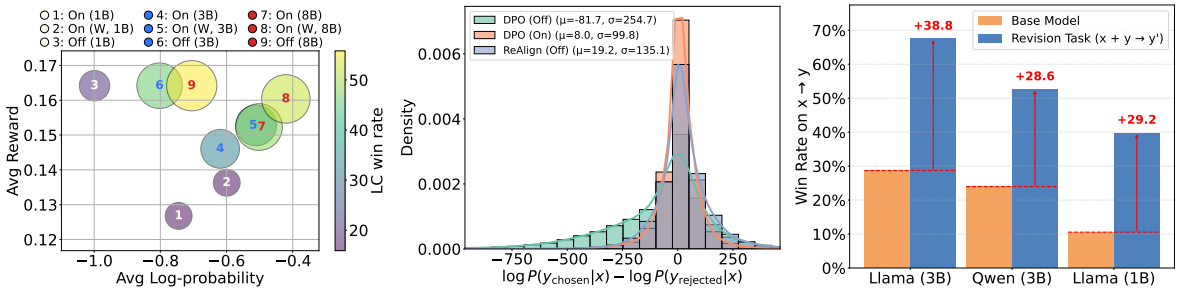
1 Introduction

Aligning small language models (SLMs) with human preferences is challenging, especially when using on-policy methods such as reinforcement learning from human feedback (RLHF) (Ouyang et al., 2022). These methods rely on model-generated outputs; however, small models often produce uniformly low-quality responses in the early training, resulting in weak and noisy supervision signals.

Two broad strategies have been explored to address this problem. The first involves supervised fine-tuning (SFT) on high-quality, human-annotated or model-generated data to bootstrap the model's performance, followed by on-policy optimization (Ouyang et al., 2022). The second strategy collects high-quality preference data from stronger models and employs off-policy optimization methods such as Direct Preference Optimization (DPO) (Rafailov et al., 2024; Teknium, 2023; Zhu et al., 2023). However, since these responses are sampled independently of the weak model, they often exhibit a distributional mismatch. To mitigate this, an intermediate SFT step is commonly applied to reduce the gap and enhance the effectiveness of subsequent off-policy learning (Xu et al., 2024; Tang et al., 2024). Figure 1a illustrates how the two training routes interact with model scale. For smaller models (e.g., 1B), off-policy training accounts for the majority of performance gains, while on-policy updates contribute only marginal improvements. However, as model capacity increases from 1B to 3B and then to 8B, the incremental benefit of on-policy optimization becomes more pronounced, with the largest gains ultimately observed at 8B. In other words, off-policy data provide high-reward signals that weak models cannot generate on their own, while on-policy optimization grows increasingly effective and reinforces the model's existing strengths (Li and Khashabi, 2025).

These observations naturally raise the question: *Can we integrate the strengths of on-policy and off-policy methods into a unified framework?* Unfortunately, directly combining on-policy and off-policy supervision within a standard DPO objective can lead to unstable optimization. As shown in Figure 1b, off-policy pairs often lie far outside the policy's current distribution and exhibit large negative log-probability margins and higher variance than on-policy. The corresponding loss term therefore dominates the aggregate gradient, swamping the on-policy signal and driving the model toward regions it cannot reach (Zhou et al., 2024; Tajwar et al., 2024). In contrast, on-policy training yields small, positive margins that align more closely with the model's generation behavior (Yan et al., 2024).

To address this incompatibility, we introduce a *revision task* into the policy training process.



(a) On-policy vs. off-policy (b) Log-probability margin distribution (c) Revision task impact

Figure 1: **(a)** Comparison of on-policy and off-policy DPO training. X-axis shows the average log-probability of chosen responses; Y-axis shows their average reward. Bubble size and color indicate the LC win rate on AlpacaEval-2. "On (W)" denotes on-policy optimization initialized with SFT using strong model chosen responses. **(b)** Distribution of log-probability margins under different training setups. We compute the log-probability margin using each method's reference model. For ReAlign, x denotes the prompt with initial response ($x = x + y_0$). **(c)** Impact of training solely on revision task preference data. Blue bars show the AlpacaEval-2 win rate ranked by ArmoRM after DPO training using only the revision task preference data; grey bars denote the original win rate.

Rather than forcing the policy to imitate responses from strong models, which often lie outside the policy model's distribution, we train the policy to *revise its own responses* toward improved alternatives. Specifically, the policy takes both a prompt and its self-generated response as input, and learns to prefer stronger candidates over weaker ones. By anchoring the learning signal to the model's own outputs, the revision task ensures distributional consistency with on-policy data while enabling supervision from higher-quality responses (see Figure 1b). This setup allows preference information to be integrated more smoothly, even when the initial generations are suboptimal. The revision task serves two key purposes. First, it enables context-aware preference learning by conditioning on the initial response. Second, it provides more informative supervision in early training, when the model's outputs are weak and standard on-policy learning lacks useful gradients. Our preliminary experiments (Figure 1c) show that training solely on the revision task can yield substantial improvements, suggesting its potential as an effective alignment objective.

Then the question becomes: *How do we obtain the improved responses required for the revision task?* A direct approach is to reuse high-quality answers from a strong model. However, these responses are generated independently of the policy and often diverge in style and structure. When paired with self-generated outputs, they tend to produce inconsistent log-probability margins, which destabilize training and reduce the effectiveness.

To mitigate this, we introduce a lightweight *Reviser* model that generates improved responses conditioned on the policy's initial output. Compared to raw strong-model responses, these revisions are closer to the original generation while maintaining high quality. Empirically, preference pairs constructed from such revisions exhibit smaller and more consistent log-probability margins (Figure 1b), and resemble the low-gap, high-quality pairs studied by Wu et al. (2024). Building on this design, we propose **ReAlign**, a unified training framework that combines on-policy learning with reviser-guided off-policy supervision. ReAlign trains the policy using both self-sampled preference pairs and revision-task pairs generated by the reviser, enabling more stable and effective alignment, especially for small language models.

We evaluate ReAlign on the UltraFeedback dataset (Cui et al., 2023) by fine-tuning several small policy models using alignment methods such as SFT, DPO, SimPO (Meng et al., 2024), WPO (Zhou et al., 2024), and SIMPLEMIX (Li and Khashabi, 2025). The aligned models are then assessed on the AlpacaEval-2 (Dubois et al., 2024) and Arena-Hard (Li et al., 2024a) benchmarks. Experimental results demonstrate that ReAlign effectively enhances the ability of SLMs to generate high-quality responses, making it a promising approach for aligning SLMs with human preferences.

2 Methodology

We begin by outlining the preliminaries of preference-based alignment. As shown in Figure 2, we then introduce **ReAlign**, a unified framework that integrates on-policy and off-policy supervision through a structured *revision task*. The core components of ReAlign are described in the following subsections: the reviser model and its training procedure (§ 2.2), and the integration of revision-based supervision into policy optimization (§ 2.3).

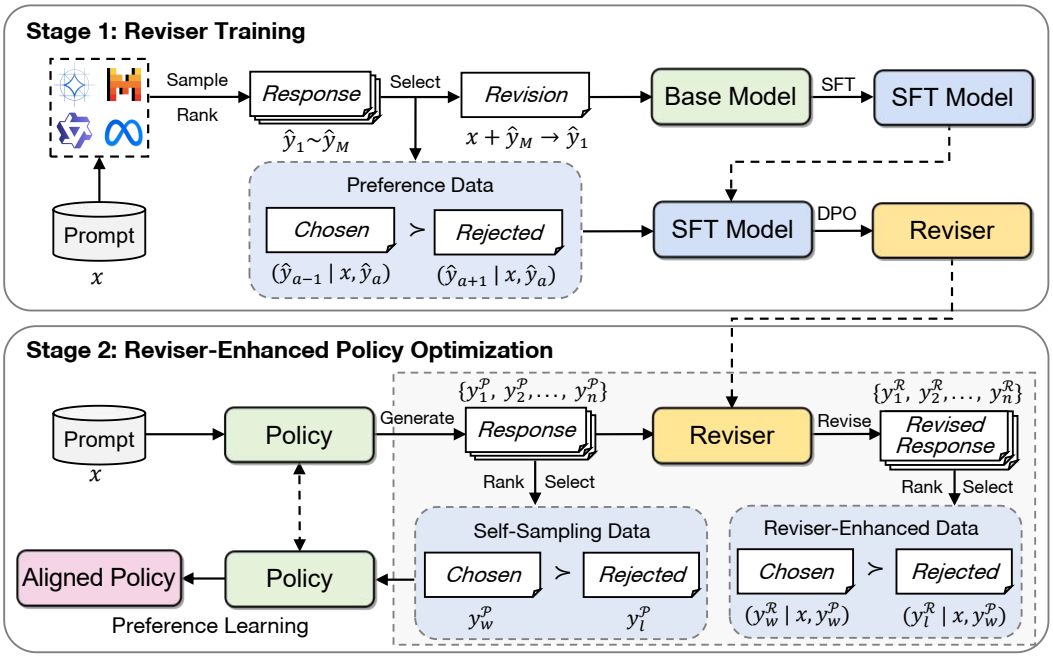


Figure 2: Overview of the proposed ReAlign. ReAlign consists of two stages: reviser training and policy optimization. The reviser is first trained to refine low-quality responses into high-quality ones using paired responses. In the second stage, the policy is optimized using both its own responses and the revised outputs from the reviser. This dual approach enables continuous improvement of the policy from both self-generated and reviser-enhanced responses.

2.1 Preliminaries

Preference-based alignment generally seeks a policy $\pi_\theta(y \mid x)$ that maximizes expected preference scores based on pairwise response comparisons. Formally, this objective can be expressed as:

$$\max_\theta \mathbb{E}(x, y_w, y_l) \sim D\Big[s\big(x, y_w, y_l\big)\Big], \tag{1}$$

where (x, y_w, y_l) denotes a prompt and a preference-labeled response pair, and s is a scoring function that evaluates the alignment of the policy with the preferences. Existing methods differ mainly in how the preference data D is constructed. *On-policy* methods generate response pairs from the current policy, ensuring distributional consistency but suffering from low sample quality in small models. In contrast, *off-policy* methods rely on responses from stronger sources, which offer clearer supervision but often exhibit significant distributional mismatch with the target model. This trade-off motivates the need for a unified approach that can combine the stability of on-policy learning with the signal quality of off-policy supervision.

2.2 Reviser Training

To mitigate the distributional mismatch introduced by directly imitating stronger models, we introduce a lightweight external **reviser model** $\mathcal{R}_\theta$. The reviser generates improved responses by refining the policy's initial outputs, thereby providing reward-enhancing yet distribution-consistent supervision.

Motivation Given a prompt x, let $y_0 \sim \pi_\theta(\cdot \mid x)$ be the response from the policy π_θ, and $\hat{y} \sim \pi_\psi(\cdot \mid x)$ be a response sampled from a strong model π_ψ. Since the strong model response $\hat{y}$ is generated independently of y_0, it often deviates from the policy's distribution, making direct off-policy optimization unstable.

To alleviate this issue, the reviser $\mathcal{R}_\theta$ is trained to condition its generation on an initial response:

$$\mathcal{R}_\theta : (x,\, y^{\mathrm{L}}) \mapsto y^{\mathrm{H}}, \quad r(x, y^{\mathrm{H}}) \succ r(x, y^{\mathrm{L}}), \tag{2}$$

where y^{L} denotes a lower-quality response and y^{H} a higher-quality response, and r is a reward model. Although the reviser's output is not sampled from the policy, its conditional nature ensures higher overlap with the policy distribution than π_{s}, providing a stronger yet learnable training signal.

Phase 1: Warm-up Training Reviser training begins by supervised fine-tuning (SFT) to establish a basic revision capability. Specifically, for each prompt x, we first collect M responses $\hat{y}_{1:M}$ from strong models, ranked in descending order of reward, with $\hat{y}_1$ receiving the highest reward and $\hat{y}_M$ the lowest. The reviser model is then trained to maximize the likelihood of mapping the lowest-

ranked response to the highest-ranked response:

$$\mathcal{L}_{\text{SFT}} = -\mathbb{E}_{(x,\hat{y}_1,\hat{y}_M)}\big[\log P_\theta(\hat{y}_1 \mid x, \hat{y}_M)\big]. \quad (3)$$

Phase 2: Preference-Consistent Optimization The reviser is further refined through a preference-consistent optimization approach. Responses collected from strong models are grouped into quantiles according to their reward scores. Let $k > 1$ be the number of quantiles, and define the anchor indices $\mathcal{A} = \{\, a_j = \lfloor jM/k \rfloor \mid j = 1, \ldots, k-1\}$. For each quantile-based anchor response $\{\hat{y}_a | a \in \mathcal{A}\}$, the reviser learns to generate a slightly better response $\hat{y}_{a-1}$ rather than a worse one $\hat{y}_{a+1}$: $\big(x, \hat{y}_a\big) \rightarrow \hat{y}_{a-1} \succ \big(x, \hat{y}_a\big) \rightarrow \hat{y}_{a+1}$.

The reviser training then employs a Direct Preference Optimization (DPO)-inspired objective adapted explicitly for incremental improvements:

$$\mathcal{L}_{\text{reviser}}(\theta; \theta_{\text{ref}}) = -\mathbb{E}\Bigg[\log \sigma\Big(\beta \log \frac{\mathcal{R}_\theta(\hat{y}_{a-1} \mid x, \hat{y}_a)}{\mathcal{R}_{\theta_{\text{ref}}}(\hat{y}_{a-1} \mid x, \hat{y}_a)} - \beta \log \frac{\mathcal{R}_\theta(\hat{y}_{a+1} \mid x, \hat{y}_a)}{\mathcal{R}_{\theta_{\text{ref}}}(\hat{y}_{a+1} \mid x, \hat{y}_a)}\Big)\Bigg]. \quad (4)$$

This formulation aligns the reviser's training objective with the intuitive goal of consistently improving responses rather than degrading them, thereby ensuring stable and incremental enhancements to the policy's response generation capability. The training details are provided in Appendix B.

2.3 Policy Optimization

Having trained and fixed the parameters of the reviser $\mathcal{R}$, we now optimize the policy π_θ using two complementary types of preference supervision: *on-policy* pairs sampled directly from the policy, and *revision-based* pairs constructed from the reviser. These two sources of training data are referred to as *Self-Sampling Data* and *Reviser-Enhanced Data*, respectively, as shown in Figure 2.

Given a prompt x, the policy π_θ first generated a set of responses $\{y_j^{\mathcal{P}}\}_{j=1}^n \sim \pi_\theta(\cdot \mid x)$. These responses are ranked according to scores from an external reward model r, and we select the responses with the highest and lowest scores, denoted as $y_w^{\mathcal{P}}$ and $y_l^{\mathcal{P}}$, to construct the on-policy preference pair $(y_w^{\mathcal{P}} \succ y_l^{\mathcal{P}} \mid x)$. To obtain revision-based supervision, each policy response $y_j^{\mathcal{P}}$ is revised by the trained reviser: $y_j^{\mathcal{R}} = \mathcal{R}(x, y_j^{\mathcal{P}})$. These revised outputs are similarly ranked, and we select the best and worst revised responses $y_w^{\mathcal{R}}, y_l^{\mathcal{R}}$. Together with the anchor $y_w^{\mathcal{P}}$, we form a revision-task preference pair $(y_w^{\mathcal{R}} \succ y_l^{\mathcal{R}} \mid x, y_w^{\mathcal{P}})$, which preserves the structure and context of the policy's original outputs.

Reviser-Enhanced Policy Optimization To effectively utilize both types of training data, we formulate a dual-objective optimization strategy. For each prompt x, the model is trained on two types of preference pairs: (1) an on-policy pair $(y_w^{\mathcal{P}} \succ y_l^{\mathcal{P}} | x)$ sampled directly from the policy distribution, and (2) a revision task pair $(y_w^{\mathcal{R}} \succ y_l^{\mathcal{R}} | x, y_w^{\mathcal{P}})$, where the revised responses are both conditioned on the policy responses. This setup reflects two complementary learning signals: the first encourages the policy to improve its own response ranking based on internal variation, while the second provides additional supervision by leveraging the reviser's structured improvements over a fixed policy sample. Formally, each prompt contributes:

$$\underbrace{(x,\ y_w^{\mathcal{P}} \succ y_l^{\mathcal{P}})}_{\text{Self-Sampling Data}} \quad \text{and} \quad \underbrace{(x,\ y_w^{\mathcal{P}}, y_w^{\mathcal{R}} \succ y_l^{\mathcal{R}})}_{\text{Reviser-Enhanced Data}}.$$

The final objective is the sum of both losses:

$$\mathcal{L}(\theta) = \mathbb{E}_{(x, y_w^{\mathcal{P}}, y_l^{\mathcal{P}})}\big[\mathcal{L}_{\mathcal{O}}(\theta; x, y_w^{\mathcal{P}}, y_l^{\mathcal{P}})\big] + \mathbb{E}_{(x, y_w^{\mathcal{P}}, y_w^{\mathcal{R}}, y_l^{\mathcal{R}})}\big[\mathcal{L}_{\mathcal{O}}(\theta; x, y_w^{\mathcal{P}}, y_w^{\mathcal{R}}, y_l^{\mathcal{R}})\big], \quad (5)$$

where $\mathcal{L}_{\mathcal{O}}$ is a generic preference-based loss, which can be instantiated by a specific preference optimization algorithm (e.g., DPO or SimPO)

Rationale for the Revision Task Early on-policy learning in SLMs yields limited progress: responses are often nearly indistinguishable, leading to narrow preference margins and shallow updates that offer weak learning signals.

To provide stronger supervision, higher-quality responses are needed. While using outputs from a strong model offers such quality, they often lie far outside the weak model's distribution, potentially destabilizing training. In contrast, revision-task pairs are conditioned on the policy's own responses, retaining margin scales similar to on-policy pairs while offering clearly improved answers.

3 Experimental Setup

Training Datasets Following prior work (Meng et al., 2024; Zhou et al., 2024), we conduct all experiments based on the UltraFeedback dataset (Cui et al., 2023), which contains 64K diverse instruction prompts covering a wide range of real-world tasks. To reflect current model capabilities, we augment the dataset by re-sampling responses from four strong open-source models[1]. Each model

[1]Gemma-2-27B-It (Team et al., 2024), Mistral-Large-Instruct-2407, Qwen-2.5-72B-Instruct (Yang et al., 2024), and Llama-3.1-70B-Instruct (Dubey et al., 2024)

generates five responses per prompt, the collected responses are then pooled and ranked using the ArmoRM-Llama3-8B-v0.1 reward model (Wang et al., 2024). We select the top- and bottom-ranked responses to construct high-quality preference pairs, which are used to train both the reviser and policy models following the procedure in § 2.2.

Training Setup The reviser is initialized from LLaMA-3-8B-Instruct (Grattafiori et al., 2024) and trained on preference pairs constructed from re-ranked completions of strong models, as described in Section 2.2. For policy training, we consider three small policy backbones: Llama-3.2-3B-Instruct, Llama-3.2-1B-Instruct, and Qwen2.5-3B-Instruct (Yang et al., 2024). Policy training proceeds in two stages. We first apply supervised fine-tuning on 30% of the training prompts using top-ranked responses or the revised responses from the reviser. The remaining 70% is used for preference optimization based on self-sampled and reviser-enhanced response pairs, as described in Section 2.3. A comprehensive description of the training procedure is available Appendix C.

Baselines and Evaluation We compare **ReAlign** against several strong alignment baselines, including: (i) **SFT**, supervised fine-tuning using top-ranked responses from strong models; (ii) **DPO-On / SimPO-On**, on-policy preference optimization using self-sampled responses; (iii) **DPO-Off / SimPO-Off / WPO-Off**, off-policy optimization using strong-model responses after SFT warm-up; and (iv) **SIMPLEMIX** (Li and Khashabi, 2025), which randomly mixes on-policy and off-policy data after SFT warm-up. We evaluate all methods on two widely used instruction-following benchmarks: (i) **AlpacaEval-2** (Dubois et al., 2024), which reports length-controlled (LC) and raw win rates (WR) against GPT-4-Preview-1106; and (ii) **Arena-Hard** (Li et al., 2024a), a curated set of challenging reasoning tasks from Chatbot Arena. GPT-4-Preview-1106 is used as the judge model, and we report win rate (WR) and style-controlled (SC) win rate against GPT-4-0314 as reference. Additional results on domain-specific tasks (e.g., QA, math, coding) are included in Appendix E.

4 Main Results

Table 1 presents the performance of our proposed ReAlign method compared with several baselines on AlpacaEval-2 and Arena-Hard.

On-policy vs. Off-policy Optimization We first compare on-policy and off-policy optimization across all policy models. On AlpacaEval-2, off-policy optimization consistently outperform on-policy alternatives. For example, DPO-Off improves the LC win rate from 33.23% to 45.67% on Llama-3.2-3B-Instruct, supporting the hypothesis that weak models struggle to generate high-quality on-policy data early in training. In contrast, off-policy data provides stronger guidance with high-quality responses from stronger models. However, on Arena-Hard, Qwen2.5-3B-Instruct achieves better performance with on-policy DPO (38.2%) than with off-policy DPO (35.3%). This suggests that on-policy optimization becomes more effective when the policy is sufficiently capable or better aligned with the target distribution, possibly as a result of pretraining differences. This observation aligns with the findings of Song et al. (2024).

Limits of Off-policy Learning Off-policy optimization provides strong supervision but often fails to align with a weak model's learning capacity. For instance, on Llama-3.2-3B-Instruct, DPO-Off achieves 45.67% LC win rate, whereas ReAlign further improves it to 50.17%. This gap indicates that stronger responses alone are insufficient, and effective supervision must also align with the model's learning capacity. Methods like WPO and SIMPLEMIX aim to alleviate this by weighting or mixing on-policy and off-policy data, and they do offer improved stability over DPO-Off. However, because they treat the two sources independently, the off-policy signals may still fall outside the model's reachable distribution and thus remain underutilized. This limitation is especially pronounced for smaller models. On Llama-3.2-1B-Instruct, DPO-Off reaches only 7.5% LC win rate on Arena-Hard, lower than even the SFT baseline (7.9%). These results highlight the need for approaches like ReAlign, which anchor high-quality supervision in the model's own outputs to improve learnability.

Advantages of ReAlign As shown in Table 1, ReAlign outperforms all baselines in most settings across different model types and preference optimization methods. On AlpacaEval-2, it improves the LC win rate from 20.36% (DPO-Off) to 25.36% on Llama-3.2-1B-Instruct, and from 17.55% (SimPO-Off) to 25.53%. On Arena-Hard, it raises the SC win rate from 7.5% (DPO-Off) to 9.3%, with similar trends observed for 3B and Qwen models. Its effectiveness stems from two core design choices. First, the reviser produces responses that are not only higher quality but also

Method	AlpacaEval-2			Arena-Hard		
	LC(%)	WR(%)	Avg. Len.	SC(%)	WR(%)	Avg. Len.
Llama-3.2-3B-Instruct						
SFT	27.59	27.09	1,965	20.40	21.10	2,826
WPO-Off	49.78	**52.99**	2,217	19.30	20.70	3,026
SIMPLEMIX	45.72	50.95	2,278	26.50	28.10	3,139
DPO/SimPO-On	33.23/31.00	32.86/32.16	1,967/2,025	22.90/20.20	23.70/21.40	2,732/2,652
DPO/SimPO-Off	45.67/43.88	47.09/30.38	2,120/1,423	23.10/16.10	23.30/16.80	2,871/2,326
ReAlign (DPO/SimPO)	**50.17**/47.50	51.26/49.26	2,073/2,100	**28.50**/26.30	**29.20**/26.60	2,933/2,955
Qwen2.5-3B-Instruct						
SFT	20.89	18.45	1,874	22.80	23.00	2,933
WPO-Off	39.19	43.22	2,197	35.00	35.00	3,052
SIMPLEMIX	33.57	37.56	2,176	34.70	34.80	3,277
DPO/SimPO-On	24.81/28.26	34.10/31.27	2,468/2,138	**38.20**/36.50	**38.40**/36.60	3,441/3,347
DPO/SimPO-Off	37.72/37.81	39.92/27.99	2,144/1,552	35.30/30.60	35.70/30.90	3,167/2,310
ReAlign (DPO/SimPO)	44.04/**45.30**	**44.95**/43.40	2,086/1,999	35.50/36.60	35.20/37.00	2,866/3,160
Llama-3.2-1B-Instruct						
SFT	12.28	11.62	1,841	7.90	8.10	2,890
WPO-Off	21.72	24.63	2,159	6.00	6.70	2,811
SIMPLEMIX	20.14	23.05	2,286	7.30	7.60	2,796
DPO/SimPO-On	15.95/18.61	19.06/15.04	2,191/1,696	7.00/7.20	7.70/8.80	3,125/2,129
DPO/SimPO-Off	20.36/17.55	22.06/12.84	2,076/1,567	7.50/7.40	7.70/7.50	2,822/2,618
ReAlign (DPO/SimPO)	25.36/**25.53**	**29.06**/27.35	2,185/2,066	**9.30**/8.90	**9.70**/9.40	2,949/2,804

Table 1: Performance comparison results of ReAlign with other baselines on AlpacaEval-2 and Arena-Hard.

structurally similar to policy outputs, resulting in preference pairs with moderate log-probability gaps that are easier to learn from. Second, ReAlign organizes these signals as a structured revision task, grounding supervision in the model's own generation process. Together, these features allow ReAlign to combine the strength of off-policy supervision with the stability of on-policy optimization.

5 Ablation Studies

To analyze the contributions of different components in ReAlign, we conduct ablation experiments along two orthogonal axes: (i) Pair format: Standard direct (D) format $(y_w \succ y_l \mid x)$ and revision-task anchored (A) format $(y_w \succ y_l \mid x, y^{\mathcal{P}})$ where $y^{\mathcal{P}}$ is the policy's initial response. (ii) Response source: Specifies where the preferred and dispreferred responses (y_w, y_l) in each pair are sampled from, including (P) policy self-sampled responses, (R) reviser outputs, and (S) responses from the strong model. All experiments are using Llama-3.2-3B-Instruct. The results of ablation studies are presented in Table 2.

Impact of the Revision Task We evaluate whether structuring supervision as a revision task improves learning. Comparing P-D + R-D and P-D + R-A, the latter achieves a higher LC win rate (50.17% vs. 47.67%) using the same reviser outputs. A similar gain is observed with strong-model completions (P-D + S-A > P-D + S-D), confirming that conditioning on policy outputs yields more learnable signals. Even without on-policy data, revision-only training (R-A) performs competitively, supporting our claim that grounding preference pairs in the model's generation space improves learnability, especially for weaker policies.

Training Data	LC(%)	WR(%)
Llama-3.2-3B-Instruct	20.00	23.28
Stepwise Ablation		
P-D	33.23	32.86
+ S-D → (P-D + S-D)	44.65	50.74
+ S-D → (P-D + S-A)	45.53	46.45
+ S-A → (P-D + R-D)	47.67	48.08
+ R-D → (P-D + R-A)	50.17	51.26
Individual or Alternative Variants		
S-D only	45.67	47.09
R-D only	48.00	47.82
R-A only	47.57	48.07
R-D + R-A	46.06	53.72

Table 2: Ablation results for ReAlign (DPO) on AlpacaEval-2. P-D means on-policy preference pairs. S-D and S-A use strong responses in direct and anchored formats, respectively. R-D and R-A use reviser responses in direct and anchored formats. '+' denotes an additive combination of preference data types used jointly in a single training epoch.

Model	Version	SFT		ReAlign (DPO)	
		WR (%)	Avg. L	WR (%)	Avg. L
Llama-3B	Initial	50.0	2137	84.2	2073
	Revised	81.2	2058	83.9	2130
Qwen-3B	Initial	48.8	1872	74.9	2086
	Revised	66.2	1981	75.8	2228
Llama-1B	Initial	20.6	2043	59.1	2186
	Revised	44.6	2130	56.0	2149

Table 3: Self-revision performance of ReAlign (DPO) on AlpacaEval-2. Each policy generates an initial response and then performs a single self-revision. WR against GPT-4-0314 is evaluated by ArmoRM score.

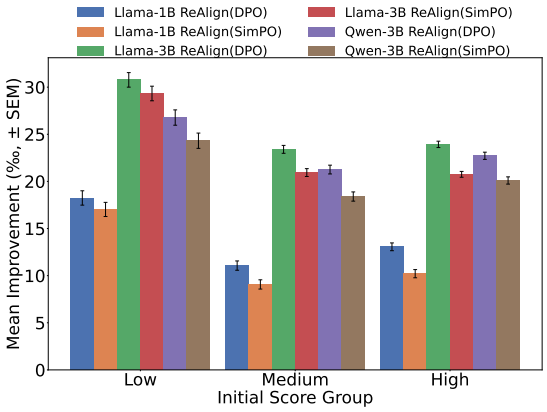


Figure 3: Reward improvement from self-revision across initial response quality. We collect 8 external models [2]responses to AlpacaEval prompts and group them into low/medium/high bins based on ArmoRM scores. Each ReAlign policy performs one-step revision per response. Bars show mean reward gain $\Delta r = r_{\text{revised}} - r_{\text{initial}}$ in each bin (± SEM).

Impact of the Reviser Reviser outputs yield better performance than strong-model responses. For instance, R-D outperforms S-D (48.00% vs. 45.67%), indicating that reviser-generated responses are both higher quality and more learnable for weak policies. Similarly, P-D + R-A surpasses P-D + S-A (50.17% vs. 45.53%), showing that revision supervision is more effective when conditioned on the policy's own outputs. This policy-awareness keeps the learning signal within the model's reachable distribution.

On-policy Learning with Revision Task Removing on-policy pairs and training only on revision-based data (R-D + R-A) leads to a noticeable drop in performance compared to P-D + R-A (46.06% vs. 50.17%), suggesting that decoupling supervision from current policy impairs learning. In P-D + R-A, revisions are grounded in preferred policy responses, ensuring alignment with the model's behavior. By contrast, R-D + R-A combines two sources detached from the policy inference, introducing conflicting signals. A similar mismatch occurs when adding direct reviser pairs to revision-only training (R-D + R-A vs. R-A), where performance slightly degrades due to incompatible optimization signals.

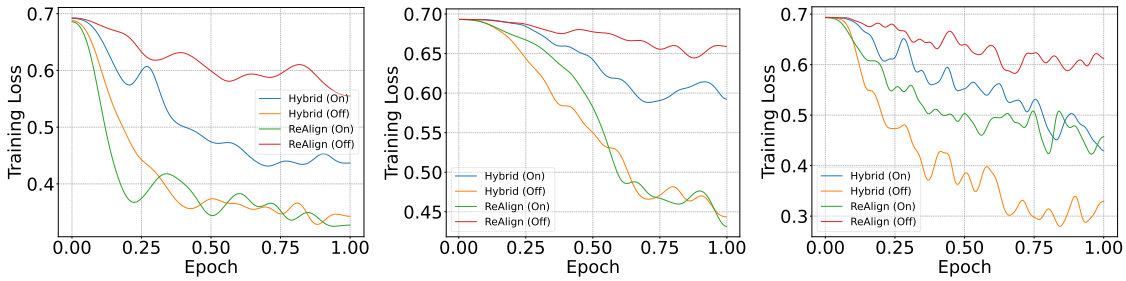
6 Discussion and Analysis

Diminishing Returns of Self-Revision Table 3 shows that self-revision substantially improves SFT models. For example, Llama-3B improves from 50.0% to 81.2% win rate, where the initial responses contain many fixable flaws. In contrast, the same revision procedure yields little or no gain after ReAlign training (e.g., 84.2% to 83.9% for Llama-3B), suggesting that the initial outputs are already near the policy's upper bound. Figure 3 confirms this interpretation. When responses from external models are grouped by quality, ReAlign policies deliver the largest reward gain Δr on low-quality inputs, and diminishing gains on medium and high-quality ones. This indicates that the learned revision ability remains effective, but it naturally declines as generation quality improves.

Training Behavior for ReAlign To investigate learning behavior under different strategies, we analyze training loss trajectories between ReAlign and a hybrid baseline that combines on-policy supervision with direct off-policy pairs from strong models. Figure 4 shows that ReAlign achieves faster loss reduction on on-policy data, especially in early training, indicating that its preference signals are better aligned with the model's distribution and easier to optimize. By conditioning off-policy supervision on policy outputs through a revision task, ReAlign avoids the gradient inconsistency that can arise when directly mixing distributions. In contrast, the hybrid baseline quickly reduces off-policy loss due to large reward gaps but yields slower improvements on on-policy data. This mismatch suggests that such external data may offer overly incompatible signals for weak policies.

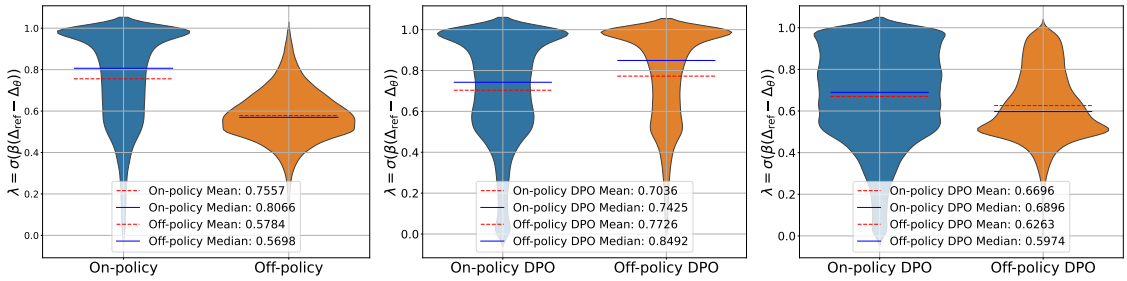
We find that ReAlign's off-policy loss decreases more gradually because revision-task pairs, conditioned on policy outputs, maintain distributional proximity and exhibit smaller margins, similar to on-policy data. These samples remain effective but lead to slower learning. Overall, the training dynamics support our hypothesis that revision tasks

[2]Qwen2.5-3B-Instruct, Llama-3.2-1B-Instruct, Llama-3.2-3B-Instruct, Qwen2-72B-Instruct, Meta-Llama-3.1-405B-Instruct-Turbo, Meta-Llama-3.1-70B-Instruct-Turbo, Meta-Llama-3.1-8B-Instruct-Turbo, and GPT-4o-2024-05-13.



(a) Llama-3.2-1B-Instruct (b) Llama-3.2-3B-Instruct (c) Qwen2.5-3B-Instruct

Figure 4: Training loss under different strategies. We compare ReAlign and a hybrid baseline on three policy models. ReAlign uses revision-task pairs (from the reviser) as off-policy data, while hybrid mixes on-policy data with direct strong-model off-policy data. Loss is reported separately for on-policy (On) and off-policy (Off) data.



(a) ReAlign (b) Hybrid (c) DPO

Figure 5: Distribution of $\lambda = \sigma(\beta(\Delta_{\text{ref}} - \Delta_\pi))$ values on on-policy and off-policy data after training. We visualize the λ distributions induced by the final policy models trained under different schemes on Llama-3.2-1B. For ReAlign (a), we separately compute λ values for on-policy and revision-task off-policy data. For hybrid (b), we compute λ over on-policy and strong model's off-policy data. For standard DPO (c), we report λ values under models trained only on on-policy or off-policy data, respectively. Higher λ indicates more confident preference alignment.

provide more compatible and effective guidance. By structuring off-policy data around the model's generation space, ReAlign enables steady optimization without destabilizing gradients.

Analyzing λ Distributions We analyze the λ distributions (detailed in Appendix D) to understand how different training schemes modulate preference weighting. As shown in Figure 5, higher λ values indicate stronger policy–reference agreement and thus higher learning confidence. ReAlign exhibits sharply peaked λ values on on-policy data (mean = 0.76, median = 0.81), showing strong agreement with its own generation space. In contrast, its off-policy revision pairs have lower mean λ (0.58), suggesting these are harder but still informative examples. This reflects ReAlign's ability to guide learning without departing from the model's distribution. Hybrid, however, shows higher λ on off-policy than on-policy (0.77 vs. 0.70), indicating that strong-model completions dominate learning. While this may speed up optimization, it risks overwhelming the model with mismatched signals, especially for weaker policies. Overall, ReAlign adapts supervision strength to the model's capacity, promoting improvement without forcing imitation. This selective weighting helps avoid overfitting and supports more stable policy learning.

7 Conclusion

In this work, we presented ReAlign, a novel framework for aligning small language models that combines the stability of on-policy learning with the guidance of reviser-assisted supervision. By introducing a revision task grounded in the model's own generations, ReAlign mitigates the distributional mismatch issues commonly seen in off-policy preference optimization and provides richer, more stable learning signals during early training. The integration of a lightweight reviser enables the creation of high-quality, preference-aligned revisions that remain close to the policy's distribution, allowing small language models to benefit from stronger supervision without destabilizing gradients. Our empirical results on challenging benchmarks such as AlpacaEval-2 and Arena-Hard demonstrate that ReAlign consistently outperforms existing baselines across various small model sizes. These results highlight the importance of conditioning supervision on the model's own behavior and suggest that structured revision-based training offers a scalable and effective strategy for preference alignment.

Limitations

In this paper, we propose ReAlign to address the challenge of small language model alignment. However, we do not explore the performance of ReAlign when combined with online RL algorithms such as PPO and REINFORCE. The reason lies in the fact that ReAlign learns to refine responses generated by the reviser, which are not sampled directly from the policy itself. This setting introduces a certain inconsistency with online RL algorithms. Nevertheless, we believe that the potential of ReAlign can be further unlocked when integrated with online RL algorithms. To achieve this, it is necessary to first address this inconsistency and enable ReAlign to better align with online RL algorithms. This will be the focus of our future work.

References

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

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, and 1 others. 2022a. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, and 1 others. 2022b. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*.

Ruijun Chen, Jiehao Liang, Shiping Gao, Fanqi Wan, and Xiaojun Quan. 2024. Self-evolution fine-tuning for policy optimization. *arXiv preprint arXiv:2406.10813*.

Jiale Cheng, Xiao Liu, Cunxiang Wang, Xiaotao Gu, Yida Lu, Dan Zhang, Yuxiao Dong, Jie Tang, Hongning Wang, and Minlie Huang. 2024. Spar: Self-play with tree-search refinement to improve instruction-following in large language models. *arXiv preprint arXiv:2412.11605*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. 2023. Ultrafeedback: Boosting language models with high-quality feedback. *arXiv preprint arXiv:2310.01377*.

Qingxiu Dong, Li Dong, Xingxing Zhang, Zhifang Sui, and Furu Wei. 2024. Self-boosting large language models with synthetic preference data. *arXiv preprint arXiv:2410.06961*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B Hashimoto. 2024. Length-controlled alpacaeval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475*.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. 2024. Kto: Model alignment as prospect theoretic optimization. *arXiv preprint arXiv:2402.01306*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Shangmin Guo, Biao Zhang, Tianlin Liu, Tianqi Liu, Misha Khalman, Felipe Llinares, Alexandre Rame, Thomas Mesnard, Yao Zhao, Bilal Piot, and 1 others. 2024. Direct language model alignment from online ai feedback. *arXiv preprint arXiv:2402.04792*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Jiaming Ji, Boyuan Chen, Hantao Lou, Donghai Hong, Borong Zhang, Xuehai Pan, Tianyi Qiu, Juntao Dai, and Yaodong Yang. 2024. Aligner: Efficient alignment by learning to correct. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Tianjian Li and Daniel Khashabi. 2025. Simplemix: Frustratingly simple mixing of off-and on-policy data in language model preference learning. *arXiv preprint arXiv:2505.02363*.

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph E Gonzalez, and Ion Stoica. 2024a. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. *arXiv preprint arXiv:2406.11939*.

Yafu Li, Xuyang Hu, Xiaoye Qu, Linjie Li, and Yu Cheng. 2025. Test-time preference optimization: On-the-fly alignment via iterative textual feedback. *arXiv preprint arXiv:2501.12895*.

Yixing Li, Yuxian Gu, Li Dong, Dequan Wang, Yu Cheng, and Furu Wei. 2024b. Direct preference knowledge distillation for large language models. *arXiv preprint arXiv:2406.19774*.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Truthfulqa: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252.

Tianqi Liu, Yao Zhao, Rishabh Joshi, Misha Khalman, Mohammad Saleh, Peter J Liu, and Jialu Liu. 2023. Statistical rejection sampling improves preference optimization. In *The Twelfth International Conference on Learning Representations*.

Hantao Lou, Jiaming Ji, Kaile Wang, and Yaodong Yang. 2025. Stream aligner: Efficient sentence-level alignment via distribution induction. *arXiv preprint arXiv:2501.05336*.

Jianqiao Lu, Wanjun Zhong, Wenyong Huang, Yufei Wang, Fei Mi, Baojun Wang, Weichao Wang, Lifeng Shang, and Qun Liu. 2023. Self: Language-driven self-evolution for large language model. *arXiv preprint arXiv:2310.00533*.

Yu Meng, Mengzhou Xia, and Danqi Chen. 2024. Simpo: Simple preference optimization with a reference-free reward. *arXiv preprint arXiv:2405.14734*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. 2023. Gpqa: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

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

Yuda Song, Gokul Swamy, Aarti Singh, J Bagnell, and Wen Sun. 2024. The importance of online data: Understanding preference fine-tuning via coverage. *Advances in Neural Information Processing Systems*, 37:12243–12270.

Zayne Sprague, Xi Ye, Kaj Bostrom, Swarat Chaudhuri, and Greg Durrett. 2024. Musr: Testing the limits of chain-of-thought with multistep soft reasoning. In *Proceedings of the International Conference on Learning Representations*. International Conference on Learning Representations.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and 1 others. 2023. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13003–13051.

Fahim Tajwar, Anikait Singh, Archit Sharma, Rafael Rafailov, Jeff Schneider, Tengyang Xie, Stefano Ermon, Chelsea Finn, and Aviral Kumar. 2024. Preference fine-tuning of llms should leverage suboptimal, on-policy data. In *International Conference on Machine Learning*, pages 47441–47474. PMLR.

Yunhao Tang, Daniel Zhaohan Guo, Zeyu Zheng, Daniele Calandriello, Yuan Cao, Eugene Tarassov, Rémi Munos, Bernardo Ávila Pires, Michal Valko, Yong Cheng, and 1 others. 2024. Understanding the performance gap between online and offline alignment algorithms. *arXiv preprint arXiv:2405.08448*.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, and 1 others. 2024. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*.

Teknium. 2023. Openhermes 2.5: An open dataset of synthetic data for generalist llm assistants.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, and 1 others. 2023. Zephyr: Direct distillation of lm alignment. *arXiv preprint arXiv:2310.16944*.

Haoxiang Wang, Wei Xiong, Tengyang Xie, Han Zhao, and Tong Zhang. 2024. Interpretable preferences via multi-objective reward modeling and mixture-of-experts. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 10582–10592.

Junkang Wu, Yuexiang Xie, Zhengyi Yang, Jiancan Wu, Jinyang Gao, Bolin Ding, Xiang Wang, and Xiangnan He. 2024. β-dpo: Direct preference optimization with dynamic β. *Advances in Neural Information Processing Systems*, 37:129944–129966.

Shusheng Xu, Wei Fu, Jiaxuan Gao, Wenjie Ye, Weilin Liu, Zhiyu Mei, Guangju Wang, Chao Yu, and Yi Wu. 2024. Is dpo superior to ppo for llm alignment? a comprehensive study. *arXiv preprint arXiv:2404.10719*.

Yuzi Yan, Yibo Miao, Jialian Li, Yipin Zhang, Jian Xie, Zhijie Deng, and Dong Yan. 2024. 3d-properties: Identifying challenges in dpo and charting a path forward. *arXiv preprint arXiv:2406.07327*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, and 1 others. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Eunseop Yoon, Hee Suk Yoon, SooHwan Eom, Gunsoo Han, Daniel Nam, Daejin Jo, Kyoung-Woon On, Mark Hasegawa-Johnson, Sungwoong Kim, and Chang Yoo. 2024. Tlcr: Token-level continuous reward for fine-grained reinforcement learning from human feedback. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 14969–14981.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. 2024. Self-rewarding language models. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800.

Rongzhi Zhang, Jiaming Shen, Tianqi Liu, Haorui Wang, Zhen Qin, Feng Han, Jialu Liu, Simon Baumgartner, Michael Bendersky, and Chao Zhang. 2024. PLaD: Preference-based large language model distillation with pseudo-preference pairs. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 15623–15636, Bangkok, Thailand. Association for Computational Linguistics.

Wenxuan Zhou, Ravi Agrawal, Shujian Zhang, Sathish Reddy Indurthi, Sanqiang Zhao, Kaiqiang Song, Silei Xu, and Chenguang Zhu. 2024. Wpo: Enhancing rlhf with weighted preference optimization. *arXiv preprint arXiv:2406.11827*.

Banghua Zhu, Evan Frick, Tianhao Wu, Hanlin Zhu, and Jiantao Jiao. 2023. Starling-7b: Improving llm helpfulness & harmlessness with rlaif.

A Related Work

Traditional Methods of Alignment Aligning language models to human preferences often employs techniques like RLHF (Ouyang et al., 2022), which incorporates human feedback into the training process. It relies on complex reinforcement learning techniques such as Proximal Policy Optimization (Schulman et al., 2017), making it not only difficult to implement but also unstable during training. To overcome these issues, numerous offline RLHF methods have emerged. Representative works include DPO (Rafailov et al., 2024), which simplifies the alignment process by directly optimizing the model using a preference dataset without the need for an explicit reward model.

Further research has sought to address potential limitations of DPO. IPO (Azar et al., 2024) mitigates the risk of overfitting by optimizing a nonlinear preference function, thereby avoiding the conversion of pairwise preferences into pointwise rewards. KTO (Ethayarajh et al., 2024) introduced a new alignment objective called human-aware loss (HALO), which maximizes the utility of generations from a binary signal instead of maximizing the likelihood of preferences. In SimPO (Meng et al., 2024), the reward component in DPO is modified to utilize the average log probability of positive or negative responses from the policy model. Despite these advances, off-policy methods still suffer from a distributional mismatch between training samples and the model, potentially leading to performance degradation.

To resolve this issue, several on-policy preference alignment methods have been proposed. For instance, Yuan et al. (2024) proposed a self-rewarding language model, where the model provides rewards of its own responses via LLM-as-a-Judge prompting. Meanwhile, OAIF (Guo et al., 2024) samples two responses and employs an LLM annotator to label the positive and negative samples. RSO (Liu et al., 2023) uses rejection sampling to sample preference data from the optimal policy, enabling a more accurate estimation of it. However, the quality of on-policy samples is constrained by the capability of the policy model, which in turn limits training effectiveness.

The challenges mentioned above are particularly pronounced for small models, which face inherent limitations in parameter size and training data. Their reduced capacity to capture complex human-language patterns often results in weaker alignment performance compared to larger LLMs (Bai et al., 2022a). So in our work, we leverage responses from strong models to enhance the upper limit of the capabilities of weak models.

External Guidance for Alignment Considering the high cost of manually collecting alignment data, many current works use a different model to assist in policy training. The most common approach is distillation, which involves using a larger and more powerful teacher model to provide feedback on the preferences of responses. For instance, Zephyr (Tunstall et al., 2023) employs GPT-4 to rank responses from multiple LLMs to obtain preference data. In DPKD (Li et al., 2024b) and PLaD (Zhang et al., 2024), the teacher's outputs and the student's outputs are treated as preferred and dispreferred responses seperately for preference learning.

Moreover, some recent works focus on incorporating refinement to the policy optimization stage. For example, Constitutional AI (Bai et al., 2022b) uses refinement data for reward models, while SELF (Lu et al., 2023) enables the models to self-evolve iteratively, equipping them with the ability to self-refine during inference. Test-Time Preference Optimization (TPO) (Li et al., 2025) translates reward signals into textual critiques and utilizes them as textual rewards to iteratively refine its response. But these methods may bring interfering factors since they directly use responses sampled from the policy model. Regarding the introduction of an external refiner, most previous works only utilized it in the inference phase (Ji et al., 2024; Lou et al., 2025). Recently, however, there have been quite a few works that use it to assist in policy optimization. Yoon et al. (2024) proposed Token-Level Continuous Reward (TLCR), which uses GPT-4 as a reviser to refine responses then assign token-wise preference labels for discriminator training. SynPO (Dong et al., 2024) employs an iterative mechanism wherein a self-prompt generator creates diverse prompts, and a response improver refines model responses progressively. SPAR (Cheng et al., 2024) introduces a refiner to judge the generated responses to collect negative data then employs a tree-search algorithm to refine them, which are then used for model training. Apart from that, in Self-Evolution Fine-Tuning (SEFT) (Chen et al., 2024), the policy undergoes internal and external evolution by being fine-tuned with enhanced responses generated by a trained adaptive reviser. Compared to the methods mentioned above, our method incor-

porates revisions from the reviser for policy optimization, enabling weak policies to learn from both their own responses and externally refined outputs.

B Reviser Training

The process of training the reviser is illustrated in Algorithm 1.

Algorithm 1 Reviser Training Pipeline

Require: initial reviser $\mathcal{R}$, strong models $M_1 \sim M_m$, prompt dataset $\mathcal{D}_p = \{x_i\}_{i=1}^{N}$ of size N, each strong model samples n responses per prompt, number of quantiles k, and a reward model r.

Ensure: Trained Reviser $\mathcal{R}'$

Training Dataset $\mathcal{D}_r = \phi$
for x in $\mathcal{D}_p$ **do**
  Initialize an empty list $Responses$
  for $i = 1$ to m **do**
    for $j = 1$ to n **do**
      Sample a response r from model M_i given prompt x
      Append r to $Responses$
    end for
  end for
  Score and rank the list $Responses$ to get $\hat{y}_1 \sim \hat{y}_{mn}$
  Calculate the k-quantile of $m * n$ to select positions $a_1 \sim a_{k-1}$
  for $i = k - 1$ downto 1 **do**
    if $i == k - 1$ **then**
      $chosen = (x + \hat{y}_{a_{k-1}}, \hat{y}_{a_{k-2}})$
      $rejected = (x + \hat{y}_{a_{k-1}}, \hat{y}_{a_{mn}})$
    else if $i > 1$ **then**
      $chosen = (x + \hat{y}_{a_i}, \hat{y}_{a_{i-1}})$
      $rejected = (x + \hat{y}_{a_i}, \hat{y}_{a_{i+1}})$
    else
      $chosen = (x + \hat{y}_{a_1}, \hat{y}_{a_1})$
      $rejected = (x + \hat{y}_{a_1}, \hat{y}_{a_2})$
    end if
    Append $(chosen, rejected)$ to $\mathcal{D}_r$
  end for
end for
Optimize $\mathcal{R}$ with $\mathcal{D}_r$ according to Eq.(4) to obtain the $\mathcal{R}'$
return the trained reviser $\mathcal{R}'$

C Implementation Details

All experiments are implemented using the TRL library.[3] We train both the reviser and policy models on the augmented UltraFeedback dataset from Section 3, which is randomly split into 30% for SFT and 70% for preference-based optimization.

Reviser Training The reviser is trained in two stages In the SFT warm-up stage, it learns to revise low-quality responses into higher-quality ones using top-vs-bottom ranked responses from strong models. In the second stage, we apply a preference-consistent optimization strategy using quantile-ranked response pairs (with $k = 4$ quantiles), as detailed in Section 2.2.

Policy Training Policy optimization also follows a two-stage procedure. First, SFT is performed on 30% of prompts using either strong model responses or reviser outputs. The remaining 70% of prompts are then used to construct two types of preference pairs: (i) **Self-sampling data**, where the policy samples 5 responses per prompt, and the top vs. bottom-ranked responses (scored by the reward model) form the on-policy pairs; (ii) **Reviser-enhanced data**, where each policy response is revised by the trained reviser, and the revisions are ranked to form revision-based preference pairs.

Baselines We compare ReAlign against the following baselines: (i) **SFT**: Fully supervised tuning using top-ranked strong model responses on all prompts. (ii) **DPO-On / SimPO-On**: On-policy preference optimization using responses sampled from the current policy. (iii) **DPO-Off / SimPO-Off / WPO-Off**: Off-policy preference optimization using strong model responses. These methods begin with 30% SFT warm-up and then train on preference pairs from the remaining 70% prompts. (iv) **SIMPLEMIX**: A hybrid approach that performs 30% SFT warm-up, followed by preference optimization on a random 50/50 mixture of on-policy and off-policy pairs. On-policy samples are drawn from the SFT model.

Hyperparameters Detailed hyperparameter settings are provided in Table 4. We train all models using a maximum sequence length of 2048 and adopt the adam optimizer (adam_torch) with a cosine learning rate scheduler and a warm-up ratio of 0.03. During the supervised fine-tuning (SFT)

[3] https://github.com/huggingface/trl

Table 4: Hyperparameter configurations used for each model and alignment method.

Method	Batch Size	Learning Rate	β	γ
Llama-3.2-3B-Instruct				
SFT	128	5.0e-6	-	-
DPO-On	128	5.0e-7	0.01	-
DPO-Off	64	5.0e-7	0.01	-
SimPO-On	128	1.0e-6	10	3
SimPO-Off	64	5.0e-7	10	3
WPO-Off	32	8.0e-7	0.03	-
SIMPLEMIX	64	5.0e-7	0.01	-
ReAlign (DPO)	128	5.0e-7	0.01	-
ReAlign (SimPO)	128	5.0e-7	10	3
Qwen2.5-3B-Instruct				
SFT	128	5.0e-6	-	-
DPO-On	128	5.0e-7	0.01	-
DPO-Off	128	5.0e-7	0.01	-
SimPO-On	128	1.0e-6	10	3
SimPO-Off	128	1.0e-6	10	3
WPO-Off	32	8.0e-7	0.03	-
SIMPLEMIX	128	5.0e-7	0.01	-
ReAlign (DPO)	64	1.0e-6	0.01	-
ReAlign (SimPO)	128	1.0e-6	10	3
Llama-3.2-1B-Instruct				
SFT	128	5.0e-6	-	-
DPO-On	32	5.0e-7	0.01	–
DPO-Off	128	5.0e-7	0.01	–
SimPO-On	32	5.0e-7	10	3
SimPO-Off	128	1.0e-6	10	3
WPO-Off	32	8.0e-7	0.03	-
SIMPLEMIX	128	5.0e-7	0.01	-
ReAlign (DPO)	32	1.0e-6	0.01	-
ReAlign (SimPO)	128	1.0e-6	10	3

stage, models are trained for 3 epochs. In contrast, all preference optimization stages, including DPO, SimPO, WPO, SIMPLEMIX, and ReAlign, are trained for 1 epoch.

D Derivation and Interpretation of the λ

D.1 From the DPO Objective to Data-Dependent Weights

Most preference–based alignment objectives can be written in the form

$$\mathcal{L} = \mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}} \Big[-\log\sigma\Big(\beta\big[\log\pi_\theta(y_w \mid x) - \log\pi_\theta(y_l \mid x)\big] - \beta\big[\log\pi_{\text{ref}}(y_w \mid x) - \log\pi_{\text{ref}}(y_l \mid x)\big]\Big)\Big], \tag{6}$$

where π_θ is the policy, π_{ref} the (frozen) reference model, $\beta > 0$ controls the strength of the KL divergence term constrains the deviation of the policy π_θ from the reference model. Letting $\Delta_\pi := \log\pi_\theta(y_w \mid x) - \log\pi_\theta(y_l \mid x)$ and $\Delta_{\text{ref}} := \log\pi_{\text{ref}}(y_w \mid x) - \log\pi_{\text{ref}}(y_l \mid x)$, one obtains the gradient.

$$\nabla_\theta\mathcal{L} \;=\; -\beta\,\mathbb{E}_{\mathcal{D}}\big[\underbrace{\sigma\big(\beta(\Delta_{\text{ref}} - \Delta_\pi)\big)}_{\lambda(x,y_w,y_l)}\;\nabla_\theta\big(\Delta_\pi\big)\big]. \tag{7}$$

Hence every preference sample is re-weighted by

$$\boxed{\lambda \;=\; \sigma\big(\beta(\Delta_{\text{ref}} - \Delta_\pi)\big)} \qquad \in (0,1).$$

Intuitive meaning.

- If the reference already *strongly* prefers y_w over y_l ($\Delta_{\text{ref}} \gg \Delta_\pi$), then $\lambda \to 1$ and the sample receives a large gradient step.
- If the policy is aligned with the reference ($\Delta_{\text{ref}} \approx \Delta_\pi$), then $\lambda \approx 0.5$, yielding a moderate update.
- If the policy *over-prefers* y_w relative to the reference ($\Delta_{\text{ref}} < \Delta_\pi$), λ becomes small, down-weighting a potentially harmful sample.

Consequently, the distribution of λ after training reveals which subset of data the model finally learns from the most.

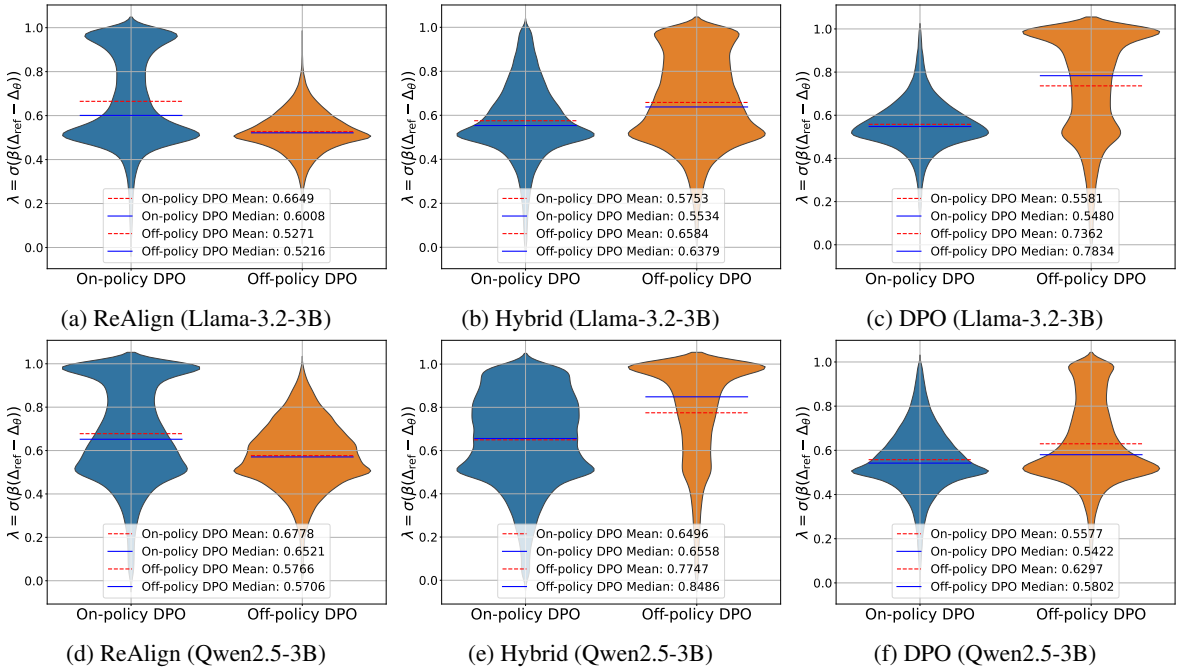
D.2 Practical Computation in Our Experiments

For every final checkpoint we recompute λ on all training pairs with $\beta = 0.01$:

- **ReAlign**: π_{ref} equals the warm-up SFT model; we report λ separately for on-policy pairs $(x, y_w^{\mathcal{P}}, y_l^{\mathcal{P}})$ and revision-task pairs $(x, y_w^{\mathcal{P}}, y_w^{\mathcal{R}}, y_l^{\mathcal{R}})$.
- **Hybrid**: π_{ref} also equals the warm-up SFT model, but pairs are a *direct* mixture of on-policy and strong-model off-policy data.
- **DPO**: two single-source baselines, trained on only on-policy or only off-policy data and evaluated on their respective training sets.

D.3 Empirical λ Distributions

Figure 5 and Figure 6 (Llama-3.2-3B-Instruct / Qwen2.5-3B-Instruct) plot the resulting λ distributions. Aggregate statistics for the 1B model are shown in Table 5.



(a) ReAlign (Llama-3.2-3B) (b) Hybrid (Llama-3.2-3B) (c) DPO (Llama-3.2-3B)

(d) ReAlign (Qwen2.5-3B) (e) Hybrid (Qwen2.5-3B) (f) DPO (Qwen2.5-3B)

Figure 6: Distribution of λ values on on-policy and off-policy data after training of Llama-3.2-3B-Instruct and Qwen2.5-3B-Instruct.

Scheme	Source	Mean	Median
ReAlign	On-policy	0.756	0.807
	Revision-task	0.578	0.570
Hybrid	On-policy	0.704	0.743
	Strong Off-policy	0.773	0.849
DPO	On-policy only	0.670	0.690
	Off-policy only	0.626	0.597

Table 5: Mean and median λ on Llama-3.2-1B-Instruct.

ReAlign The on-policy density is sharply peaked around $\mu = 0.756$, median $= 0.807$, indicating high reference–policy agreement. Revision-task pairs are harder ($\mu = 0.578$) yet still centred well above 0.5, showing that ReAlign retains informative off-policy signals without conflicting with its own distribution.

Hybrid Hybrid assigns *higher* weight to strong-model pairs ($\mu = 0.773$) than to on-policy ones ($\mu = 0.704$), confirming that the external completions dominate training. Coupled with the loss curves in Section 6, this suggests a risk of distributional overstretch.

Standard DPO When trained solely on off-policy data, the mean λ is $\mu = 0.626$ (median 0.597), lower than the on-policy counterpart ($\mu = 0.670$). This indicates that the weak policy does obtain non-negligible gradients on external pairs, yet these signals remain less aligned than on-policy ones. Conversely, on-policy-only DPO maintains moderate weights (≈ 0.67), confirming better reference–policy agreement within its own distribution.

These findings substantiate our claim that revision-task off-policy supervision offers learnable yet non-conflicting guidance, yielding stable improvement across model scales.

E Results of Downstream Benchmarks

To assess out work on downstream tasks, we conducted experiments across eight tasks spanning general knowledge, mathematics, and programming domains. The tasks are described as follows:

MMLU (Hendrycks et al., 2021): A multiple-choice dataset to evaluate knowledge capability, which covers 57 tasks including elementary mathematics, US history, computer science, and more.

HellaSwag (Zellers et al., 2019): A commonsense reasoning benchmark requiring models to choose the most plausible continuation.

GSM8K (Cobbe et al., 2021): A dataset of 8.5K linguistically diverse grade-school math word problems to evaluate mathematical reasoning capability.

BBH (Suzgun et al., 2023): A subset of the BIG-Bench benchmark comprising 23 challenging tasks

Method	MMLU 5-shot Acc	Hellaswag 10-shot Acc Norm	GSM8K 8-shot, CoT Acc	BBH 3-shot Acc Norm	GPQA 0-shot Acc Norm	MuSR 0-shot Acc Norm	Winogrande 5-shot Acc	TruthfulQA 0-shot Acc	Average
Llama-3.2-3B-Instruct									
SFT	58.33	73.83	74.37	39.56	30.37	35.98	70.17	41.24	52.98
DPO-On	60.18	75.74	75.89	41.99	31.96	36.90	62.43	42.58	53.46
DPO-Off	58.94	75.73	74.22	42.04	29.70	39.29	67.72	36.24	52.99
SimPO-On	60.28	75.55	77.03	42.39	32.47	38.10	67.48	44.95	54.78
SimPO-Off	60.01	76.51	67.17	42.35	30.70	38.62	65.75	54.81	54.49
WPO-Off	59.18	76.01	74.37	42.74	30.54	35.05	67.09	42.58	53.45
SIMPLEMIX	59.64	75.49	73.39	41.12	31.63	35.32	67.72	44.04	53.54
ReAlign (DPO)	59.69	74.87	75.36	41.12	31.80	35.85	69.06	43.25	53.88
ReAlign (SimPO)	59.17	76.05	73.09	42.04	30.45	38.62	67.32	39.93	53.33
Qwen2.5-3B-Instruct									
SFT	65.65	75.01	77.10	42.56	32.13	41.14	69.46	45.40	56.06
DPO-On	66.39	75.67	81.05	42.34	31.12	39.68	66.06	48.99	56.41
DPO-Off	65.68	76.40	79.38	43.69	31.80	41.53	67.64	44.92	56.38
SimPO-On	66.35	75.47	81.27	43.69	32.55	41.14	68.35	50.54	57.42
SimPO-Off	65.83	76.87	72.10	44.32	31.04	41.40	68.11	48.67	56.04
WPO-Off	65.67	76.69	79.83	42.84	31.29	40.48	65.27	42.03	55.51
SIMPLEMIX	66.39	75.64	81.35	44.56	30.45	39.02	62.27	45.54	55.65
ReAlign (DPO)	66.40	75.71	79.68	44.78	30.37	41.80	67.56	48.59	56.86
ReAlign (SimPO)	65.64	76.07	77.79	43.78	32.55	41.80	68.98	47.44	56.76
Llama-3.2-1B-Instruct									
SFT	43.54	61.71	42.91	32.46	26.76	33.60	60.62	38.09	42.46
DPO-On	46.48	62.89	42.46	30.57	24.50	37.57	56.43	36.18	42.14
DPO-Off	45.28	62.75	42.76	32.96	25.92	32.01	60.46	31.96	41.76
SimPO-On	46.59	63.48	40.26	33.97	27.68	32.01	57.22	41.21	42.80
SimPO-Off	45.83	63.07	41.09	33.00	25.25	32.01	61.64	34.60	42.06
WPO-Off	45.03	64.57	39.12	31.56	25.50	33.20	60.46	32.44	41.49
SIMPLEMIX	45.81	63.64	44.35	32.77	26.51	30.29	59.04	37.99	42.55
ReAlign (DPO)	46.01	63.93	41.32	31.31	26.26	30.82	60.06	37.76	42.18
ReAlign (SimPO)	45.23	63.98	39.95	31.92	27.27	30.95	60.30	31.22	41.35

Table 6: Performance comparison results across multiple benchmarks.

where prior LLMs performed poorly. It tests emergent abilities such as multistep reasoning, hierarchical planning, and nuanced understanding.

GPQA (Rein et al., 2023): A challenging knowledge benchmark crafted by PhD-level domain experts in biology, physics, and chemistry.

MuSR (Sprague et al., 2024): A dataset comprising algorithmically generated complex problems, such as murder mysteries, object placement challenges, and team allocation optimizations.

Winogrande (Sakaguchi et al., 2021): A set of adversarial and difficult Winograd benchmarks for commonsense reasoning.

TruthfulQA (Lin et al., 2022): A benchmark dataset for evaluating the truthfulness of LLMs and their ability to avoid falsehoods.

The results summarized in Table 6 reveal several trends regarding general task performance. Across most benchmarks, on-policy methods such as DPO-On and SimPO-On better preserve the model's original capabilities. This holds across model scales and supports the intuition that on-policy optimization maintains the model's native behavior by training within its own generation distribution.

While ReAlign does not always surpass the strongest on-policy baselines on every task, it consistently delivers competitive results across models and benchmarks. These outcomes suggest that the revision-task supervision introduced by ReAlign enables effective preference alignment without substantially degrading performance on broader reasoning and language tasks.